

# Mopsi Geo-tagged Photo Search

Tania Akter

Master's thesis



ITÄ-SUOMEN YLIOPISTO

School of Computing

Computer Science

December 2020

UNIVERSITY OF EASTERN FINLAND, Faculty of Science and Forestry, Joensuu  
School of Computing  
Computer Science

Opiskelija, Tania Akter: Mopsi Geo-tagged Photo Search

Master's Thesis, 94 p., 1 appendix (7 p.)

Supervisors of the Master's Thesis: Pasi Fränti and Radu Marinescu-Istodor

December 2020

**Abstract:** Millions of databases generate massive information every year, and it is an important necessity to search that massive information by sophisticated search tools named search engines. There are numerous search tools worldwide, but most search tools cannot extract essential information effectively. Mopsi is one of them that only work for the inclusion of the keyword within the data description. To assist this Mopsi web searching system, we have developed a tool that can measure the string's syntactic similarity using character-level and token-level similarity measures. This thesis focuses on analyzing Mopsi data and developing a web tool to search the geo-tagged photo from the Mopsi database by applying the syntactic string similarity measures. To search the geo-tagged photo, we adopted an experimental case study and observation method to find the optimal threshold for each string similarity measures, flexibility and quality of each string similarity measures, and the correlation between physical distance and string similarity measures. The experimental results revealed that most of the string similarity measures perform better in optimal threshold 0.7 or above, and the Smith-Waterman-Gotoh is the best similarity measure based on data characteristics, which also produces better results than inclusion. Our tool only supports syntactic similarity measures, so there remains a gap between human intuition of what they expect to see and the results from this tool. In the future, we have a plan to integrate semantic similarity measures into our tool.

**Keywords:** Search technique, search tool, inexact search, syntactic similarity, semantic similarity, string similarity, geo-tagged photo, Mopsi database

**CR Categories (ACM Computing Classification System, 1998 version):** Computer Uses in Education, Computer and Information Science Education

## **Acknowledgment**

This thesis was done at the School of Computing, University of Eastern Finland, during autumn 2020.

I am grateful to the University of Eastern Finland administration and to all the teachers for their support during the journey of my master's degree. I would like to express my gratitude to my thesis supervisor Professor Pasi Fränti and Dr. Radu Marinescu-Istodor, for guiding me over the years. I believe my presentation skills and time management while maintaining a work-life balance improved significantly because of their constructive feedback, advice, and guidance.

I am also grateful to Dr. Oili Kohonen for her enormous support and suggestions on every matter. And special thanks to my friend Nancy for always being by my side whenever I needed her to have a conversation or making any decisions on anything.

Finally, my heartiest gratitude and love goes to my family and friends for their moral support and encouragement.

## List of abbreviations

|         |  |
|---------|--|
| UEF     | University of Eastern Finland  |
| CSE     | Computer Science Engineering   |
| IEEE    | Institute of Electrical and Electronics Engineers  |
| GIS     | Geographic Information Systems   |
| LBS     | Location-based service   |
| GPS     | Global Positioning System  |
| GLONASS | Global Navigation Satellite System   |
| BeiDou  | Big Dipper   |
| MOPSI   | Mobiilit Paikkatieto-Sovellukset ja Internet or<br>Mobile location-based applications and Internet |
| WWB     | World Wide Web   |
| JSON    | JavaScript Object Notation   |
| CSS     | Cascading Style Sheets   |
| Ajax    | Asynchronous JavaScript and XML  |
| API     | Application Programming Interface  |
| PHP     | Hypertext Preprocessor   |
| HTML    | Hyper Text Markup Language   |
| URL     | Uniform Resource Locator   |
| PMI     | Pointwise mutual information   |
| LSA     | Latent semantic analysis   |
| SWG     | Smith-Waterman-Gotoh   |



# Contents

|       |                                     |    |
|-------|-------------------------------------|----|
| 1     | Introduction.....                   | 1  |
| 1.1   | Research background.....            | 2  |
| 1.2   | MOPSI.....                          | 4  |
| 1.3   | Structure of this thesis .....      | 4  |
| 2     | Searching techniques .....          | 6  |
| 2.1   | Linear or sequential search .....   | 8  |
| 2.2   | Binary or interval search.....      | 9  |
| 2.3   | Hybrid search.....                  | 11 |
| 2.4   | Exact search .....                  | 12 |
| 2.5   | Inexact search .....                | 14 |
| 3     | Relevance factor .....              | 16 |
| 3.1   | Keyword relevance .....             | 17 |
| 3.2   | Location relevance.....             | 18 |
| 4     | Semantic similarity .....           | 20 |
| 4.1   | Corpus based measures.....          | 23 |
| 4.2   | Knowledge-based measures.....       | 24 |
| 5     | Syntactic similarity .....          | 26 |
| 5.1   | Character-level measures.....       | 27 |
| 5.1.1 | Levenshtein distance.....           | 28 |
| 5.1.2 | Damerau-Levenshtein distance.....   | 29 |
| 5.1.3 | Needleman–Wunsch algorithm.....     | 29 |
| 5.1.4 | Smith–Waterman algorithm.....       | 30 |
| 5.1.5 | Smith–Waterman–Gotoh algorithm..... | 31 |
| 5.1.6 | Hamming distance .....              | 32 |
| 5.1.7 | Jaro distance.....                  | 32 |
| 5.1.8 | Jaro–Winkler distance.....          | 33 |
| 5.1.9 | Longest common substring.....       | 33 |
| 5.2   | Token-level measures .....          | 37 |
| 5.3   | Soft measures.....                  | 38 |
| 5.3.1 | Simpson similarity .....            | 39 |
| 5.3.2 | Jaccard similarity .....            | 40 |
| 5.3.3 | Soft-Cosine similarity .....        | 40 |
| 5.3.4 | Euclidean distance .....            | 41 |
| 5.3.5 | Manhattan distance .....            | 42 |
| 6     | Implementation .....                | 45 |
| 6.1   | Tool description .....              | 45 |
| 6.2   | Technology .....                    | 49 |
| 7     | Experiment.....                     | 52 |
| 7.1   | Datasets.....                       | 52 |

|       |  |    |
|-------|--|----|
| 7.2   | Experimental setup .....                                   | 53 |
| 7.2.1 | Optimal threshold for each string similarity measure.....  | 61 |
| 7.2.2 | Quality of the measures .....                              | 67 |
| 7.2.3 | Correlation to physical distance and similarity measure .. | 70 |
| 8     | Conclusions.....   | 73 |
|       | References.....  | 74 |

## **Appendices**

[Appendix 1](#): Checklist (7 pages)

# 1 Introduction

In the 21st Century, Location-Based Services (LBS) widely apply in computing systems and applications. Technological advances like the World Wide Web (WWW), Global Positioning Systems (GPS), and mobile devices make location-based services feasible (Brimicombe & Li, 2009). By incorporating satellite navigation data, cellular networks, and mobile computing, location-based systems provide user's geographical locations (A. Ahson & Ilyas, 2011). Location-based services, including Nokia Ovi Service, YellowPages, and Google Maps, are usually based on databases that specifically georeferenced all records while saving throughout the database (Fränti, Tabarcea, Kuittinen, & Hautamäki, 2010). Recently the research of location-based services increasing in both the academic and commercial sectors (Tabarcea, 2015).

The United States defense department designed the Global Positioning System (GPS) throughout the 1970s, and it had released publicly to users around the globe throughout the 1980s (Schiller & Voisard, 2004). GPS is a satellite-based navigation and positioning system mainly used for vehicle navigation, smartphones, and land surveys. Besides United States-based GPS, Russian GLONASS, European Galileo, and Chinese BeiDou use to navigating moving objects (Nls, 2020). It provides quick, accurate, and economical services to determine the position and velocity of any objects on the earth at any place with the help of signals received from satellites put in earth-centered orbits (Xie, 2019). Users can conveniently acquire a large number of trajectory data through moving objects to advance satellite positioning technologies.

By looking at various things, such as the position and navigations of moving objects, time, places, contents, or social networks, LBS and GPS can determine each user's context. Similarly, using LBS and GPS characteristics, we have developed a tool to assist the Mopsi<sup>1</sup> web searching tool for searching the user's query. The Mopsi is a location-based application designed to collect, process, and represent location-based

---

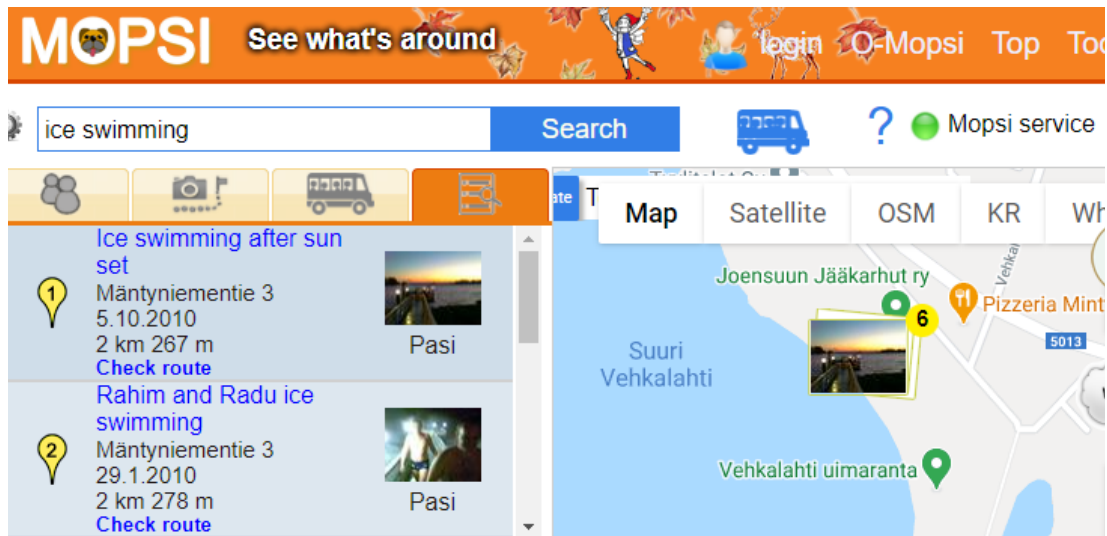
<sup>1</sup> <http://cs.uef.fi/mopsi/>

data as geo-tagged photos and trajectories (Tabarcea, 2015). The geo-tagging applies to accurate global positioning data to a web page or photograph found online using certain latitude and longitude information. It is the easiest way to incorporate localized data into a website by inserting image and photo information (Romain, 2019).

## **1.1 Research background**

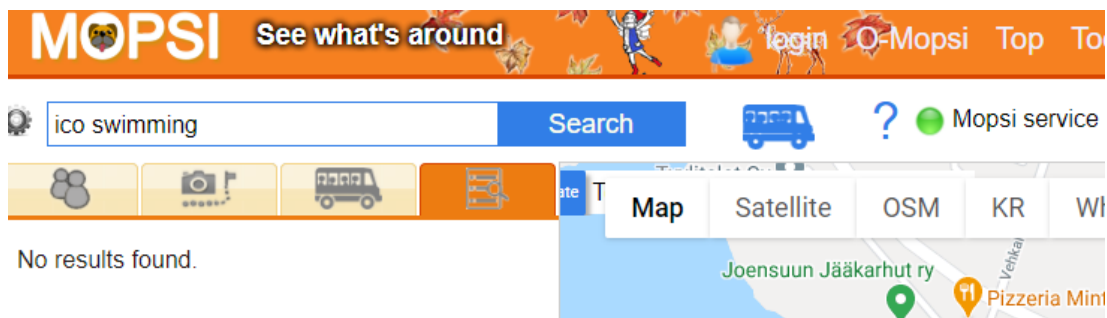
In data or information retrieval from the web, each user has a different factual aspiration (Khatter & Ahlawat, 2020) while seeking information on web pages. The web search engine can extract the essential data or information from the web page based on the search query. Effective syntactic comparable designs, image retrieval, vibrant preparation, and assessment tools are needed to improve the syntactic search capability (Tabarcea, 2015). In the same way, the traditional keyword-based syntactic search algorithms can understand user's expectations compared to the semantic search algorithm (Kaur, 2015). By taking the above advantage of keyword-based syntactic search, we have decided to develop a tool that can assist the Mopsi for approximate searching besides inclusion.

The current Mopsi website provides location-based and navigational services (Xie, 2019). The search system of Mopsi works based on inclusion, which means it will show results for any search keywords that are exactly included with the Mopsi data descriptions. The keyword is compared with the description of the Mopsi database's photo to perform the matching and to produce the output. The maximum number of results is predefined as 10, and a binary search is performed to find these results. The ordering of the results has conducted based on the physical distance because the similarity of searching keywords is the same for all results. We have shown an example of the inclusion based Mopsi searching in Figure 1 below. Where the output results for the keyword "ice swimming" is exactly included in the Mopsi database.



**Figure 1. Inclusion based keyword searching in Mopsi (<http://cs.uef.fi/mopsi/>).**

If a user typed a misspelled keyword (i.e., "ico swimming") while searching in the Mopsi, what will happen? Previously we knew that the existing searching in Mopsi only works for inclusion. The problem with inclusion is that it does not show any results for misspelled keywords and misspelled data description. An example of a misspelled scenario has shown in Figure 2 for the Mopsi web searching. Figure 2 shows that the existing Mopsi does not show any output results for inexact or spelling mistakes. The spelling mistake is directly indicating the approximate or inexact searching.



**Figure 2. Searching with misspelled keyword in Mopsi (<http://cs.uef.fi/mopsi/>).**

Another scenario might be if the data contains some relevant information other than syntactically similar keyword, inclusion based searching will fail to consider those data as relevant. By analyzing the above limitations of existing Mopsi, we have developed a web searching tool considering syntactic similarity measures to perform an approximate search. Where, the syntactic similarity is a metric that measures the

grammatical structure of words, short text, and sentences (Harispe, Ranwez, Janaqi, & Montmain, 2015).

Our tool can help users to find a nearby location, distance, and photos from their smartphones or computers. Users can obtain or view the list of geo-tagged photos based on their given address or location. This tool can calculate the physical distance between the user and each data source from the Mopsi database to find the expected geo-tagged photo or output. It can also calculate the syntactic similarities between two strings. In custom search options, users can select multiple parameters such as numbers of results, ordering, strings similarity measurement methods, string similarity threshold, distance radius based on their preference.

## **1.2 MOPSI**

We have done every experiment of our thesis work on the Mopsi website. This website was established by the University of Eastern Finland's computing department's speech and image processing group. Mopsi can provide location-based services such as geo-tagged photos, location mining and data processing, filtering and retrieval of GPS trajectories, user's activity, and moving object exposure from GPS trajectory (Xie, 2019). It helps users find locations, distances, geo-tagging photos, and trajectories for walking, running, cycling, and skiing. It also offers for capturing, sorting, presenting, and combining location-based data. The Mopsi website is convenient for all operating systems of phone and computer (Tabarcea, 2015).

## **1.3 Structure of this thesis**

This thesis is constructed of eight sections and prepared according to the following: Section 1 is an introduction, which covers the research background, Mopsi, and structure of this thesis; Section 2 describes searching technique that presents linear or sequential search, binary or interval search, hybrid search, exact search, and inexact search; Section 3 describes relevance factor, which consists of keyword relevance and location relevance, Section 4 is about semantic similarity that covers corpus-based measures and knowledge-based measures; Section 5 describes syntactic similarity which covers character-level measures, token-level measures, and soft

measures; Section 6 is an implementation which includes tool description and technology; Section 7 is an experiment which interprets and describes datasets, libraries, experimental setup, and results; Section 8 is a conclusion, which includes observations and future tasks.

## 2 Searching techniques

Searching is an important technique for processing large databases. It can use on internal or external data structures or any list of values with their appropriate indexing. It is a process of finding the index of a given element in the array. Searching considers success if the elements are found and unsuccessful if not found, but the element is there, or it exists at the wrong index. It is also the algorithmic way of detecting a specific item in a list of components (TutorialRide, 2017). The searching can be a feature of seeking files, documents, records, data, folders, reports, websites, weblinks, blog posts, and other information (ComputerHope, 2018). For example, the University of Eastern Finland's search box is at the top of their page (see Figure 1).

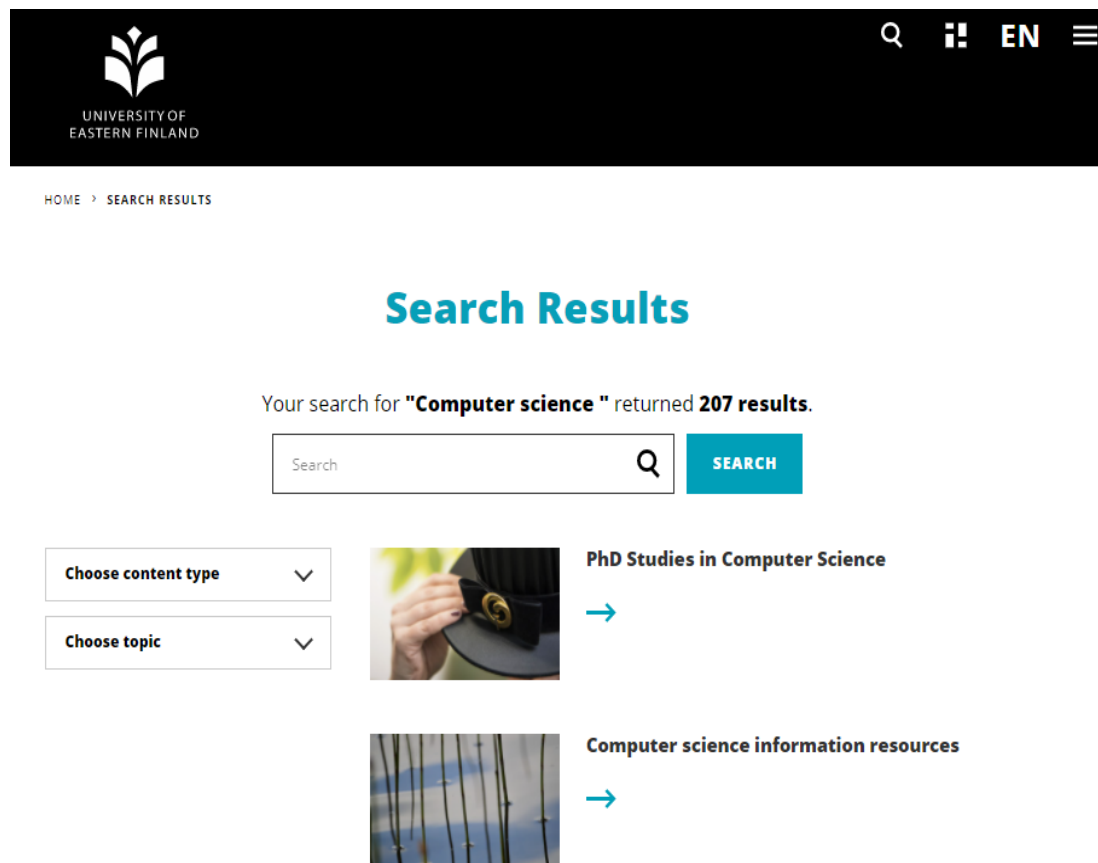


Figure 3. The search box for the University of Eastern Finland webpage (uef.fi).

When users access the internet, they often use a search engine (CodeOrg, 2017). The first thing to note is that the search engine is not actually traversing the World Wide Web to run a query in real-time because there are over a billion websites on the internet, and hundreds more are created every single minute (Sheela & Jayakumar,



2019). Therefore, search engines are continuously running a program called spider that crosses through web pages in advance to record the information and index information for later search. As an example, the crawler (spider) indexes many pages that have to do with space travel or the planet Mars and then the search looks through those indices. When you search about travel to Mars (see Figure 2), the search engine already has what it needs to give you an answer in real-time.

|   |                       |  |
|---|-----------------------|--|
| Keyword <b>how long does it take to travel to Mars?</b> |                       | SEARCH ENGINE                            |
| SEARCH INDEX  |                       |  |
| how   | 8,420,000,000 results |  |
| long  | 6,150,000,000 results | DETERMINE THE BEST MATCHES TO SHOW FIRST |
| does  | 3,880,000,000 results |  |
| take  | 4,900,000,000 results | 27,684,000,000<br>TOTAL RESULTS          |
| travel  | 3,480,000,000 results |  |
| Mars  | 854,000,000 results   |  |

Figure 4. The search engine came up with the results (CodeOrg, 2017).

When a user accessed a search engine, all relevant page's database lists use an algorithm (Marsden, 2020) to organize the relevant pages hierarchically into a result list. On the other hand, for a searching query, the search engine applies other essential data to obtain results, along with the location (as 'pizza shop near me' or 'sunset time's or 'travel time to Mars. '), language detection (as English, Finnish, French, etc.), earlier search information (what users have previously looked for?), and machine (as the machine that the query has created).

The methods of information retrieval from stored datasets or documents are called search techniques. A large number of data or information may store on the web in a structured, semi-structured, or non-structured way (Mala & Lobiyal, 2016). So, it is hard to find exact information from the search engine. To handle databases or the internet appropriately, you can use multiple techniques for information search or retrieval. It is also possible to precise search output by applying advanced search

options (as boolean logic, phrase search, etc.) (Erasmus, 2020). Linear search and Binary search are the most popular algorithms for searching for an object in a database (Jacob, Ashodariya, & Dhongade, 2017). The following search techniques are mostly used for information retrieval: linear or sequential search, binary or interval search, hybrid search (the combination of different search technique), exact search, inexact search, semantic search, syntactic search or keywords based search, content-based search, etc. Figure 3 is an example of an advanced (Boolean) search in IEEE Xplore:

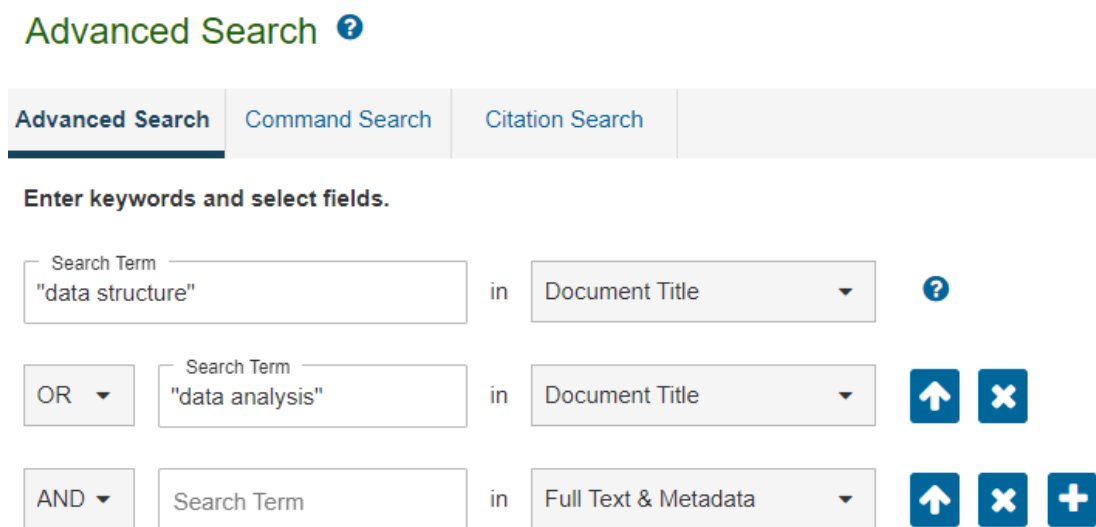


Figure 5. Advanced (boolean) search in IEEE Xplore (ieeexplore).

