Generation of Overview-Oriented Search Results Using Ordered Structure of Word Occurrences in Technical Blogs

Masaru Hakii, Daisuke Kitayama

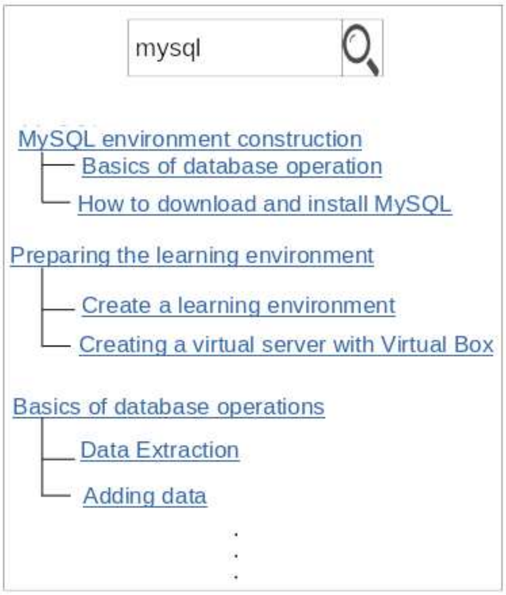
***Abstract*—For a given search topic, a user without prior knowledge may have difficulty fully understanding the topic in the search results. In addition, when a user wants to gain a deeper knowledge of the topic, the user can not create appropriate search queries with the existing search engines because the user does not know the surrounding topics. To solve this problem, we propose a method for presenting search results to help users grasp the full scope of a topic. Specifically, we present the user with search results, based on the tree structure and using words with prerequisite knowledge relations from the set of search results. By presenting search results using prerequisite knowledge relationships the user can traverse the tree structure when they want to know more about the topic. Thus, the user can grasp the full topic. This paper proposes a method of presentation that searches for results to help provide the full scope of a topic to the user. Proposed method is that obtaining a set of articles from search results and create a tree structure of words from the set of articles and generating overview oriented search results on the basis of a tree structure. As a result of the evaluation experiment, the precision of the detailed relation, meaning the depth direction of the tree was 70.5% and the Spearman's rank correlation coefficient of the order of appearance, meaning the width direction of the tree, was 0.41. This result indicates that it may help the user fully understanding, but there is a problem with the extraction of the title sentence.**

***Index Terms*—web-search, overview-oriented search result, web-UI, tree structure.**

I. Introduction

IN recent years, improvements in search technology have made it easier for users to obtain the information they want. However, if the user has no knowledge of the topic, the search becomes extremely difficult, as they may not know what keywords to enter. In addition, it is not easy to glean important keywords from multiple pages. For example, consider an uninformed user who wants to learn about MySQL. The user enters a simple keyword search for MySQL in the normal web search results, and the search returns a summary of MySQL and some of its functions. It would not be clear which pages are most useful or should be viewed first. In addition, to obtain an overview, the user has to keep adding keywords to the search.

We hypothesize that it would be possible to solve this problem by presenting search results that allow the user to easily grasp the entire search query topic. When the search keyword "MySQL" is entered, the title "MySQL environment construction" is displayed first, and then the titles "Basics of database operation" and "How to download and install MySQL" are displayed as child elements. By so structuring the result, we expect that similar to a table of contents in a book, users would be able to comprehensively grasp the subtopics of the result through the headings on the same level, and they would be able to learn more about the topics of interest by following the child elements.



mysql

- MySQL environment construction
  - Basics of database operation
  - How to download and install MySQL
- Preparing the learning environment
  - Create a learning environment
  - Creating a virtual server with Virtual Box
- Basics of database operations
  - Data Extraction
  - Adding data

⋮

Fig. 1. Proposed overview-oriented search result

We refer to such displaying of the search result as the "overview-oriented search result," and it is illustrated in Figure 1. As shown in the figure, moving down the same hierarchy of output results would increase the knowledge of the search keyword, and moving down the technical blog hierarchy would provide deeper knowledge of the topic.

