Personalized Query Auto-Completion for News Search

Lorand Dali[1], Blaž Fortuna[2], Jan Rupnik[2]

[1]Jožef Stefan International Postgraduate School

[2]Artificial Intelligence Laboratory - Jožef Stefan Institute

[1,2]Jamova cesta 39, 1000 Ljubljana, Slovenia

Tel: +386 1 4773419; fax: +386 1 4251038

e-mail: lorand.dali@ijs.si, blaz.fortuna@ijs.si, jan.rupnik@ijs.si

ABSTRACT

In this paper we study the problem of guessing what search query the user intends to type into a search engine based on the first few characters of the query, also known as prefix based query auto-completion. We train and evaluate two personalized auto-completion models on search logs from an online news portal. The personalization comes from using demographic and location information specific to the user. Our experiments show that we can guess the query the user intended to type and rank it among the top three suggestions over 75% of the time. Moreover, the methods described can decrease the number of keystrokes by about 40%, thus saving the user a lot of typing.

1. INTRODUCTION

Search has certainly become one of the most frequently used computer applications. Keyword search got established as the standard way to do search on the internet. Although keyword search may have its disadvantages, it is appealing to a wide variety of users because it is a very fast and easy to use form of search. One feature which further contributes to the usability of search engines, and which most of the search engines nowadays have, is query auto-completion. Query auto-completion consists of the search engine trying to guess what query the user intends to type before the user finishes typing the query. Most often query auto-completion is prefix based, which means that while the user types the beginning of the query, like for instance 'mo', the search engine suggests possible continuations like '**mo**rning', '**mo**ther', '**mo**nastery', '**mo**on', etc. The main goal of query auto-completion is to save time so that the user does not have to type the full query but can choose from the suggestion instead. Such a timesaving feature is always convenient, but it is a must-have especially on mobile devices where the keyboards are often small, slow and awkward to use, and typing a query can take a lot of effort [1]. With the large increase in usage of mobile devices, auto-completion becomes more and more important and has turned into a necessity, rather than a mere convenience.

Because the main goal of query auto-completion is to save time, it has to satisfy two important requirements. First, the search engine should try to guess what the user wants to type after just a few characters in order to save the user as much typing as possible. Secondly, the suggested queries should be sorted such that the query the user most likely intended to type is at the top of the list. This is to prevent the user losing the time which he gained in typing, to search through the list of suggestions. A user study on mobile search [2] shows that query auto-completion is very appreciated by the users who select a correct suggestion about 90% of the time. This user study also shows that query suggestion increases the cognitive load of a user which results in a 30% increase of time needed to type the query. This means that the number of keystrokes saved due to auto-completion has to be high enough to make the tradeoff worthwhile.

The most straightforward way to satisfy the requirements of query auto-completion is to suggest past queries which begin with the given prefix, and sort them by popularity. In this paper we explore ways of improving on this simple idea. We focus our attention on news search and will try to accomplish better query auto-completion by tailoring the ranking of the suggestions to the specific user who is typing the query. In order to personalize the query completion, we shall take into account several features of the user such as age, gender, job, income, location, etc. It is easy to imagine that users with different attributes, in different contexts will prefer different queries. As an example form our news search data let's take a look at the most preferred queries of a user from France (*michael jackson*, *french*, *paris*, *kenken*, *swine flu*, *paul krugman*, *sarkozy*) versus the preferred queries of a user form the US (*crossword*, *warren buffet*, *sudoku*, *buffett*, *maureen dowd*, *michael jackson*). Additionally to preferring quite different queries, the users from France would also need different suggestions for query completion. For example if a French user starts a query with the letter 's' then he should be suggested *swine flu* and s*arkozy* first, while if the user is from the US then *sudoku* should be suggested. This difference makes sense as the importance which location has in search has been studied [3]. How important information about location is for query auto-completion specifically, is shown in [4]. Additionally we also look at other demographic features and use them to do better query auto-completion. News search lends itself to such experimentation because as opposed to web search, users are often logged in to the news website when doing search and so more personal information can be leveraged.

