

AdScope: Intelligent Scoping of Paid Search Campaigns using Relevance Feedback

A thesis submitted to the
Graduate School of Natural and Applied Sciences

by

Kevser Nur OĐALMIŐ

in partial fulfillment for the
degree of Master of Science

in

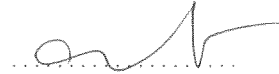
Electronics and Computer Engineering



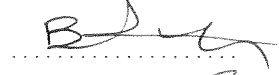
This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science in Electronics and Computer Engineering.

APPROVED BY:

Assist. Prof. Ahmet Bulut
(Thesis Advisor)



Assist. Prof. Barış Arslan



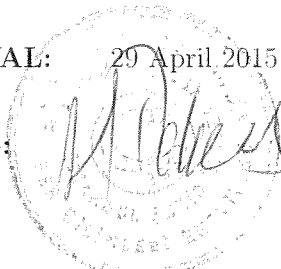
Prof. Dr. Aslihan Nasır



This is to confirm that this thesis complies with all the standards set by the Graduate School of Natural and Applied Sciences of İstanbul Şehir University:

DATE OF APPROVAL: 29 April 2015

SEAL/SIGNATURE:



Declaration of Authorship

I, Kevser Nur OGALMIŐ, declare that this thesis titled, 'AdScope: Intelligent Scoping of Paid Search Campaigns using Relevance Feedback' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Kevser

Date: 29 APRIL 2015

“Most people say that it is the intellect which makes a great scientist. They are wrong: it is character.”

Albert Einstein

“Everything is theoretically impossible, until it is done.”

Robert A. Heinlein

AdScope: Intelligent Scoping of Paid Search Campaigns using Relevance Feedback

Kevser Nur OĐALMIŐ

Abstract

In this thesis, we propose a semi-supervised online tool called AdScope for search engine marketing. AdScope can be used for filtering out unprofitable user queries from the search campaign while at the same time allowing profitable queries only. AdScope uses relevance feedback for classifying user queries broadly into two categories as relevant or non-relevant. All queries labeled as non-relevant are excluded from the search campaign; no ad is shown to a user posing an excluded query in the future. All queries labeled as relevant are included in the search campaign as regular campaign keywords. In order to label queries, two sources of relevance feedback are used: user feedback comes in the form of clicks and conversions which are available in the search query log provided by ad broker. Advertiser feedback is collected interactively. For this purpose, we designed an active learning step where advertiser is asked to label a selected set of unlabeled queries. The feedback received is incorporated into the classification model in real time using Bayesian update. In performance tests, we observed that AdScope had the highest classification accuracy of 89.25% for queries that contain at least two terms. Furthermore, three domain experts agreed substantially with a Fleiss' agreement score of 0.79 on the selections made by our actively learning system.

Keywords: Online Advertising, Search Campaign Optimization, Active Learning, Relevance Feedback

AdScope: Ücretli Arama Kampanyaları İçin İlişkili Geri Bildirimleri Kullanarak Akıllı Kapsam Belirleme

Kevser Nur ÇOĞALMIŞ

ÖZ

Bu tez çalışmasında, arama motoru bazlı pazarlama kampanyaları için AdScope adında yarı-denetimli bir çevrimiçi araç sunulmaktadır. AdScope herhangi bir kampanya için kazançsız olan kullanıcı sorgularını elerken, aynı zamanda kazanç sağlayabilecek kullanıcı sorgularını eklemek için de kullanılabilir. AdScope kullanıcı sorgularını ilişkili ve ilişkisiz olmak üzere iki ayrı kategoride sınıflandırmak için ilişkili geri bildirim bilgisini kullanır. İlişkisiz (non-relevant) olarak işaretlenen sorgular kampanyanın kapsamı dışında bırakılır; ileride bu sorgu cümleleri kullanıcıya tekrar gösterilmez ve sorgulanmaz. İlişkili (relevant) olarak işaretlenen sorgular ise kampanyaya ait anahtar kelimeler olarak dahil edilir. Sorguları işaretlemek için iki ayrı ilişkili geri bildirim kaynağı kullanılır. Bu kaynaklardan biri, tıklama sonucu elde edilen kullanıcı geri bildirim ve kullanıcının oturumu satın alma aksiyonu ile bitirmesidir. Bu bilgiler reklam sağlayıcı tarafından arama sorgusu kayıtlarında tutulur. Reklamcı geri bildirimini interaktif bir şekilde elde edilir. Bu amaçla, reklamcının işaretlenmemiş sorguları işaretleyebileceği aktif bir öğrenme adımı tasarlanmıştır. Bu adımda elde edilen geri bildirim, Bayes eşitliği kullanılarak sınıflandırma modeline gerçek zamanlı olarak entegre edilir. Yapılan performans testlerinde, en az iki tane kelime içeren sorgular için, AdScope %89.25 sınıflandırma doğruluğu göstermiştir. Ayrıca reklamcı geri bildirimleri kullanılarak bu sistem tarafından önerilen sorgu seçimlerinin değerlendirilmesinde, Fleiss' Kappa skoru ile üç ayrı yorumcunun büyük ölçüde aynı fikirde olduğu saptanmıştır.

Anahtar Sözcükler: İnternet Reklamcılığı, Arama Kampanya Optimizasyonu, Aktif Öğrenme, İlgili Geri bildirim

*This thesis is dedicated to my family, especially to my twin sister for
being always with me.*

Acknowledgments

I would like to express my sincere gratitude to my advisor Prof. Ahmet Bulut for his guidance, support, motivation and time during my whole graduation study. He was always tend to be helpful for any kind of problem about my work. His sympathy was there all time even I did not know or decide what to do next. He never said it is enough and you can give up on working. He pushed me go over one step next in each time to make my results better and certain. I learned from him that there is always a way to make a research better than previous version. I am thankful for his faith on me.

I would like to thank my thesis committee: Prof. Dr. Aslihan Nasır and Assist. Prof. Barış Arslan for their time and comments.

I thank my fellows in Data Science Lab, Oğuzhan Sağoğlu for his contributions on Ad-Scope system, and Aslan Bakirov for his endless help and patience while teaching me distributed programming. I am grateful the knowledge that I gain from him in this field.

I appreciate my parents for being supportive throughout my education and every single opportunity they provided.

This thesis is supported by Turkish National Science Foundation (Tübitak) under grant number 113E243.

Contents

Abstract	ii
Öz	iii
Acknowledgments	v
List of Figures	viii
List of Tables	ix
Abbreviations	x
1 Introduction	1
1.1 Our contribution	3
2 Literature Review	5
3 Methodology	9
3.1 Computation of Relevance Status Value	10
3.2 User feedback and bootstrapping in AdScope	11
3.3 Active learning in AdScope	12
3.3.1 Streaming algorithm for incorporating advertiser feedback	12
3.3.2 Getting advertiser feedback	13
3.4 Complexity analysis of AdScope	15
4 Experimental Evaluation	18
4.1 Details of our datasets	18
4.2 Testing methodology	18
4.3 Self comparison	19
4.3.1 With pre-processing vs. without pre-processing	19
4.3.2 Active learning in AdScope	19
4.4 Comparison with the state of the art	21
4.4.1 Multinomial Naive Bayes	22
4.4.2 Binary classifiers	22
4.4.3 Markov chain	23
4.4.4 Comparative results	24
4.4.5 Performance on the second dataset	26
5 Phrase discovery in AdScope	27

6	Conclusions	30
A	Computation of Jaccard score for trigrams	32
	Bibliography	34

List of Figures

3.1	The split-paned query suggestions page	14
3.2	Dragging illustration in AdScope	15
3.3	AdScope's computation time in seconds vs. number of queries	16
3.4	AdScope's computation time in seconds vs. query length	17
4.1	The distribution of RSVs	24

List of Tables

3.1	The table of notations	10
4.1	Accuracy of AdScope	19
4.2	Inter-annotator agreement between three domain experts	20
4.3	Cohen’s kappa results	21
4.4	Markov Chain vs. Multinomial Naive Bayes vs. AdScope	25
4.5	SVC vs. LR vs. AdScope	25
5.1	Bigram phrases by Jaccard Score	29
5.2	Trigram phrases by Jaccard Score	29

Abbreviations

AD	A dvertisement
IR	I nformation R etrieval
BIM	B inary I ndependence M odel
PRP	P robability R anking P inciple

Chapter 1

Introduction

Internet usage is growing at a rapid rate. In the US, the number of minutes spent online increased from 497 billion minutes in May 2010 to 958 billion minutes in May 2013 [1]. During the time spent online, Internet users keep searching the web for more and more relevant information, such as where to find the best online deals, where to go on a vacation at this time of the year, what to wear to a wedding, and many others. Online sales started to compete with offline sales in many business segments. Therefore, companies started to pay an increasing amount of attention and budget in advertising online. A survey that was run in 2012 found out that while 78% of online users trust peer recommendations, only %14 of users trust online advertisement [2].