However, in recent years, technical blog articles have remarkably improved. In technical blogs, such as Qiita[1] and Developers.IO[2], that present knowledge of programming, various terms and concepts are often systematically written in units of articles. By using the structure of such technical blogs, it is possible to obtain a parent–child relationship in which, for example, "data" and "SELECT statement" are placed under "MySQL". Using this structure, we propose a method that constructs a tree structure from a set of search results; the tree uses words with prerequisite knowledge rela-

This work was supported by ISPS KAKENHI of Grant-in-Aid for Scientific Research(C) Grant Number 21K12147.

Masaru Hakii and Daisuke Kitayama are Graduate School of Engineering, Kogakuin University, Tokyo, Japan e-mail:em21018@ns.kogakuin.ac.jp, kitayama@cc.kogakuin.ac.jp

[1] https://qiita.com

[2] https://dev.classmethod.jp

tionships, extracts search results based on this structure, and presents the entire topic to the user. Specifically, it extracts the parent–child relations of terms from the dependency relations of the headings of technical blogs, such as headings and subheadings, and extracts the corresponding titles in the technical blogs, according to the parent–child relations of the words. Thus, a structure similar to chapter titles and section titles is created, and the search results are presented in a structured manner.

The contributions of this study are as follows:

- From a technical blog, automatically structuring the parent-child relationship of terms and constructing a type of ontology.
- As an approach to structuring search results, the effect and nature of structuring based on parent-child relationships of terms are demonstrated.
- Challenge structuring including the order in which it should be viewed.

This paper is structured as follows. In section II, we discuss related works. In section III, we detail the proposed method. The experimental study is detailed in section IV. Finally, in section V, we provide the summary and future tasks.

II. Related work

A. Web search support

Aktolga et al.[1] described a method to boost rarely clicked queries in a system wherein limited clickthrough data are available for all queries. They utilized information from co-click queries, subset queries, and synonym queries to estimate the clickthrough for a sparse query.

In an empirical study of three highly frequented websites, Keller et al.[2] analyzed how website taxonomies influence the next browsing steps of users arriving from a search engine. This reveals that users do not randomly explore the destination site but proceed to the direct child nodes of the landing page with a significantly higher frequency compared with the other linked pages.

Saito et al.[3] investigated how users carefully search the Web to obtain credible and accurate information. They revealed that users' attitudes toward using verification strategies in Web search are positively correlated to their need for cognition (NFC), educational background, and search expertise; users with strong attitudes are likely to click lower-ranked search results than those with intermediate levels of attitude, and users with strong attitudes are more likely to use the terms such as "evidence" or "truth" in their queries, possibly to scrutinize the uncertain or incredible information.

Umemoto et al.[4] proposed an interface to visualize browsing information only for recommendation queries. They proposed an approach for exhaustiveness-oriented tasks; the approach evolved from exploratory search and scored the amount of unreviewed information in terms of importance, relevance, and novelty.

B. Clustering web search results

Motohiro et al.[5] proposed a semi-automatic method for extracting topic maps from a set of web pages. In addition to the conventional clustering method focusing on the network structure, they introduce a weighting system–using web page content similarity, based on web site directory structure, and the links between Web pages–to estimate the meanings of the links between pages, by considering the topic relationships contained in the links between webpages. In this method, the meaning of the links between pages is estimated by considering the topic relations contained in the links between web pages; not only the topics but also the topic relationships are extracted.

Sumida et al.[6] proposed a method to automatically acquire a quantity of super-subordinate relationships, using the Wikipedia article structure as a knowledge source. They proposed a method to extract a large number of candidate super-subordinate relations from clause and bullet headings in the article structure of Wikipedia, and filter them using machine learning to obtain highly accurate super-subordinate relations.

Sara et al.[7] presented an approach for search engine results clustering that relies on the semantics of the retrieved documents rather than the terms in those documents. This approach considers both lexical and semantic similarities among documents and applies the activation spreading technique to generate semantically meaningful clusters. This allows documents that are semantically similar to be clustered together, rather than clustering documents based on similar terms.