2. RELATED WORK

Quite a few papers [5][6][7][8][9][10][11][12][13] have been written on mining query logs for automatic query suggestion, also called query recommendation. Given a query of a user, query recommendation consists of suggesting related or alternate queries. In this case the suggestion is made after the user has finished typing the query, and the suggestions are not necessarily morphologically related to the query. Several approaches taking into account the previous queries of the user have been tried. Association rule mining is proposed in [5]. In [6] query-flow graphs are proposed, and in [7] an optimization framework using query-flow graphs is evaluated. Information contained in query sessions is leveraged in [8] and [9] by using query co-occurrence, and in [10], where the similarity between two queries is influenced by how many queries appear chronologically between them in the query history of the user. The queries and search

results clicked after the queries are represented as a bipartite graph in [11] and [12]. In [12] a random walk model for query similarity is proposed. In [13], to do clustering of queries into topics, queries are represented by a term-weight vector taking into account not only the query terms but also terms from the documents which were clicked. All related papers mentioned so far focus on query recommendation. The query recommendation problem is slightly different from the problem we tackle because our approach completes the query before the user has finished typing it. Much less research has been done on prefix based query auto-completion. Although the most popular web search engines use auto-completion, there is no detailed description on which models they use and how they work. CONQUER [14] is a context aware prefix-based query suggestion system which uses time of day and location as contextual features for personalizing the query completion. The behavior of users when using auto-completion in mobile search is studied in [2]. In [4], a series of experiments done, studying query auto-completion in the mobile search case. These experiments look at how often a correct query completion can be suggested and how many keystrokes that saves the user on average. The experiments also evaluate the impact of features such as hour, day and location.

Our contribution is a personalized query auto-completion based on user-specific demographic features. As far as we know, an evaluation of the influence of demographic features for query auto-completion or query recommendation was not done previously.

3. DESCRIPTION OF THE DATA

The data we use to experiment with personalized query completion consists of search logs from an online news portal for the period 18th of August – 30th of August 2009. After filtering out some of the entries the data amounts to 380000 searches. The users who perform the searches are also tracked and uniquely identified. From a log entry we can determine what query was asked, which user did the search, and at what time did the search happen.

There are around 250000 unique users in the dataset, and about each of them we have some additional demographic information: gender, age, job, industry, income and the location (city and country) from which the user made the search. We divide the age into ranges: under 21, 21-30, 21-40, 41-50, 51-60 and over 60. The age is quite uniformly distributed, with slightly more younger users. Similarly, the yearly income is also given in ranges. A user can have one of several different types of jobs in one of several different industries. For the location we chose to take into account only the countries and cities which appear at least one thousand times in the logs. If a user has a location which is rarer than this, then the location is considered to be unknown.

The users are of two types, those who have an account on the website and those who do not. For the users who have an account we know all the demographic data which the user has filled out in a form when registering to the site. For the other users we can extract only the location from the IP address, and demographic data is unavailable. Over one third of the users, more exactly 112000, are registered users. Hence the demographic data has many missing values.

The analyzed data contains about 80000 unique queries. There are a few very frequent queries, and a long tail of very rare queries. This power-law distribution is plotted in Figure 1 on a log-log scale. The number of words in a query is usually small. The most queries, 40.78% of them, have two words, 33.44 % are made of one word and 13.79% have three words. Thus almost 90% of the queries are three words or shorter.