Online advertising is used for two main goals: (1) reaching a broad set of online users, and (2) increasing online conversions, e.g., sales. Reaching a broad set of users is necessary for raising brand awareness. For this purpose, rich media ads such as banner ads and video ads are used. For branding, the advertiser usually ignores whether or not the user has an intention to buy. However for conversions, the advertiser is keenly interested in the user's "intention to buy". With a carefully crafted keyword such as "how to create a social network for cheap", the advertiser has a better chance of engaging with a potential buyer with better odds to convert [3–5]. Focus of this thesis is increasing sales with keyword ads.

If a search advertiser wants to advertise on say Google, she has to express her product with a well-chosen set of advertising keywords. These keywords will be used to target customers, as they are searching the Internet for an answer to their need. The user's need

should be exactly what the company's product caters to. It is important to select the correct set of keywords for targeting the correct set of customers. Once the right keywords are determined, an appropriate ad message is put together, the keyword campaign can go live after being certified by the ad broker (usually the search engine itself).

Once a search campaign keyword matches with a user's search query, then the corresponding ad is eligible for entering into a bid-auction alongside with other matching ads. Eligible ads are ranked by their bid times their quality score (the details of how to compute this score are proprietary to the ad brokers) and are displayed in that order to the user. If the user decides to click on one of these ads, she will be re-directed to the destination URL designated by the advertiser. In return, the ad broker charges the advertiser using the available information such as the bid amount, the quality score, and the second highest bid in the auction.

Search campaign management dashboards provided by most ad brokers give detailed information on per-campaign spend, total conversions, total users, total impressions and information about what search terms¹ are used by users. Experienced advertisers periodically go through the long list of search terms and manually cherry-pick the relevant (i.e., positive) search terms to expand, and non-relevant (i.e., negative) search terms to shrink the reach and scope of their campaigns. This manual and at the same time tedious process has the following drawbacks:

- i. Each search term is individually evaluated. Since this process is far from being principled and consistent across time, it tends to result in suboptimal performance compared to a global evaluation across many keywords. For example, a search term may be labeled as negative six months ago, while a "similar" search term may be labeled as positive more recently. In order to detect such inconsistencies within a continually growing portfolio of search terms is challenging. The manual process is inadequate to provide a clear solution.
- ii. Within a large set of search terms classified as either positive or negative, common phrases start to arise within each set of positive and negative search terms. It is hard to detect such phrases manually, and depends highly on advertiser experience. If an advertiser is responsible for managing many campaigns for many products for many seasons and special occasions, relying solely on advertiser experience is neither

¹Search query and search term is used interchangeably.

sustainable and nor scalable. If an automated tool detects such phrases, then a positive or negative phrase match would expand or shrink the campaign scope more aggressively. Furthermore, such effective scoping would reduce the frequency and the amount of time it takes to do a manual evaluation, and as a result would increase the total number of conversions.

- iii. The vocabulary used by the advertiser to describe a product and its segment verbally, and the vocabulary used by the users searching online for that product may be different. This problem is called impedance mismatch, and makes it harder for advertisers to do effective scoping [6]. The problem arises because an advertiser does not have a-priori knowledge of the set of all relevant search terms. Inability to target effectively would result in market share going to the competition. The only way to alleviate the impedance mismatch problem is to take the conversions data into account. This effectively means that a conversion event, i.e., user feedback for a relevant search term, has to be used while building a classifier. If we can unify the domain expert's feedback with user feedback, then we can build a classifier for scoping without suffering from impedance mismatch.
- iv. The manual nature of the process makes it hard to seamlessly build upon prior knowledge and experience.

1.1 Our contribution

A semi-automated system (with potential to be fully automated) was built that can alleviate the major drawbacks of a purely manual process outlined above. We unified the user feedback in the form of conversions with the feedback provided by the domain expert. The collected feedback is used to build a conversion model. The model is built up on the Binary Independence Model (BIM), which has good performance in information retrieval (IR) tasks on short documents and abstracts. Since search terms usually do not exceed more than ten words, the model is an appropriate choice for the task at hand. The BIM model uses the probability ranking principle (PRP), and at its core is the multivariate Bernoulli Naive Bayes model [7]. In order to integrate advertiser feedback into the model, We used a Bayesian update process with weighing prior belief in an

iterative manner. The model parameters are log of odds-ratio per feature. The resulting model can:

- i. sustain the know-how built over time within the model and incorporate new advertiser feedback using a well-defined update procedure.
- ii. be used to discover and rank phrases existing within search terms in order of importance, and present only the top few of them to an advertiser, who usually has limited availability because of managing multiple campaigns.
- iii. evaluate each search term holistically (while considering all other search terms) rather than individually.
- iv. provide a solution to inherent impedance mismatch, because user provided feedback is used for building the model.

Chapter 2

Literature Review

Search advertising is an active research area. Selecting the right set of keywords is one of the most popular research topics besides budget optimization, and also is the closest in scope to the focal point in that work. Common phrases that occur in a given set of search queries are frequently used as keyword candidates. In a fairly recent work, a feature selection algorithm was used for ranking the common phrases in search queries according to their performance using historical data [8]. The top phrases in the ranked list were used in order to extend campaign keywords and to make them more specific. Their goal was to increase overall campaign profitability by making keywords more relevant. Only user feedback was used in this work while ignoring the advertiser side of the problem. This is a crucial difference between this and our work since this thesis strongly advocates the use of advertiser opinion, i.e., expert opinion. When combined with bag of words features, human expert tips such as whether an ad has a call-to-action and whether it contains free as keyword result in a better estimation of the relevance order of a set of competing ads to a query keyword [9].

The use of both advertiser and user feedback was considered for recommending keywords that are relevant to an input set of seed terms [10]. The relevance relationship between two keywords was established by whether advertisers co-bid on those two keywords. In addition, search click logs were used as indicators of end user preference for establishing relevance of a given search term to a target URL. The user's decision to click on a specific URL was considered as user feedback. A logistic regression model was learned on features that represent associative relationships between terms such as the number of times a

pair of terms target the same URL, and features that represent term specificity such as whether a given term targets many URLs vs. just a few URLs. In their experiments, the logistic regression model performed as good as a standard collaborative filter.

Similar to our methodology, active learning was used for keyword suggestion in online advertising [11]. First, a seed term was determined. Then, a search engine query with this seed term was made for retrieving the search results to the query. From the search results, keywords with the highest TF-IDF scores were extracted out, and a select few of them were presented to human annotators for evaluation. The keyword selection was based on transductive experimental design [12]. In the experiments, it led to a minor improvement when compared to using random sampling. Furthermore, active learning was used for getting conversion labels faster by targeting consumers, who provide the most information to improve the quality of the predictive model to be learned [13]. The intuition behind their approach was that users, who share similar Web browsing behaviors, tend to have similar preference over ads.

Keyword relevance to a target webpage was computed using the information found in the page such as where in the page a potential keyword occurs [14]. A keyword occurrence within bold tags was given higher importance than an occurrence in plain text within the body. Similarly, if a keyword appeared within meta section, that specific occurrence was considered as more significant compared to an occurrence within anchor text. The experiments showed comparable results to commercial tools in use today [15].

Popular keyword suggestion methods generally use statistical information and leaves the semantics aside. In order to improve the quality of the suggestions made, the semantic relationships between keywords were taken into account [16]. The Open Directory Project¹ (ODP) ontology holds semantic relationships between entities, which belong to well-defined concepts. Each concept within ODP contains a set of webpages that are categorized under that concept. The textual contents of these webpages were used as a repository for suggesting new keywords. In order to use ODP for keyword suggestion, a seed term was first matched to a concept within ODP. Once the corresponding concept was found, the concept hierarchy was traversed to find other concepts that were relevant to the primary concept.

¹<http://www.dmoz.org>

A more advanced approach for keyword suggestion is to use a generative translation model together with a suitable language model. The translation model is used to formulate the probability of a given term matching with another term in a target text. The higher is the probability, the more relevant is the term to the target. On top of this, an n -gram based language model can be used in order to learn meaningful phrases. The two models together were used for generating non-intuitive keywords for a given keyword portfolio [17]. This approach is complementary to our work as it can be used to create new keywords for expanding a keyword portfolio horizontally.

Using contextual relationships between potential keywords for keyword suggestion was considered in TermsNet [18]. For each keyword, first a search engine query was made with that keyword. Then, a context for the keyword was built using the first fifty search results to the query. Using keyword context, a directed graph was constructed where keywords represent nodes, and edges represent suggestiveness. The weight on a directed edge $A \rightarrow B$ represents the frequency of the keyword B in the context of the keyword A . The larger is the weight on the edge, the stronger is the suggestiveness. That is, the keyword A suggested the concatenated keyword phrase $A B$, or that the keyword A was a potential keyword recommendation for the query keyword B . Search engine results to a query were further used in Wordy for establishing a relationship between seemingly distant terms [19]. For example, given three terms A , B , and C , if A and B co-occur and B and C co-occur separately, then A and C can also be associated with each other even though they never co-occur.