## 2.1 Linear or sequential search

In computing, various search algorithms apply with a dataset, and the linear search (Jacob, Ashodariya, & Dhongade, 2017) is one of them. A linear search is a technique of seeking an item within a set of the dataset in computing. Each member of the dataset sequentially examines before matching is identified or the entire dataset is checked. But in linear search, every element of the dataset is examined separately and sequentially, so it takes a huge time for a search. A Linear or sequential search algorithm has given bellow:

**Input:** X is a shorted array and Y is a value to find in the array.

**Output:** True if Y is in X, else false.

```
function linear_search(list, value)
```

```

for each item in the list
    if match item == value
        return the item's location
    end if
end for
end procedure

```

The time complexity of linear search at worst case is  $O(n)$ , best-case is  $O(1)$ , and the average case is  $O\left(\frac{n}{2}\right)$ . On the other hand, the space complexity is  $O(n)$ . A linear search example has shown in Figure 4 below, where an array  $X[i] = \{10, 54, 31, 75, 82, 60, 20, 91, 44\}$  and searched value  $Y = 20$ . Now, we are going to find value 20 is present in X or not.

| Search Y=20 |    |    |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|----|----|
| Index i     | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  |
| X[i]        | 10 | 54 | 31 | 75 | 82 | 60 | 20 | 91 | 44 |

Figure 6. Linear or sequential search.

## 2.2 Binary or interval search

In computing, various search algorithms apply with a dataset, and the binary or interval search (Jacob, Ashodariya, & Dhongade, 2017) one of them [re-write sentence because of repetition]. Binary search segments the arrays or datasets between segments and matches each segment's or dataset's central element to the searchable key element. Binary or interval search algorithms need to store and order the dataset, which consumes a long time. The time consuming happens once when the data is inserted or need to update. On the other hand, it is beneficial to find the item in  $O(\log(n))$  time instead of  $O(n)$  time. A binary search algorithm has given below:

**Input:** X is a sorted array and Y is a value to find in the array.

**Output:** True if Y is in X, else false.

**function** *binary\_search*(list, value)

*mid* = *X.size*/2

**if** *X.size* == 0 **then return false**

**if** *X.size* == 1 **then return** *X*[0] == *Y*

**if** *X*[*mid*] == *Y* **then return true**

**if** *X*[*mid*] < *Y* **then return** *binary\_search*(*X*[*mid* + 1 ... end], *Y*)

**if** *X*[*mid*] > *Y* **then return** *binary\_search*(*X*[0 ... *mid* - 1], *Y*)

The time complexity of binary search at worst case is  $O(\log n)$ , best-case is  $O(1)$ , and the average case is  $O(\log n)$ . On the other hand, space complexity is  $O(1)$ . A binary search example has shown in Figure 5 below, where an array  $X[i] = \{3, 4, 8, 13, 17, 20, 30, 45, 52, 60\}$ , and searched value  $Y = 20$ . Now, we are going to find value 20 is present in  $X$  or not.

|                        |         |          |   |   |    |          |          |          |          |    |     |
|------------------------|---------|----------|---|---|----|----------|----------|----------|----------|----|-----|
| Search<br>Y=20         | Index i | 0        | 1 | 2 | 3  | 4        | 5        | 6        | 7        | 8  | 9   |
|                        | X[i]    | 3        | 4 | 8 | 13 | 17       | 20       | 30       | 45       | 52 | 60  |
| 20>17 take<br>2nd half |         | 1st half |   |   |    |          | 2nd half |          |          |    |     |
|                        | Index i | L=0      | 1 | 2 | 3  | M=4      | 5        | 6        | 7        | 8  | R=9 |
|                        | X[i]    | 3        | 4 | 8 | 13 | 17       | 20       | 30       | 45       | 52 | 60  |
| 20>45 take<br>1st half |         |          |   |   |    | 1st half |          |          | 2nd half |    |     |
|                        | Index i | 0        | 1 | 2 | 3  | 4        | L=5      | 6        | M=7      | 8  | R=9 |
|                        | X[i]    | 3        | 4 | 8 | 13 | 17       | 20       | 30       | 45       | 52 | 60  |
| Found 20,<br>Return 5  |         |          |   |   |    | 1st half |          | 2nd half |          |    |     |
|                        | Index i | 0        | 1 | 2 | 3  | 4        | L=5=M    | R=6      | 7        | 8  | 9   |
|                        | X[i]    | 3        | 4 | 8 | 13 | 17       | 20       | 30       | 45       | 52 | 60  |

**Figure 7. Binary or interval search.**

## 2.3 Hybrid search

A hybrid search is a combination (Jacob, Ashodariya, & Dhongade, 2017) of the linear and binary search algorithms, which carried out both algorithm's advantages. This algorithm takes less time to the unsorted array or dataset compared with the linear search. A hybrid search algorithm has given bellow:

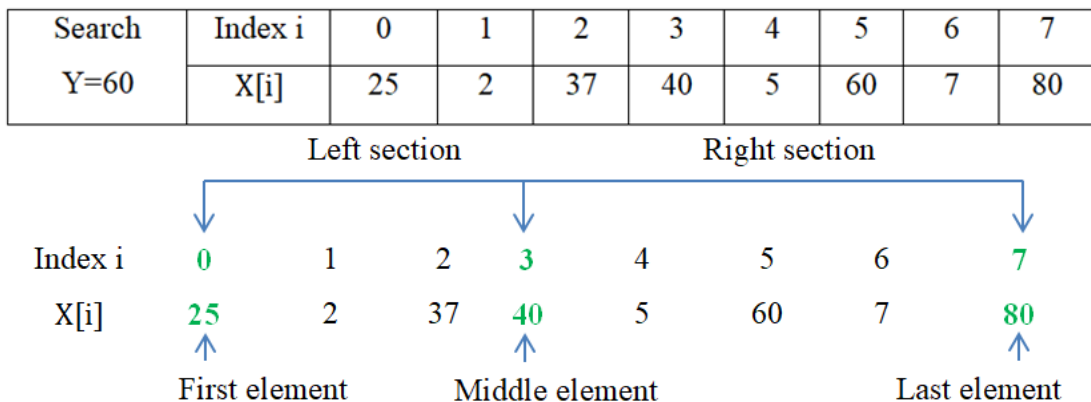
**Input:** Array X, lower index low, higher index high, and value to be searched Y.

**Output:** returns index of the element if the element is present else if the element is not present then it returns -1.

```

function hybrid_search(X, low, high, Y)
    mid = (low + high)/2
    if X[low] = Y then return low
    else if X[high] = Y then return high
    else if X[mid] = Y then return mid
    else if low ≥ high - 2 then return - 1
    else p = hybrid_search(X, low + 1, mid - 1, Y)
    if p = -1
        p = hybrid_search(mid + 1, high - 1, Y)
    return p
  
```

The time complexity of hybrid search at worst case is  $O(n)$ , best-case is  $O(1)$ , and the average case is  $O(\log_2 n)$ . On the other hand, space complexity is constant because it only requires memory to store an array. A hybrid search example has shown in Figure 6 below, where an array  $X[i] = \{25, 2, 37, 40, 5, 60, 7, 80\}$  and searched value  $Y = 60$ . Now, we are going to find value 60 is present in X or not.



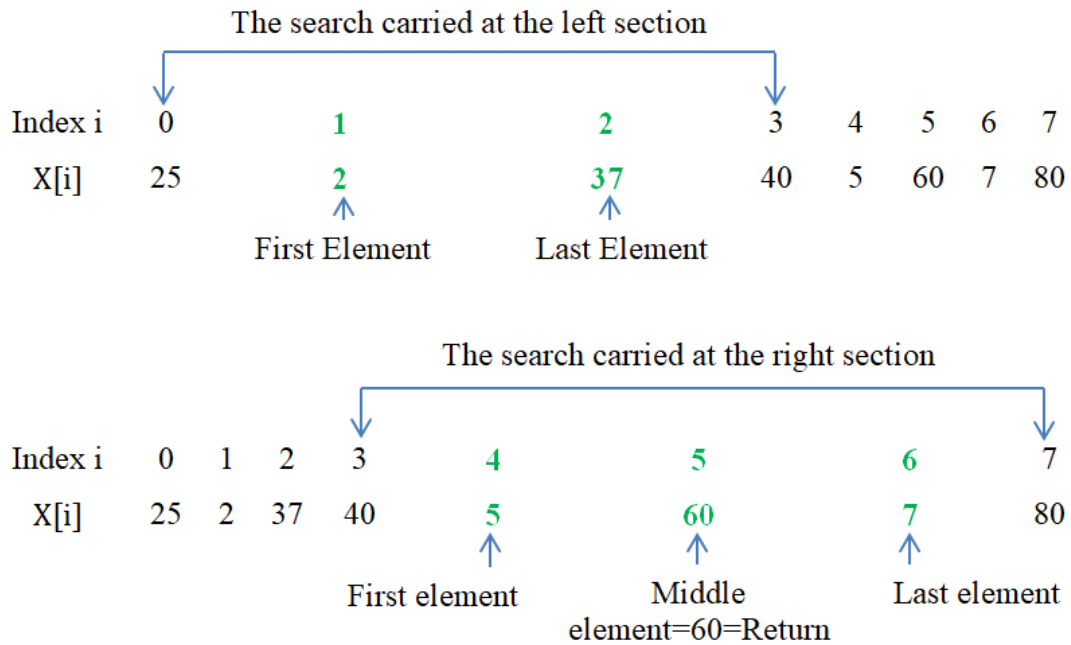


Figure 8. Hybrid search.

## 2.4 Exact search

Exact searching or matching is a highly developed search function that can be useful if you only have to see the database that matches your quest (Kehrer, 2018). An exact search or match keyword shows the correct keyword exactly represents search terms, relevant keywords, or web address. You can present your advertisement precisely to clients who are looking for their exact keywords or similar variants. In exact matching method (Hakak, 2019), the given text T detects from a specific sequence or pattern P. The length of both sequences of T and P must be the same where the text characters T are compared with the pattern of window characters P. There are two types of exact matching algorithms: a single pattern-matching algorithm and multiple patterns matching algorithms. Single pattern matching algorithm allows one sequence as input and searches it from the target databases. On the other hand, the multiple patterns matching algorithms allow one input and search it multiple ways from target databases. The character-based algorithm is a single pattern-matching algorithm that matches characters to determine string matching problems. There are several types of character-based matching algorithms, and the brute force pattern matching algorithm is one of them.

The Brute force algorithm is the most straightforward algorithm (Abdeen, 2019) to apply for pattern searching. For pattern and text, it does not any step for training and testing. It checks for all the places within 0 and  $n-m$  in the text. In this algorithm, the searching performs by character between the pattern and a text. The time complexity of this algorithm is  $O(nm)$ . If the pattern is  $P[0..m-1]$  and text is  $T[0..n-1]$ , then the brute force pattern matching algorithm (Janani & Vijayarani, 2006) will be:

**Input:** Strings  $T$  with  $n$  character and  $P$  with  $m$  character.  
**Output:** String index of the first substring of  $T$  matching  $P$ ,  
or an indication that  $P$  is not a substring of  $T$ .  
**for**  $i := 0$  **to**  $n - m$  **do**  
     $\{j := 0$   
        **while** ( $j < m$  **and**  $T[i + j] = P[j]$ ) **do**  $j := j + 1$   
        **if**  $j = m$  **then return**  $i$   
     $\}$   
**return** "there is no substring of  $T$  matching  $P$ ."

If a logical binary outcome gives an exact match: 1 = exactly the same string, 0 = no common string. This logical binary outcome is the standard way to analyze strings while collecting information (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). An exact searching or matching example has shown in Figure 7 below.

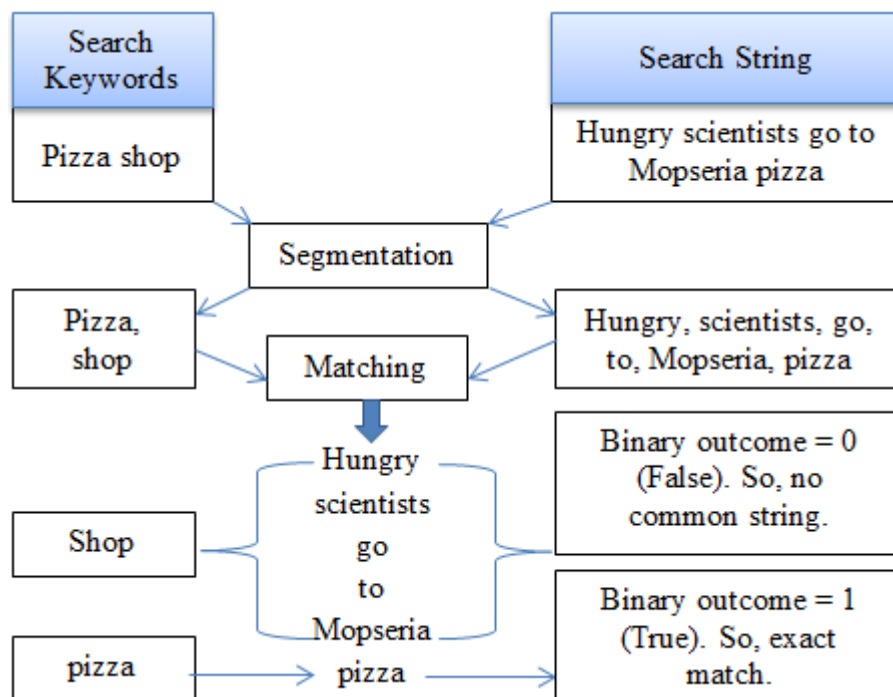


Figure 9. Exact matching (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

## 2.5 Inexact search

The measurements or determinations of two entities structural relationships have some random noise, leading to the study of the approximate or inexact string matching (Shapiro & Haralick, 1981). In computing, the method of identifying a string that roughly matches a pattern is called inexact or approximate or fuzzy string matching (Navarro, Baeza-Yeats, Sutinen, & Tarhio, 2001). It is the most significant problem of similarity calculation, data processing, computer vision, image processing, and many computing branches (Benza-Yeats & Ganzalo, 1998). Close variants include keyword queries, regardless of spelling or syntax similarities between the question and the keyword with the same meaning as the same keywords (Manage-ads, 2020). When anyone looks for your keyword or similar-ups of your keyword, it will reveal your advertisement. Similar variants can contain (Manage-ads, 2020): incorrect spelling of a word, types of those are single or plural, originate from (as roof and roofing), shortcut keys (as IEEE, UEF, and CSE), a defining style of speech of a language, similar meanings of expressions (as "pizza shops" and "shops pizza"), including or eliminating words (as in, to, for, but, a, an, the) from function (as "pizza shops" and "shops of pizza"), antonyms, synonyms, and phrasing.

Usually, the query of inexact string matching is classified into two sub-problems: looking substring into the main string and looking dictionary string into the pattern roughly. Two main levels can apply for precise searching for approximate string matching (Kumar, Bibhu, Islam, & Bhardwaj, 2010) as string similarity measures and phonetic coding.

**String similarity measures:** It examines the numerical calculation of the number of characters matched between two strings or the number of actions required to change one string to another string. Those examinations are usually associated with edit distances. Usually, approximate string measures can apply to classify a series by a dictionary concerning a query string. These measures are typically considered suitable for spelling correction, with around most random mistakes being a single insertion, deletion, or swap.



**Phonetic coding:** In similarity measures, there is considering each string has a phonetic code. If two strings have the same code, then they are considered similar otherwise dissimilar. Phonetic coding is relevant to individual name matching because it can have a unique writing format and a similar sound.

An example of approximate string searching has bellowed:

## Search results

---

This wiki is using a new search engine. ([Learn more](#))

[Content pages](#) [Multimedia](#) [Translations](#) [Everything](#) [Advanced](#)

Did you mean: ***andré emotions***

Figure 10. Approximate string matching (<https://en.wikipedia.org>).

### 3 Relevance factor

Today's consumers have strong aspirations, and the query's relevance factor (Blandineau, 2020) is vital to the customer experience. The importance of the relevance factor of search is the tests of precision of the search query concerning the search outcome. It is the optimization of the search query that applies to measures that aim to enhance search outcomes. There are various steps to fine-tune the search relevance factor to order the search outcome. To fine-tune the search results, search keywords, user location, distance, textual and spelling correctness, recency of data, quality of the content, and popularity can be good relevance factors. A combination of those factors helps users to find the best search results. We have explained some scenarios below based on keyword, location, distance, syntactic, and semantic similarity to understand the impacts and importance of relevance searching factors.

For example, suppose a user searches for a "car repair shop" from any city center. In that case, the results will more likely be the nearest car repair shops from the user-specified location or the user's current location. Suppose the results show some car repair shops which are 100 kilometers away. In that case, these are not the expected results in general, especially as the user stands in a city center and there should be other shops closer than the results. But if the user is in some remote/rural area, and there are no car repair shops within 100 kilometers radius from the user's location. In that case, this should be the best result considering the situation. In another case, some results may include "grocery shop" instead of "car repair shop" because both have the common word "shop," but these are not expected results. Here, the keyword is one major factor to determine the results because the user only needs to find the "car" repair shop, not any "grocery" shops even though it is nearby. Here, both the semantic and syntactic similarity between the keywords and the results strings would work for the keyword matching.

If someone searches for "pizza," and there is no pizza place closer to them, the user can get results for the nearest burger place. In that case, semantic similarity for string matching would be a better option to solve this issue.

From the above examples, we can determine that every navigational search engine must contain both keyword and location as independent relevance factors to find the most matching output on searching. In this thesis, we will discuss and define both relevant factors for analyzing their influences on searching. We have shown some of the relevant factors in Figure 11.

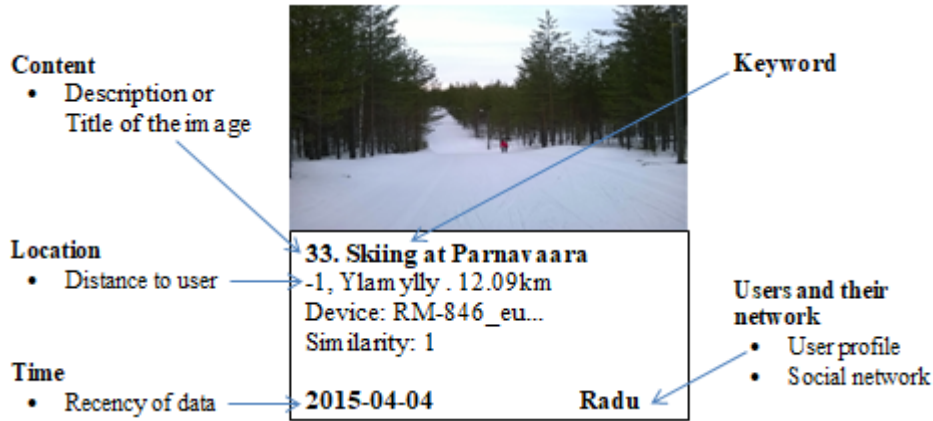


Figure 11. Relevance factors.

### 3.1 Keyword relevance

Keywords are search statements that customers require to type on a search engine to look for a product or service (Kapoor, 2019). To determine the relevance between the search queries and the output results, keywords are most important (Kent, 2019). It will not perform at all, except the web page is essential to the search strings. It helps the web page to rank more relevant content. We have explained some keyword-based searching scenario below.

The scenario might be, a user enters the keyword as "resturant" instead of "restaurant" and found no result because of a typing mistake while entering the keyword. On the other hand, there might be some data in the database which contains a typo (misspelled title), so in that case, even if the user put the correct keyword, still some true positive result might be missing. The syntactic string similarity measurement can be useful for keyword-based matching or searching to overcome those above limitations. Now, a syntactic keyword-based searching scenario has shown in Figure 12.a.

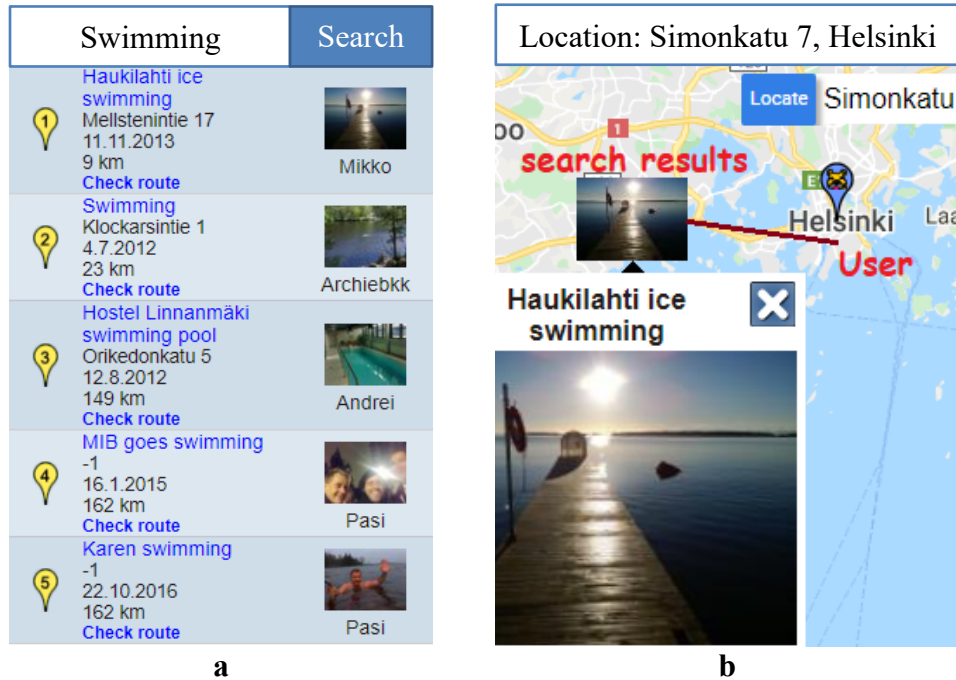


Figure 12. Keyword and location based searching (<http://cs.uef.fi/mopsi/>).

### 3.2 Location relevance

Location relevance is considered to determine the physical distance between the user's defined location and the location of the targeted place. From the user's current location to the targeted place can define a distance. A location a location-based searching scenario has discussed below.

In our developed tool, users can choose the distance radius to filter their results. For example, the query can be something like, "find swimming within 150 kilometers from user's current location". Here, "swimming" is the keyword, and "150km" is the distance radius. If the user does not specify the distance radius, then the results may contain data from all over the world. There is a default value of distance, or a distance can be set by default in the main search. We have shown a location-based searching scenario in Figure 12.b.

We have ranked both factors from the discussion of keyword and location relevance, which significantly influence the searching process.

1. For keywords-based ranking, the output results will be categorized according to the search terms or keywords. If two search output have the same keywords

(similarly will be one), then the output results will be categorized according to the nearest physical distance.

2. For location-based ranking, the output results will be only categorized according to the nearest physical distance, not keywords.

Besides keyword and location, we have also analyzed some other factors such as, recency of data, quality of the content, popularity, and social network. Those factors are also essential to determine search results. A short description of those relevant factors has given below.

**Recency of data:** The recency of data or updated data mainly important for news or trending topic search. For example, if someone searches for a query as "today's events near me," the date of the event must be the current date. Another example, if someone searches for "weather information" or "stock price," which means they want to know the most updated information on those topics.

**Quality of the content:** Quality of the content helps users to find the most relevant searching results based on provided information with data or user's ratings. For example, when we search for a place to visit or landmarks, places with high user ratings or recommendations will be the top results.

**Popularity:** Popularity or prominence refers to how well known the searching queries. Some places are more prominent in the offline world, and search results reflect this in the local ranking. For example, famous museums, landmark hotels, or well-known store brands can be prominent in local search results.

**Social network:** Social networks can also influence search results. For example, if a member from a group of people (as classmates) looking for some popular thing, other group members can be influenced by that member to looking for the same thing.

Based on the type and structure of the Mopsi database, we can conclude that recency of data, quality of the content, popularity, social networks are not that essential factors for Mopsi data searching. For developing our tool, we have given preference to keyword and location as our relevance factor.

## 4 Semantic similarity

The World Wide Web grows every day, making it hard to obtain the necessary information. One efficient aspect which can get information from the web is search engines. The discovery by search engines allows users to discover information on the web. But, the precise information from this vast volume of data on the web is no simple job (Hussan, 2020). Search engines are essentially used for the extraction and collection of a specific type of web content. Users can obtain the required data by using the search engine based on their application. The data created by the search engine may be documents, script, message, file, letter, picture, article, audio files, and video files (Sheela & Jayakumar, 2019). However, all created data types are stored as a binary digit.