Jiyang et al.[8] reformalized the clustering problem as a word-sense discovery problem. Given a query and a list of result pages, our unsupervised method detects word-sense communities in the extracted keyword network. The documents were assigned to several refined word-sense communities to form clusters.

Chung et al.[9] proposed a new method for clustering the search results. Similarities between documents were calculated, based on similarities between these topics. Subsequently, an affinity propagation clustering algorithm was employed to cluster web search results.

Dikan et al.[10] proposed a novel algorithm known as a deep classifier to classify the search results into detailed hierarchical categories with higher effectiveness than previous approaches.

These related studies aimed to obtain the entirety of a topic for search keywords; however, they did not consider the order in which users should browse the results. In this study, the objective of grasping the entirety of a topic was the same as in extant studies; the difference was in the order in which users can easily grasp the entirety of a topic.

III. Proposed method

A. Overview

The flow of the proposed method can be roughly divided as follows.

1) Obtaining a set of articles from search results
2) Creating a tree structure of words from the set of search result articles
3) Generating overview oriented search result on the basis of a tree structure

Figure 2 illustrates the system overview. First, the user enters the search term into the system. The system then obtains a set of search results. From the result set, the system creates a tree structure of dependent words, and presents the results using this structure.

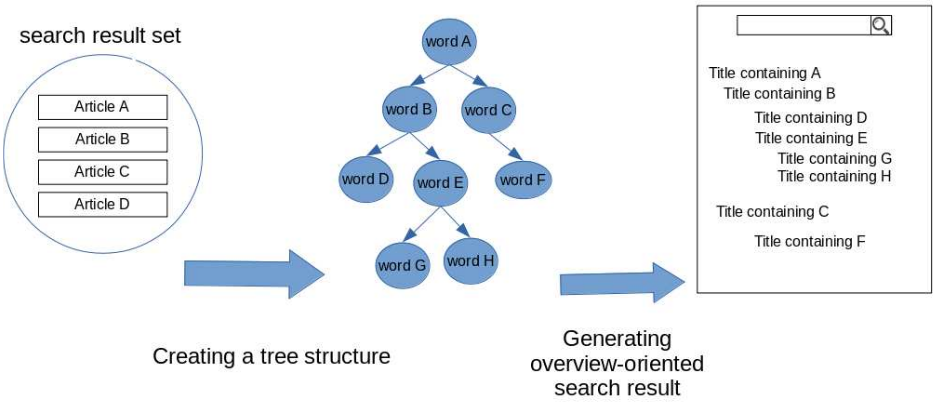


Fig. 2. System overview

B. *Obtaining a set of articles from search results*

First, for the search keywords entered by the user, we obtained articles that contained the search keywords in the title or the tags of articles. The title and headline were extracted from the text of the article, and only nouns were extracted using the morphological analyzer, MeCab[11]. In this study, we used ipadic-neologd[3] as a dictionary.

As regards the dataset, we used an article on Qiita[4]. As aforementioned, Qiita is a technical information-sharing service for programmers that allows them to record and publish their know-how and notes on programming and other topics. We used QiitaAPIv2, an API provided by Qiita, to retrieve 520,000 articles from June 7, 2014, to August 3, 2020. The retrieved contents are the title, body of the article, URL, and tags of the article. Articles are written in markdown, and headings are expressed by headings such as the h1 and h2 tags. There is no explicit parent–child relationship between these headings; however, there is often a semantic dependency between the h1 tag and the h2 tag after it. We hypothesize that we can extract the parent–child relationship of the words by using this observation. The body of an article is often written according to a work or a time series, and the words that appear first are often the prerequisite work or prerequisite knowledge for the words that appear later.