The search logs and the webpage contents provide the relevance association between the search queries and the result pages [20]. For popular webpages, there is abundant data available in the search logs since such pages appear frequently in search results. Furthermore, there is rich contextual information readily available for popular pages in the form of metadata. Therefore, it is easy to compute the relevance score of a popular page by using a ranking algorithm. However for less popular pages, the available data is sparse. Such tail pages rarely occur in search results, and there is less context information. The empirical data shows that 75% of tail pages have less than one anchor text. Due to their ranking low in relevance order, there is less click data available, which complicates the relevance computation further. The authors proposed a search-focused key n -gram approach in order to improve relevance score of tail pages by exploring such pages. They proposed to extract search-focused information from popular pages. Search

focused information corresponds to mapping each web page to a set of keywords by using page title, anchor data, and the body of the page. From these keywords, key n -grams were extracted to be used as features in the learning step. The learning model was trained using the search log and the key n -grams extracted. The experiment results showed that the relevance score of tail pages indeed improved by using search focused information.

Users choice of what to click depends on micro and macro factors. The total number of ads displayed (ad depth), the interaction between ads and the query (query diversity), and the types and the qualities of ads next to a target ad were considered as micro factors [21]. The interaction between organic search results and sponsored search results were considered as a macro factor. The empirical studies showed that organic search and sponsored search were negatively correlated. Therefore, diversity in both results should be preferred. In organic search, it is likely that the top website is clicked due to its high relevance, while in sponsored search this is not always true. The ad depth and the purchase intent behind the query affected the click behavior.

Chapter 3

Methodology

Given a user query $q \in Q$ that contains an information need, the probability of q being relevant to the campaign is denoted by $P(R = 1 | q)$. The probability value can be used for ranking queries Q in a decreasing order of relevance to the campaign. This ranking principle based on probabilities is known as PRP. Similarly, the probability of q being *not* relevant to the campaign is denoted by $P(R = 0 | q)$.

In the term vector space of dimension K for an ordered vocabulary V of K terms, the query q is represented by the term vector $\vec{q} = (e_1, \dots, e_i, \dots, e_K)$, where each e_i corresponds to the occurrence of a term $t_i \in V$ in query q . The value e_i is equal to 1 if the corresponding term occurs in the query, and it is 0 otherwise. Since BIM assumes independence of terms, the order of terms in the query is not taken into account. In order to ease the comprehension of the formal model, the full table of notations is given in Table 3.1. Using Bayes rule, the probability of q being relevant to the campaign can be computed as follows:

$$\begin{aligned} P(R = 1 | q) &\approx P(R = 1 | \vec{q}) \\ &\approx P(\vec{q} | R = 1) \times P(R = 1) \end{aligned}$$

Similarly, the probability of q being *not* relevant to the campaign can be computed as follows:

$$\begin{aligned} P(R = 0 | q) &\approx P(R = 0 | \vec{q}) \\ &\approx P(\vec{q} | R = 0) \times P(R = 0) \end{aligned}$$

TABLE 3.1: The table of notations

Entity	Symbol
a search query	q
a term in a search query	t
training corpus (term vocabulary)	V
the set of queries	Q
the set of relevant queries suggested by AdScope	Q^+
the set of non-relevant queries suggested by AdScope	Q^-
the set of true positives in Q^+ confirmed by the advertiser	Q_c^+
the set of true positives in Q^- confirmed by the advertiser	Q_c^-
the number of queries in Q_c^+ that include term t	Z
the number of queries in Q_c^- that include term t	Y
relevance threshold for RSV	τ
the weight of prior belief in active learning	κ

For each term $t_i \in V$, we compute two probability values p_{t_i} and u_{t_i} . The value p_{t_i} corresponds to the probability of t_i being present in relevant queries, while u_{t_i} corresponds to the the probability of t_i being present in a non-relevant queries. More formally:

$$p_{t_i} = P(e_i = 1 \mid R = 1)$$

$$u_{t_i} = P(e_i = 1 \mid R = 0)$$

3.1 Computation of Relevance Status Value

Given p_t and u_t for term t , the odds of the term appearing in a relevant document equals to $\frac{p_t}{(1-p_t)}$, and the odds of the term appearing in a non-relevant document equals to $\frac{u_t}{(1-u_t)}$. The ratio of these two odds is called the odds ratio, the log of which corresponds to the log odds ratio. The log odds ratio c_t for term t is computed as follows:

$$c_t = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{(1-u_t)}{u_t} \quad (3.1)$$

Using per term log odds ratios, the relevance status value (RSV) of an unlabelled query q is computed as follows [7]:

$$RSV_q = \log \prod_{t:e_t=1} \frac{p_t(1-u_t)}{u_t(1-p_t)} \quad (3.2)$$

$$= \sum_{t:e_t=1} \log \frac{p_t(1-u_t)}{u_t(1-p_t)} \quad (3.3)$$

$$= \sum_{t:e_t=1} c_t \quad (3.4)$$

where the log odds ratios for the terms appearing in the query are summed up in order to obtain the RSV of the query.

3.2 User feedback and bootstrapping in AdScope

The search query log provided by ad broker contains several types of information per search query such as the number of clicks received, the total cost, the average cost per click, the average position, the number of converted clicks, the number of conversions, the average cost per converted click, the click conversion rate (CR), the click-through rate (CTR), and many other characteristics. For a subset of the queries, there is also the advertiser's annotation as relevant or non-relevant to the scope of the search campaign. We used only three types of information: the search query, the advertiser's annotation if any, and the number of converted clicks cc . The conversion information was used as user relevance feedback. All queries that were converted can be considered as relevant while all queries that did not convert after receiving enough clicks can be considered as non-relevant. The user feedback is collected periodically by pulling the query log from the ad broker. In the latest pull, only the delta change from the previous pull is taken into account.

For each query in the log that contains enough information for relevance judgment, a class label is assigned using the following decision rules:

$$\begin{aligned} & \text{If } (cc > 0) \vee (\text{advertiser annotation} = \text{relevant}) \text{ then } R = 1 \\ & \text{If } (cc = 0) \wedge (\text{advertiser annotation} = \text{nonrelevant}) \text{ then } R = 0 \end{aligned} \quad (3.5)$$

Using initial user feedback and advertiser feedback, AdScope bootstraps the RSV-based model by computing p_t , u_t , and c_t values for all terms in the vocabulary. For an unlabeled query q , AdScope can compute its RSV and make a relevance judgment for it as follows:

$$\text{Query } q \text{ is relevant if and only if } RSV_q > \tau$$

where the threshold τ is initialized using a linear classifier [22].

3.3 Active learning in AdScope

Active learning is used for training classifiers with less training data than required in a regular supervised approach. The key idea behind active learning is that when the learning algorithm is allowed to choose the data from which it learns, then it can perform up to par with less training data. This is valuable in situations where unlabeled data is abundant, but labeling them is expensive. In AdScope, RSV is used as a selection measure for choosing a subset of queries to learn from. AdScope identifies unlabeled queries with very high RSV scores and unlabeled queries with very low RSV scores, and presents them to the advertiser for relevance feedback. The advertiser feedback is used to update c_t values for all terms in real time (see Section 3.3.1 for details). The advantage of this approach is that it does not require re-training from scratch using the query log and the newly provided feedback since the computation of RSV relies on c_t values being updated incrementally.

3.3.1 Streaming algorithm for incorporating advertiser feedback

Assume that AdScope suggests to the advertiser two sets Q^+ and Q^- as relevant and non-relevant queries respectively. From these two sets, the advertiser selects the correct ones as Q_c^+ and Q_c^- . With this new feedback obtained, the previous p_t and u_t values for term t at time i can be updated at time $i + 1$ as follows:

$$p_t^{i+1} = \frac{Z + \kappa \times p_t^i}{Q_c^+ - Z + \kappa} \quad (3.6)$$

$$u_t^{i+1} = \frac{Y + \kappa \times u_t^i}{Q_c^- - Y + \kappa} \quad (3.7)$$

where Z denotes the number of queries that include term t among the correctly labeled relevant queries Q_c^+ , Y denotes the number of queries that include term t among the correctly labeled non-relevant queries Q_c^- , and κ denotes the weight of prior belief. For large values of κ , the prior probability value is strongly weighted whereby the new estimates do not change too much from the evidence provided by a small number of suggestions. The c_t at time $i + 1$ can easily be computed using the updated probability values in the following way:

$$c_t^{i+1} = \log \frac{p_t^{i+1}}{(1 - p_t^{i+1})} + \log \frac{(1 - u_t^{i+1})}{u_t^{i+1}} \quad (3.8)$$

3.3.2 Getting advertiser feedback

In order to collect advertiser feedback, we developed a web application for advertisers using Django [23]. In the front-end tier, we used Twitter Bootstrap [24] and jQuery UI [25]. In the database tier, we used MySQL. Note that Django supports other database vendors such as PostgreSQL, Oracle, and SQLite.