People may look for a title, location, image, name, and other brand recognition in several web services using keywords that include different orthography, combinations, and various forms comparable, but not the same term as the required individual. Within two titles, it should be possible to decide if the markers reflect the same individual through an appropriate test of similarities (Gali, Mariescu-Istodor, & Fränti, 2016). A series of similarities is expected in biotechnology, microbiology, image analysis, data processing, neural network, machine learning, speech recognition, image recognition, big data analysis, quantum computing, robotics, virtual reality, and data extraction. For example, in data extraction, a similarity measure is used to extract the related data to a user's request (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

In similarity measure, the title name of information is essential to extract the related data. A title is a simple summary of a post, item, documents, picture, object, or website characterizing it from other entities (Gali, Mariescu-Istodor, & Fränti, 2016). There are different numbers of methods that have been developed to extract the similarity between the titles. The major ones are semantic and syntactic similarities. Semantic similarity is a measurement specified over various data sources or concepts that exclude grammatical similarities through the idea of distinction among objects (Harispe, Ranwez, Janaqi, & Montmain, 2015). Semantic web technologies can em-

phasize through information instead of sentence structure, allowing search engines to determine the significance of keywords rather than the keyword sentence structure. Therefore, the data's reliability obtained from a search result will contribute to an efficient role in traditional search engines (Hussan, 2020).

The Semantic Search Engines are the smart engines that query for keywords according to their relevance (Hussan, 2020). The semantic web provides a concise and relevant result because it is capable of meaningful analysis of the query (Sheela & Jayakumar, 2019). Furthermore, they ensure the findings relevant to the context of the keywords sought. They use conceptual frameworks to obtain significant findings and maintain a high level of precision results, and link with associated data (El-gayar, Mekky, & Atwan, 2015). They also differentiate between trustworthy data sources rather than the single data source connections are different forms of linked references (Hussan, 2020). The following requirements should be taken into account by the semantic search engine: user interface, productivity, efficiency, performance, quality, reliability, flexibility, time, method classification, usability, and economic efficiency (El-gayar, Mekky, & Atwan, 2015). Different types of semantic search engines exist, such as Kosmix, Hakia, Congition, DuckDuckGo, Lexxe, and Swoogle (Sheela & Jayakumar, 2019). For example, the semantic similarity between two strings "child" and "kid" will be 100% due to the same meaning. On the other hand, those stings are syntactically 40% similar due to common characters "i" and "d". Existing semantic similarity measures can be categorized into two main categories as text semantic similarity and similarity of words (Mihalcea, Corley, & Strapparava, 2006; Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

**Text semantic similarity:** Estimates of semantic similarities generally identify (Mihalcea, Corley, & Strapparava, 2006) among words or concepts and far less among two or more texts. The importance of word-to-word similitudes is apparently due to information that expresses connections among terms and ideas and the different testing grounds that permit their assessment. Furthermore, the theorem of a text-to-text similarity measure associated with a word-based similitude measure may not be easy. Subsequently, most studies of text similarities have widely thought implementations of the general framework. For any two input texts, we want to determine their semantic level.

$$Sim(T1, T1) = \frac{1}{2} \left( \frac{\sum_{w \in \{T1\}} (maxSim(w, T2) * idf(w))}{\sum_{w \in \{T1\}} idf(w)} + \frac{\sum_{w \in \{T2\}} (maxSim(w, T1) * idf(w))}{\sum_{w \in \{T2\}} idf(w)} \right)$$

Where T1=Input text 1, T2=Input text 2, Similarity score 0=No semantic similarity between two text, and similarity score 1=Identical semantic similarity between two texts.

**Similarity of words:** It measures similarities or connections or the association of words. You can see the similarity and distinctions between two words when you compare them. In various literatures, there is a relatively high number of word-to-word similarity metrics (Mihalcea, Corley, & Strapparava, 2006). The similarity of a word can be categorized into two main categories as corpus-based measures and knowledge-based measures (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019; Mihalcea, Corley, & Strapparava, 2006).

A semantic similarity measures algorithm has given bellow (Benharzallah, Kazar, & Caplat, 2011):

**Require:** *Ontology*  $O_1$  and  $O_2$ ,  $e1 \in O_1$ ,  $e2 \in O_2$

Calculation SimN of e1, e2,

Calculation SimC of e1, e2,

Calculation SimV of e1, e2,

Calculation SimR of e1, e2,

$$SimTer(e1, e2) = \alpha_1 \times SimN + \alpha_2 \times SimC$$

$$SimStruc(e1, e2) \leftarrow \beta_1 \times SimV + \beta_2 \times SimR$$

$$Sim(e1, e2) \leftarrow \alpha \times SimTer + \beta \times SimStruc$$

**End**

Where, e1,e2=Two elements, Organizational unit= $\alpha \in [0,1]$ ,  $\beta \in [0,1]$ ,  $\alpha_1 \in [0,1]$ ,  $\beta_1 \in [0,1]$ ,  $\alpha_2 \in [0,1]$ ,  $\beta_2 \in [0,1]$ , SimTer=Terminological similarity, SimStruc=Structural similarity, SimN=Name similarity, SimC=Comments similarity, SimV=Vicinity similarity (surrounding area), SimR=Roles similarity.



## 4.1 Corpus based measures

Corpus-based string similarity measures the semantic similarity between strings and leads to determine the level of similarity throughout terms with data from extensive corpora. We have analyzed (Mihalcea, Corley, & Strapparava, 2006; Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019) three corpus-based metrics, namely: Latent semantic analysis (LSA) metrics, pairwise mutual information (PMI) metrics, and Word2Vec metrics.

**Latent semantic analysis (LSA) metrics:** LSA analysis is artificial intelligence, data, language, machine learning, and speech recognition processing (Landauer, Foltz, & Laham, 1998) mathematical or statistical technique for analyzing connections among various documents and their features through multiple aspects of the documents and terms. It assumes that close-to-meaning words will appear in related text parts. It does not allow the use of humanly built dictionaries, information bases, semantic databases, grammars, syntax compilers, microstructure, etc. A word paragraph matrix generates wherever each meaning shows how frequently the word in this subparagraph transpires. The singular value decomposition applies for seeking a reduced dimension of the matrix (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). The similarity of words is determined by taking the cosine angle of two vectors that match the comparable words (Mihalcea, Corley, & Strapparava, 2006). It also uses written data as an input, divided into relevant sequences or examples, such as phrases or sections, and established as a single element sequence (Landauer, Foltz, & Laham, An introduction to latent semantic analysis, 1998). LSA can be construed in two ways: one is calculating the contextual use substitution features of terms for broader documents, and the second one is the development and application of skills.

**Pointwise mutual information (PMI) metrics:** PMI is a measurement used in theory and statistics of information (Church & Hanks, 1990). In pointwise mutual information, the mutual information consists of the average of each event that occurs. Pointwise mutual information is obtained by incorporating data by information retrieval (PMI-IR) to determine the semantic similarity of words (Mihalcea, Corley, & Strapparava, 2006). The equation below describes the principle of PMI-IR similarity measures.

$$PMI - IR(w1, w2) = \log_2 \frac{p(w1, w2)}{p(w1) * p(w2)}$$

Where, W1=Input word 1, B=Input word 2, and  $p(w_i) = \frac{hits(w_i)}{WebSize}$

**Word2Vec metrics:** Word2vec is a mining and natural language analysis method. This technique requires a neural network and computational framework to recognize terms from a broad corpus of texts. After training, this model can identify terms that are synonyms or propose other words for an incomplete expression. Although this name suggests, word2vec provides each distinguishable word with a specific number set named a vector. The vectors are designed to signify the degree of semantic similarity within the terms defined by this kind of vectors throughout the basic mathematical framework (cosine similarity) (Khatter & Ahlawat, 2020). Usually, the training of the Word2vec model classifies (GoogleCodeArchive, 2016) into a two-way approach. One of them is hierarchical softmax, which uses a Huffman tree to minimize the computation to determine the conditional log-likelihood. It is also more effective for uncommon words. Another negative sampling approach: This approach maximizes the problem by reducing the log-likelihood of sampled negative cases, which is more effective for frequent word and low dimensional vectors.

## 4.2 Knowledge-based measures

A variety of measures have been established to determine the similarity between words, texts, or short sentences, and the knowledge-based measure is one of them. It measures quantifying the degree to which two words or strings are linked semantically through handling data obtained from semantic webs (Mihalcea, Corley, & Strapparava, 2006; Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). We have analyzed one knowledge-based semantic network, namely: WordNet. The WordNet is a conventional lexicographic dictionary of semantically connections between words in advanced computing (Miller, 1995). WordNet relates words in semantic connections, including synonyms, hyponyms, and meronyms (Miller, 1995; WordNets in the World, 2010). It is a lexicographic web dataset for data management functions (Miller, 1995). The synonyms divide into concise explanations and instances of use.

Usually, words are organized into sets of synonyms, hyponyms, and meronyms for-  
mate, and each represents a lexicographic idea (Miller, 1995). The semantic relations  
in WordNet has bellowed.

**Table 1. Semantic relations in WordNet.**

| Semantic relation              | Examples                                |   |                                    |                    |
|--------------------------------|---|---|------------------------------------|--------------------|
|                                | Noun                                    | Verb  | Adjective                          | Adverb             |
| Synonymy (similar meaning)     | bed, couch, berth, the sack, kip        | go to bed, retire, call it a day, sleep           |                                    |                    |
| Hyponymy (subordinate meaning) | spoon, cutty, spatula, ladle            | spoon   |                                    |                    |
| Meronymy (part)                | face, mouth, mug, courage, surface, top | look out on, look toward, open out over, overlook |                                    |                    |
| Antonymy (Opposite)            | Good, welfare, benefit, blessing        |   | Excellent, strong, helpful, Honest | To advantage, well |

## 5 Syntactic similarity

The syntactic similarity measures the grammatical structure of words, short text, and sentences (Harispe, Ranwez, Janaqi, & Montmain, 2015). It defines how similar two data elements are without considering the type of the language or the meaning of the content. Measuring syntactic similarity between words, short text in big text data, statistical analysis, research process, statistical analysis, machine learning, language processing, and data extraction plays an important role (Kaur, 2015).

Syntactic search engines signify a valuable and essential approach to gather data from the web. It cannot interpret the user question interpretation, and therefore the output does not provide a particular structure. There are numerous researches, studies, and work on the technology of syntactic search engines. The extracted findings are always dependent on the keyword match concerning the Syntactic web. Several web page's query results do not even have significance or meaning against the keyword match (Sheela & Jayakumar, 2019). Besides semantic search engines, syntactic search engines also should consider the following criteria: user interface, productivity, efficiency, performance, quality, reliability, flexibility, time, method classification, usability, and economic efficiency (El-gayar, Mekky, & Atwan, 2015).

Different types of existing syntactic search engines are Google, Yahoo, and Ask (Sheela & Jayakumar, 2019). For example, two strings, "Best" and "Rest," are given. Now, the syntactic similarity between two strings is 75%. Both strings are syntactically similar because "Best" and "Rest" have three matching characters; "e," "s," and "t." On the other hand, semantically 0% similar because "Best" and "Rest" has not the same meaning. There are two main categories of syntactic similarity measures as character-level measures and token-level measures. Some measurement techniques combine these two techniques called soft measures (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

The syntactic similarity measurements are case-sensitive, and all the not relevant characters are not taken into account. This similarity measurement approach combines the Longest Common Substring (LCS) and 2-grm algorithms. A syntactic simi-

larity measures algorithm has given below (Kaur, 2015; Taib, Abbou, & Alam, 2008).

**Input:** Two string A and B

**Output:** String matching value between 0 to 1

```
Case is ignored
L1 = A .getAllPairs()           //List of substring L1
L2 = B .getAllPairs()           //List of substring L2
same = L1 ∩ L2                   //Intersection of intended metric
all = L1 ∪ L2                     //Disjoint union of intended metric
X = 2 * same.length/all.length() //2 – gram algorithm
Y = 2 * LCS(A, B)/(A.length() + B.length()) //LCS algorithm
return Max(X, Y)                 //Return maximum string matching value
```

Where, A, B=Two strings, L1, L2=Length of two strings, return 0= In an exact match, and return 1= Exact match.

## 5.1 Character-level measures

In character-level measures, the string can define as a series of characters (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019) that interprets as a sequence. Measures of character-level apply whether the strings are words, short texts, phrases, expressions, or short sentences. Character-level measures apply in words, short texts, phrases, expressions, or short sentences to deal with spelling mistakes, typography mistakes, and structural variations. There are three different types of character-level measures, namely: exact match, transformation measures, and longest common substring (LCS) measures.

**Exact match:** The matching feature is highly optimized and can be beneficial if you only need to see the databases that match your search (Kehrer, 2018). An exact matching keyword exhibits the same matching keyword according to search terms (Manage-ads, 2020). A logical binary outcome gives you an exact match: 1 = exactly the same string, 0 = no common string. This logical binary outcome is the standard way to analyze strings while collecting information (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

**Transformation measures:** It assesses two strings by calculating the number of processes required to transform one string to the next (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019) and obtain it in numerous ways. The following edit distance functions apply to edit operations based on a number, type, and cost. These are Levenshtein distance (Levenshtein, 1966), Damerau-Levenshtein distance (Damerau, 1964), Needleman-Wunsch distance (Needleman & Wunsch, 1970), Smith-Waterman distance (Smith & Waterman, 1981), Smith Waterman-Gotoh distance (Gotoh, 1982), Hamming distance (Hamming, 1950), Jaro distance (Jaro, 1989), and Jaro-Winkler distance (Winkler, 1990).

**Longest common substring (LCS) measures:** The description of this LCS (Friedman & Sideli, 1992) measures distance represents the fact that it determines the longest combination of characters between the two strings by following letter's order (Navarro G. , 2001). LCS obtains the most lengthened strings, which is a substring of two or more strings. It has been developed for applying to medical record matching in a hospital environment and text resuming. However, short text comparison may apply (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

### 5.1.1 Levenshtein distance

The Levenshtein distance is a metric that determines the difference or similarities of two strings or sequences (Apostolico & Galill, 1997). A minimum number of operations (edit distance) are needed to transform one string into another string (Gali, Mariescu-Istodor, & Fränti, 2016; Navarro G. , 2001). To measure the Levenshtein distance between string A and B, it applies the insertion, deletion, or substitution of a single character (Apostolico & Galill, 1997; Malakasiotis & Androutsopoulos, 2007). The equation below (Levenshtein, 1966) describes the principle of Levenshtein similarity measures.

$$Levenshtein(A,B) = 1 - \frac{edit(A,B)}{\max(|A|,|B|)}$$

Where, A=Input String 1, B=Input String 2, and edit (A, B)=Operational cost of two strings. For example, the Levenshtein distance between two strings A = "zokin", and

string B = "rocking" is 4. But the following three changes are needed for edit operation:

1. zokin → rokin (substitution of "z" by "k")
2. rokin → rockin (Insertion of "c")
3. rockin → roking (Insertion of "g" at the end)

### 5.1.2 Damerau-Levenshtein distance

Damerau–Levenshtein is a metric that determines the edit distance of two-character or sequences (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). It allows the swapping of two adjacent characters ( $XY \leftrightarrow YX$ ) at a minimum cost. To measure the Damerau-Levenshtein distance between sequences of characters, it applies the insertion, deletion, or substitution of a single character or transposition of two adjacent characters. The equation below (Damerau, 1964) describes the principle of Damerau-Levenshtein similarity measures.

$$Damerau - Levenshtein(A, B) = 1 - \frac{edit(A, B)}{\max(|A|, |B|)}$$

Where A=Input String 1, B=Input String 2, and edit (A, B)=Operational cost of two strings. For example, the Damerau-Levenshtein distance between two strings, "OP" and "POC" is 2. But the following three changes are needed for edit operation:

1. **OP** → **PO** (Transposition of "OP" to "PO")
2. **PO** → **POC** (Insertion of "C" at the end)

### 5.1.3 Needleman–Wunsch algorithm

In 1970, Needleman and Wunsch presented an algorithm to measure the optimum global protein or nucleotide or biological sequence standardization (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). Needleman-Wunsch applies the cost of the addition and deletion of two modules and one for replacement. Usually, this type of edit distance applies to calculate syntactical mistakes (as "pizza" shop and "pizzashop"), not abbreviated mistakes (as Tangail Cricket Academy and Tangail CA). The equa-

tion below (Needleman & Wunsch, 1970) describes the principle of Needleman-Wunsch similarity measures.

$$\text{Needleman - Wunsch } (A, B) = 1 - \frac{\text{edit}(A, B)}{2 \times \max(|A|, |B|)}$$

Where A=Input String 1, B=Input String 2, and edit (A, B)=Operational cost of two strings. As an example, globally aligned sequences of two strings, "ATGCT" and "AGCT," according to Needleman-Wunsch has shown in Figure 12. For this algorithm match=1, mismatch= -1, and gap = -2.

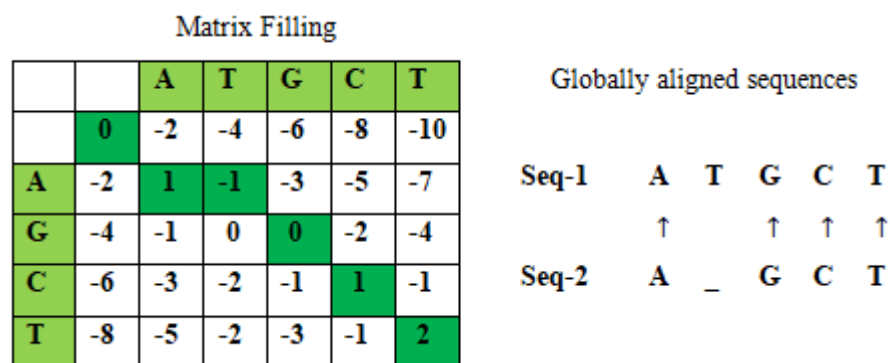


Figure 13. Needleman-Wunsch sequence matching.

#### 5.1.4 Smith–Waterman algorithm

In 1981 Smith-Waterman presented an algorithm to measure the optimum local alignment of sequences (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). It calculates a similar position for two sequences rather than the whole sequence. Smith-Waterman applies the least cost for mismatch at the beginning and middle of the sequences. The equation below (Smith & Waterman, 1981) describes the principle of Smith-Waterman similarity measures.

$$\text{Smith - Waterman } (A, B) = \frac{\text{edit}(A, B)}{\min(|A|, |B|)}$$

Where, A=Input String 1, B=Input String 2, edit (A, B)=Operational cost of two strings. As an example, locally aligned sequences of two strings, "ATGCT" and "AGCT," according to Smith-Waterman has shown in Figure 13. For this algorithm match=1, mismatch= -1, gap = -2, and negative value= 0.



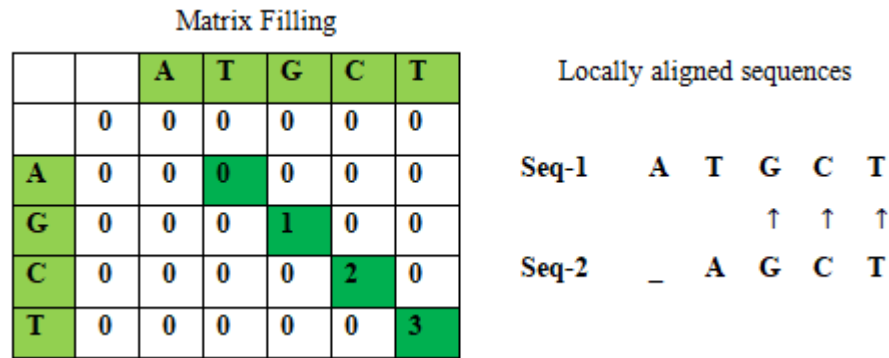


Figure 14. Smith-Waterman sequence matching.

### 5.1.5 Smith–Waterman–Gotoh algorithm

In 1982, Smith-Waterman-Gotoh proposed an algorithm (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019) to improve the optimal local sequence alignment. It permits the affinity of the distance to facilitate local sequence synchronization. It introduced an open gap and expanded gap costs for inclusion. It gives a high score to substitute identical characters then incompatible characters. The equation below (Gotoh, 1982) describes the principle of Smith-Waterman-Gotoh similarity measures.

$$Smith - Waterman - Gotoh (A, B) = \frac{edit(A, B)}{\min(|A|, |B|)}$$

Where, A=Input String 1, B=Input String 2, edit (A, B) =Operational cost of two strings. For example, globally aligned biological or DNA sequences of two strings, "TGTTACGG" and "GGTTGACTA," according to Smith-Waterman-Gotoh has shown in Figure 14. For this algorithm match=1, mismatch= -1, negative value=0, gap opening=5, gap extension=1.

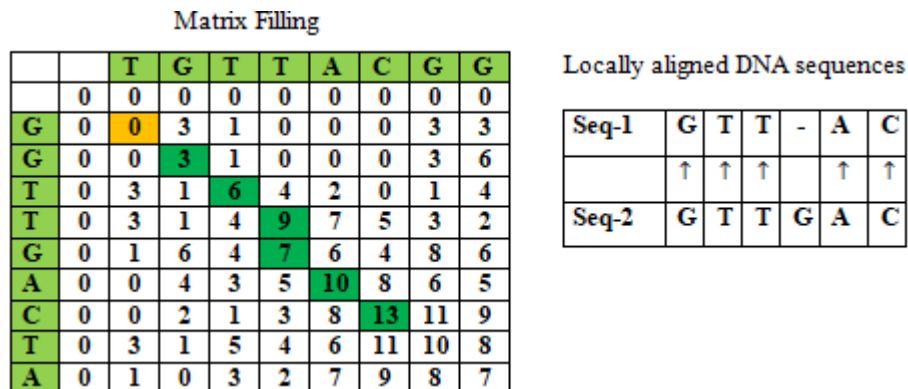


Figure 15. Smith-Waterman-Gotoh DNA sequence matching (en.wikipedia.org).

### 5.1.6 Hamming distance

The Hamming distance is a modification of edit distance since substitution is the most basic editing action, and the cost is one. It deals with strings of the same length (Boytssov, 2011; Navarro G. , 2001). The equation below (Hamming, 1950) describes the principle of Hamming distance similarity measures.

$$\text{Hamming distance } (A, B) = 1 - \frac{\text{edit}(A, B)}{\max(|A|, |B|)}$$

Where A=Input String 1, B=Input String 2, and edit (A, B)=Operational cost of two strings. For Example, the Hamming distance between two strings, "topic" and "tofel," is 3.

### 5.1.7 Jaro distance

The Jaro distance is a measure of similarity between two strings. It introduced incorrect text fields linked to data and determines the amount of transposed and matching characters (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). The higher the Jaro distance for two strings is, the more similar the strings are. The score is normalized such that 0 equates to no similarity and 1 is an exact match.

The equation below (Jaro, 1989) describes the principle of Jaro similarity measures.

$$\text{Jaro } (A, B) = \frac{1}{3} \times \left( \frac{m}{|A|} + \frac{m}{|B|} + \frac{m - x}{m} \right)$$

Where, A=Input string 1, B=Input string 2, m=Number of matching character, and x=Number of transposed character. For example, the Jaro distance between two strings, "topic" and "tofel" is 0.6. The calculation is:

$$\text{Jaro distance}(\text{Str1}, \text{Str2}) = \frac{1}{3} \times \left( \frac{2}{|5|} + \frac{2}{|5|} + \frac{2 - 0}{2} \right) = 0.6$$

The Input string 1=topic=5 characters, Input string 2=tofel=5 characters, m=Number of matching character=2, x=Number of transposed character=0.