C. *Creating a tree structure with prerequisite knowledge relations*

1) Creating dependency word pairs: This system creates word pairs with dependency relations from the search result set. We consider that there is a dependency relationship between headings in an article. Specifically, there is a relationship between the title word and the word in the h1 tag, and between the word in the h1 tag and the word in the h2 tag. Using this relationship, we create a tree structure in which summary words are located at the top, and more detailed words appear as we move through the nodes. Nouns were extracted from the title text using a morphological analyzer, and pairs were created using nouns in a dependency relationship. At this time, URLs and symbols are deleted as stop words. The next step is to remove the noise. The pairs with the top N occurrences in the pair are excluded, and among the other pairs, those containing words whose occurrences are below the threshold are removed. Table I shows an example of creating a word pair with the search keyword "MySQL."

TABLE I
EXAMPLE OF CREATING A WORD PAIR

Word (parent)	Word (child)	Count
MySQL	environment	675
MySQL	setting	627
MySQL	reference	596
MySQL	installation	508
MySQL	create	497
MySQL	confirmation	461
MySQL	table	324
MySQL	startup	304
MySQL	database	273
MySQL	procedure	272
MySQL	error	234
installation	MySQL	236
MySQL	assumption	214
MySQL	command	158
environment	installation	173
build	environment	189
build	MySQL	182
connect	MySQL	157
docker	environment	130
environment	build	120

2) Creating a tree structure: This system creates a tree structure using the created word pairs. Before the occurrence of the word is considered to be the parent of the pair, the number of pairs is counted. This count is the support of the parent word of the pair. In this case, word pairs with the same constituent words but different orders appear, and thus, the word pairs with the lower support are eliminated. If a tree structure is created, it is possible that inappropriate words will appear in the tree when the nodes are traced.

[3] https://github.com/neologd/mecab-ipadic-neologd
[4] https://qiita.com

Therefore, we cluster the words that appear in the tree. Then, we merged tree structures with word pairs consisting only of words in the clusters. The root of the tree is the search keyword if the parent word of the word pair has the search keyword; otherwise, the word with the highest number of occurrences is the parent. The root of the tree structure is the search keyword, and the tree structure created earlier is joined to its child nodes. The tree structure was created by extracting word pairs using the root word as the parent. The tree structure is created by extracting word pairs with the root word as the parent, and then creating the tree structure using the following rules: only words in pairs are used as parent-child relations, words that appear once are not used again, and nodes directly below the root node with a depth of one are deleted. At this time, the nodes in the tree structure are given priority attributes. To control which of the sibling nodes in the tree structure is displayed first, the display priority of the words is created from the search result set. For a given word, the words that appear in the first half of the same heading in an article are expected to appear higher in the results display. Therefore, when the number of headings of the same size as the headings in which the target word appears is M and the appearance position of the headings in which the target word appears (the order of appearance in headings of the same size) is i, formula (1) is calculated. Then, after calculating it for each article, it is averaged.

$$\frac{M-i+1}{M} \tag{1}$$

Each word is given a display priority, and the word with the higher value is displayed higher. Figure 3 shows an example of creating a tree structure with the search keyword “MySQL.”

In the tree structure that we created, there is a possibility that words that are similar between siblings may appear. For example, words such as “data” and “records” appear in the tree structure. To solve this problem, we use the word vectors learned from the text and title sentences of all articles and merge the nodes of words whose similarity exceeds a threshold. The words of the merged nodes are the words with the highest display priority.

D. Extraction of titles based on tree structure and display of results

Extracting title sentences based on the structure of the created tree structure. From the created tree structure, extract titles that contain the word of the target node and the word of the parent node. The reason for not using ancestor nodes is that the number of words used increases as the depth of the node increases, and it is thought that concise titles cannot be extracted. Title sentences that contain words from the child node or words not related to the child node are not desirable; therefore, priority is given to selecting title sentences that are short and concise, containing as few words from the child node as possible. For this reason, the title sentences to be extracted should be short and concise and should contain as few words as possible that appear in the child nodes. The decision method is the one with the shortest string length from the set of title documents that contain the text of the parent node and the text of its own node.

The extracted title sentences are presented as search results. It is presented similarly to a table of contents of a book. The title text of the child node is located one paragraph down from the parent node’s title text. This makes it possible to obtain the necessary information by following the paragraphs to obtain more in-depth results. The order of presentation in the same hierarchy is ranked in the order of priority of the hierarchy.