Advertiser uses a page that is split into two columns as shown in Figure 3.1. The left pane on the page shows unlabeled queries that are suggested for inclusion in the campaign. The right pane shows unlabeled queries that are suggested for exclusion from the campaign. The suggestions are determined by first sorting the RSVs of each unlabeled query. From this sorted list, we choose the top ten queries with the highest RSVs as the relevant suggestions and the bottom ten queries with the lowest RSVs as the non-relevant suggestions. We made use of colors for guiding the advertiser during the process. For example, the default background color for the left pane is blue. When the advertiser approves a given suggestion as relevant by clicking on it, the background color of that suggestion turns to green in order to indicate approval. On the other hand, the default background color for the right pane is sunburst. When the user approves a given suggestion as indeed non-relevant by clicking on it, the background color of that suggestion turns to red. A sample scenario is depicted in Figure 3.2. If the advertiser clicks on a suggestion by mistake, then she can click on it again in order to revert it back to its unapproved state.

As an additional feature, if the advertiser thinks that a query which is suggested for inclusion is actually non-relevant and therefore should be excluded, she can drag the

+	-
better more private social networking site 13.4575357126	who can help me build a social media networks -19.4348932407
social community software for musicians free use to build 13.4389629021	site has all your social media site and activity info -19.3539828384
the latest social networks to launch 2013 13.4232633553	small groups with purpose how to create healthy communities -19.262162257
visitors a month should build a social network 13.4163409821	what is a social networking site that you can chat on for teens -19.2413702517
what website should you use to make a social network site that you can have your own domain name 13.3968478865	http ning it 13w7zsd -19.2048186733
steps to start a social networking site 13.3741064244	all in one forum creation -19.0935450944
how to start a blogging business 13.3738633366	on linemembership application -19.0884381793
turn your site into a social network 13.3342319116	create personal wiki in visual studio 2012 vb -19.086989651
create wiki leaks style site from scratch 13.3225302012	membership site wp plugin -19.0302523791
ready made membership turnkey websites 13.3222447581	free wp membership theme -19.0289538102

Next

FIGURE 3.1: The split-pane query suggestions page, where the symbol + denotes relevant query suggestions and the symbol – denotes the non-relevant query suggestions

suggestion and drop it on the appropriate side as shown in Figure 3.2. Its color will change automatically.

When the advertiser is done with the approval, she can click on the button at the bottom of the page in order to confirm her changes, export them into the search campaign through the API provided by the broker, and get the next batch of “improved” suggestions. The new suggestions are supposed to be improved because the feedback provided by the advertiser is incorporated into the model in real time. This iterative process continues until no query is left for evaluation. If that happens, a new query log which also contains new user feedback, is pulled from the broker, and the process continues as usual.

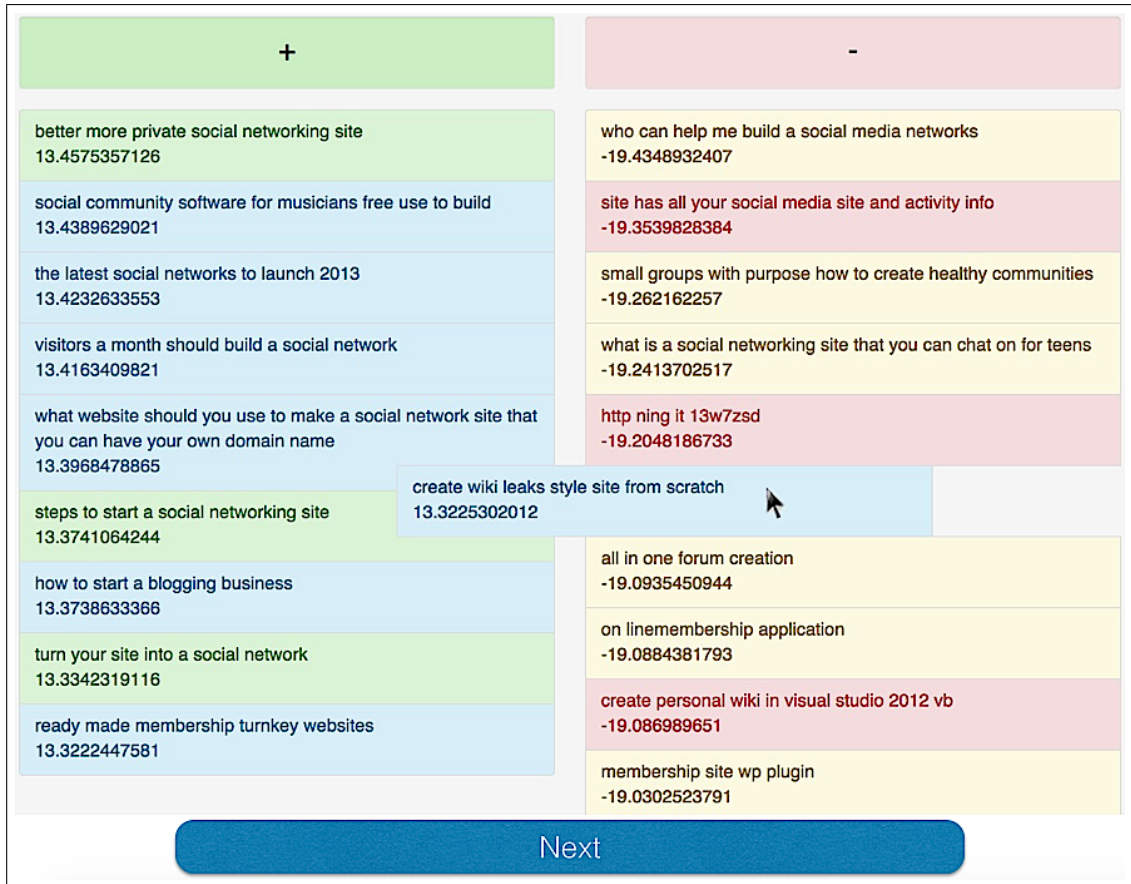


FIGURE 3.2: The illustration of the advertiser dragging a query suggestion made by AdScope and dropping it on the appropriate category according to her expert opinion

3.4 Complexity analysis of AdScope

We analysed the time and storage complexity of AdScope. We varied the number of queries and the length of queries. The time complexity includes all calculation steps: reading data, splitting data into training and test sets, finding c values of each term in each query, and labeling queries in the test set. In the first set of experiments, we synthetically generated queries that contain five distinct terms, i.e., the length of each query is five. No term occurs more than once in any experiment setting. This is the worst case for look up and storage. The number of queries varied in $\{6000, 12000, \dots, 42000\}$. Figure 3.3 shows the computation time in seconds with varying number of queries. These empirical results confirm that the runtime complexity of AdScope is linear in the number of queries N . For each query, a constant number of lookups for the c values have to be made. Each lookup takes $O(1)$ time. Therefore, it takes $O(N)$ time to label N queries,

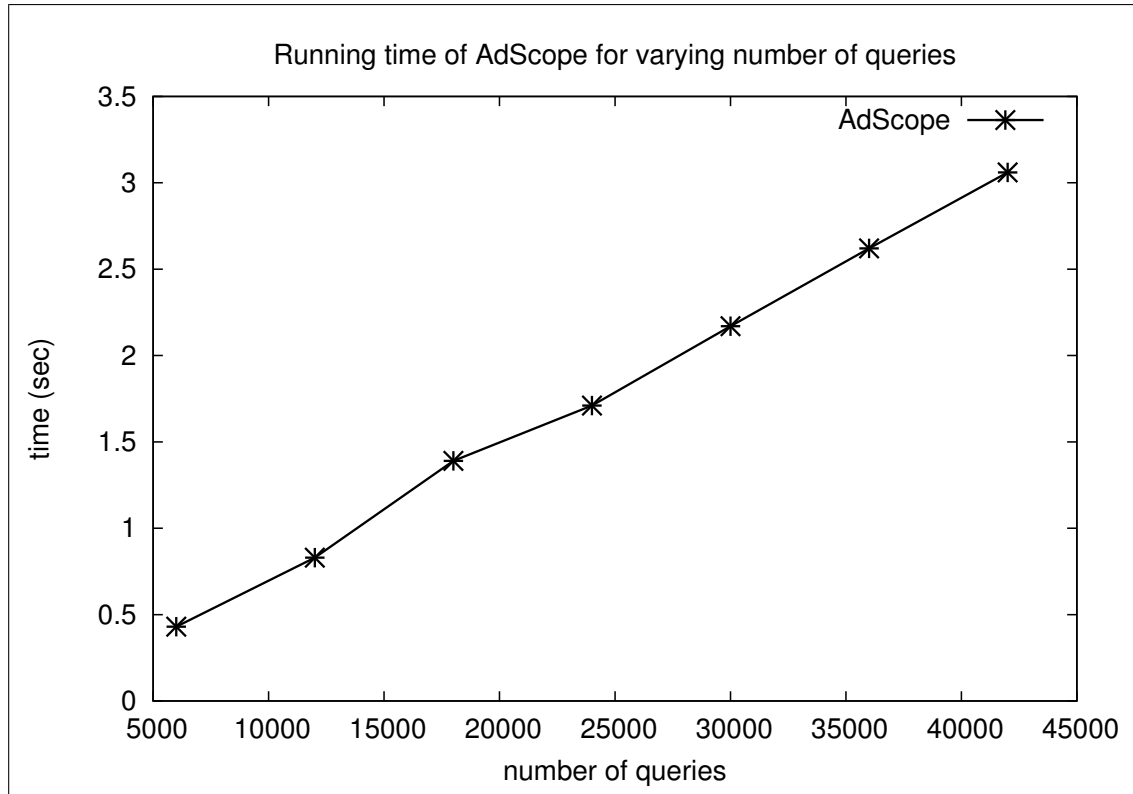


FIGURE 3.3: AdScope’s computation time in seconds vs. number of queries

which corresponds to the amortized cost of labeling. This indicates that it takes $O(1)$ time to label each query.