### 5.1.8 Jaro–Winkler distance

The modification of Jaro distance by Winkler is familiar with the name of Jaro-Winkler distance (Malakasiotis & Androutsopoulos, 2007). It measures the edit distance between two strings and gives greater weight to match the prefix (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). The equation below (Winkler, 1990) describes the principle of Jaro-Winkler similarity measures.

$$\text{Jaro – Winkler } (A, B) = J(A, B) + (l \times p(1 - J(A, B)))$$

Where, A=Input string 1, B=Input string 2, l=Length of the common prefix between string, p= Scaling factor 0.1, J(A,B)=Jaro distance between two strings, and Prefix weight=(l×p(1-J(A,B))). For example, the Jaro-Winkler distance between two strings, "topic" and "tofel," is 0.68. The calculation is:

$$\text{Therefore, the Jaro distance}(\text{Str1}, \text{Str2}) = \frac{1}{3} \times \left( \frac{2}{|5|} + \frac{2}{|5|} + \frac{2 - 0}{2} \right) = 0.6$$

$$\text{So, Jaro – Winkler distance } (\text{Str1}, \text{Str2}) = 0.6 + (2 \times 0.1(1 - 0.6)) = 0.68$$

The Input string 1=topic=5 characters, Input string 2=tofel=5 characters, m=Number of matching character=2, x=Number of transposed character=0, l=Length of the common prefix between string=2, x=Number of transposed character=0, and p=Scaling factor 0.1

### 5.1.9 Longest common substring

The longest common substring (LCS) identifies the largest adjacent series of characters that is a substring of other strings. It was developed for uses such as patient records and text summaries in a medical environment and can also compare short text (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). It only recognizes insertions and deletions by cost 1. Moreover, the LCS distance is symmetrical unpaid characters (Navarro G. , 2001). The equation below (Friedman & Sideli, 1992) describes the principle of the Longest common substring.

$$LCS(A, B) = \frac{|sub(A, B)|}{\max(|A|, |B|)}$$

Where A=String 1, B=String 2, and sub=Substring. For example, the length of LCS of two strings, "ABCDGH" and "ACDGHRT," is 4 due to the common substring "CDGH." The algorithm of LCS has given below:

**Input:** Two string A and B, length n and m.

**Output:** Length of the LCS of A and B.

```

for  $i \leftarrow 0$  to  $n$ 
     $L[i, 0] \leftarrow 0$ 
end for
for  $i \leftarrow 0$  to  $m$ 
     $L[0, j] \leftarrow 0$ 
end for
for  $i \leftarrow 1$  to  $n$ 
    for  $j \leftarrow 1$  to  $m$ 
        if  $a_i = b_j$  then  $L[i, j] \leftarrow L[i - 1, j - 1] + 1$ 
        else  $L[i, j] \leftarrow \{L[i, j - 1], +L[i - 1, j]\}$ 
        end if
    end for
end for
return  $L[n, m]$ 

```

We have also performed some case studies to measure the syntactic similarity of character-level by applying a Java toolkit named "Stringsim". This toolkit can calculate the syntactic similarity of multiple strings using character and token level functions (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). To execute our case study, we have selected three pairs of strings from MOPSI databases. Those chosen strings are String 1a: *Koti pizza ravintola*, and String 1b: *ravintola*; String 2a: *National park*, and String 2b: *National library at night*; String 3a: *Swimming*, and String 3b: *Ice swimming experience*. The syntactic similarity measurement values of character-level have shown in Table 2.

Table 2. Syntactic similarity measures of character-level.

| Syntactic similarity | Unit similarity method | Similarity value-1 |           | Similarity value-2 |           | Similarity value-3 |           |
|----------------------|------------------------|--------------------|-----------|--------------------|-----------|--------------------|-----------|
|                      |                        | String 1a          | String 1b | String 2a          | String 2b | String 3a          | String 3b |
| Character-level      | Levenshtein            | 0.45               |           | 0.44               |           | 0.30               |           |
|                      | Damerau-Levenshtein    | 0.45               |           | 0.44               |           | 0.30               |           |
|                      | Needleman-Wunsch       | 0.73               |           | 0.52               |           | 0.50               |           |
|                      | Smith-Waterman         | 1.00               |           | 0.69               |           | 0.88               |           |
|                      | Smith-Waterman Gotoh   | 1.00               |           | 0.72               |           | 0.88               |           |
|                      | Hamming                | 0.00               |           | 0.00               |           | 0.00               |           |
|                      | Jaro                   | 0.00               |           | 0.73               |           | 0.73               |           |
|                      | Jaro-Winkler           | 0.00               |           | 0.89               |           | 0.73               |           |

**Case study-1:** Similarity value-1 has shown the different similarity value for String 1a: *Koti pizza ravintola*, and String 1b: *ravintola*.

- Levenshtein and Damerau-Levenshtein method applies high cost if one string is tiny from another string. The Java toolkit has given 45% string similarity results between two strings because the length of string 1a is 2.5 times higher than string 1b.
- For the Needleman-Wunsch method, the similarity results between two strings are 73% because the sequence of the string 1b exactly matches the ending sequence of the string 1a.
- Smith-Waterman and Smith Waterman-Gotoh methods apply the least cost for mismatch at the beginning and end of the string sequence. For both methods, the Java toolkit has given 100% similar results between two strings because the matching happened at the end of both strings.
- The Hamming method only works if both strings have the same length so it produces 0% similarity results between two strings due to their non-similar length.
- Jaro and Jaro-Winkler methods apply least cost for string similarity matching if both strings are identical or no farther than  $\lceil \frac{\max(|S1|, |S2|)}{2} - 1 \rceil$ . The Java toolkit has shown 0% similarity results between two strings because both strings are not identical and no farther than  $\lceil \frac{\max(|S1|, |S2|)}{2} - 1 \rceil$ .

**Case study-2:** Similarity value-2 has shown the different similarity value for String 2a: *National park*, and String 2b: *National library at night*.

- Levenshtein and Damerau-Levenshtein both method produces 44% string similarity results between two strings because the length of string 2a is 2 times higher than string 2b.
- For the Needleman-Wunsch method, it produces 52% similarity results between two strings because the starting sequence of the string 2a exactly matches the starting sequence of the string 2b.
- Smith-Waterman and Smith Waterman-Gotoh measures produce 69%, and 72% similarity between two strings because the matching happened at the beginning of both strings.
- The Hamming measure produces 0% similarity between two strings due to their non-similar length.
- Jaro and Jaro-Winkler measures produce 73% and 89% similarity results between two strings because both strings are semi- identical.

**Case study-3:** Similarity value-2 has shown the different similarity value for String 3a: *Swimming*, and String 3b: *Ice swimming experience*.

- Levenshtein and Damerau-Levenshtein measures produce 30% string similarity results between two strings because the length of string 3b is 3 times higher than string 3a.
- For the Needleman-Wunsch method, the Java toolkit has given 50% similarity results between two strings because the sequence of the string 3a exactly matches the middle sequence of the string 3b.
- Smith-Waterman and Smith Waterman-Gotoh measures produce 88% similar results between two strings because the matching happened at the beginning of string 3a and middle of the string 3b.
- The Hamming method produce 0% similarity between two strings due to their non-similar length.
- Jaro and Jaro-Winkler measures produce 73% similarity between two strings.

## 5.2 Token-level measures

One of the string segmentation methods is token level measurements (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019), and it deals with typographic patterns contributing to word reordering. It splits a string into tokens with whitespaces and punctuations, and the strings analyze as a set of tokens rather than only characters; these measures are known as token level measures. The token level measures apply to dataset maintenance, and it uses data at the token level to address swap and incomplete token problems. There are two ways to solve the token ordering (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019) problem. These are sorting heuristic and permuting heuristic.

**Sorting heuristic:** Each string is tokenized alphabetically arranged, reconnected, and added edit distance to the modified strings in heuristic sorting (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

**Permuting heuristic:** In permuting heuristic, all token transformations drive from the first string, and a distinction performs between all permuting strings and the second string and the largest similarity value select (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019).

Suppose two strings A and B can be interpreted as multiple sets of words (Cohen, Ravikumar, & Fienberg, 2003). Now, we have considered two different token-based distance metrics. These are Jaccard similarity measures and Token frequency-Inverse Document Frequency.

**Jaccard similarity measures:** The Jaccard similarity between the word sets A and B is simply (Galhardas, 2013; Cohen, Ravikumar, & Fienberg, 2003) define as:

$$S_{JACCARD} = \frac{|A \cap B|}{|A \cup B|}$$

Where A=String 1, and B=String 2.

**Token frequency-Inverse Document Frequency (TFIDF) or cosine similarity measures:** The information retrieval group commonly uses (Galhardas, 2013; Cohen, Ravikumar, & Fienberg, 2003), and it can be defined as:

$$S_{TFIDF \text{ or } COSINE} = \sum_{j=1}^{|D|} \frac{vA(j) * vB(j)}{|A| * |B|}$$

Where, A=String 1, B=String 2, TFW=Higher weights to token appearing in documents, IDFW=Lower weights to token appearing in the whole set of documents= $|D|/nw$ , D= Database that contain the word w, nw=Number of records , w=Word, and vStri(w)=String separation into words and assign a weight to each word  $w = \log_{[f_0]}(TFw+1) * \log_{[f_0]}(IDFW)$ . For example, the tokenization of the string "Janny is a student of UEF" will be "Janny, is, a, student, of, and UEF."

### 5.3 Soft measures

Combine character-level and token level measures are referred to as soft measures. The theory of a soft measure is to implement a character-level measure to all sets of tokens between strings and to recognize those tokens that fulfill a certain requirement. The soft measure exhibited higher similarity values than anticipated due to its capability to recognize the similarity between related and identical tokens (Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). There are five different types of soft measures, namely: Simpson similarity (Choi, Cha, & Tappert, 2010), Jaccard similarity (Rezaei & Franti, 2016), Soft-cosine similarity (Cohen, Ravikumar, & Fienberg, 2003), Euclidean distance (Malakasiotis & Androustopoulos, 2007), and Manhattan distance (Malakasiotis & Androustopoulos, 2007).

In soft measures, the binary feature vector presents a significant role in interpreting the pattern and measuring the similarity and distance of many problems such as clustering, classification, etc. The binary similarity and distance measure apply for data analysis and analysis of the clustering method. We have shown an operational taxonomic unit of binary measure (Choi, Cha, & Tappert, 2010) bellow:



**Table 3. Operational Taxonomic Units Expression of Binary Instance  $i$  and  $j$  (Choi, Cha, & Tappert, 2010).**

| $(ij)$             | <b>1 (Present)</b>            | <b>0 (Absence)</b>                  | <b>Sum</b>          |
|--------------------|-------------------------------|-------------------------------------|---------------------|
| <b>1 (Present)</b> | $a = i \cdot j = (1,1)$       | $b = \bar{i} \cdot j = (0,1)$       | $a + b$             |
| <b>0 (Absence)</b> | $c = i \cdot \bar{j} = (1,0)$ | $d = \bar{i} \cdot \bar{j} = (0,0)$ | $c + d$             |
| <b>Sum</b>         | $a + c$                       | $b + d$                             | $n = a + b + c + d$ |

Where,

$i, j$ =Objects in vector form

$n$ =Number of features (attributes)or dimension of the feature vector

$a$ =Number of attributes  $((i, j) = (1, 1) = \text{presence}) = \text{Positive match}$

$b$ =Number of attributes  $((i, j) = (0, 1) = i \text{ absence Mitch match})$

$c$ =Number of attributes  $((i, j) = (1, 0) = j \text{ absence Mitch match})$

$d$ =Number of attributes  $((i, j) = (0, 0) = \text{absence}) = \text{negative Mitch match}$

$(a+d)$ ,  $(a+b)$ , and  $(a+c)$  =Total number of matches between  $(i, j)$

$(b+c)$ =Total number of mismatches between  $(i, j)$

### 5.3.1 Simpson similarity

Simpson's similarity method is used to measure the similarity of a set of community samples and to measure if their species distribution is identical or distinct (Choi, Cha, & Tappert, 2010). The equation below describes the principle of Simpson similarity measures.

$$S_{SIMPSON} = \frac{a}{\min(a + b, a + c)}$$

Where,  $S$ =Similarity measures,  $a=(1,1)$ = Positive match,  $b= (0,1)$ = $i$  absence Mitch match,  $c= (1,0)$ = $j$  absence Mitch match,  $(a+b)$ ,and  $(a+c)$ =Total number of matches between  $(i,j)$ . For example, the Simpson similarity between two strings "acdfg", and "bcefg" is 0.6. The operation and calculation of Simpson similarity has given below:

Now, the distance matrix will be

|          | <b>a</b> | <b>b</b> | <b>c</b> | <b>d</b> | <b>e</b> | <b>f</b> | <b>g</b> |
|----------|----------|----------|----------|----------|----------|----------|----------|
| String 1 | 1        | 0        | 1        | 1        | 0        | 1        | 1        |
| String 2 | 0        | 1        | 1        | 0        | 1        | 1        | 1        |

|          | <b>1</b>    | <b>0</b>    |
|----------|-------------|-------------|
| <b>1</b> | $a(1,1)= 3$ | $b(0,1)=2$  |
| <b>0</b> | $c(1,0)=2$  | $d( 0,0)=0$ |

$$S_{SIMPSON}(Str\ 1, Str\ 2) = \frac{a}{\min(a+b, a+c)} = \frac{3}{\min(3+2, 3+2)} = \frac{3}{\min(5, 5)}$$

$$= \frac{3}{\frac{(5+5-|5-5|)}{2}} = \frac{3}{5} = 0.6$$

### 5.3.2 Jaccard similarity

The Jaccard similarity measure applies for the segmentation of indigenous items, and Forbes suggested a factor for the segmentation of indigenous related items. It can be explained as the size of the intersection divided by the size of the union of two sets (Choi, Cha, & Tappert, 2010). The equation below describes the principle of Jaccard similarity measures.

$$S_{JACCARD} = \frac{a}{a+b+c}$$

Where, S=Similarity measures, a= (1,1)=presence)=Positive match, b=(0,1)=i absence Mitch match, c=(1,0)=j absence Mitch match, and (a+b+c)=Total number of matches between (i,j). For example, the Jaccard similarity between two strings "acdfg", and "bcefg" is 0.43. The operation and calculation of Jaccard similarity has given below:

Now, the distance matrix will be

|          | a | b | c | d | e | f | g |
|----------|---|---|---|---|---|---|---|
| String 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| String 2 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |

|   | 1         | 0         |
|---|-----------|-----------|
| 1 | a(1,1)= 3 | b(0,1)=2  |
| 0 | c(1,0)=2  | d( 0,0)=0 |

$$S_{JACCARD}(Str\ 1, Str\ 2) = \frac{a}{a+b+c} = \frac{3}{3+2+2} = \frac{3}{7} = 0.43$$

### 5.3.3 Soft-Cosine similarity

A soft-cosine or soft-similarity between two vectors takes into account correlations between characteristics combinations. The soft cosine meets the measurements of character as well as the token level to align the name. Soft cosine incorporates the cosine for matching the token with sentences for measuring the character grades

(Gali, Mariescu-Istodor, Hostettler, & Fränti, 2019). The equation below (Choi, Cha, & Tappert, 2010) describes the principle of soft-cosine similarity measures.

$$S_{COSINE} = \frac{a}{\sqrt{(a+b)(a+c)^2}}$$

Where, S=Similarity measures, a=(1,1)=presence)=Positive match, b=(0,1)=i absence Mitch match, c=(1,0)=j absence Mitch match, and  $\sqrt{(a+b)(a+c)^2}$  =Square root multiply between two matches (i, j). For example, the soft-cosine similarity between two strings "acdfg" and "bcefg" is 0.12. The operation and calculation of soft-cosine similarity has given below:

Now, the distance matrix will be

|          | <b>a</b> | <b>b</b> | <b>c</b> | <b>d</b> | <b>e</b> | <b>f</b> | <b>g</b> |
|----------|----------|----------|----------|----------|----------|----------|----------|
| String 1 | 1        | 0        | 1        | 1        | 0        | 1        | 1        |
| String 2 | 0        | 1        | 1        | 0        | 1        | 1        | 1        |

|          | <b>1</b>  | <b>0</b>  |
|----------|-----------|-----------|
| <b>1</b> | a(1,1)= 3 | b(0,1)=2  |
| <b>0</b> | c(1,0)=2  | d( 0,0)=0 |

$$S_{COSINE}(Str\ 1, Str\ 2) = \frac{a}{\sqrt{(a+b)(a+c)^2}} = \frac{3}{\sqrt{(3+2)(3+2)^2}} = \frac{3}{25} = 0.12$$

### 5.3.4 Euclidean distance

Euclidean distance is the square root of the sum of the respective elements of two vectors. The equation above (Choi, Cha, & Tappert, 2010) describes the principle of Euclidean distance measures.

$$D_{EUCLIDEAN} = \sqrt{b+c}$$

Where, D=Distance measures, b=(0,1)=i absence Mitch match, c=(1,0)=j absence mitch match, and  $\sqrt{b+c}$  =Root summation of two attributes (i,j). Now the Euclidean distance between two strings "acdfg" and "bcefg" is 2. The operation and calculation of Euclidean distance have given below:

Now, the distance matrix will be

|          | <b>a</b> | <b>b</b> | <b>c</b> | <b>d</b> | <b>e</b> | <b>f</b> | <b>g</b> |
|----------|----------|----------|----------|----------|----------|----------|----------|
| String 1 | 1        | 0        | 1        | 1        | 0        | 1        | 1        |
| String 2 | 0        | 1        | 1        | 0        | 1        | 1        | 1        |

|          | <b>1</b>  | <b>0</b>  |
|----------|-----------|-----------|
| <b>1</b> | a(1,1)= 3 | b(0,1)=2  |
| <b>0</b> | c(1,0)=2  | d( 0,0)=0 |

$$S_{EUCLIDEAN}(Str\ 1, Str\ 2) = \sqrt{b + c} = \sqrt{2 + 2} = 2$$

### 5.3.5 Manhattan distance

The Block Distance between two elements is the sum of their respective component's differences is called manhattan distance (Choi, Cha, & Tappert, 2010; Boytsov, 2011). The equation below describes the principle of manhattan distance measures.

$$D_{MANHATTAN} = b + c$$

Where, D=Distance measures, b=(0,1)=i absence Mitch match, c=(1,0)=j absence mitch match, and (b+c)=Summation of two attributes (i,j). Now the Manhattan distance between two strings "acdfg" and "bcefg" is 2. The operation and calculation of Manhattan distance have given below:

Now, the distance matrix will be

|          | <b>a</b> | <b>b</b> | <b>c</b> | <b>d</b> | <b>e</b> | <b>f</b> | <b>g</b> |
|----------|----------|----------|----------|----------|----------|----------|----------|
| String 1 | 1        | 0        | 1        | 1        | 0        | 1        | 1        |
| String 2 | 0        | 1        | 1        | 0        | 1        | 1        | 1        |

|          | <b>1</b>  | <b>0</b>  |
|----------|-----------|-----------|
| <b>1</b> | a(1,1)= 3 | b(0,1)=2  |
| <b>0</b> | c(1,0)=2  | d( 0,0)=0 |

$$S_{MANHATTAN}(Str\ 1, Str\ 2) = b + c = 2 + 2 = 4$$

The Block Distance between two elements is the sum of their respective component's differences is called manhattan distance (Choi, Cha, & Tappert, 2010; Boytsov, 2011). The equation below describes the principle of manhattan distance measures.

$$D_{MANHATTAN} = b + c$$

Where, D=Distance measures, b=(0,1)=i absence Mitch match, c=(1,0)=j absence mitch match, and (b+c)=Summation of two attributes (i,j). Now the Manhattan distance between two strings "acdfg" and "bcefg" is 2. The operation and calculation of Manhattan distance have given below:

Table 4. Syntactic similarity of soft measures.

| Syntactic similarity | Group matching method + Levenshtein method | Similarity value-1 |           | Similarity value-2 |           | Similarity value-3 |           |
|----------------------|--|--------------------|-----------|--------------------|-----------|--------------------|-----------|
|                      |  | String 1a          | String 1b | String 2a          | String 2b | String 3a          | String 3b |
| Soft measures        | Simpson                                    | 0.67               |           | 0.68               |           | 0.72               |           |
|                      | Jaccard                                    | 0.50               |           | 0.51               |           | 0.46               |           |
|                      | Soft-cosine                                | 0.60               |           | 0.57               |           | 0.46               |           |
|                      | Euclidean                                  | 0.55               |           | 0.29               |           | 0.61               |           |
|                      | Manhattan                                  | 0.55               |           | 0.00               |           | 0.25               |           |

**Case study-1:** The similarity value-1 has shown the different similarity value for String 1a: *Morning exercise*, and String 1b: *Exercise place*.

- Simpson's method applies to calculate the similarity between a pair of community samples and to measure if their species distribution is identical or distinct. The Java toolkit has given 67% string similarity results between two strings because both strings are identical.
- Jaccard similarity measure method applies to measure the similarity and diversity of strings. It produces 50% string similarity between two strings because the second segment of string 1a is similar to the first segment of string 1b.
- Soft-cosine similarity measure applies for calculating the features and characteristics combinations of pair strings. It produces 60% similarity between two strings because both strings have the same features.
- The Euclidean distance similarity measure applies to measuring the length of a line segment between strings so it has 55% similarity results between two strings because the line segment of both strings is 55% similar.
- Manhattan distance similarity is applied to measure the block distance between two strings and it produces 55% similarity between two strings because the block distance between two strings is 55%.

**Case study-2:** The similarity value-2 has shown the different similarity values for String 2a: *Morning walk*, and String 2b: *Walking street*.

- Simpson's method produces 68% string similarity between two strings because both strings are identical.

- Jaccard similarity measure has given 51% string similarity between two strings because the second segment of string 2a is almost similar to the first segment of string 2b.
- Soft-cosine similarity measure produces 57% similarity between two strings because both strings have the same features.
- The Euclidean distance similarity measure produces 29% similarity because the lengths of two string's line segments are less similar.
- Manhattan distance produces 0% similarity between these strings because there is no block distance between two strings.

**Case study-3:** The similarity value-3 has shown the different similarity value for String 3a: *Railway station statue*, and String 3b: *Tram and railway bridge*.

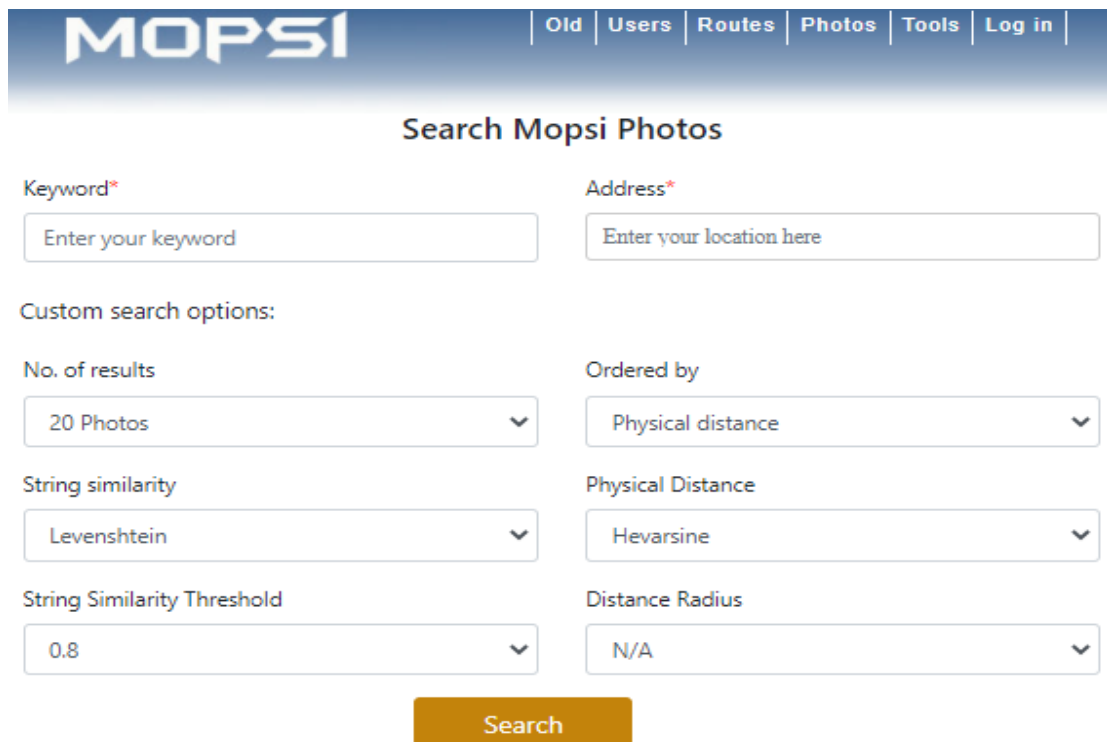
- Simpson's method produces 72% string similarity results between two strings because both strings have identical pairs.
- Jaccard similarity measure has given 46% string similarity between these strings because the starting segment of string 3a is similar to the second segment of string 3b.
- Soft-cosine similarity measure produces 46% similarity between these two strings because both strings have small features matching.
- The Euclidean distance similarity measure produces 61% similarity results as the two-line segment string's length is two-thirds similar.
- Manhattan distance has 25% similarity between two strings because the block distance between them is very small.