In conclusion, for a given search from the log we have the following information:

- The query
- Time when the query was made (hour)
- Information about the user who made the query (gender, age, job, industry, income, city, country)

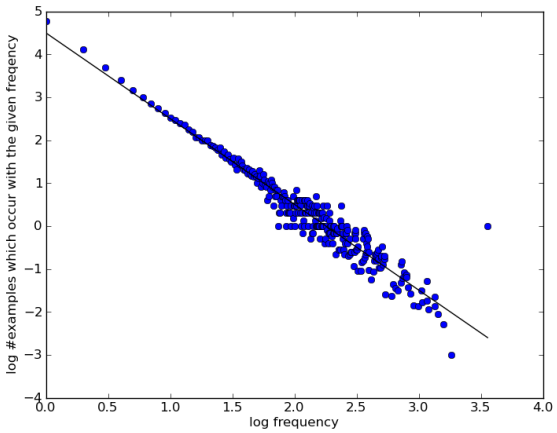

Figure 1 Power law distribution of the query frequencies plotted on a log-log scale

4. QUERY AUTO-COMPLETION

Query auto-completion takes as input a prefix $x$of a query and a set of features of the user $\{F_1, F_2, \cdots F_N\}$ and suggests queries which start with the prefix x and are likely to be asked by a user with the given features. Suggestions can be only queries which were asked by some user in the past. First we get the set of candidate queries Q which consists of the queries which start with x, and then for each query $q \in Q$ we compute its score by estimating the probability

$$p(q|F_1, F_2, \cdots F_N)$$

The query with the highest score will be suggested as the first option for auto-completion, and so on. Due to sparseness and missing values we cannot directly compute the joint conditional probability to obtain the score. Instead, we shall approximate the score by taking the product of the probabilities:

$$score(q) = p(q) \cdot p(q|F_1) \cdot p(q|F_2) \cdot \ldots \cdot p(q|F_N)$$

The factor $p(q)$, which is the frequency of the query q in the past, penalizes the score if q is rare and boosts it if q is frequent. We do this in order to get a very popular query among the suggestions even if it does not fit the user very well. Intuitively, the score for a query is high if the query is popular by itself and also popular among the queries asked by users who have features in common with this user.

4.1 Baseline Ranking of Query Suggestions

As the baseline query auto-completion we rank the candidate queries according to frequency only. So the first suggestion for a prefix will be the most popular query which begins with that prefix. The baseline is not personalized as it does not take into account any of the user's features.

5. EXPERIMENTS

This section describes several experiments used to evaluate the performance of personalized automatic query completion in general, to identify the best features and subsets of features and to estimate how useful the auto-completion is for the user. In order to be consistent, all the experiments use the same training data and the measurements are made with the same testing examples. The training data consists of the chronologically first 300000 entries in the log. The test examples are selected from the later entries such that they contain no missing values for any of the features. This is because to evaluate a model which takes into account all features we need the values for those features to be present. Although models which take into account only a subset of features may have missing values for the other features we decided to keep the testing data the same to enable a more reliable comparison of the results. We have chosen about 2200 such testing points.

The general idea of the experiments is the following. Given a testing example for which we know the query and the features of the user we take a prefix of the test query and try to suggest queries from the training data. We measure how often the actual query will be found and how high it will be ranked in the list of suggestions. We also look into how long a prefix is required to get good results and which features play a greater role in making correct query completions. The following subsections address each of these problems in detail.

5.1 Measuring Ranking Performance

In order to measure the performance of query auto-completion we look at which rank the correct query has in the list of query suggestions. The correct query is the query from the test example, and we try to guess it from prefixes of length 1, 2, 3 and 4. We evaluate separately for registered and unregistered users. For unregistered users we know: city, country and the time of day when the query was made. For registered users we know additional features like: age, gender, industry, job and income.

The columns R1, R2, R3 show in percentages how often the suggested auto-completion which is correct is on the first place, on the second place and on the third place. The column TOP3 shows how often the correct suggestion makes it into the top three in the suggestion list. The Prefix column shows what length a prefix was used to guess the query. The first column shows which feature set has been used. Knowing this, we can read for instance that the probability product model puts the correct query as the first suggestion 44% of the time when given a prefix of length three and the full feature set.