In the second set of experiments, we set the number of queries to 6,000 and varied the query length in $\{5, 10, \dots, 35\}$. Similar to the previous experiment, no term occurs more than once in any experiment setting. Figure 3.4 shows the computation time in seconds with varying query length. For a query of length m , a total of m lookups have to be made. Each lookup takes $O(1)$ time. These empirical results confirm that the runtime complexity of AdScope is linear in the length of queries m .

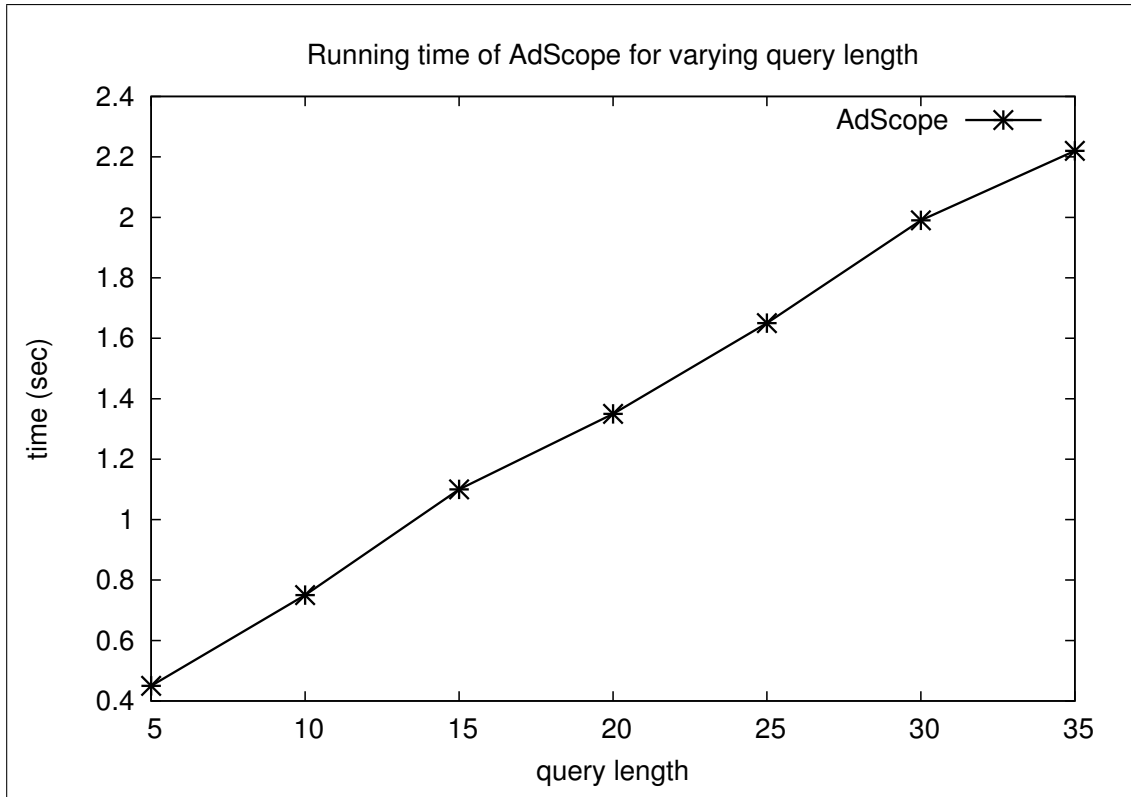


FIGURE 3.4: AdScope's computation time in seconds vs. query length

Altogether, if we assume that there are N queries with length m , the time complexity of AdScope is $O(Nm)$. The storage complexity of AdScope is linear in the size of data. Our implementation uses a HashMap, which maps each term t to its per class counts, p_t , u_t , and c_t values as $\langle K, V \rangle$ pairs. For an average query length of m and a total of N queries, each of which contains distinct terms, the storage complexity of AdScope is $O(Nm)$.

Chapter 4

Experimental Evaluation

4.1 Details of our datasets

We used two different datasets in our experiments for AdScope. The first dataset is the search query log of an online business-to-consumer service provider offering tools for creating a custom social network online. The second dataset is from a software-as-a-service startup that provides electronic invoicing service to small and mid-size businesses. The first dataset has 13761 search queries, 3388 of which were labeled using the decision rules given in Equation 3.5 of Section 3.2. The second dataset has 14258 search queries, 1016 of which were labeled similarly. Unless otherwise stated, the first dataset was used by default in our performance tests.

4.2 Testing methodology

In order to train a classifier that can estimate the class label of an unlabeled search query, we divided the labeled data containing 3,388 records into two sets as training and test sets. During performance measurements, we used six-way cross validation in order to measure the accuracy of each method in estimating the correct label per query in the test set.

4.3 Self comparison

4.3.1 With pre-processing vs. without pre-processing

In most text-processing applications, raw data is first pre-processed in order to normalize it. Discarding stop-words and/or punctuations, and stemming a word in order to find its root are common pre-processing steps. In order to see the effect of pre-processing in our application, we made some initial tests on raw data and pre-processed data. The stemming library in Natural Language Toolkit (NLTK) [26] was used to find the root of each word, and NLTK’s English corpus was used to eliminate stop-words and non-English words.

The effects of stop-word elimination, lemmatization, singularization, and stemming are shown separately in Table 4.1. The accuracy on pre-processed data was lower compared to using raw data. This finding makes intuitive sense in search advertising where even words with typos in them are valid search keywords. For example, keywords with typos are examples of non-intuitive search keywords. Similarly, certain word forms may be indicators of information seekers rather than product buyers. Our finding indicates that search engine marketers and practitioners should use pre-processing sparingly for maintaining valuable information as much as possible.

TABLE 4.1: The classification accuracy of AdScope on pre-processed data vs. raw data.

<i>AdScope</i>	Classification Accuracy
No pre-processing	83%
All words lemmatized	82%
No stop-words	81%
All nouns singularized	81%
Stemming	80%
No stop-words and all nouns singularized	79.4%

4.3.2 Active learning in AdScope

Three different domain experts named E_1 , E_2 , and E_3 evaluated the suggestions made by AdScope. All experts have the same priori-knowledge about the campaign. We had one expert who tagged queries that have conversion potential in the future as relevant. Also, one of the experts was more strict in relevance judgment compared to the other two.

We designed two experiments. In the first experiment, E_1 and E_2 carried out 50 consecutive sessions separately. In each session, the system made ten suggestions per category for inspection, and the experts confirmed or rejected these suggestions. The average accuracy of the system over 50 sessions with E_1 was 78.2%. The accuracy on the relevant suggestions was higher compared to the accuracy on the non-relevant suggestions, 81.6% vs. 74.8%. The average accuracy of the system over 50 sessions with E_2 was much higher, i.e., 85.9% compared to E_1 . Similarly, the accuracy on the relevant suggestions was again higher compared to the accuracy on the non-relevant suggestions, 96% vs. 72.5%. Since

TABLE 4.2: Inter-annotator agreement between three domain experts

(a) Agreements of Expert E_1 and Expert E_2

		E_2		
		Yes	No	Total
E_1	Yes	315	12	328
	No	30	42	72
Total		345	55	400

(b) Agreements of Expert E_1 and Expert E_3

		E_3		
		Yes	No	Total
E_1	Yes	320	8	328
	No	37	35	72
Total		357	43	400

(c) Agreements of Expert E_2 and Expert E_3

		E_3		
		Yes	No	Total
E_2	Yes	333	12	345
	No	24	31	55
Total		357	43	400

the expert opinion varied between E_1 and E_2 , we setup a second experiment in order to measure the agreement of the experts on the system's performance. In this test, each of the experts inspected 400 suggestions in a single session. A total of 200 suggestions were made for each of the two categories. The expert opinions are shown in Table 4.2. For measuring the degree of agreement between different experts, we used Cohen's kappa [27]. This statistical value is calculated as follows:

$$\kappa = \frac{P(a) - P(e)}{1 - P(e)} \quad (4.1)$$

where $P(a)$ denotes the observed proportion of the times the experts agreed, and the value $P(e)$ denotes the probability that the two experts agreed by chance. Consider the kappa statistic $\kappa(E_1, E_2)$ between the experts E_1 and E_2 . Using the Table 4.2(a), the values of $P(a)$ and $P(e)$ can be computed as follows:

- $P(a) = \# \text{ of agreements} / \# \text{ of suggestions}$, which is equal to $(315 + 42)/400 = 0.8925$.
- The pooled marginals for relevant and non-relevant suggestions are $P(\text{relevant}) = \frac{328+345}{800} = 0.8416$ and $P(\text{non-relevant}) = \frac{72+55}{800} = 0.1588$ respectively.
- The probability of agreement by chance $P(e)$ is equal to the sum of the squares of the pooled marginals, i.e., $P(e) = P(\text{relevant})^2 + P(\text{non-relevant})^2$. That is, $P(e) = 0.8416^2 + 0.1588^2 = 0.7335$.
- Finally, we can compute the statistic $\kappa(E_1, E_2)$ as $(0.8925 - 0.7335)/(1 - 0.7335) = 0.60$.