## 6 Implementation

This section will present the function of inexact matching of the Mopsi web search tool<sup>2</sup>. Section 6.1 will explain different parameters of tools for searching geo-tagged photos, physical distance measurement, and syntactic similarity measurement. Section 6.2 will describe the several technologies that we have used for backend and frontend development.

### 6.1 Tool description

We have applied multiple parameters to develop the inexact web searching tool to customize the user search query. We will introduce the following parameters in detail: numbering of results, ordering, strings similarity measurement methods, string similarity threshold, and distance radius. A user interface of the Mopsi web searching tool has shown in Figure 16.



The screenshot shows the Mopsi web search tool interface. At the top, there is a navigation bar with the Mopsi logo and links for Old, Users, Routes, Photos, Tools, and Log in. Below the navigation bar, the main heading is "Search Mopsi Photos". The interface contains several input fields and dropdown menus for custom search options. The "Keyword\*" field has a placeholder "Enter your keyword". The "Address\*" field has a placeholder "Enter your location here". Under "Custom search options:", there are four dropdown menus: "No. of results" (set to "20 Photos"), "Ordered by" (set to "Physical distance"), "String similarity" (set to "Levenshtein"), and "Physical Distance" (set to "Hevarsine"). Below these are two more dropdown menus: "String Similarity Threshold" (set to "0.8") and "Distance Radius" (set to "N/A"). A prominent orange "Search" button is located at the bottom center of the form.

---

<sup>2</sup> [http://cs.uef.fi/mopsi\\_dev/tools/inexact\\_search.php](http://cs.uef.fi/mopsi_dev/tools/inexact_search.php)

Total photos for **pizza**: 20



Figure 16. The user interface of the web searching tool.

In this searching tool, the first input textbox is a keyword where users can put their search string to find their query. Besides the keyword input, there is another input field named address, where users place their location to get nearby searching outcome. An example of search string and a location address have shown in Figure 17.

Keyword\*

Address\*

- 📍 Joensuu Finland
- 📍 Joensuun tori Joensuu, Finland
- 📍 Joensuu Sipoo, Finland
- 📍 Joensuun lentoasema (JOE) Lentoasemantie, Joensuu, Finland
- 📍 Joensuu Keskusta Kauppakatu, Joensuu, Finland

Figure 17. User choosing a keyword and location address.



The user can choose the number of search outcomes for geo-tagged photos in the custom search option, where the number of geo-tagged images can be 20, 30, 50, or all matching search results. In this way, the user can filter the search results to avail the searching outcome based on his requirement. Figure 18.a. has shown the options for selecting the number of searching outcomes for geo-tagged photos.

| No. of results | Distance Radius | String similarity    |
|----------------|-----------------|----------------------|
| 20 Photos      | N/A             | Levenshtein          |
| 30 Photos      | 5km             | Damerau-Levenshtein  |
| 50 Photos      | 10km            | Needleman-Wunsch     |
| All Photos     | 20km            | Smith-Waterman       |
|                | 50km            | Smith-Waterman-Gotoh |
|                | 100km           | Hamming              |
|                | 500km           | Jaro                 |
|                |                 | Jaro-Winkler         |
|                |                 | Inclusion            |

Figure 18. Advanced search option of the tool.

Ordering means a ranking. Figure 18.b. presents two options to select the search outcome based on physical distance and string similarity. If users prefer physical distance, then the Mopsi search tool will show comparative search results according to the location of all matching Mopsi data. On the other hand, if the users choose string similarity, it will show a relative search outcome according to string matching with all Mopsi data.

Users can choose the searching outcome based on different string similarity measures. They have a few options to select the similarity measurement method. Figure 18.d. represents a few similarity measures as Levenshtein, Damerau-Levenshtein, Smith-Waterman, Smith-Waterman Gotoh, Jaro, Jaro Winkler and Inclusion. We have discussed multiple similarity measurement methods in section 2 and 3 and choose these measures according to their comparative performance with other methods to develop our tool.

When the users choose the searching outcome based on all Mopsi data's physical distance, Haversine distance measurement method is used to calculate this physical

distance between the user's location and the data from the database to set the search outcome.

The Haversine method is a distance determination method that determines the least distance between two positions on a sphere using their latitude and longitude measurements on the surface (Kettle, 2017). It is mainly used in GPS applications development and navigation (Prakhar, 2018). The Haversine can express in trigonometric function as:

$$Haversine(\theta) = \text{Sin}^2\left(\frac{\theta}{2}\right)$$

The Haversine function can also be express into latitude and longitude coordinates.

$$a = \text{Sin}^2\left(\varphi_B - \frac{\varphi_A}{2}\right) + \cos\varphi_A * \cos\varphi_B * \text{Sin}^2\left(\lambda_B - \frac{\lambda_A}{2}\right)$$

$$c = 2 * \text{atan2}\left(\sqrt{a}, \sqrt{1 - a}\right)$$

$$d = R * c$$



The geographic distance on earth (photo. iGISMAP)

Where,  $\varphi$ =Latitude,  $\lambda$ =Longitude,  $R$ =Earth radius = 6,371km.

For string similarity measurement, users can apply threshold values to filter out the search outcome of the Mopsi tool corresponds with the search keyword. The different string similarity threshold values are as follows 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9. For example, if the user sets the threshold value 0.7, it will show a minimum of 70% to a maximum of 100% string matching search results compared with the search keyword.

Users can set the distance radius in kilometers to filter out the Mopsi's search outcome. Distance radius does the similar operation as string similarity threshold does. It shows the controlled results within the selected circular area. Figure 18.c has shown multiple distance radiuses as N/A (Not Applicable), 5km, 10km, and 20km. If the users set N/A, then it will show the search output at any geographic location on earth, either the area nearby them or not. On the other hand, if the users set the dis-

tance radius as 5km, 10km, and 20km, Mopsi will show search output within the confined radius.

## 6.2 Technology

We have used various technologies for developing the Mopsi photo searching tool in backend and front-end including Google Map API, Autocomplete address API, Geolocation, Geocoding, Reverse geocoding, Ajax, JSON, PHP, HTML, CSS, and JavaScript.

The Ajax is Asynchronous JavaScript and XML (Xie, 2019). Ajax is employed within the server in the back-end to build a swift and interactive web page via transmitting data (Morris, 2020; Xie, 2019). When an update is needed for a conventional web page, then the entire web page is reloaded (Morris, 2020). On the other hand, you may update a section of the web page asynchronously for Ajax without updating the whole web page (Xie, 2019). We applied Ajax due to its efficiency, accuracy of the process, and little data processing time, mainly to integrate our tool to Mopsi website. An Ajax model of a MOPSI web searching tool has shown in Figure 18. When the Mopsi web searching tool produces an event, the Ajax engine generates an HTTP request and sends it to the server-based system. The server-based system process the data of HTTP request and return it to the Ajax engine. The Mopsi web searching tool processes the return data through JavaScript and updates the web page section. Besides that, the return data also enclose in JavaScript Object Notation (JSON) format. JSON is a lightweight data storage and transportation format that is applied while data is needed to transmit to a web page from a server (Shin, 2018). It is also self-descriptive and easy for people to understand. In the end, we have used Hypertext Preprocessor (PHP) as a scripting language for the development of the MOPSI web searching tool that can set into HTML (Prokofyeva & Boltunova, 2017).

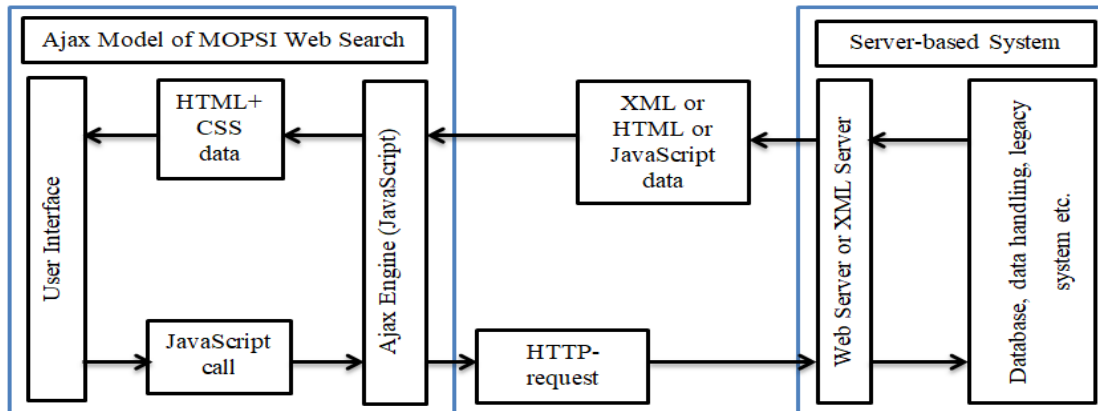


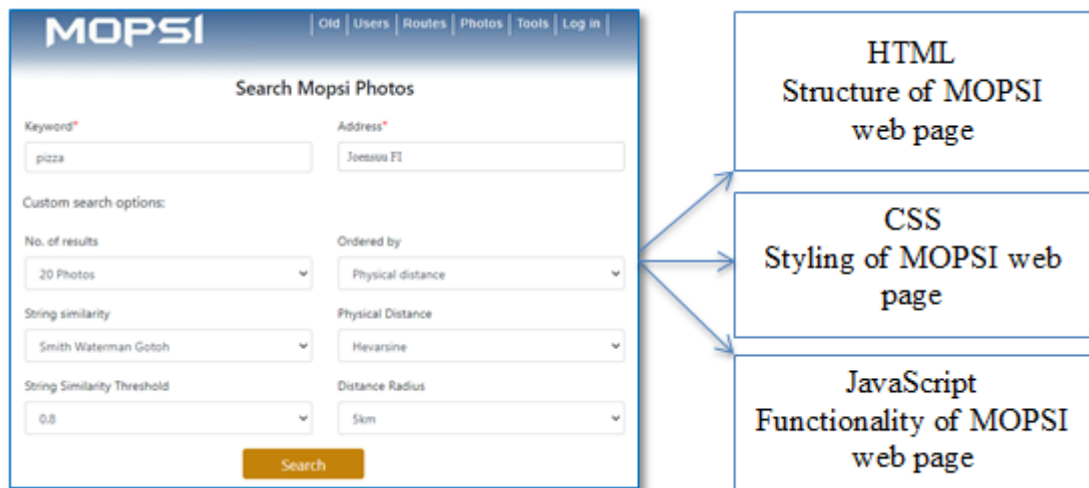
Figure 19. An Ajax model of a Mopsi web searching tool (Ajax-vergleich-en.svg).

Google Maps is a Google-designed web-mapping tool (Xie, 2019). It delivers comprehensive information on geographical areas and locations all over the earth on a Web-based system. It also provides aerial and satellite images of different places in addition to standard route maps. In certain areas, it also offers street views with vehicle images (Dodsworth & Nicholson, 2012). It provides a different kind of APIs for various purposes. Google Map APIs services have used to implant Google Maps into MOPSI web pages, recover data, adjusting marker, line, and trajectory plotting (Xie, 2019; GoogleMapsPlatform, 2020).

In Google Maps JavaScript API (Application Programming Interface), we have used an Autocomplete address (GoogleMapsPlatform, 2020) for positioning the database. It has given the type-ahead-search function to the Google Map for the MOPSI search field. It also provides the substrings, position titles, directions, location address, and code facilities to Mopsi. To develop the Mopsi web searching tool, we have tested geo-location to estimate an object's physical distance and track individuals based on latitude and longitude coordinates. Where the Geolocation API employs to transfer data to the server system and return a response from the server system to the client by sending an HTTP request. We have also used the Geocoding API to transform a street address into geographic coordinates (such as latitude and longitude) in which the marker can set on the map. It also turns the geographical coordinates into a readable address for humans.

We have used Hypertext Markup Language (HTML) to develop and produce web content for document design and show in the MOPSI web browser (Ferguson, 2020). To styling the Mopsi web page, we have employed Cascading Style Sheets (CSS)

that guides HTML. Besides HTML and CSS, we have also applied the JavaScript (JS) language that interacts with the functional work of the Mopsi tool (Kononenko, 2018). Together they build everything that is displayed for Mopsi visually while a person visits the webpage. A relational image of HTML, CSS, and JS has shown in Figure 17.



**Figure 20. Relational image of HTML, CSS, and JS.**

## 7 Experiment

This section will present the analysis and observation of the Mopsi data set and experimental details. Section 7.1 will explain the Mopsi data collection, data set descriptions, and data analysis procedure. Section 7.2 will explain some case studies, experiments, and observations briefly.

### 7.1 Datasets

The Mopsi data set mainly contains two types of data: geotagged photos and trajectories (Xie, 2019). The geotagged photos carry location information and recorded time; trajectories have a fixed interval sequence of GPS coordinates. The Mopsi data set has approximately 65694 geotagged photos and around 2400 users since December 2020. A data set summary has been given in Table 5 (Mariescu-Istodor, 2017).

Table 5. Summary of data set.

| Data set     | Size  | Type              | Language           | Length of string |    |     |                  |    |     |
|--------------|-------|-------------------|--------------------|------------------|----|-----|------------------|----|-----|
|              |       |                   |                    | Token length     |    |     | Character length |    |     |
|              |       |                   |                    | Min              | Av | Max | Min              | Av | Max |
| Mopsi photos | 65694 | Photo description | English<br>Finnish | 1                | 3  | 26  | 3                | 17 | 65  |

The large number (as 29612) of the Mopsi data does not contain any descriptions (as title) which are not usable for our experiment. As we are comparing the similarity between keywords and descriptions so the usable data must contain descriptions. Moreover, the Mopsi data carries some artificial data for some specific users for their experimental purposes. To clean the dataset, we have filtered these non-usable and artificial data from the database after data preprocessing and the size of usable data reduced to near 35000 photos.

Dataset contains mostly brief English or Finnish descriptions. A label for a time, and the photos' physical location has given for each Mopsi photo. After taking photos, the Mopsi users can write a description instantly. Then the Mopsi app provides the user with pre-written explanations for using. The pre-written explanations are gener-

ated from photos around to the user. As a result, photos taken in the same position seem to have similar descriptions with the same feature they address. In Mopsi data, typing mistakes in the descriptions are expected. The maximum number of geotagged photos and trajectories are collected from mainly Joensuu, Finland. The properties of the Mopsi data set have shown in Table 6.

**Table 6. Mopsi data properties.**

| <b>Column</b> | <b>Type</b> | <b>Description</b>          | <b>Example</b>     |
|---------------|-------------|-----------------------------|--------------------|
| Description   | varchar     | Title of the photo          | Skiing chess       |
| Street Name   | varchar     | Street name of that point   | Niskakatu          |
| Street Number | Int.        | Street number of that point | 1                  |
| Commune       | varchar     | Commune of that point       | Joensuu            |
| Date          | date        | Date for the point          | 2010-10-12         |
| User ID       | varchar     | User for that point         | Radu               |
| Phone         | varchar     | Users' Device               | Nokia N95          |
| Latitude      | Double      | Latitude value of point     | 62.92              |
| Longitude     | Double      | Longitude value of point    | 23.18              |
| Timestamp     | String      | Timestamp for the point     | 1559983789b second |

## 7.2 Experimental setup

This section contains different experiments for performing case studies based on our tool's<sup>3</sup> structure. We use the following experimental setup to analyze all the measure's performance compared to the inclusion of Mopsi concerning the different parameters such as different threshold values for similarity measures and physical distance. We aimed to find

1. Optimal threshold for each string similarity measure.
2. Flexibility of string similarity measures under the same similarity threshold.
3. Quality of the string similarity measures.
4. Correlation to physical distance and similarity measure.

---

<sup>3</sup> [http://cs.uef.fi/mopsi\\_dev/tools/inexact\\_search.php](http://cs.uef.fi/mopsi_dev/tools/inexact_search.php)

To perform these experiments, we have chosen a few test sets. For each test set, we have chosen a different keyword from the most popular keywords<sup>4</sup> from the Mopsi dataset. Some parameters are fixed for these experiments. These fixed parameters are address (user location), the number of results, ordered by (string similarity or physical distance), physical distance method, and distance radius. These parameters are mainly used to limit the search results and to customize the ordering based on user preference. We have set the number of results as "all photos," and the distance radius is not limited to obtain the maximum possible results for an entry and to make the observations based on the whole dataset. Table 7 shows an example of an experimental test set structure.

**Table 7. Structure of test set -1.**

| <b>Parameters</b>       | <b>Value</b>      |
|-------------------------|-------------------|
| Keyword                 | Chess             |
| Address (user location) | Joensuu, Finland  |
| Number of results       | All photos        |
| Ordered by              | String similarity |
| Physical distance       | Haversine         |
| Distance radius         | None              |

Other test sets only differ the keyword, for example, test-set 2 - "Lenkkireitin maisemia", test-set 3 - "Lenkkireitti", test-set 4 - "pizza" and so on. All string characters in the keywords and in the descriptions were converted to lowercase as a pre-processing step in all tests to avoid case sensitivity, which means "skiing" and "SKI-ING" will produce the same results. We also suppressed the spaces in the beginning and at the end of the strings if there were any in the pre-processing step for testing.

---

<sup>4</sup> [http://cs.uef.fi/mopsi\\_dev/tools/popularkeywords.php](http://cs.uef.fi/mopsi_dev/tools/popularkeywords.php)



For every test set, we have done some case studies based on some character level, string similarity measures as Levenshtein, Damerau-Levenshtein, Smith-Waterman, Smith-Waterman-Gotoh, Jaro, and Jaro-Winkler.

Prior to conducting our experiment, we have selected some known data from Mopsi to create a subset for a selected test set-1 where the keyword is "chess." We have chosen this keyword because all of the event dates concerning chess are known, and we gathered all the entries from those dates and created a new dataset<sup>5</sup> of 1050 data. For visualization, the live data can be found in the dataset<sup>6</sup> webpage. The structure of this dataset is similar to the Mopsi dataset. Additionally, it has a column named "Label," set to as binary type 0 or 1 to represent the label status. Here, the label is 1, whether the data is true or relevant for the keyword "chess," and 0 if the data is false or not relevant. Table 8 shows a few examples of defining the label for each data based on their descriptions.

**Table 8. Example of labeling in chess dataset.**

| <b>Description</b>             | <b>Status</b> | <b>Label</b> |
|--------------------------------|---------------|--------------|
| Chess table                    | True          | 1            |
| Cristina                       | False         | 0            |
| Swim chess tourney in progress | True          | 1            |
| Night in Iasi                  | False         | 0            |

We have collected these dates from various sources. Initially, dates are collected from the chess event webpage<sup>7</sup>. Some of the events have been organized before developing the Mopsi, so the Mopsi dataset does not contain any data from those dates. After collecting the event dates, we searched into Mopsi using some keywords related to chess. As the Mopsi data is mostly in English and Finnish language, we used the keyword "chess" for the English and "shakki," for the Finnish language. Our keywords while searching has collected the unique dates from all the retrieved data. For example, Figure 21 shows some unique dates 2018-04-24 and 2014-03-14 from

---

<sup>5</sup> [http://cs.uef.fi/mopsi\\_dev/chess/](http://cs.uef.fi/mopsi_dev/chess/)

<sup>6</sup> [http://cs.uef.fi/mopsi\\_dev/chess/chessdata.php](http://cs.uef.fi/mopsi_dev/chess/chessdata.php)

<sup>7</sup> <http://cs.uef.fi/chess/>

the Mopsi dataset, which are not included in the event list and we have added these dates in our newly created dataset.

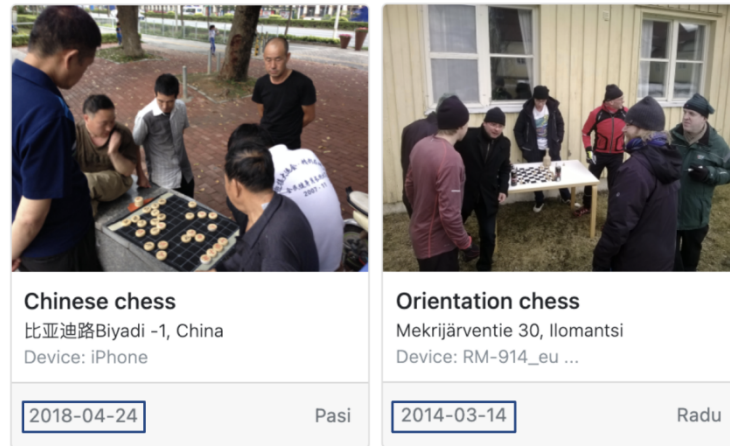


Figure 21. Data collection by selecting unique dates.

A full list of these dates is described in Table 9.

Table 9. Unique dates for chess data.

| Event                                 |            | Non- event |            |
|---------------------------------------|------------|------------|------------|
| Name                                  | Date       | User       | Date       |
| Football Chess Tournament             | 2019-09-19 | Radu       | 2014-03-14 |
| Football Chess Tournament             | 2012-06-19 | Pasi       | 2014-10-12 |
| Pulkashakki                           | 2018-02-13 | Pasi       | 2017-06-28 |
| Skiing Chess                          | 2016-02-21 | Pasi       | 2011-04-09 |
| Skiing Chess Mekrijärvi Championships | 2013-03-05 | Pasi       | 2017-07-02 |
| Skiing Chess Mekrijärvi Championships | 2012-03-20 | Pasi       | 2014-07-11 |
| Uintoshakki-turnaus                   | 2011-04-15 | Pasi       | 2016-08-16 |
| Uintoshakki-turnaus                   | 2010-10-29 | Pasi       | 2017-04-22 |
| 1st Running Chess Tournament          | 2003-08-12 | Pasi       | 2017-08-07 |
| Basketball Chess                      | 2020-11-14 | Pasi       | 2017-08-03 |
| Frisbee Chess                         | 2017-11-11 | Pasi       | 2017-08-04 |
| Ice Swimming Chess                    | 2015-11-09 | Pasi       | 2018-04-24 |
| Orienteering Chess                    | 2014-03-13 | Pasi       | 2015-12-10 |
| Beachball Chess Tournament            | 2013-06-12 | Andrei     | 2009-12-11 |

After analyzing the data, we have found some data which have different descriptions than "chess." For example, "Church," "Eteläkatu," "Lunch place" do not have chess in the description but added in the same date or (event), and we consider these are not relevant. Figure 22 shows an example of relevant and non-relevant data for chess.



**Figure 22.** Example of relevant and non-relevant data for "Chess."

It also happened because some other users uploaded photos in Mopsi on the same date, except those involved in any chess events. We have manually annotated these data and estimated ground truth as 108 for the chess dataset, while the total number of data is 1050. As the false values are 942, so the proportion of the true and false values in the chess dataset is not balanced, as we can observe from the number.

