Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Курсовой проект по курсу «Информационный поиск»

Студент: П. А. Харьков
Преподаватель: А. А. Кухтичев
Группа: М8О-406Б

Дата:
Оценка:
Подпись:

Москва, 2023

Курсовой проект

Необходимо добавить в поисковой индекс информацию о зонах, в которых встретились термины. Как минимум, нужно сделать отдельные зоны для заголовков документов. Также, необходимо учесть эти зоны в ранжировании, причем таким образом, чтобы поиск стал искать лучше.

В отчете нужно привести:

- Побитовое описание индекса с зонами.
- Формулу ранжирования, подобранные веса.
- Оценку качества поиска после внедрения зон.

1 Описание

Для того, чтобы выполнить курсовой проект, нам необходимо изменить работу всех частей кода:

1. Необходимо обработать «сырые» html страницы заново, в данном случае выделять три зоны: заголовок статьи, первый абзац статьи, остальной текст.
2. При индексации обработать поотдельности каждую зону в документе.
3. При поиске, необходимо производить поиск по определенным зонам.

2 Исходный код

```
word_id SearchEngine::get_word_id(const string& word) {
    static word_id next_word_id = 0;
    auto it = words_dict.find(word);
    if (it == words_dict.end()) {
        words_dict[word] = next_word_id;
        return next_word_id++;
    }
    return it->second;
}

vector<string> SearchEngine::search(const string& query, Zone search_zone, int k) {
    istringstream iss_query(query);
    string word;
    unordered_map<doc_id, double> doc_scores;
    vector<word_id> query_word_ids;
    unordered_set<doc_id> candidate_docs;

    while (iss_query >> word) {
        word_id id = get_word_id(word);
        query_word_ids.push_back(id);

        if (inverted_index.count(word) == 0) {
            return vector<string>();
        }

        for (int zone = TITLE; zone <= search_zone; ++zone) {
            if (inverted_index[word].count((Zone)zone) == 0) {
                continue;
            }

            const vector<doc_id>& doc_ids = inverted_index[word][(Zone)zone];
            candidate_docs.insert(doc_ids.begin(), doc_ids.end());
        }
    }

    for (const auto& doc : candidate_docs) {
        if (!all_words_exists(doc, search_zone, query_word_ids)) {
            continue;
        }

        double total_tf_idf_value = 0.0;
        for (word_id id : query_word_ids) {
            for (int zone = TITLE; zone <= search_zone; ++zone) {
                total_tf_idf_value += tf_idf_index[doc][(Zone)zone][id];
            }
        }

```

```
48          doc_scores[doc] = total_tf_idf_value;
49      }
50
51      vector<string> result = get_top_k(doc_scores, k);
52      return result;
53  }
54
55  bool SearchEngine::all_words_exists(doc_id doc, Zone search_zone, const vector<word_id
        >& query_word_ids) {
56      for (word_id word_id : query_word_ids) {
57          bool word_present = false;
58          for (int zone = TITLE; zone <= search_zone; ++zone) {
59              if (find(forward_index[doc][(Zone)zone].begin(),
60                       forward_index[doc][(Zone)zone].end(),
61                       word_id) != forward_index[doc][(Zone)zone].end()) {
62                  word_present = true;
63                  break;
64              }
65          }
66          if (!word_present) {
67              return false;
68          }
69      }
70      return true;
71  }
72
73  vector<string> SearchEngine::get_top_k(const unordered_map<doc_id, double>& doc_scores
        , int k) {
74      vector<pair<doc_id, double>> ranked_docs(doc_scores.begin(), doc_scores.end());
75      sort(ranked_docs.begin(), ranked_docs.end(),
76           [](const auto& a, const auto& b) { return a.second > b.second; });
77
78      vector<string> result;
79      for (int i = 0; i < k && i < ranked_docs.size(); i++) {
80          result.push_back(docs_dict[ranked_docs[i].first]);
81      }
82
83      return result;
84  }
```

3 Выводы

Выполнив курсовую работу по курсу «Информационный поиск», я научился реализовывать зонный поиск, благодаря ему скорость выдачи некоторых запросов возрасла, так как пользователь ищет не по всем областям документов, а только в определенной. Также возрасла точность запросов, так как при ранжировании TF-IDF увеличивается точность, так как наибольший вес приходится в заголовке, потом в первом абзаце, а затем лишь в остальном тексте.

Список литературы

[1] Маннинг, Кристофер Д. Введение в информационный поиск [Текст] / Кристофер Д. Маннинг, Прабхакар Рагхаван, Хайнрих Шютце ; пер. с англ. М. Л. Суркова. - Москва : Вильямс, 2020. - 528 с.