学位報告 4

報告番号	※甲　　第　　号

主　論　文　の　要　旨

論文題目　Efficient Text Autocompletion for Online Services
（オンラインサービスのための効率的なテキスト自動補完）

氏　　名　HU Sheng　(胡 晟)

論 文 内 容 の 要 旨

Query autocompletion (QAC) is an important interactive feature that assists user in formulating queries and saving keystrokes. Due to the convenience it brings to users, QAC has been adopted in many applications, including Web search engines, integrated development environments (IDEs), and mobile devices.

However, there are many challenges laid on various applications other than Web search engines. Thus, we focus on the solutions of Autocompletion for Online Services.

First, we investigate location-aware query autocompletion. As mobile devices become more and more popular, one of the main applications is location-aware service, such as Web mapping. Although there have been several solutions to location-aware query autocompletion that are based on a combination of spatial and textual indexes to process queries, all of them suffer from inefficiency when the dataset is large or when large amount of simultaneous queries occur. Most existing works can be classified into text-first, space-first, and tightly-combined methods, according to how the indexes are combined. The text-first methods first index the text descriptions and then apply spatial constraints as filters to verify the objects. E.g., if a user searches for “Starbucks” around “New York”, text-first methods will first find all the objects matching “Starbucks” and then verify whether they are around “New York”. The space-first methods adopt the reverse order. The tightly-combined methods will transfer “Starbucks” and “New York” into one combined token and then use it as a key for lookups. In this work, we propose a new solution to location-aware query autocompletion. We devise a trie-based index structure and integrate spatial information into trie nodes. Our method is able to answer both range and top-k queries.

In addition, we discuss the extension of our method to support the error-tolerant feature in case user's queries contain typographical errors. Experiments on real datasets show that the proposed method outperforms existing methods in terms of query processing performance.

Second, we study a novel QAC paradigm. For existing QAC methods, users have to manually type delimiters to separate keywords in their inputs and then the system takes the input characters as the prefixes of keywords to match. Hence a limitation is that these methods are unable to handle the case when users prefer not to manually separate keywords in the input or it is inconvenient to do so. In this work, we propose a novel QAC paradigm through which users may abbreviate keywords by prefixes and do not have to explicitly separate them. Such paradigm is useful for applications where it is inconvenient to specify delimiters, such as desktop search, text editors, input method editors, as well as the t asks of searching long proper names comprising multiple morphemes. E.g., in an IDE, users may input "getnev" and we suggest "GetNextValue".

We show that the query processing method for traditional QAC, which utilizes a trie index, is inefficient under the new problem setting. A novel indexing and query processing scheme is hence proposed to efficiently complete queries. To suggest meaningful results, we devise a ranking method based on a Gaussian mixture model, taking into consideration the way in which users abbreviate keywords, as opposed to the traditional ranking method that merely considers popularity. Such a Gaussian mixture model is utilized to predict the probability that a user abbreviates keywords into a given set of prefixes observed in the input. We also present a top-k query processing algorithm to efficiently compute the top-k answers with respect to the new ranking method by integrating a series of early termination techniques. Experiments on real datasets demonstrate the effectiveness of the new QAC paradigm and the efficiency of the proposed query processing method.

Finally, we explore the problem of code completion, which is a traditional popular feature for API access in integrated development environments (IDEs). It not only frees programmers from remembering specific details about an API but also saves keystrokes and corrects typographical errors. Existing methods for code completion usually suggest APIs based on statistics in code bases described by language models. However, they neglect the fact that the user's input is also very useful for ranking, as the underlying patterns can be used to improve the accuracy of predictions of intended APIs.

To improve users' satisfactions, we propose a novel method to improve the quality of code completion by incorporating the users' acronym-like input conventions and the APIs' scope context into a discriminative model. The users' input conventions are learned using a logistic regression model by extracting features from collected training data. The weights in the discriminative model are learned using a support vector machine (SVM). To improve the real-time efficiency of code completion, we employ a trie

to index and store the scope context information. An efficient top-k algorithm is developed. Experiments show that our proposed method outperforms the baseline methods in terms of both effectiveness and efficiency.

Generally, an overall view of autocompletion across different application domains is provided. We believe that our contributions are practical and easy to be applied in many other applications. At first, we think our efficient index design and early termination pruning techniques can be applied in either geo-graphical or textual databases. Secondly, our proposed ranking method and novel autocompletion paradigm can practically improve the top-k accuracy and effective performance in textual editors, mapping services and any other Web retrieval systems. Last but not least, extensive experimental evaluations are conducted to illustrate the performance on real-world datasets.