We have used another approach to estimate the ground truth is taking the largest true positive number as the "golden standard." For example, in the test set 3, where the keyword is "pizza," the largest true positive number is 98 for all possible queries, which are taken as ground truth for our calculation. Table 10 shows the list of golden standard values for some selected test sets.

**Table 10.** Estimated golden standard for each test set.

| Test set number | Keyword               | Golden standard |
|-----------------|-----------------------|-----------------|
| 1               | Chess                 | 108             |
| 2               | Lenkkireitin maisemia | 391             |
| 3               | Lenkkireitti          | 391             |
| 4               | Pizza                 | 98              |
| 5               | Lounas                | 115             |
| 6               | Marathon              | 262             |
| 7               | Swimming              | 131             |
| 8               | Beach                 | 297             |
| 9               | Kuhasalo King         | 36              |
| 10              | Kuntorastit           | 150             |
| 11              | Hölkkä                | 130             |

|    |            |     |
|----|------------|-----|
| 12 | Skiing     | 127 |
| 13 | Barbeque   | 19  |
| 14 | Restaurant | 733 |
| 15 | Kahvila    | 590 |
| 16 | Sauna      | 140 |
| 17 | Hotelli    | 359 |
| 18 | Apteekki   | 10  |

Here, our usable data from the Mopsi dataset is around 35000, which we used for testing, so the ratio of true and false values are not balanced for any of the keywords as the number of false counts is significantly higher than the true counts.

We calculate true positive and false positive values manually based on human intuition. Here, true positive means the returned result is true or expected, and false-positive means the returned result is not expected for that specific query. Note that these true positive and false positive values can be biased for different users because some results might be expected to one user but unexpected from another user's perspective.

For example, Figure 23 and Figure 24 represent a few positive values for the keyword "chess" using the Smith-Waterman-Gotoh measure where the similarity threshold is set as 0.8. The results in Figure 23 are considered to be true positives because the content and string such as "Swim chess tourney in progress" and "Football chess tournament 2012" are relevant for the specific keyword. The results in Figure 24 are considered to be false positives because none of the content and string such as "Tavrichesky garden," "Historisches museum," and "Chestnuts" are relevant for the keyword "chess," but they are positive results because the string in the description of those results and the keyword has string similarity 0.8 according to chosen measure.

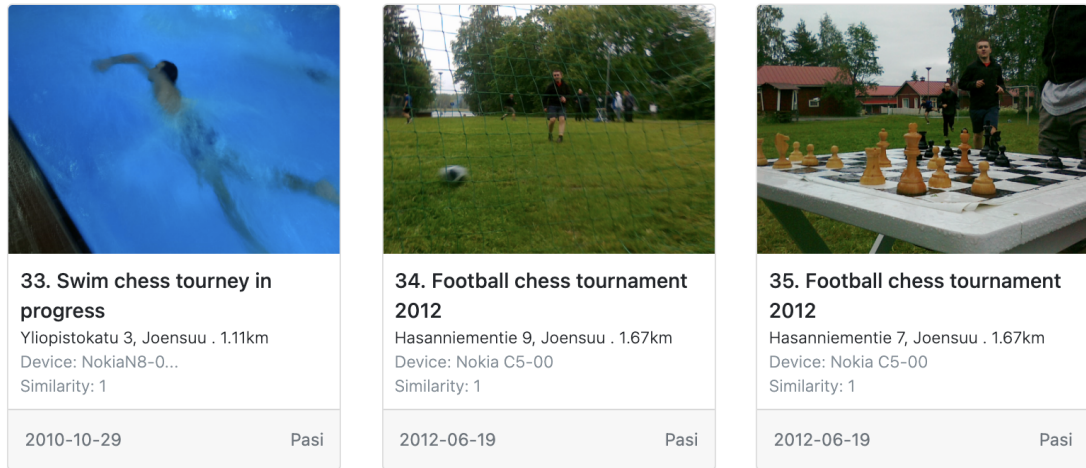


Figure 23. Example of "true positive" results for keyword "chess."

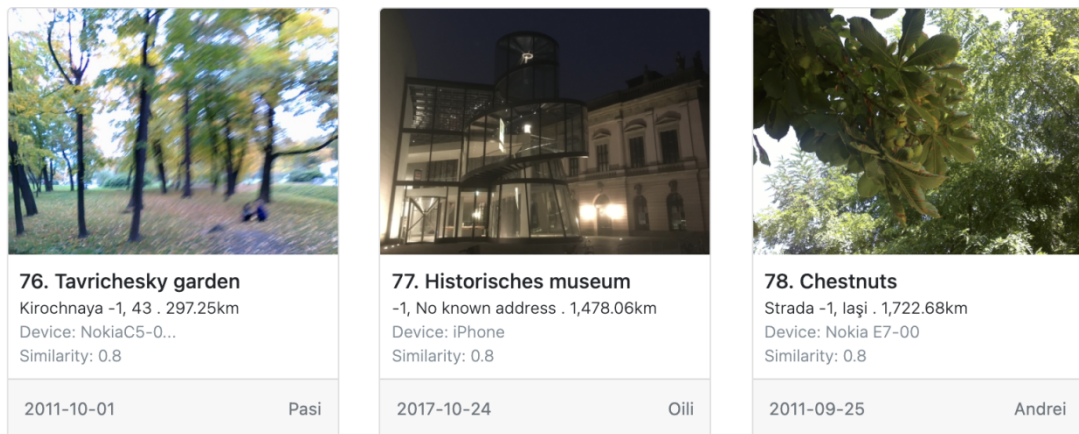


Figure 24. Example of "false positive" results for keyword "chess."

For defining the measures, we also need to calculate the number of false-negative results. Here false-negative represents the missed entries from the ground truth dataset. The calculation of true positive, false positive and false-negative for a few test sets has shown in Table 11. We can observe from Table 11, in the case of Levenshtein, Damerau-Levenshtein, Jaro, and Jaro-Winkler, the false positive is near to zero at a similarity threshold of 0.7 to 1.0, but the false negative is extremely higher. So, it means these measures produce a smaller number of results, but most of them are true. Meanwhile, Smith-Waterman-Gotoh produces fewer false-negative results even in the similarity threshold of 0.8 to 1. The calculation of true positive, false positive and false negative values for the rest of the test-sets has given in [Appendix 1](#).

**Table 11. True positive, false positive and false negative for a few test sets**

| Keywords                           | Threshold | Total true positive result |                     |                      |      |              | Total false positive result |                     |                      |       |              | Total false negative result |                     |                      |      |              |
|------------------------------------|-----------|----------------------------|---------------------|----------------------|------|--------------|-----------------------------|---------------------|----------------------|-------|--------------|-----------------------------|---------------------|----------------------|------|--------------|
|                                    |           | Levenshtein                | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro  | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Chess (test set-1)                 | 1         | 1                          | 1                   | 85                   | 1    | 1            | 0                           | 0                   | 0                    | 0     | 0            | 107                         | 107                 | 23                   | 107  | 107          |
|                                    | 0.9       | 1                          | 1                   | 85                   | 1    | 2            | 0                           | 0                   | 1                    | 0     | 1            | 107                         | 107                 | 23                   | 107  | 106          |
|                                    | 0.8       | 1                          | 1                   | 85                   | 5    | 10           | 0                           | 0                   | 20                   | 1     | 22           | 107                         | 107                 | 23                   | 103  | 98           |
|                                    | 0.7       | 1                          | 1                   | 85                   | 19   | 12           | 0                           | 0                   | 60                   | 180   | 1253         | 107                         | 107                 | 23                   | 89   | 96           |
|                                    | 0.6       | 1                          | 1                   | 85                   | 40   | 12           | 14                          | 0                   | 592                  | 2285  | 2916         | 107                         | 107                 | 23                   | 68   | 96           |
|                                    | 0.5       | 3                          | 3                   | 86                   | 70   | 22           | 38                          | 44                  | 1734                 | 9388  | 8096         | 105                         | 105                 | 22                   | 38   | 86           |
| Lenkkireitin maisemia (test set-2) | 1         | 280                        | 280                 | 293                  | 290  | 290          | 0                           | 0                   | 16                   | 0     | 0            | 111                         | 111                 | 98                   | 101  | 101          |
|                                    | 0.9       | 290                        | 290                 | 293                  | 292  | 292          | 0                           | 0                   | 26                   | 0     | 83           | 101                         | 101                 | 98                   | 99   | 99           |
|                                    | 0.8       | 290                        | 290                 | 293                  | 292  | 293          | 0                           | 0                   | 112                  | 88    | 138          | 101                         | 101                 | 98                   | 99   | 98           |
|                                    | 0.7       | 295                        | 295                 | 293                  | 293  | 293          | 5                           | 5                   | 145                  | 180   | 1302         | 96                          | 96                  | 98                   | 98   | 98           |
|                                    | 0.6       | 296                        | 295                 | 293                  | 293  | 293          | 28                          | 29                  | 272                  | 1936  | 8497         | 95                          | 96                  | 98                   | 98   | 98           |
|                                    | 0.5       | 366                        | 366                 | 293                  | 293  | 293          | 104                         | 109                 | 688                  | 14743 | 19090        | 25                          | 25                  | 98                   | 98   | 98           |
| Lenkkireitti (test set-3)          | 1         | 65                         | 65                  | 73                   | 65   | 65           | 0                           | 0                   | 0                    | 0     | 0            | 326                         | 326                 | 318                  | 326  | 326          |
|                                    | 0.9       | 65                         | 66                  | 73                   | 66   | 360          | 0                           | 0                   | 0                    | 0     | 0            | 326                         | 325                 | 318                  | 325  | 31           |
|                                    | 0.8       | 66                         | 66                  | 369                  | 362  | 366          | 0                           | 0                   | 7                    | 0     | 98           | 325                         | 325                 | 22                   | 29   | 25           |
|                                    | 0.7       | 66                         | 69                  | 369                  | 367  | 367          | 2                           | 0                   | 14                   | 123   | 598          | 325                         | 322                 | 22                   | 24   | 24           |
|                                    | 0.6       | 67                         | 70                  | 369                  | 369  | 367          | 32                          | 30                  | 100                  | 1516  | 3535         | 324                         | 321                 | 22                   | 22   | 24           |
|                                    | 0.5       | 363                        | 363                 | 369                  | 369  | 367          | 89                          | 92                  | 455                  | 12163 | 15249        | 28                          | 28                  | 22                   | 22   | 24           |
| Pizza (test set-4)                 | 1         | 3                          | 3                   | 59                   | 3    | 3            | 0                           | 0                   | 0                    | 0     | 0            | 95                          | 95                  | 39                   | 95   | 95           |
|                                    | 0.9       | 3                          | 3                   | 59                   | 3    | 9            | 0                           | 0                   | 35                   | 0     | 1            | 95                          | 95                  | 39                   | 95   | 89           |
|                                    | 0.8       | 3                          | 3                   | 98                   | 13   | 36           | 1                           | 1                   | 38                   | 1     | 31           | 95                          | 95                  | 0                    | 85   | 62           |
|                                    | 0.7       | 3                          | 3                   | 98                   | 40   | 39           | 1                           | 1                   | 39                   | 49    | 158          | 95                          | 95                  | 0                    | 58   | 59           |
|                                    | 0.6       | 3                          | 3                   | 98                   | 55   | 39           | 27                          | 28                  | 53                   | 584   | 978          | 95                          | 95                  | 0                    | 43   | 59           |
|                                    | 0.5       | 11                         | 11                  | 98                   | 66   | 40           | 36                          | 42                  | 128                  | 5662  | 4591         | 87                          | 87                  | 0                    | 32   | 58           |
| Lounas (test set-5)                | 1         | 4                          | 4                   | 99                   | 4    | 4            | 0                           | 0                   | 0                    | 0     | 0            | 111                         | 111                 | 16                   | 111  | 111          |
|                                    | 0.9       | 4                          | 4                   | 99                   | 4    | 5            | 0                           | 0                   | 0                    | 0     | 0            | 111                         | 111                 | 16                   | 111  | 110          |
|                                    | 0.8       | 4                          | 4                   | 102                  | 7    | 34           | 0                           | 0                   | 4                    | 5     | 20           | 111                         | 111                 | 13                   | 108  | 81           |
|                                    | 0.7       | 4                          | 4                   | 102                  | 45   | 42           | 0                           | 0                   | 4                    | 316   | 557          | 111                         | 111                 | 13                   | 70   | 73           |
|                                    | 0.6       | 4                          | 4                   | 102                  | 67   | 54           | 3                           | 4                   | 103                  | 2357  | 2630         | 111                         | 111                 | 13                   | 48   | 61           |
|                                    | 0.5       | 14                         | 14                  | 115                  | 110  | 72           | 164                         | 168                 | 1081                 | 11641 | 11069        | 101                         | 101                 | 0                    | 5    | 43           |

For testing, we have calculated precision, recall and F-score for each domain as:

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \dots\dots\dots 1$$

$$Recall = \frac{True\ positive}{True\ positive + False\ negative} \dots\dots\dots 2$$

$$F - score = 2 * \frac{Precision * Recall}{Precision + Recall} \dots\dots\dots 3$$

In the statistical analysis, the F-score or F-measure is a measure of a test's quality. F-score is calculated from the precision and recall of the test, where the precision is the

number of correctly identified positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of correctly identified positive results divided by the number of all samples that should have been identified as positive. Higher precision means that an algorithm returns more relevant results than irrelevant ones, and high recall means that an algorithm returns most of the relevant results (whether or not irrelevant ones are also returned). The highest possible value of an F-score is 1, indicating perfect precision and recall, and the lowest possible value is 0 if either the precision or the recall is zero.

### 7.2.1 Optimal threshold for each string similarity measure

We first examined which string similarity threshold is optimal for which similarity measure based on precision, recall, and F-score. Table 12 shows F-score values for different similarity measures and for Inclusion in the case of the test set 1 (Chess), and Table 13 for test set 2 (Lenkkireitin maisemia). Due to a large number of tested measures, we only plot selected measures in Table 12, 13. The rest of the measures, including individual precision, recall, and F-score for each test-set, have been added in [Appendix 1](#).

Table 12. F-score (%) for different measures for "Chess"

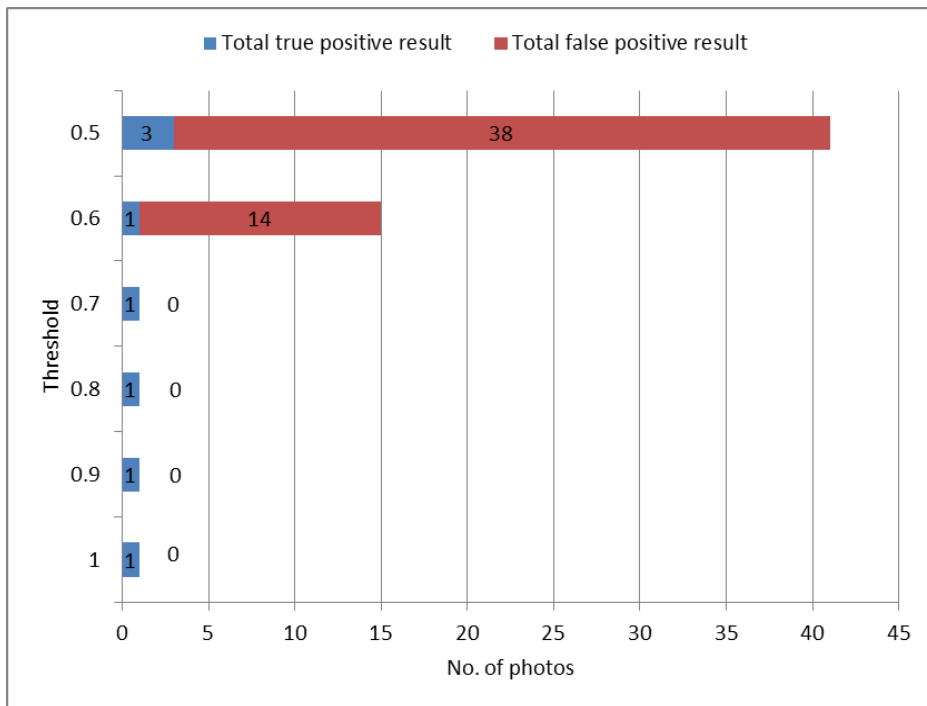
| Threshold | Le-<br>venshtein | Damerau-<br>Le-<br>venshtein | Smith Wa-<br>terman-<br>Gotoh | Jaro | Jaro-<br>Winkler | Inclu-<br>sion |
|-----------|------------------|------------------------------|-------------------------------|------|------------------|----------------|
| 1.0       | 1.8              | 1.8                          | 88.1                          | 1.8  | 1.8              | 88.1           |
| 0.9       | 1.8              | 1.8                          | 87.6                          | 1.8  | 3.6              |                |
| 0.8       | 1.8              | 1.8                          | 79.8                          | 8.8  | 14.3             |                |
| 0.7       | 1.8              | 1.8                          | 67.2                          | 12.4 | 1.7              |                |
| 0.6       | 1.6              | 1.8                          | 21.7                          | 3.3  | 0.8              |                |
| 0.5       | 4.0              | 3.9                          | 8.9                           | 1.5  | 0.5              |                |

Table 13. F-score (%) for different measures for "Lenkkireitin maisemia"

| Threshold | Le-<br>venshtein | Damerau-<br>Le-<br>venshtein | Smith Wa-<br>terman-<br>Gotoh | Jaro | Jaro-<br>Winkler | Inclu-<br>sion |
|-----------|------------------|------------------------------|-------------------------------|------|------------------|----------------|
| 1.0       | 83.5             | 83.5                         | 85.7                          | 85.2 | 85.2             | 85.7           |
| 0.9       | 85.2             | 85.2                         | 82.5                          | 85.5 | 76.2             |                |
| 0.8       | 85.2             | 85.2                         | 73.6                          | 75.7 | 71.3             |                |
| 0.7       | 85.4             | 85.4                         | 70.7                          | 67.8 | 29.5             |                |
| 0.6       | 82.8             | 82.5                         | 61.3                          | 22.4 | 6.4              |                |
| 0.5       | 85.0             | 84.5                         | 42.7                          | 3.8  | 3.0              |                |



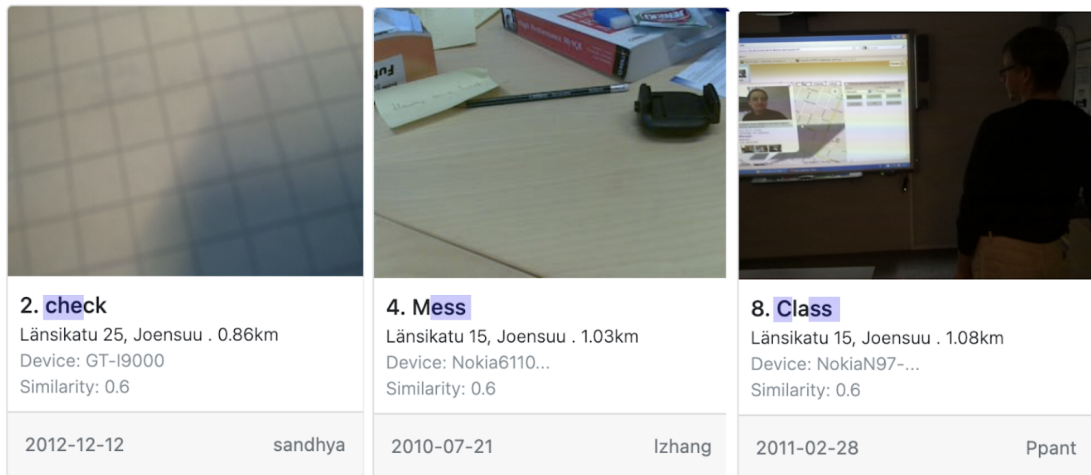
Figure 25 shows a graph of the true and false-positive results in case of different string similarity thresholds for the Levenshtein method for the keyword "Chess." We get the maximum number of true positive results for the similarity threshold 0.5 for "Chess," which is still poor results as we only get maximum 3 true positives where the golden standard is set to 108 (table 10), and for 0.7, we get only 1 with no false positive. So, the precision would be the highest for the similarity threshold greater than 0.7.



**Figure 25. Different string similarity threshold for Levenshtein method for "chess."**

We know from section 4 that Levenshtein and Damerau-Levenshtein method applies high cost only if one string is very smaller than another string. For example, in Figure 26, the data contains descriptions such as "check," "class," "mess," "cross," which are completely irrelevant for "chess," but the string similarity is 0.6 as they have at least 3 same characters at the same index within the total number of characters 5 within both of the keyword and descriptions which is greater than the similarity threshold.

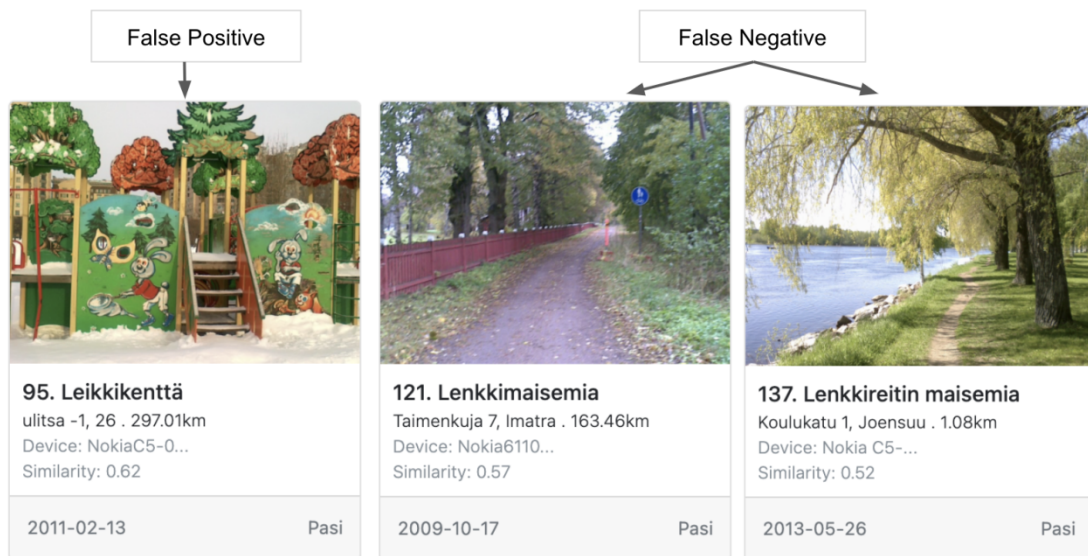




**Figure 26. False positive results for Levenshtein for "Chess" at threshold 0.6.**

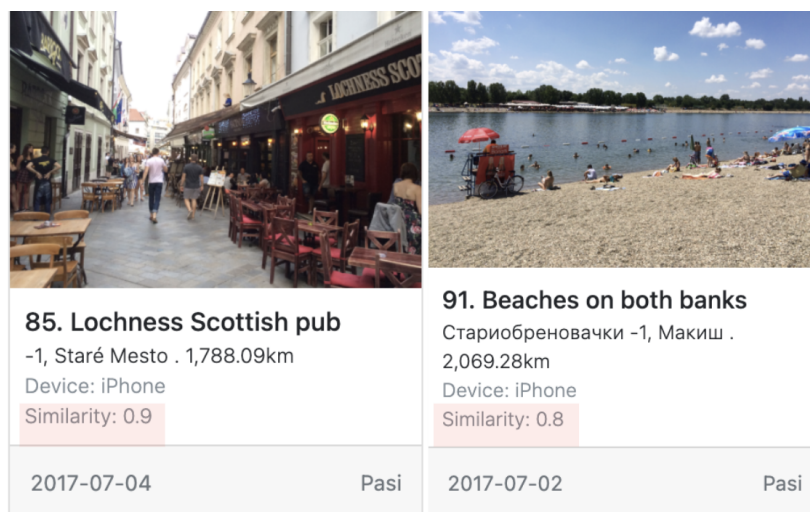
At the same time, some data in the dataset contains descriptions such as "Skiing chess competition," "Simultaneous chess in progress," which are relevant, but the string similarity is approximately 0.2, which is lower than the threshold as the dissimilar number of characters is much higher than the keyword.