When comparing to the baseline, it looks like personalized auto-completion is most useful when the prefix is short. As the length of the prefix increases it gets easier to guess the correct query, and the margin by which the baseline is beaten decreases. In some cases the baseline even wins in these situations. Another observation is that our method tends to put the correct query as the first suggestion more often when compared to the baseline. The improvement which the extra features available for registered users give is consistent but not very big. Knowing these extra features improves the performance by 1-2%.

5.2 Dominant Features

In this section we discuss which features are the most important in predicting the correct query completion. In order to do that we define the concept of dominant feature as being the feature F_i for which the probability $p(q|Fi)$ is highest. For each test example we keep just a short prefix from the query and try to guess the rest using personalized query auto-completion

Table 1 Auto-completion Evaluation

		RANK OF SUGGESTION (%)			
FEATURES	**Prefix**	**R1**	**R2**	**R3**	**TOP3**
UNREG.	**1**	11.50	5.08	2.47	19.06
	2	23.44	10.12	5.39	38.96
	3	43.96	13.53	7.56	65.05
	4	56.87	14.59	5.30	76.78
REGISTERED	**1**	12.07	6.01	2.16	20.25
	2	25.56	9.86	4.90	40.33
	3	44.44	13.79	6.54	64.79
	4	57.71	13.44	4.86	76.02
BASELINE	**1**	9.06	6.19	2.34	17.60
	2	23.07	9.81	5.92	38.83
	3	42.94	14.55	6.59	64.05
	4	57.49	13.48	6.19	77.17

. If the correct suggestion is in the top three suggestions then the dominant feature of this ranking is remembered. At the end we have for each feature the number of times it has been the dominant feature. The piechart in Figure 2 shows the distribution obtained using a prefix of 2 and the probability merging auto-completion model. It seems that the city has a slight edge over the other features accounting for about a quarter of the correct suggestions in the top three. Next are industry, job and age which are roughly equal and are the dominant features between 15% and 20% of the time each. Frequency performs poorly compared to the other features. This means that very rarely the correct auto-completion appears in the top three suggestions because of query frequency, and other features play a much greater role.

5.3 Keystrokes Saved

Because the goal of query auto-completion is to save time by guessing the query before the user has completely typed it, the natural question to ask is how much time the proposed approach actually saves for the users. Therefore we counted how many keystrokes our method saved. For each query from the test set we first give the first letter as prefix and run auto-completion. If the actual query gets suggested in the top three suggestions then we assume that the user very likely notices it, chooses it and stops typing, so we have saved the user the length of the query minus one keystrokes.

If the actual query is not one of the top three suggestions, then we assume that the user continues typing and we run auto-completion with prefix of the first two letters, maybe now the correct query makes it into the top three and gets chosen. We go so forth increasing the length of the prefix up to length 4. If for a prefix of length 4 the auto-completion still does not suggest the correct query in the top three suggestions, then we assume that the auto-completion has failed, and we have not saved any keystrokes on this particular query.

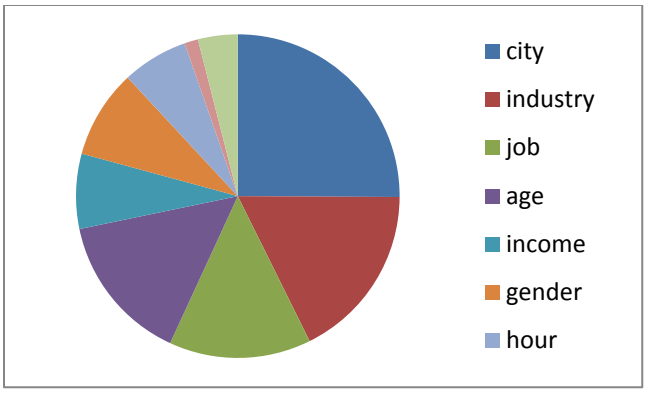

Figure 2 Relative importance of Demographic Features

It turns out that our approach can save on average 5.7 keystrokes per query. Considering that a query in our dataset has on average 13.9 characters it looks like we can reduce the average number of keystrokes used to search to 8.2. This is a huge timesaver if the user uses a mobile device, especially if he has a multitap keyboard.