E. Execution example

In this section, we present an example of the execution of the proposed method described in Section III. The results in Section III-C2 are shown in Figure 4. This show an example of a word-based execution. The results in Section III-D are shown in Figure 5. This illustrates an example of a title-based execution. Bold text denotes keywords, implying a word-based execution. In both cases, the search keyword is “MySQL” the minimum number of pairs is 25, the number of N that explained section III-C is 10000, the clustering method is k-means, and the number of clusters is 10.

The example output shows the MySQL environment construction, DB creation, and database connection in the DB creation hierarchy.

IV. Experiment

We performed an experiment to investigate whether the search results generated by the proposed method are valid as detailed relations and backward/forward relations.

A. Experimental settings

This experiment was conducted on 10 university students majoring in information engineering. Two types of experiments were conducted. The parameters of this experiment are the same as those in Section III-E. The first is on detailed relations, where the deeper the output hierarchy, the more detailed the relation is, or the more it assumes the information of the parent node. We extracted six title sentences from the title based on the tree structure we created and conducted an experiment on detailed relations. After presenting the title sentence of a node, the subjects were shown the title sentences of their child nodes and asked to select the one they think had a detailed relationship. The percentage of correct answers is then calculated. A correct answer rate of more than 50% was considered the correct answer. The second is the order of occurrence of the title sentences. For the order of appearance experiment, we used five title sentences that contained at least four child title sentences from the title sentences used in the detailed relation experiment. In the output results, we investigate whether sentences that should be in the first half of the set of sentences in the same hierarchy or that do not require more prerequisite knowledge, appear in the first half. Subjects are presented with the title sentence of a node and are asked to select a sentence from the title sentences of its child nodes that they think would be more effective in reading the sentence. The data were then aggregated and reordered in the order of the number of selections made. We calculated the Spearman’s rank correlation coefficient between the reordered data and the data produced by this system.

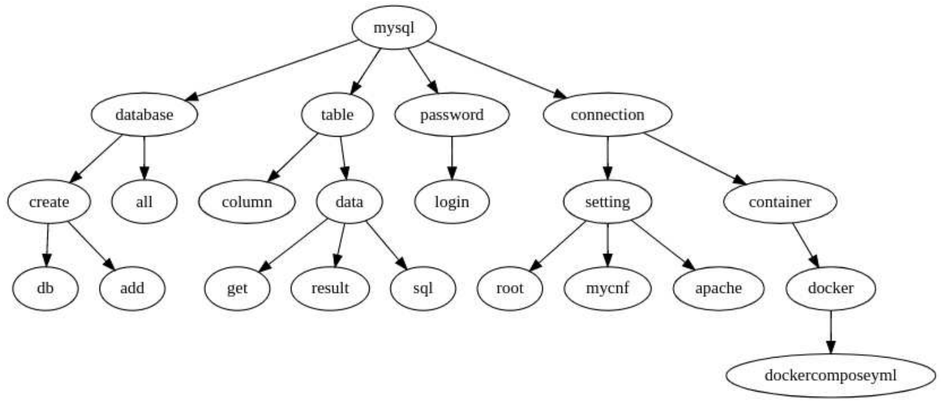


Fig. 3. Example of creating a tree structure

```
mysql
├── environment
│   ├── directory
│   └── install
│       ├── download
│       ├── repogitory
│       ├── version
│       └─ command
├── database
│   ├── all
│   └── create
│       ├── add
│       └── db
├── table
│   └── data
│       ├── get
│       └── result
├── build
│   └── configuration
└── password
    └── login
```

Fig. 4. Tree structure of words created from MySQL search results.