On the other hand, from Table 13, we observe, Levenshtein produces good results for the keyword "Lenkkireitin maisemia" at the similarity threshold at 0.6 or more. The length of the description in the dataset is the same as a keyword for a large number of relevant data. For example, we observed that in the Mopsi dataset, there are 293 photos with the exact description as "Lenkkireitin maisemia". Suppose we test with the keyword "Lenkkireitti." In that case, Levenshtein will produce comparatively poor results for the similarity threshold at 0.6 or more because the similarity between "Lenkkireitti" and "Lenkkireitin maisemia" is 0.52 (see Figure 27), which is lower than the threshold. Still, the number of total relevant data is the same for these two keywords. So, it is false negative for threshold points greater than or equal to 0.6. If we set the threshold at 0.5, then the results would be better in this case. According to the Levenshtein measure, Figure 27 represents an example of a false positive and false negative for "Lenkkireitti" at threshold 0.6. Damerau-Levenshtein measure also produces a similar kind of results as the Levenshtein measure.



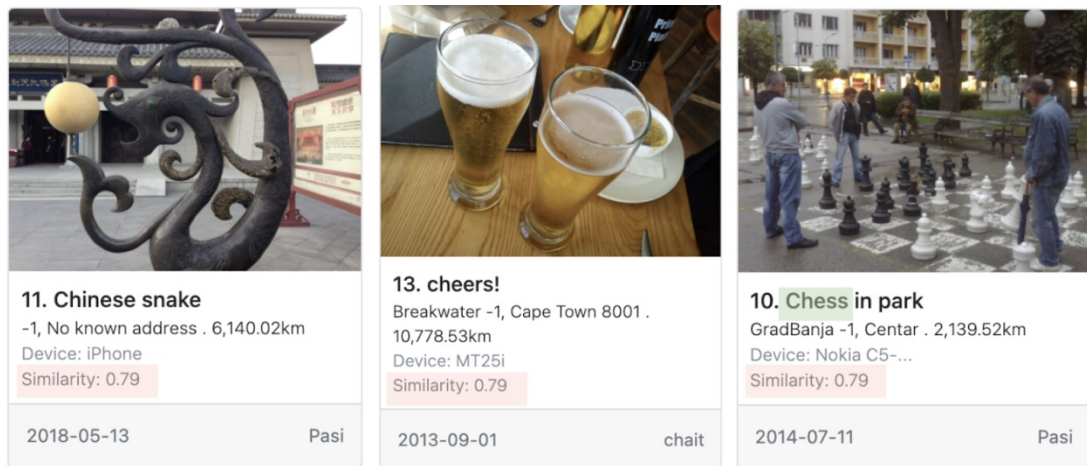
**Figure 27. False positive and negative results of Levenshtein measure for "Lenkkireitti" at threshold 0.6.**

Another character level measure Smith-Waterman-Gotoh (SWG), which applies the least cost for mismatch at the beginning and end of the string sequence. From Table 12 and 13, we observe it produces a higher F-score at the similarity threshold greater than 0.7. Still, it produces some false-positive results. For example, in Figure 28, "Lochness Scottish pub" is a false positive result for "chess" at 0.9 which has 90% of similarity in characters though it is not relevant. Also, "Beaches on both banks" has 80% of similarity which is false positive at threshold 0.8.



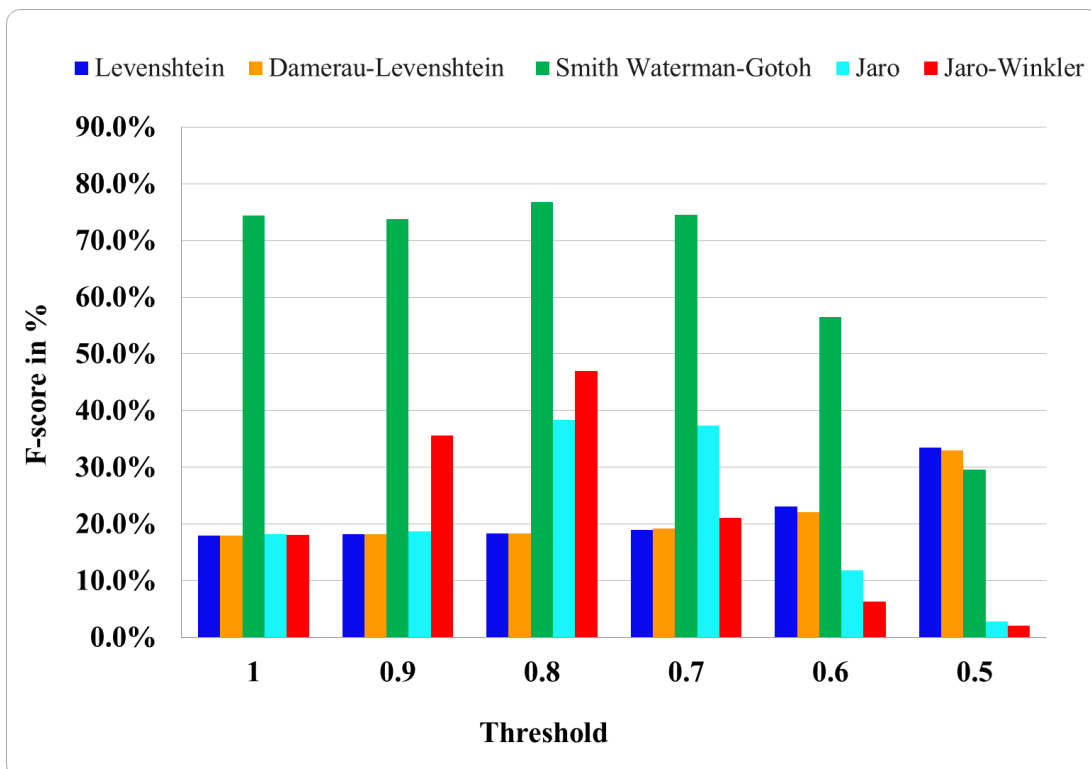
**Figure 28. Example of false positive of SWG at threshold 0.8.**

We know that Jaro and Jaro-Winkler methods apply the least string similarity matching cost if both strings are identical. For example, in Figure 29, "Chess in the park" and "chess" has a similarity score of 0.79, which would be a false negative in the case of similarity threshold 0.8. Still, at the same time, "Chinese snake" or "cheers!" with "chess" also has the same similarity as 0.79, which is false-positive in the case of similarity threshold 0.7.



**Figure 29. Example of Jaro at threshold 0.7 for "chess."**

The average F-score of different similarity measures at different threshold is visualized in a chart diagram in Figure 30. It indicates the optimal threshold point for all string similarity measures based on the F-score value. It also indicates that Smith-Waterman-Gotoh produces a higher F-score for any similarity threshold point.



**Figure 30. Average F-score for different similarity measures.**

The results are summarized as averages of precision, recall, and F-score measures for all domains in Table 14. As can be observed, the F-score for Levenshtein and Damerau-Levenshtein is less than 30% on average. For the Smith-Waterman-Gotoh measure, the precision decreases at a lower threshold, producing less of the false positives at a threshold equal to or greater than 0.8. Jaro measure also produces poor results on average, and the maximum average F-score value for all test sets is found at the similarity threshold of 0.8. Jaro-Winkler also produces similar results as Jaro for most of the cases, and on average, it scores a maximum F-score at the similarity threshold of 0.8. At threshold 0.7 for chess Jaro- Winkler produces 1265 results in which only 12 are true positive ([Appendix 1](#)).

Table 14. Average precision, recall, and F-score (%) for different Measures.

| Threshold | Levenshtein |           |         | Damerau-Levenshtein |           |         | Smith Waterman-Gotoh |           |         | Jaro   |           |         | Jaro-Winkler |           |         | Inclusion |           |         |
|-----------|-------------|-----------|---------|---------------------|-----------|---------|----------------------|-----------|---------|--------|-----------|---------|--------------|-----------|---------|-----------|-----------|---------|
|           | Recall      | Precision | F-score | Recall              | Precision | F-score | Recall               | Precision | F-score | Recall | Precision | F-score | Recall       | Precision | F-score | Recall    | Precision | F-score |
| 1.0       | 10          | 100       | 18      | 10                  | 100       | 18      | 63                   | 99        | 74      | 10     | 100       | 18      | 10           | 100       | 18      | 65        | 100       | 76      |
| 0.9       | 10          | 100       | 18      | 10                  | 100       | 18      | 63                   | 97        | 74      | 10     | 100       | 19      | 25           | 94        | 36      |           |           |         |
| 0.8       | 10          | 98        | 18      | 10                  | 98        | 18      | 72                   | 89        | 77      | 28     | 78        | 38      | 40           | 61        | 47      |           |           |         |
| 0.7       | 11          | 94        | 19      | 11                  | 89        | 19      | 78                   | 76        | 75      | 45     | 38        | 37      | 42           | 16        | 21      |           |           |         |
| 0.6       | 16          | 61        | 23      | 16                  | 64        | 22      | 80                   | 51        | 56      | 56     | 7         | 12      | 46           | 3         | 6       |           |           |         |
| 0.5       | 31          | 47        | 33      | 32                  | 43        | 33      | 85                   | 20        | 30      | 69     | 1         | 3       | 54           | 1         | 2       |           |           |         |

Based on the above discussion, we observe, on average, the optimal threshold value for both Levenshtein and Damerau-Levenshtein is 0.5, where F-score is 33%, Smith-Waterman-Gotoh at 0.8 with 77%, Jaro at 0.7 with 38%, and Jaro-Winkler at 0.8 with 47% F-score. In general, the optimal threshold is greater than 0.7 for most cases. The average F-score for Inclusion is 76%, which is near to the Smith-Waterman-Gotoh measure.

### 7.2.2 Quality of the measures

We tested each string similarity measure's flexibility under the same similarity threshold belonging to the same set. Figure 31 shows a few example data chosen randomly from the Mopsi database to observe how each method measures the same data under the same domain (same test set). As the average character length in Mopsi data is 17 to a maximum of 65 characters, we have chosen these data containing mostly 20 to 25 characters. Each data contains images, descriptions, addresses, and keywords, which we use to analyze the result. We tested each extracted keyword from those data in our tool and observed if these data are returned as the results or not for a fixed similarity threshold. We chose this similarity threshold as 0.8 because, from our last experiment, we observe that most of the similarity measures produce better results when the similarity threshold is set greater than 0.7. We represent the observations in Table 15, where "+" represents positive results and "-" represents negative results. We can see Levenshtein and Damerau-Levenshtein produce negative results, whereas Smith-Waterman-Gotoh (SWG) produces positive results for all data. If we would choose some data containing the exact same description as the



keyword, then all measures would produce a positive result. In data F (Figure 31), the description of the photo is "SDU Student Restaruant" which is misspelled, so the keyword "restaurant" is not exact in this case, but the similarity is greater than 0.8 for Smith-Waterman, Smith-Waterman-Gotoh, Jaro, and Jaro-Winkler.

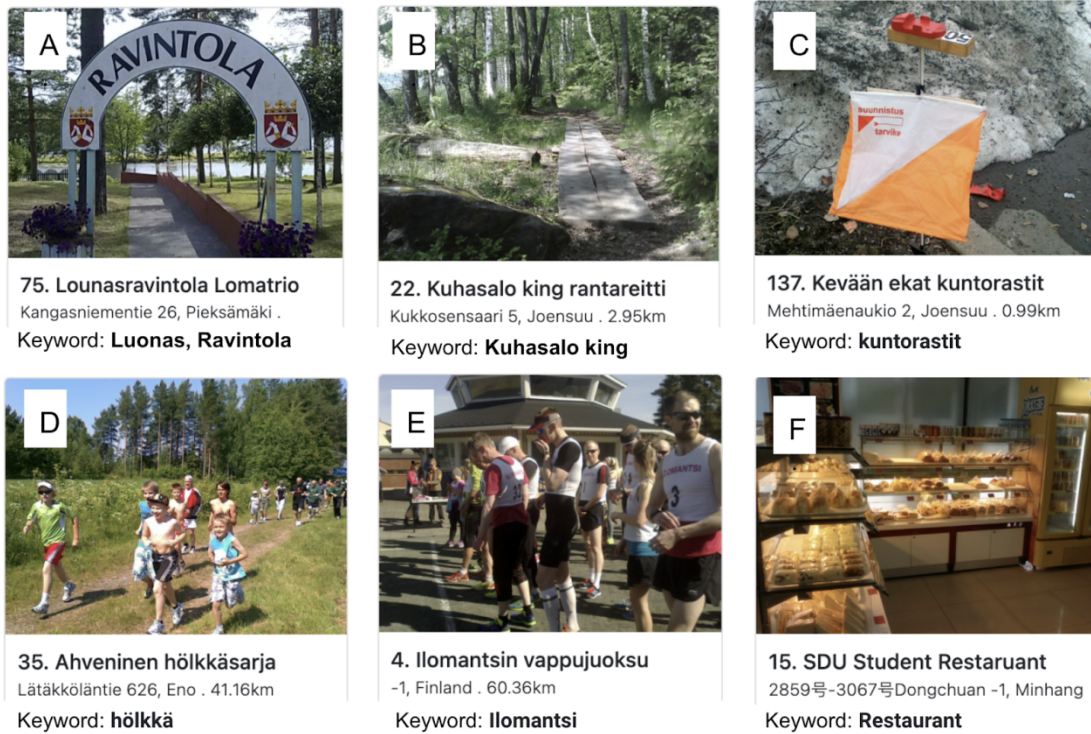


Figure 31. Random test data from Mopsi Database with keyword.

Table 15. Flexibility of string similarity measures at same similarity threshold (0.8).

| Test Data        | Inclusion | Levenshtein | Damerau-Levenshtein | SWG | Jaro | Jaro-Winkler |
|------------------|-----------|-------------|---------------------|-----|------|--------------|
| A. Luonas        | +         | -           | -                   | +   | -    | +            |
| B. Kuhasalo King | +         | -           | -                   | +   | +    | +            |
| C. kuntorastit   | +         | -           | -                   | +   | -    | -            |
| D. hölkkä        | +         | -           | -                   | +   | +    | -            |
| E. Ilomantsi     | +         | -           | -                   | +   | +    | +            |
| F. restaurant    | -         | -           | -                   | +   | -    | -            |

The quality of the measures can be determined by its ability to find matching entries from the dataset as the dataset contains descriptions that are not symmetrical by their length or number of total characters and number of tokens, so the same measure would produce different quality results for different keywords. In an ideal case, each

measure would produce similar results for any keyword. Table 16 represents which similarity measure produces what maximum scores for a specific keyword regardless their similarity threshold. For example, as for Levenshtein, which has F-score 4.1% maximum for the keyword "chess," but it scores 86.1% maximum for the keyword "Lenkkireitti." We can observe the same behavior for the Damerau-Levenshtein, Jaro, Jaro-Winkler measure. Smith-Waterman-Gotoh gives more consistent measures for most of the keywords. It also produces comparatively the most F-score values for most of the chosen keywords.

**Table 16. Maximum F-score in % (threshold) of any measure for different keywords.**

| Keyword               | Levenshtein | Damerau-Levenshtein | SWG  | Jaro | Jaro-Winkler | Inclusion |
|-----------------------|-------------|---------------------|------|------|--------------|-----------|
| Chess                 | 4.1         | 3.9                 | 88.1 | 12.4 | 14.3         | 88.1      |
| Lenkkireitin maisemia | 85.4        | 85.4                | 85.7 | 85.5 | 85.2         | 85.7      |
| Lenkkireitti          | 86.1        | 85.8                | 96.2 | 96.1 | 95.9         | 66.7      |
| Pizza                 | 15.2        | 14.6                | 83.8 | 42.8 | 43.6         | 75.2      |
| Lounas                | 9.9         | 9.4                 | 92.5 | 18.9 | 40.2         | 92.5      |
| Marathon              | 25.8        | 25.4                | 76.4 | 19.1 | 17.8         | 76.4      |
| Swimming              | 27.6        | 27.8                | 72.8 | 28.1 | 36.3         | 70.9      |
| Beach                 | 49.8        | 50.4                | 99.8 | 61.5 | 63.8         | 99.8      |
| Kuhasalo King         | 72.1        | 68.8                | 75.9 | 70.2 | 69.1         | 75.9      |
| Kuntorastit           | 50.2        | 61.4                | 92.4 | 74.2 | 74.9         | 89.3      |
| Hölkä                 | 17.7        | 8.8                 | 72.2 | 26   | 0.3          | 61.0      |
| Skiing                | 17.5        | 17.1                | 74.9 | 23.7 | 41.1         | 74.9      |
| Barbeque              | 34          | 32                  | 82.4 | 50   | 50           | 84.8      |
| Restaurant            | 37.6        | 37.2                | 77.1 | 35   | 28.1         | 78.4      |
| Kahvila               | 7           | 7.2                 | 55.8 | 24.7 | 24.5         | 44.9      |
| Sauna                 | 31.2        | 32.3                | 95.9 | 41.8 | 49.4         | 95.9      |
| Hotelli               | 18.3        | 18.3                | 93.8 | 45.5 | 59.2         | 29.5      |
| Apteekki              | 33.3        | 33.3                | 94.7 | 33.3 | 46.2         | 88.9      |

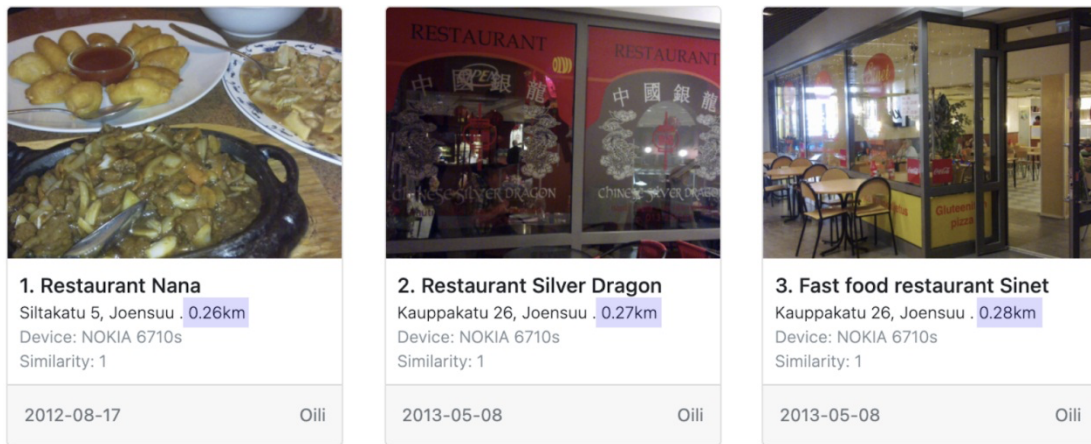
Compared to the Inclusion match Smith-Waterman-Gotoh produces greater F-score value for "Lenkkireitti," "Pizza," "Swimming," "Kuntorastit," "Hölkä," "Kahvila," "Hotelli," "Apteekki" at string similarity threshold 0.8 whereas both measures produce same F-score value for other keywords except "Barbeque," and "Restaurant."

### 7.2.3 Correlation to physical distance and similarity measure

When we search for something, we first check whether the data's content is relevant to the search query. For example, when someone searches for "grocery store," the search engine shifts through all the possible matches in the database and lists them in the index. And to order these results, we consider the physical distance along with the content relevance. Now we observe if our tool can find the most relevant results based on the user's query and return those data on the top of the results or not. Ordering is important as if people search for something, and they tend to look at the first few results, not the entire list of results. For example, if a search query produces 500 results, and 200 is true positive among them, it will be expected that those true positive results would be at the top of the entire list. Otherwise, if users see top results are mostly unexpected, they may not continue looking at the data until the end of the list.

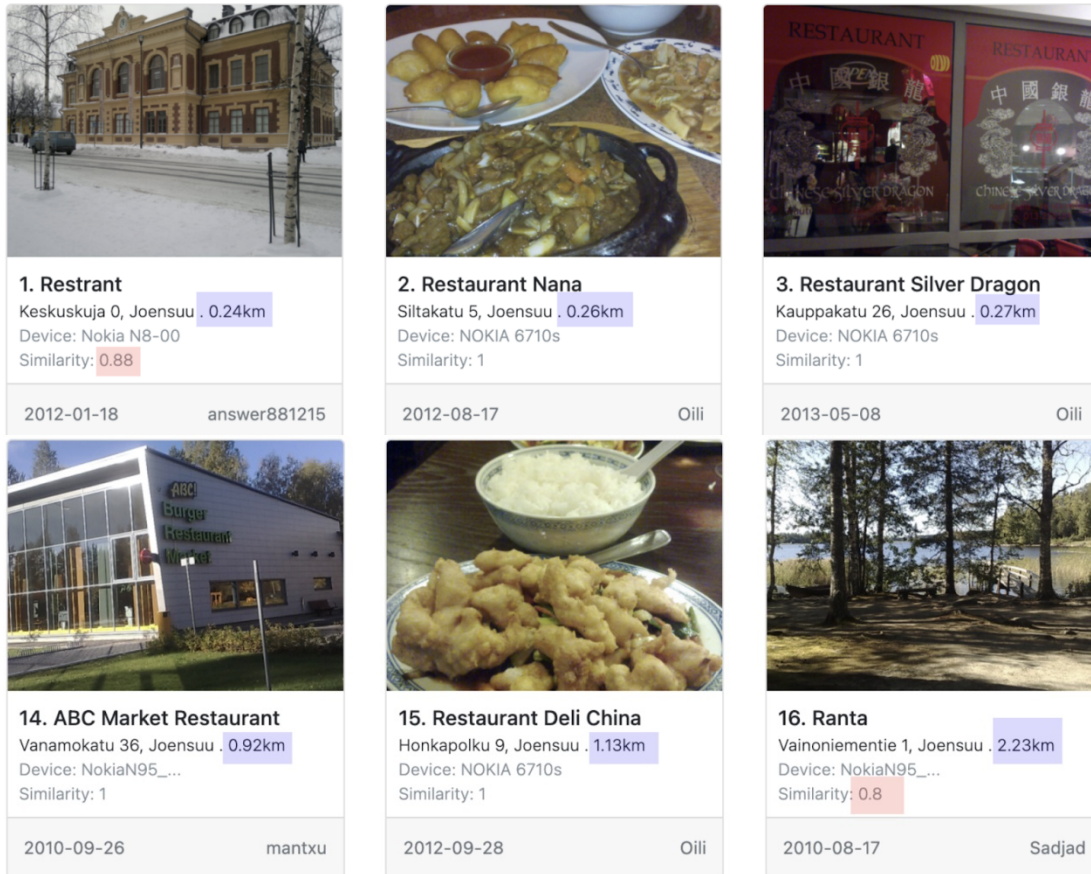
For this experiment, we consider users limit the search result, i.e., 20 photos, so our tool should return the top 20 relevant photos based on the user's specification. There might be a few different scenarios and such as users want to see the results within a given distance. For example, we have selected a search query as keyword "restaurant," the maximum number of results is 20, the string similarity measure is Smith-Waterman Gotoh, similarity threshold is 0.8, ordered by string similarity, distance radius is 10km, and address is Niskakatu, Joensuu, Finland. For this query, we get a maximum of 20 results, and all of them will be within a 10km radius from a specified location, and they will be ordered by string similarity. Some of the results may have the same string similarity, and in that case, this tool will perform multi-sort, so the results with the same string similarity will be sorted by physical distance. Figure 32 shows that all the top results have string similarity 1 for the abovementioned query, and they are internally sorted by physical distance, i.e., 0.26km, 0.27km, and 0.28km at index 1, 2, 3, respectively. And if we change the address to Helsinki, Finland, then it will return only those results which are located within a 10km radius from Helsinki.