5.4 Query Suggestion

Automatic query recommendation is a problem very similar to query auto-completion. The difference is that based on the current query and past queries the user gets suggestions of queries which are not necessarily morphologically related to the current query. Although query recommendation is not the focus of our paper, we have to notice that the auto-completion approach which we propose could be applied to the personalized query recommendation scenario as well. We could obtain related queries by dropping the constraint that the suggested query has to start with the prefix. As an example consider a male user, in his thirties, from New York who issued the query *crossword*. Our approach would suggest him the following related queries: *kenken*, *crossword puzzle*, *today's crossword*, *sudoku*, *crosswords*, *puzzle*, *modern love*, *daily crossword*, etc. It is clear that the suggested queries are highly relevant. Most of them are alternative searches for crossword, some are queries for other puzzles and games, and the query *modern love* is not related to puzzles, but it is very specific for the given age group. Please note that the user would miss many of these related queries if the suggestions were based of prefixes only. In the future we plan to investigate this kind of query recommendation and possible combinations of it with prefix based auto-completion.

6. CONCLUSIONS

In conclusion, we have proposed and evaluated prefix-based personalized query auto-completion for news search. The context for personalization comes from user-specific demographic data, and the time of day when the query was made. The results are very promising. We can suggest the correct completion for over 75% of the queries, and we can decrease the number of characters the user needs to type by over 40%. Our work is among the few research works done on prefix-based query auto-completion, and as far as we know it is the only one which uses demographic data for personalized query completion. In the future we plan to focus more on the information which can be obtained from previous queries, and we shall try to combine some ideas from the field of query recommendation into our work.

7. ACKNOWLEDGEMENTS

This work was supported by the Slovenian Research Agency amd the ICT Programme of the EC under XLike (ICT-STREP-288342)

8. REFERENCES

[1] Kamvar, M., Baluja, S. A Large Scale Study of Wireless Search Patterns: Google Mobile Search, CHI 2006 pp 701-709

[2] Kamvar, M., Baluja, S. Query Suggestions for Mobile Search: Understanding Usage Patterns, CHI 2008

[3] Backstrom, L., Kleinberg, J., Kumar, R., , Novak, J. Spatial variation in search engine queries, WWW 2008

[4] Kamvar, M., Baluja, S. The Role of Context in Query Input: Using contextual signals to complete queries on mobile devices.CHI 2007

[5] Fonseca, B., Golgher, P., De Moura, E., Ziviani, N. Using assotiation rules to discover search engines related queries. LA-WEB 2003

[6] Boldi, P., Bonchi, F., Castillo, C., Donato, D., Vigna, S., Query Suggestions Using Query-Flow Graphs. WSCD 2009

[7] Anagnostopoulos A., Becchetti, L., Castillo, C., Gionis, A. An Optimization Framework for Query Recommendation.WSDM 2010.

[8] Huang, C., Chien L., Oyang T., Query-Session Based Term suggestion for Interactive Web Search. WWW 2001.

[9] Huang, C., Chien L., Oyang T. Relevant Term Suggestion in Interactive Web Search Based on Contextual Information in Query Session Logs. 2003. Journal of the American Society for Information Science and Technology, Volume 54, Issue 7, pp 638-649.

[10] Zhang, Z., Nasaroui, O., Mining Search Engine Query Logs for Query Recommendation. WWW 2006.

[11] Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., Li, H., Context-Aware Query Suggestion by Mining Click-Through and Session Data. KDD 2008.

[12] Mei, Q., Zhou, D., Church, K. Query Suggestion Using Hitting Time. CIKM 2008.

[13] Baeza-Yates R., Hurtado, C., Mendoza, M. Query Recommendations using Query Logs in Search Engines. Workshop on Clustering Information over the Web, EDBT 2004.

[14] Sengstock, C., Gertz, M.,CONQUER: A System for Efficient Context-aware Query Suggestions. WWW 2011