TABLE II
SUMMARY OF EXPERIMENTAL RESULTS

Type	Result
Detailed relationship	70.5% (Precision)
Order of appearance	0.41 (Rank correlation coefficient)

TABLE III
EXAMPLE OF DETAILED RELATIONSHIP RESULTS

Parent Title	Child Title	Fit
MySQL environment construction	MySQL installed in virtual environment	Positive
MySQL environment construction	Laravel environment construction with docker	Negative
MySQL environment construction	vagrant environment construction	Negative

B. Experimental results and discussion

This results of both experiments are shown in Table II. The precision of the detailed relationship was relatively high at 70.5%. This indicates that detailed titles can be appropriately extracted appropriately for the parents. However, the order relation has a moderate correlation (0.41) with respect to the order of correct answers, indicating that inappropriate judgments were included. Table III shows some of the results for detailed relations. Table III shows parent title, child title and fit. This fit column shows that if the child title is the same as the user's answer, it is positive and not the same as the user's answer, it is negative. In terms of detail relations, we can find unsuitable outputs that "MySQL environment construction" → "Laravel environment construction with docker" and "MySQL environment construction" →"vagrant environment construction" We think that there is a problem in the extraction method of the title sentences because the title sentences are not appropriate compared to those judged to be good in the experiment. This is likely because, when extracting title sentences, articles were extracted with reference to those that contained the search keyword in the tag, but because multiple tags can be attached, articles for which the search keyword was not the main topic were extracted.

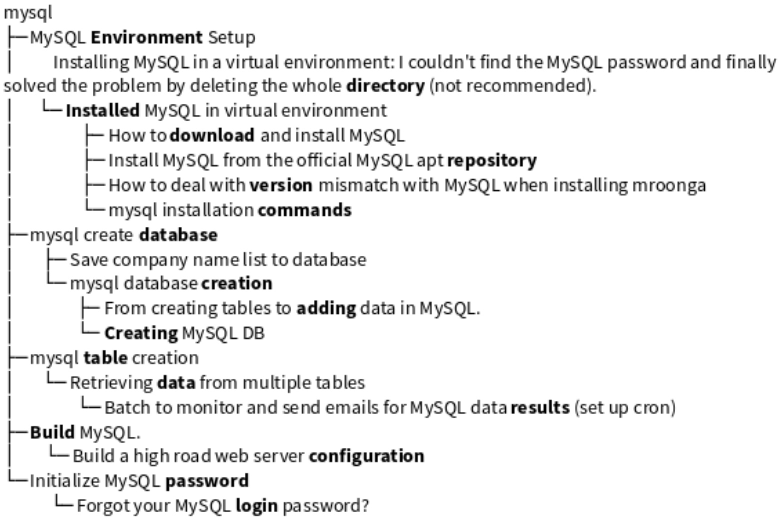
Table IV shows some of the results for the order of appearance. Table IV shows parent title, child title by the system, child title by user and rank correlation coefficient. In terms of the order of appearance, we can find the unsuitable parent title is "[MySQL] DB creation, user creation, authority setting" and this rank correlation coefficient is 0. When we look at the word extraction, we see that under "creation," it is followed by "user," "file," "add," and "data" This is apparently not a strange order of appearance to the senses. This suggests that the word extraction stage is correct; however, there is a problem with the extraction of the title sentence.

V. Concluding Remarks

Based on the idea that it is difficult for a user without prior knowledge to grasp the entirety of a topic included in the search results, we propose a search result generation method that can grasp the entirety of a search result. This is based on a tree structure of words created from the parent–child relationship of terms between the headings of technical blog articles. As a result, we were able to extract the parent–child relationship of the terms in the tree structure; however, the extraction method of the title sentence remains an issue. Future work will include improving the extraction method of title sentences and examining the threshold of the number of word pairs.

References

[1] E. Aktolga and J. Allan, "Reranking search results for sparse queries," in *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, ser. CIKM '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 173–182. [Online]. Available: https://doi.org/10.1145/2063576.2063606



```
mysql
├─MySQL Environment Setup
│     Installing MySQL in a virtual environment: I couldn't find the MySQL password and finally
solved the problem by deleting the whole directory (not recommended).
│   └─Installed MySQL in virtual environment
│        ├─How to download and install MySQL
│        ├─Install MySQL from the official MySQL apt repository
│        ├─How to deal with version mismatch with MySQL when installing mroonga
│        └─mysql installation commands
├─mysql create database
│   ├─Save company name list to database
│   └─mysql database creation
│        ├─From creating tables to adding data in MySQL.
│        └─Creating MySQL DB
├─mysql table creation
│   └─Retrieving data from multiple tables
│        └─Batch to monitor and send emails for MySQL data results (set up cron)
├─Build MySQL.
│   └─Build a high road web server configuration
└─Initialize MySQL password
     └─Forgot your MySQL login password?
```

Fig. 5. Structured search results created from MySQL search results.