Table 4.3 shows the kappa for all possible expert pairs. Cohen Kappa result changed between 0.56 and 0.6 because there are some settle difference in tagging process. A kappa value between 0.4 and 0.6 indicates moderate agreement while a value between 0.6 and 0.8 indicates good agreement between experts. From the results, we can confirm that the suggestions made by AdScope were from moderate to good [28]. Cohen’s Kappa provide well defined results mostly if there are two experts for judgement and check the agreement between two raters.

In contrast to Cohen’s kappa, which only work when assessing the agreement between two raters, Fleiss’ kappa assesses the reliability of agreement between more than two raters [29]. With respect to Fleiss’ kappa, there is a substantial agreement between our three experts with a score of 0.79.

TABLE 4.3: Quantifying inter-annotator agreement with Cohen’s kappa.

	Cohen’s kappa
E_1 and E_2	0.60
E_1 and E_3	0.54
E_2 and E_3	0.58

4.4 Comparison with the state of the art

In this section, we first describe the competing techniques and then present the results of our comparison.

4.4.1 Multinomial Naive Bayes

Multinomial Naive Bayes or multinomial NB model is a probabilistic learning method. The probability of query $q = t_1 t_2 \dots t_n$ being relevant ($R = 1$) or being non-relevant ($R = 0$) is computed as

$$P(R | q) \propto P(R) \prod_{1 \leq i \leq n} P(t_i | R)$$

where $P(t_i | R)$ is the conditional probability of term t_i occurring in a query of class R . We interpret $P(t_i | R)$ as a measure of how much evidence t_i contributes that R is the correct class. $P(R)$ is the prior probability of a query occurring in class R .

4.4.2 Binary classifiers

In order to select a set of terms for building a binary classifier on binary term features, we used G^2 score. The score performed well for identifying descriptive key phrases for text visualization [30]. The G^2 score of a term t is calculated as follows:

$$G^2(t) = 2 \times (t_{RQ} \times \log \frac{t_{RQ} \times T_Q}{T_{RQ} \times T_{RQ}} + t_{\overline{RQ}} \times \log \frac{t_{\overline{RQ}} \times T_Q}{T_{\overline{RQ}} \times T_{RQ}}) \quad (4.2)$$

where the set of queries is denoted by Q , the relevant queries is denoted by $RQ \subset Q$, and the non-relevant queries is denoted by $\overline{RQ} \subset Q$. Furthermore,

- T_Q denotes the number of distinct terms in Q .
- T_{RQ} denotes the number of distinct terms in RQ .
- $T_{\overline{RQ}}$ denotes the number of distinct terms in \overline{RQ} .
- t_{RQ} denotes the frequency of term t in RQ .
- $t_{\overline{RQ}}$ denotes the frequency of term t in \overline{RQ} .

Terms are sorted in a descending order of their G^2 scores. From this sorted list of terms, the top 1600, 1800, and 1900 terms were chosen for constructing three different feature sets. For each query in the labeled dataset, a binary feature vector is computed as follows: if the query includes a feature term, the corresponding feature value is set to

1, and 0 otherwise. This operation led to three different feature matrices of dimensions 565×1600 , 565×1800 , and 565×1900 . Support Vector Classification [31] and Logistic Regression [32] learning methods were used for training classifiers on each feature matrix.

4.4.3 Markov chain

Using the set of relevant queries, an expectation model for relevance can be built on term phrases using a Markov Chain of a pre-specified order $M \geq 1$. The expected value of a given query $q = t_1 t_2 \dots t_n$ can then be computed as follows:

$$E(q) = \mu(t_1 \dots t_M) \prod_{i=1}^{n-M} \pi(t_i \dots t_{i+M-1}, t_{i+M}) \quad (4.3)$$

where μ corresponds to the stationary state probabilities, and π correspond to the state transition probabilities. According to maximum likelihood estimation (MLE):

$$\mu(t_1 \dots t_M) = \frac{H_M(t_1 \dots t_M)}{\sum H_M} \quad (4.4)$$

$$\pi(t_i \dots t_{i+M-1}, t_{i+M}) = \frac{H_{M+1}(t_i \dots t_{i+M})}{H_M(t_i \dots t_{i+M-1})} \quad (4.5)$$

where H_M represents the frequency histogram for all phrases of length M . A similar approach was used successfully in finding anomalies in time series [33]. Similarly, an expectation model for non-relevance can be built on term phrases using the set of non-relevant queries.

In order to test our hypothesis, we created four different histograms $\{H_M\}$ for $M = \{2, 3, 4, 5\}$ on the training set by sliding a window of size M over the queries, and counting the frequency of all phrases of length M .

In order to compute the expectedness of a query $q = t_1 t_2 \dots t_n$ in the test set, we slide two windows of size M and $M + 1$ over q . For $M = 2$, we extract the following bigrams $t_1 t_2, t_2 t_3, t_3 t_4, \dots, t_{n-1} t_n$, and the following trigrams $t_1 t_2 t_3, t_2 t_3 t_4, \dots, t_{n-2} t_{n-1} t_n$. Using the histograms $\{H_M\}$, the stationary state probability $\mu(t_1 t_2)$ can be computed as $H_2(t_1 t_2) / \sum H_2$, and the state transition probability $\pi(t_i t_{i+1}, t_i t_{i+1} t_{i+2})$ can be computed as $H_3(t_i t_{i+1} t_{i+2}) / H_2(t_i t_{i+1})$ for all i in $\{1, \dots, n - 2\}$. From these constituent parts, the expectedness of q can be calculated easily using Equation 4.3. Since

queries can belong to either the relevant or the non-relevant class, we computed two $E(q)$ values per query, and then used them as the Markov features of the query in a logistic regression.

4.4.4 Comparative results

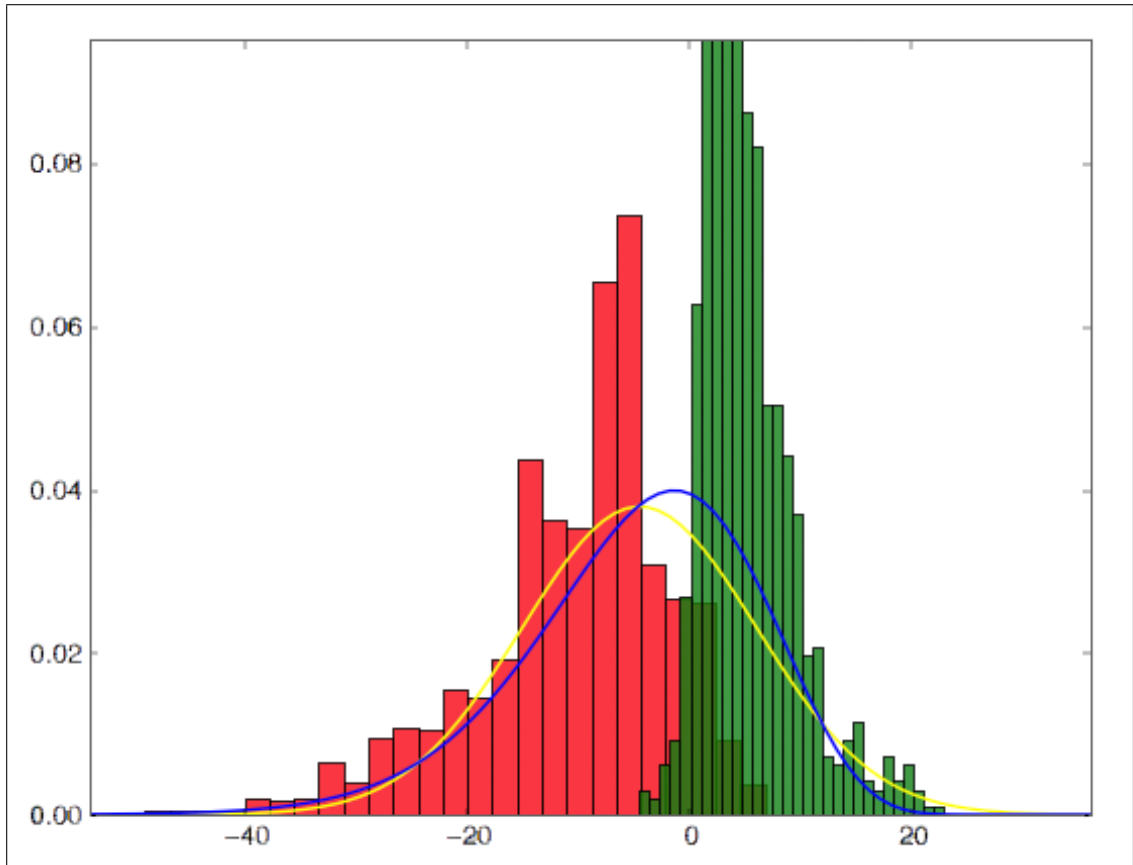


FIGURE 4.1: The distribution of RSVs computed on the campaign dataset (zoomed in). Red bars indicate RSVs of non-relevant queries while green bars indicate RSVs of relevant queries.

In order to visually inspect the efficacy of RSV as a feature, we fit a distribution on the RSVs of all queries in the training set. The distribution is shown in Figure 4.1 where red bars indicate RSVs of non-relevant queries and green bars show RSVs of relevant queries. From the graph, we can interpret that the RSV feature by itself is able to linearly separate two problem classes fairly well. Table 4.4 shows the comparison between Markov chain based method, multinomial NB, and AdScope. The performance of AdScope was the highest for $M = 1$. For longer queries ($M > 2$), the multinomial NB performed better. AdScope uses RSV, which is similar to a multivariate Bernoulli

TABLE 4.4: Performance comparison between Markov chain based method, Multinomial Naive Bayes, and AdScope. For each value of M , only queries that are of length at least $M + 1$ were considered.

<i>Min query length</i> - 1	Classification Accuracy			
	$M = 1$	$M = 2$	$M = 3$	$M = 4$
AdScope	89.25%	87.6%	84.5%	81%
Multinomial Naive Bayes	88%	87.4%	85.6%	84%
Markov	58%	56%	54%	50.7%

Naive Bayes model. In practice, a multinomial NB can handle longer documents while a multivariate Bernoulli NB works best for short documents as is demonstrated in our tests as well. Since the number of training samples decrease with increasing M , the accuracy of all methods decrease with increasing M . We used the scikit-learn library

TABLE 4.5: The classification accuracy of Support Vector Classification, Logistic Regression, and AdScope. The G^2 score is used to construct three different term vocabularies.

<i>Vocabulary size</i> $ V $	Classification Accuracy		
	1600	1800	1900
AdScope	63.5%	64.2%	84%
Logistic Regression	64%	65%	86%
LinearSVC	55%	63%	84%
SVC (kernel = RBF)	62%	62%	62%

[34] for implementing Support Vector Classification (SVC) and Logistic Regression. The default SVC corresponds to the use of an RBF kernel, while LinearSVC refers to SVC with a linear kernel. Table 4.5 summarizes the performance of the classifiers tested. The classification accuracy was lower for smaller vocabularies of size $|V| = 1600$ and $|V| = 1800$. For the larger vocabulary of size $|V| = 1900$, all algorithms except SVC had an accuracy of 84% to 86%. SVC performed the worst among all the alternatives. Due to the linear nature of our problem, non-linear features seemed to hinder the classification performance. On the other hand, the classification accuracy increased with the increasing number of binary features.

These empirical results demonstrated that AdScope, which provides incremental and online training, has a competitive classification accuracy when compared to sophisticated models that require offline training.

4.4.5 Performance on the second dataset

We trained a LinearSVC on the second dataset. The RSV score itself was used as a feature for the classifier. We used four-way cross validation in order to measure the classification accuracy. The results showed that the classifier achieved an accuracy of 72%. The second dataset contains far less labeled records compared to the first dataset; therefore, the accuracy was also lower compared to the first dataset.

Chapter 5

Phrase discovery in AdScope

Instead of labeling individual queries as relevant or non-relevant, one can discover phrases of relevance and phrases of non-relevance. These phrases can be used to dramatically extend or limit a campaign's scope. Consider the following set of non-relevant queries $\overline{RQ} = \{q_1, q_2, \dots, q_{n-1}, q_n\}$:

$$\begin{aligned}q_1 &= A B \dots t_1 \\q_2 &= t_1 t_2 \dots A B \\q_3 &= \dots A B \dots \\q_{n-1} &= t_0 A B \dots t_{n-1} \\q_n &= A B \dots t_n\end{aligned}$$

In \overline{RQ} , $A B$ is a common phrase. The $A B$ phrase can be used to limit the scope of the campaign by disregarding all queries that contain this phrase.

For phrase discovery, we used the Jaccard index [35]. The index represents the likelihood of multiple terms forming a statistically significant collocation among all possible collocations. Mathematically, it is the ratio of how many times term A and term B co-occur vs. how many times they occur individually. For a phrase of length two $t_i t_j$, the Jaccard score $J(t_i, t_j)$ is calculated as follows:

$$J(t_i, t_j) = \frac{H_2(t_i t_j)}{H_2(t_i t_j) + [H_1(t_i) - H_2(t_i t_j)] + [H_1(t_j) - H_2(t_i t_j)]} \quad (5.1)$$

where H_1 and H_2 are frequency histograms for unigrams and bigrams respectively. Each histogram can easily be constructed using a sliding window of appropriate length. For the interested reader, the score computation for phrases of length three is given in Appendix A.

Since each phrase has an RSV value and a Jaccard score, we sort all phrases by their Jaccard score times their RSV and subject the top few of them to a statistical test as follows: there are two events as (i) the occurrence of a phrase and (ii) the conversion event. These two events can be tested for dependence using chi-square statistic. If the null hypothesis is rejected, then the corresponding phrase can be suggested to the advertiser for *inclusion*. Similarly, the occurrence of a phrase and the *no*-conversion events (the opposite of conversion) can be subjected to the chi-square test. If the null hypothesis is rejected, then the corresponding phrase can be suggested to the advertiser for *exclusion*. In a statistical test, a low p -value indicates greater confidence that the observed deviation from the null hypothesis is significant. A p -value ≤ 0.05 with $\chi^2 \geq 3.84$ for one degree of freedom is often used as a bright-line cutoff between statistically significant and not-significant results.

Tables 5.1 and 5.2 show the phrases identified in this way. The advertiser can inspect these significant phrases for adjusting the campaign scope. For example, the phrase “what is” can be added as a negative phrase in the campaign¹. Similarly, “wordpress membership site” should be added as a negative phrase. The phrase “social networking site” can be added as a positive phrase. An advertiser can also come up with new keywords by combining these phrases. For example, “create your own social networking site” is a good keyword. Similarly, “how to create a social network” is a good keyword to consider as well.

¹In Google Adwords, a keyword specified in double quotes is called a phrase match keyword, which means that the keyword matches up to any query that contains the phrase.

TABLE 5.1: Phrases of length 2 sorted according to Jaccard score times RSV. The phrases shown are the ones that have a χ^2 value greater than 3.84, i.e., 95% statistical significance.

(+) for <i>inclusion</i>	(-) for <i>exclusion</i>
your own	what is
networking site	< <i>an arabic text</i> >
own social	membership plugin
social networking	is a
create a	make money
social network	is the
like facebook	on social
my own	wordpress membership
how to	

TABLE 5.2: Phrases of length 3 sorted according to Jaccard score times RSV. The phrases shown are the ones that have a χ^2 value greater than 2.71, i.e., 90% statistical significance.

(+) for <i>inclusion</i>	(-) for <i>exclusion</i>
social networking site	what is a
your own social	what is the
a social network	on social media
create your own	social network app
to create a	membership site with
how to create	money with social
make your own	wordpress membership site
start your own	social networking app
build your own	money on social

Chapter 6

Conclusions

Ad brokers provide daily reports to advertisers about which of their campaign keywords received user clicks, which of keywords converted, and what search queries matched up to the campaign. Identifying the relevant user queries is essential, because in this way, the campaign budget can be spent on the relevant users. The relevant query selection is performed by the advertiser. Depending on the daily search volume, the selection process can be labor-intensive and may require hours to complete. Furthermore, the advertiser has to keep herself up-to-date with changing search market dynamics at all times.

We proposed AdScope, which provides advertisers with relevancy suggestions for search queries. AdScope uses the log of the odds ratio of relevance to non-relevance as the base metric for relevance judgment. For a given query consisting of a set of terms, the sum of the per term log odds ratios are summed up for obtaining the query's relevance value. The novelty we introduced was to unify the user feedback in the form of conversions with the advertiser feedback in the form of relevance supervision. Both types of feedback were collected in order to account for the vocabulary mismatch between users and advertisers. The collected feedback was used in building a multivariate Bernoulli Naive Bayes relevance model. New advertiser feedback is collected using active learning, and the feedback is integrated in real time using a Bayesian update process. In this way, AdScope maintains the advertiser know-how accrued over time within the model itself. Only top few queries are suggested in each step, which eases the management of multiple campaigns. Since each term is holistically evaluated rather than individually, common phrases are detected more easily.