TABLE IV
EXAMPLE OF ORDER OF APPEARANCE RESULTS

Parent title	Child title (system output)	Child title (correct answer data)	Rank correlation coefficient
mysql	1: MySQL environment creation 2: Creating MySQL DB 3: Confirm mysql user 4: Mysql add column	1: MySQL environment creation 2: Creating MySQL DB 3: Confirm mysql user 4: Mysql add column	1.0
MySQL environment creation	1: Installed MySQL in virtual environment 2: Build Vagrant environment (Mac) 3: Build Laravel environment with docker 4: Try to use mysql in local environment 5: Creating a LAMP environment with centOS7 6: EC2 LAMP environment Introducing phpMyAdmin	1: Installed MySQL in virtual environment 2: Try to use mysql in local environment 3: EC2 LAMP environment Introducing phpMyAdmin 4: Build Vagrant environment (Mac) 5: Build Laravel environment with docker 6: Creating a LAMP environment with centOS7	0.6
[MySQL]DB creation, user creation, permission settings	1: MYSQL user creation 2: The five files created for form creation 3: From creating tables to adding data in MySQL 4: Creating Dummy Data	1: From creating tables to adding data in MySQL 2: MYSQL user creation 3: Creating Dummy Data 4: The five files created for form creation	0.0

[2] M. Keller, P. Mühlschlegel, and H. Hartenstein, “Search result presentation: Supporting post-search navigation by integration of taxonomy data,” in *Proceedings of the 22nd International Conference on World Wide Web*, ser. WWW ’13 Companion. New York, NY, USA: Association for Computing Machinery, 2013, p. 1269–1274. [Online]. Available: https://doi.org/10.1145/2487788.2488161

[3] F. Saito, Y. Shoji, and Y. Yamamoto, “Highlighting weasel sentences for promoting critical information seeking on the web,” pp. 424–440, 1 2020.

[4] K. Umemoto, T. Yamamoto, and K. Tanaka, “Scentbar: A query suggestion interface visualizing the amount of missed relevant information for intrinsically diverse search,” *SIGIR 2016 - Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 405–414, 7 2016.

[5] M. Mase and S. Yamada, “Extracting topic maps from web histories by clustering with web structure and contents,” in *2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, 2006, pp. 405–408.

[6] A. Sumida, N. Yoshinaga, and K. Torisawa, “Hyponymy relation acquisition from hierarchical layouts in wikipedia,” *Journal of Natural Language Processing*, vol. 16, no. 3, pp. 3_3–3_24, 2009.

[7] S. Saad, M. El-Sayed, and Y. Hassan, “Semantic clustering of search engine results,” *The Scientific World Journal*, vol. 2015, pp. 1–9, 12 2015.

[8] J. Chen, O. R. Zaïane, and R. Goebel, “An unsupervised approach to cluster web search results based on word sense communities,” in *2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, vol. 1, 2008, pp. 725–729.

[9] C. Tran and A. Ameljańczyk, “Clustering web search results using wikipedia resource,” *Computer Science and Mathematical Modelling*, pp. 25–29, 09 2020.

[10] D. Xing, G.-R. Xue, Q. Yang, and Y. Yu, “Deep classifier: Automatically categorizing search results into large-scale hierarchies,” in *Proceedings of the 2008 International Conference on Web Search and Data Mining*, ser. WSDM ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 139–148. [Online]. Available: https://doi.org/10.1145/1341531.1341552

[11] Y. M. Taku Kudo, Kaoru Yamamoto, “Applying conditional random fields to japanese morphological analysis,” *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, 2004.