We compared the relevance classification accuracy of Markov chain model, Multinomial NB, binary classifiers for text, and AdScope. Although in some test cases, Multinomial Naive Bayes and Logistic Regression provided better accuracy results, the ability to active learn makes AdScope more favorable. AdScope integrates new relevancy feedback to the system without shutting down and re-training the model from scratch.

In performance tests, AdScope achieved the highest classification accuracy of 89.25% for queries that contain at least two terms. Furthermore, three domain experts agreed substantially with a Fleiss' agreement score of 0.79 on the selections made by our actively learning system.

Appendix A

Computation of Jaccard score for trigrams

In order to compute the Jaccard score for trigrams, four histograms H_1, H_2, H_2^* , and H_3 are maintained¹. The histograms H_1, H_2 , and H_3 are used for maintaining frequency counts of unigrams, bigrams, and trigrams respectively. The histogram H_2^* is used for extended bigrams. The Jaccard score for an arbitrary trigram $t_i t_j t_k$ can be computed as follows:

$$J(t_i, t_j, t_k) = \frac{H_3(t_i t_j t_k)}{D} \quad (\text{A.1})$$

where

$$\begin{aligned} D = & H_3(t_i t_j t_k) + \\ & H_3(\neg t_i t_j t_k) + H_3(t_i \neg t_j t_k) + H_3(t_i t_j \neg t_k) + \\ & H_3(t_i \neg t_j \neg t_k) + H_3(\neg t_i t_j \neg t_k) + H_3(\neg t_i \neg t_j t_k) \end{aligned}$$

Each component of D can be computed iteratively using the histograms H_1, H_2, H_2^* and H_3 . As an example, $H_3(\neg t_i t_j t_k)$ is the count of occurrences of t_j and t_k consecutively without being preceded by t_i . This is exactly the bigram count of $t_j t_k$ less the trigram count of $t_i t_j t_k$. Thus, the value of $H_2(t_j t_k) - H_3(t_i t_j t_k)$ is equal to $H_3(\neg t_i t_j t_k)$.

¹Since each histogram can be implemented as a hash map, the maintenance of these histograms takes constant time per query.

All possible trigrams with exactly two word alignments to the trigram $t_i t_j t_k$ are:

$$\begin{aligned} H_3(\neg t_i t_j t_k) &= H_2(t_j t_k) - H_3(t_i t_j t_k) \\ H_3(t_i \neg t_j t_k) &= H_2^*(t_i t_k) - H_3(t_i t_j t_k) \\ H_3(t_i t_j \neg t_k) &= H_2(t_i t_j) - H_3(t_i t_j t_k) \end{aligned}$$

All possible trigrams with exactly a single word alignment to the trigram $t_i t_j t_k$ are:

$$\begin{aligned} H_3(t_i \neg t_j \neg t_k) &= H_1(t_i) - H_3(t_i t_j t_k) - H_3(t_i \neg t_j t_k) - H_3(t_i t_j \neg t_k) \\ H_3(\neg t_i t_j \neg t_k) &= H_1(t_j) - H_3(t_i t_j t_k) - H_3(\neg t_i t_j t_k) - H_3(t_i t_j \neg t_k) \\ H_3(\neg t_i \neg t_j t_k) &= H_1(t_k) - H_3(t_i t_j t_k) - H_3(\neg t_i t_j t_k) - H_3(t_i \neg t_j t_k) \end{aligned}$$

Bibliography

- [1] G. Fulgoni. The digital world in focus. In *Proceedings of the 2013 DMA, The Global Event for Data-Driven Marketers*, October 2013.
- [2] E. Qualman. Social Media Revolution. Retrieved from:
<http://www.socialnomics.net/2012/01/04/39-social-media-statistics-to-start-2012/>, 2012.
- [3] A. Ashkan and C. LA Clarke. Characterizing commercial intent. In *18th ACM Conference on Information and Knowledge Management (CIKM)*, pages 67–76. ACM, 2009.
- [4] W. Chang, P. Pantel, A.-M. Popescu, and E. Gabrilovich. Towards intent-driven bidterm suggestion. In *18th International Conference on World Wide Web (WWW)*, pages 1093–1094. ACM, 2009.
- [5] H. K. Dai, L. Zhao, Z. Nie, J.-R. Wen, L. Wang, and Y. Li. Detecting online commercial intention (OCI). In *15th International Conference on World Wide Web (WWW)*, pages 829–837. ACM, 2006.
- [6] B. Ribeiro-Neto, M. Cristo, P. B. Golgher, and E. Silva de Moura. Impedance coupling in content-targeted advertising. In *28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 496–503. ACM, August 2005.
- [7] C. D. Manning, P. Raghavan, and H. Schütze. *Probabilistic Information Retrieval*, volume 1. Cambridge University Press, 2008.
- [8] S. Kiritchenko and M. Jiline. Keyword optimization in sponsored search via feature selection. In *JMLR: Workshop and Conference Proceedings*, pages 122–134, 2008.

-
- [9] D. Hao Hu, E. W. Xiang, and Q. Yang. Get more clicks. In *Information Retrieval and Advertising Workshop*. SIGIR, June 2009.
- [10] K. Bartz, V. Murthi, and S. Sebastian. Logistic regression and collaborative filtering for sponsored search term recommendation. In *2nd Workshop on Sponsored Search Auctions*. ACM, June 2006.
- [11] H. Wu, G. Qiu, X. He, Y. Shi, M. Qu, J. Shen, J. Bu, and C. Chen. Advertising keyword generation using active learning. In *18th International Conference on World Wide Web (WWW)*, pages 1095–1096. ACM, 2009.
- [12] K. Yu, J. Bi, and V. Tresp. Active learning via transductive experimental design. In *23rd International Conference on Machine Learning (ICML)*, pages 1081–1088. ACM, June 2006.
- [13] J. G. Shanahan, N. Lipka, and D. Van den Poel. Learning to active learn. In *SIGIR Workshop: Internet Advertising (IA)*. ACM, July 2011.
- [14] S. Thomaidou and M. Vazirgiannis. Multiword keyword recommendation system for online advertising. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 423–427. IEEE, July 2011.
- [15] Google Keyword Tool. Google AdWords Support Forum. <https://adwords.google.com/o/KeywordTool>, 2012.
- [16] Y. Chen, G.-R. Xue, and Y. Yu. Advertising keyword suggestion based on concept hierarchy. In *1st International Conference on Web Search and Web Data Mining (WSDM)*, pages 251–260. ACM, February 2008.
- [17] S. Ravi, A. Broder, E. Gabrilovich, V. Josifovski, S. Pandey, and B. Pang. Automatic generation of bid phrases for online advertising. In *3rd International Conference on Web Search and Data Mining (WSDM)*, pages 341–350. ACM, February 2010.
- [18] A. Joshi and R. Motwani. Keyword generation for search engine advertising. In *Workshops in 6th International Conference on Data Mining (ICDM)*, pages 490–496. IEEE, December 2006.
- [19] V. Abhishek and K. Hosanagar. Keyword generation for search engine advertising using semantic similarity between terms. In *9th International Conference on Electronic Commerce (ICEC)*, pages 89–94. ACM, August 2007.

- [20] C. Wang, K. Bi, Y. Hu, H. Li, and G. Cao. Extracting search-focused key n-grams for relevance ranking in web search. In *5th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 343–352. ACM, 2012.
- [21] D. Yin, S. Mei, B. Cao, J.-T. Sun, and B. D. Davison. Exploiting contextual factors for click modeling in sponsored search. In *7th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 113–122. ACM, 2014.
- [22] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9: 1871–1874, June 2008.
- [23] Django. Django Software Foundation. <http://djangoproject.com/>, 2015.
- [24] R. M. Lerner. At the forge: Twitter bootstrap. *Linux Journal*, 2012(218):6, 2012.
- [25] S. Gonzalez, J. Zaefferer, and K. Borchers. jQuery User Interface. <http://jqueryui.com/>, 2015.
- [26] S. Bird. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL on Interactive Presentation Sessions*, pages 69–72. Association for Computational Linguistics, 2006.
- [27] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, pages 37–46, 1960.
- [28] D. G. Altman. *Practical Statistics for Medical Research*. CRC Press, 1990.
- [29] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382, 1971.
- [30] J. Chuang, C. D. Manning, and J. Heer. Without the clutter of unimportant words: Descriptive keyphrases for text visualization. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 19(3):19, 2012.
- [31] C.-C. Chang and C.-J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011.
- [32] H.-F. Yu, F.-L. Huang, and C.-J. Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, 2011.

-
- [33] E. Keogh, S. Lonardi, and B. Chiu. Finding surprising patterns in a time series database in linear time and space. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 550–556, 2002.
- [34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research (JMLR)*, 12:2825–2830, 2011.
- [35] M. A. Russell. *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*. O’Reilly Media, Inc., 2013.