**Figure 32. Ordered by string similarity for keyword "restaurant" from Niskakatu.**

Another condition might be whether the user wants to see the nearby top results even if the content does not have an exact match with the keyword. To satisfy this, we have selected the ordering as the physical distance for the abovementioned query, and it returns 20 results, of which 17 are true-positive, and 3 results are false positive. For example, the photo in Figure 33 contains "Restrant" which is misspelled, so the string similarity is 0.88, and still, it is at index 1 as it has the lowest distance (0.24km) than others and "ranta" at index 16. As our tool only works for syntactic similarity, it does not consider the meaning of the content, and it satisfies the condition given in the query, as mentioned earlier.



**Figure 33. Ordered by physical distance for keyword "restaurant" from Niskakatu.**

At the end of our experiments and case studies, we have some observations based on our pre-determined objectives. Those observations are as follows.

- The proportion of the true and false data in the Mopsi Dataset is not balanced for any keyword.
- Smith-Waterman-Gotoh has performed better than other similarity measures at the similarity threshold values from 0.8 to 1.0.
- Some titles of the Mopsi datasets have spelling mistakes, but our tool can still retrieve those data due to approximate searching.
- When the similarity threshold value is set from 0.1 to 0.5, most of the similarity measures produce more irrelevant results than relevant.

## 8 Conclusions

We have developed a photo searching tool for the Mopsi database based on approximate searching where people can search for the nearby photo according to their current location. As can be observed from our experimentation, based on the types and characteristics of data in the Mopsi database, our tool gives the best quality results for string similarity measures as Smith-Waterman-Gotoh compared to other similarity measures, and it produces an average F-score 77% for the selected test-sets. It also produces better results compared to Inclusion matching in most of the cases.

Our tool only supports syntactic similarity measures, so there remains a gap between human intuition of what they expect to see and the results from this tool. This work can be extended further to integrate semantic similarity measures to support this issue. Another limitation is that our tool does not support the multi-language selection so when user search for something, they may like to see also those results which are relevant but in a different language, i.e., "jogging" and "hölkkä" are relevant in meaning, but one is in English, and another one is in the Finnish language.

## References

- A. Ahson, S., & Ilyas, M. (2011). Location-based services handbook: Applications, technologies, and security. CRC Press.
- Abdeen, R. A. (2019). An algorithm for string searching. *International Journal of Computer Applications*, 17-22.
- Apostolico, A., & Galill, Z. (1997). *Pattern matching algorithms*. New York: Oxford University Press.
- Benharzallah, S., Kazar, O., & Caplat, G. (2011). Intelligent query processing for semantic mediation. *Egyptian Informatics Journal*, 151-162.
- Benza-Yeats, R., & Ganzalo, N. (1998). Fast approximate string matching in a dictionary. *University of Chili*, 1-8.
- Bislimovska, B., Bozzon, A., Brambilla, M., & Fraternali, P. (2012). Search upon UML repositories with text-matching techniques. *IEEE*, 9-12.
- Blandineau, M. (2020). *What is search relevance?* Retrieved from Algolia: <https://blog.algolia.com/what-is-search-relevance/>
- Boytssov, L. (2011). Indexing methods for approximate dictionary searching: Comparative analysis. *ACM*, 1-91.
- Brimicombe, A., & Li, C. (2009). Location-based services and geo-information engineering. Wiley-Blackwell.
- Choi, S.-S., Cha, S.-H., & Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Systemics, Cybernetics and Informatics*, 43-48.
- Church, K. W., & Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Comput. Linguist.*, 22-29.
- CodeOrg. (2017). *The internet: How search works*. Retrieved from CODE: <https://www.code.org/>

- Cohen, W. W., Ravikumar, P., & Fienberg, S. E. (2003). A comparison of string distance metrics for name matching tasks. *American association for artificial intelligence*, 1-6.
- ComputerHope. (2018). *Search*. Retrieved from Computer Hope: <https://www.computerhope.com/jargon/s/search.htm>
- Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 171-176.
- Dodsworth, E., & Nicholson, A. (2012). Academic uses of Google Earth and Google Maps in a library setting. *ResearchGate*, 102-117.
- El-gayar, M., Mekky, N., & Atwan, A. (2015). Efficient Proposed framework for semantic search engine using eew semantic ranking algorithm. *International Journal of Advanced Computer Science and Applications*, 1-9.
- Erasmus. (2020). *Search methods & techniques: search techniques*. Retrieved from Erasmus University Library: <https://libguides.eur.nl/informationsskillssearchmethods>
- Ferguson, N. (2020). *The difference between frontend and backend web development*. Retrieved from CareerFoundry: <https://careerfoundry.com/en/blog/web-development/whats-the-difference-between-frontend-and-backend/>
- Fränti, P., Tabarcea, A., Kuittinen, J., & Hautamäki, V. (2010). Location-based search engine for multimedia phones. *Proceedings of the 2010 IEEE International Conference on Multimedia and Expo*, (pp. 558-563). Singapore.
- Friedman, C., & Sideli, R. (1992). Tolerating spelling errors during patient validation. *Elsevier*, 486-509.
- Galhardas, H. (2013). *Data cleaning and transformation- record linkage*. <https://www.slideshare.net/amooool2000/token-17595472>.

- Gali, N., Mariescu-Istodor, R., & Fränti, P. (2016). Similarity measures for title matching. *International Conference on Pattern Recognition* (pp. 1549-1554). Cancún, México: IEEE.
- Gali, N., Mariescu-Istodor, R., Hostettler, D., & Fränti, P. (2019). Framework for syntactic string similarity measures. *ScienceDirect*, 169-185.
- GoogleBusinessHelp. (2020). *Improve your local ranking on Google*. Retrieved from Google my business help : <https://support.google.com/business/answer/7091?hl=en#:~:text=How%20Google%20determines%20local%20ranking,best%20match%20for%20your%20search.>
- GoogleCodeArchive. (2016). *Long-term storage for google code project hosting*. Retrieved from code.google.com: <https://code.google.com/archive/p/word2vec/>
- GoogleMapsPlatform. (2020). *Google maps platform documentation*. Retrieved from Google Maps Platform : <https://developers.google.com/maps/>
- Gotoh, O. (1982). An improved algorithm for matching biological sequences. *Journal of Molecular Biology*, 705-708.
- Guemmat, E. K., & Ouahabi, S. (2019). Towards a new educational search engine based on hybrid searching and indexing techniques. *IEEE*, 1-5.
- Hakak, S. (2019). Exact string matching algorithms: survey, issues, and future research. *IEEE access* , 1-25.
- Hamming, R. W. (1950). Error detecting and error correcting codes. *Bell Labs Technical Journal*, 147-160.
- Harispe, S., Ranwez, S., Janaqi, S., & Montmain, J. (2015). Semantic similarity from natural language and ontology analysis. *Synthesis lectures on human language technologies*. , 1-254.

- Hussan, B. K. (2020). Comparative study of semantic and keyword based search engines. *Advances in Science, Technology and Engineering Systems*, 106-111.
- Jacob, A. E., Ashodariya, N., & Dhongade, A. (2017). Hybrid search algorithm-combined linear and binary search algorithm. *IEEE*, 1543-1547.
- Janani, R., & Vijayarani, S. (2006). An efficient text pattern matching algorithm for retrieving information from desktop. *Indian Journal of Science and Technology*, 1-11.
- Jaro, M. A. (1989). Advances in record-linkage methodology as applied to matching the 1985 Census of tampa, Florida. *Journal of the American Statistical Association* , 414-420.
- Kapoor, N. (2019). *Step by step guide to keyword research*. Retrieved from Relevance : <https://www.relevance.com/step-by-step-guide-to-keyword-research/>
- Kaur, A. (2015). A novel approach for syntactic similarity. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, 216-219.
- Kehrer, R. (2018). *Exact searching*. Retrieved from FamilySearch: <https://www.familysearch.org/en/>
- Kent, A. (2019). *How search engines use keywords*. Retrieved from WordTracker: <https://www.wordtracker.com/academy/keyword-research/guides/how-search-engines-use-keywords>
- Kettle, S. (2017). *Distance on a sphere: The haversine formula*. Retrieved from Esri Community: [https://community.esri.com/t5/coordinate-reference-systems/distance-on-a-sphere-the-haversine-formula/ba-p/902128#:~:text=For%20example%2C%20haversine\(%CE%B8\),longitude%20of%20the%20two%20points.](https://community.esri.com/t5/coordinate-reference-systems/distance-on-a-sphere-the-haversine-formula/ba-p/902128#:~:text=For%20example%2C%20haversine(%CE%B8),longitude%20of%20the%20two%20points.)

- Khatter, H., & Ahlawat, A. K. (2020). An intelligent personalized web blog searching technique using fuzzybased feedback recurrent neural network. *Springer-Verlag GmbH Germany*, 9321-9333.
- Khor, T. P. (2014). Keyword relevance in search engine optimization. *Open University Malaysia*, 1-96.
- Kononenko, K. (2018). *Visual explanations of HTML, CSS and JavaScript concepts*. Retrieved from CodeAnalogies: <https://blog.codeanalogies.com/2018/05/09/the-relationship-between-html-css-and-javascript-explained/>
- Kumar, N., Bibhu, V., Islam, M., & Bhardwaj, S. (2010). Approximate string matching algorithm . *International Journal on Computer Science and Engineering* , 641-644.
- Landauer, T. K., & Dumais, S. T. (1997). A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *American Psychological Association, Inc.*, 211-240.
- Landauer, T. K., Foltz, P. W., & Laham, D. (1998). An introduction to latent semantic analysis. 259-284.
- Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 707-710.
- Mala, V., & Lobiyal, D. (2016). Semantic and keyword based web techniques in information retrieval. *IEEE*, 23-26.
- Malakasiotis, P., & Androutsopoulos, I. (2007). Learning textual entailment using SVMs and string similarity measures. *Proceedings of the Workshop on Textual Entailment and Paraphrasing*, 42-47.
- Manage-ads. (2020). *About exact match*. Retrieved from Google Ads Help: <https://support.google.com/google-ads/answer/2497825?hl=en#:~:text=With%20exact%20match%2C%20you%20can,the%20query%20and%20the%20keyword.>



- Mariescu-Istodor, R. (2017). Efficient management and search of GPS routes. *University of Eastern Finland*, 1-163.
- Marsden, S. (2020). *How do search engines work?* Retrieved from DeepCrawl: <https://www.deepcrawl.com/knowledge/technical-seo-library/how-do-search-engines-work/>
- Mihalcea, R., Corley, C., & Strapparava, C. (2006). Corpus-based and knowledge-based measures of text semantic similarity. *American Association for Artificial Intelligence*, 1-6.
- Miller, G. A. (1995). WordNet: A lexical database for english. *ACM*, 39-41.
- Morris, S. (2020). *Ajax—What it is, How it works, and What it used for?* Retrieved from skillcrus: <https://skillcrush.com/blog/what-is-ajax/>
- Navarro, G. (2001). A guided tour to approximate string matching. *ACM Computing Surveys*, 1-82.
- Navarro, G., Baeza-Yeats, R., Sutinen, Y., & Tarhio, J. (2001). Indexing methods for approximate string matching. *IEEE*, 1-8.
- Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 443-453.
- Nls. (2020). *Satellite positioning*. Helsinki: National Land Survey of Finland .
- Prakhar. (2018). *Haversine formula to find distance between two points on a sphere*. Retrieved from geeksforgeeks: <https://www.geeksforgeeks.org/haversine-formula-to-find-distance-between-two-points-on-a-sphere/#:~:text=The%20Haversine%20formula%20calculates%20the,important%20for%20use%20in%20navigation.>
- Prokofyeva, N., & Boltunova, V. (2017). Analysis and practical application of PHP frameworks in development of web information systems. *ResearchGate*, 51-56.

- Rezaei, M., & Franti, P. (2016). Set matching measures for external cluster validity. *IEEE*, 2173 - 2186.
- Romain, S. (2019). *Geotagging and SEO: How your location matters*. Retrieved from Romain Berg: <https://www.romainberg.com/geotagging-and-seo-how-your-location-matters/#:~:text=Geotagging%20is%20the%20process%20of,information%20to%20photos%20and%20images>.
- Schiller, J., & Voisard, A. (2004). Location-based services. *ELSEVIER*, 1-255.
- Shapiro, L. G., & Haralick, R. M. (1981). Structural descriptions and inexact matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 504-516.
- Sheela, A. S., & Jayakumar, D. (2019). Comparative study of syntactic search engine and semantic search engine: A survey. *IEEE Xplore*, 1-4.
- Shin, S. (2018). Introduction to JSON (JavaScript Object Notation). Java Technology Architect.
- Smith, T., & Waterman, M. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, 195-197.
- Tabarcea, A. C. (2015). Location-based Web search and mobile applications. University of Eastern Finland.
- Taib, F., Abbou, F. M., & Alam, M. J. (2008). A matching approach for object oriented formal specifications. *JOURNAL OF OBJECT TECHNOLOGY*, 141-153.
- Turney, P. D. (2001). *Mining the web for synonyms: PMI-IR versus LSA on TOEFL*. M-50 Montreal Road, Ottawa, Ontario, Canada: Institute for Information Technology, National Research Council of Canada.
- TutorialRide. (2017). *Searching in data structure*. Retrieved from TutorialRide.com: <https://www.tutorialride.com/data-structures/searching-in-data->

structure.htm#:~:text=Searching%20is%20the%20process%20of,or%20on%  
20external%20data%20structure

Watters, C., & Amoudi, G. (2018). GeoSearcher: Location-based ranking of search. *Journal of the American Society for Information Science and Technology*, 140-151.

Winkler, W. E. (1990). String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. *In Proceedings of the section on survey research methods*, 354-359.

WordNets in the World. (2010). *Global WordNet association*. Retrieved from <http://globalwordnet.org/resources/wordnets-in-the-world/>

Xie, M. (2019). Trajectories medoid and clustering. *University of Eastern Finland* , 1-56.

# Appendix 1: Important data

Table 1a. Calculation of true positive, false positive and false negative values for different test set.

| Keywords                            | Threshold | Total true positive result |                     |                      |      |              | Total false positive result |                     |                      |       |              | Total false negative result |                     |                      |      |              |
|-------------------------------------|-----------|----------------------------|---------------------|----------------------|------|--------------|-----------------------------|---------------------|----------------------|-------|--------------|-----------------------------|---------------------|----------------------|------|--------------|
|                                     |           | Levenshtein                | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro  | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Chess (test set-1)                  | 1         | 1                          | 1                   | 85                   | 1    | 1            | 0                           | 0                   | 0                    | 0     | 0            | 107                         | 107                 | 23                   | 107  | 107          |
|                                     | 0.9       | 1                          | 1                   | 85                   | 1    | 2            | 0                           | 0                   | 1                    | 0     | 1            | 107                         | 107                 | 23                   | 107  | 106          |
|                                     | 0.8       | 1                          | 1                   | 85                   | 5    | 10           | 0                           | 0                   | 20                   | 1     | 22           | 107                         | 107                 | 23                   | 103  | 98           |
|                                     | 0.7       | 1                          | 1                   | 85                   | 19   | 12           | 0                           | 0                   | 60                   | 180   | 1253         | 107                         | 107                 | 23                   | 89   | 96           |
|                                     | 0.6       | 1                          | 1                   | 85                   | 40   | 12           | 14                          | 0                   | 592                  | 2285  | 2916         | 107                         | 107                 | 23                   | 68   | 96           |
|                                     | 0.5       | 3                          | 3                   | 86                   | 70   | 22           | 38                          | 44                  | 1734                 | 9388  | 8096         | 105                         | 105                 | 22                   | 38   | 86           |
| Lenkkireitin maiseimia (test set-2) | 1         | 280                        | 280                 | 293                  | 290  | 290          | 0                           | 0                   | 16                   | 0     | 0            | 111                         | 111                 | 98                   | 101  | 101          |
|                                     | 0.9       | 290                        | 290                 | 293                  | 292  | 292          | 0                           | 0                   | 26                   | 0     | 83           | 101                         | 101                 | 98                   | 99   | 99           |
|                                     | 0.8       | 290                        | 290                 | 293                  | 292  | 293          | 0                           | 0                   | 112                  | 88    | 138          | 101                         | 101                 | 98                   | 99   | 98           |
|                                     | 0.7       | 295                        | 295                 | 293                  | 293  | 293          | 5                           | 5                   | 145                  | 180   | 1302         | 96                          | 96                  | 98                   | 98   | 98           |
|                                     | 0.6       | 296                        | 295                 | 293                  | 293  | 293          | 28                          | 29                  | 272                  | 1936  | 8497         | 95                          | 96                  | 98                   | 98   | 98           |
|                                     | 0.5       | 366                        | 366                 | 293                  | 293  | 293          | 104                         | 109                 | 688                  | 14743 | 19090        | 25                          | 25                  | 98                   | 98   | 98           |
| Lenkkireitti (test set-3)           | 1         | 65                         | 65                  | 73                   | 65   | 65           | 0                           | 0                   | 0                    | 0     | 0            | 326                         | 326                 | 318                  | 326  | 326          |
|                                     | 0.9       | 65                         | 66                  | 73                   | 66   | 360          | 0                           | 0                   | 0                    | 0     | 0            | 326                         | 325                 | 318                  | 325  | 31           |
|                                     | 0.8       | 66                         | 66                  | 369                  | 362  | 366          | 0                           | 0                   | 7                    | 0     | 98           | 325                         | 325                 | 22                   | 29   | 25           |
|                                     | 0.7       | 66                         | 69                  | 369                  | 367  | 367          | 2                           | 0                   | 14                   | 123   | 598          | 325                         | 322                 | 22                   | 24   | 24           |
|                                     | 0.6       | 67                         | 70                  | 369                  | 369  | 367          | 32                          | 30                  | 100                  | 1516  | 3535         | 324                         | 321                 | 22                   | 22   | 24           |
|                                     | 0.5       | 363                        | 363                 | 369                  | 369  | 367          | 89                          | 92                  | 455                  | 12163 | 15249        | 28                          | 28                  | 22                   | 22   | 24           |
| Pizza (test set-4)                  | 1         | 3                          | 3                   | 59                   | 3    | 3            | 0                           | 0                   | 0                    | 0     | 0            | 95                          | 95                  | 39                   | 95   | 95           |
|                                     | 0.9       | 3                          | 3                   | 59                   | 3    | 9            | 0                           | 0                   | 35                   | 0     | 1            | 95                          | 95                  | 39                   | 95   | 89           |
|                                     | 0.8       | 3                          | 3                   | 98                   | 13   | 36           | 1                           | 1                   | 38                   | 1     | 31           | 95                          | 95                  | 0                    | 85   | 62           |
|                                     | 0.7       | 3                          | 3                   | 98                   | 40   | 39           | 1                           | 1                   | 39                   | 49    | 158          | 95                          | 95                  | 0                    | 58   | 59           |
|                                     | 0.6       | 3                          | 3                   | 98                   | 55   | 39           | 27                          | 28                  | 53                   | 584   | 978          | 95                          | 95                  | 0                    | 43   | 59           |
|                                     | 0.5       | 11                         | 11                  | 98                   | 66   | 40           | 36                          | 42                  | 128                  | 5662  | 4591         | 87                          | 87                  | 0                    | 32   | 58           |
| Lounas (test set-5)                 | 1         | 4                          | 4                   | 99                   | 4    | 4            | 0                           | 0                   | 0                    | 0     | 0            | 111                         | 111                 | 16                   | 111  | 111          |
|                                     | 0.9       | 4                          | 4                   | 99                   | 4    | 5            | 0                           | 0                   | 0                    | 0     | 0            | 111                         | 111                 | 16                   | 111  | 110          |
|                                     | 0.8       | 4                          | 4                   | 102                  | 7    | 34           | 0                           | 0                   | 4                    | 5     | 20           | 111                         | 111                 | 13                   | 108  | 81           |
|                                     | 0.7       | 4                          | 4                   | 102                  | 45   | 42           | 0                           | 0                   | 4                    | 316   | 557          | 111                         | 111                 | 13                   | 70   | 73           |
|                                     | 0.6       | 4                          | 4                   | 102                  | 67   | 54           | 3                           | 4                   | 103                  | 2357  | 2630         | 111                         | 111                 | 13                   | 48   | 61           |
|                                     | 0.5       | 14                         | 14                  | 115                  | 110  | 72           | 164                         | 168                 | 1081                 | 11641 | 11069        | 101                         | 101                 | 0                    | 5    | 43           |
| Swimming (test set-7)               | 1         | 0                          | 0                   | 162                  | 0    | 0            | 0                           | 0                   | 0                    | 0     | 0            | 262                         | 262                 | 100                  | 262  | 262          |
|                                     | 0.9       | 0                          | 0                   | 162                  | 0    | 22           | 0                           | 0                   | 0                    | 0     | 0            | 262                         | 262                 | 100                  | 262  | 240          |
|                                     | 0.8       | 0                          | 0                   | 162                  | 25   | 31           | 0                           | 0                   | 98                   | 5     | 55           | 262                         | 262                 | 100                  | 237  | 231          |
|                                     | 0.7       | 0                          | 0                   | 162                  | 48   | 31           | 0                           | 0                   | 112                  | 192   | 506          | 262                         | 262                 | 100                  | 214  | 231          |
|                                     | 0.6       | 13                         | 14                  | 162                  | 88   | 34           | 2                           | 2                   | 170                  | 1956  | 3036         | 249                         | 248                 | 100                  | 174  | 228          |
|                                     | 0.5       | 47                         | 47                  | 162                  | 98   | 64           | 56                          | 61                  | 822                  | 12462 | 13136        | 215                         | 215                 | 100                  | 164  | 198          |
| Marathon (test set-6)               | 1         | 1                          | 1                   | 75                   | 1    | 1            | 0                           | 0                   | 0                    | 0     | 0            | 130                         | 130                 | 56                   | 130  | 130          |
|                                     | 0.9       | 1                          | 1                   | 75                   | 2    | 16           | 0                           | 0                   | 0                    | 0     | 0            | 130                         | 130                 | 56                   | 129  | 115          |
|                                     | 0.8       | 1                          | 1                   | 75                   | 18   | 29           | 0                           | 0                   | 1                    | 2     | 0            | 130                         | 130                 | 56                   | 113  | 102          |
|                                     | 0.7       | 2                          | 2                   | 77                   | 37   | 60           | 0                           | 0                   | 8                    | 95    | 142          | 129                         | 129                 | 54                   | 94   | 71           |
|                                     | 0.6       | 20                         | 20                  | 80                   | 56   | 66           | 3                           | 4                   | 17                   | 888   | 1280         | 111                         | 111                 | 51                   | 75   | 65           |
|                                     | 0.5       | 35                         | 36                  | 131                  | 91   | 79           | 88                          | 92                  | 181                  | 6361  | 7777         | 96                          | 95                  | 0                    | 40   | 52           |

| Keywords                   | Threshold | Total true positive result |                     |                      |      |              | Total false positive result |                     |                      |       |              | Total false negative result |                     |                      |      |              |
|----------------------------|-----------|----------------------------|---------------------|----------------------|------|--------------|-----------------------------|---------------------|----------------------|-------|--------------|-----------------------------|---------------------|----------------------|------|--------------|
|                            |           | Levenshtein                | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro  | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Beach (test set-8)         | 1         | 80                         | 80                  | 296                  | 84   | 80           | 0                           | 0                   | 0                    | 0     | 0            | 217                         | 217                 | 1                    | 213  | 217          |
|                            | 0.9       | 80                         | 80                  | 296                  | 84   | 99           | 0                           | 0                   | 0                    | 0     | 0            | 217                         | 217                 | 1                    | 213  | 198          |
|                            | 0.8       | 84                         | 84                  | 297                  | 107  | 147          | 6                           | 6                   | 16                   | 7     | 17           | 213                         | 213                 | 0                    | 190  | 150          |
|                            | 0.7       | 84                         | 84                  | 297                  | 154  | 147          | 6                           | 6                   | 32                   | 50    | 112          | 213                         | 213                 | 0                    | 143  | 150          |
|                            | 0.6       | 84                         | 84                  | 297                  | 167  | 149          | 11                          | 12                  | 186                  | 753   | 743          | 213                         | 213                 | 0                    | 130  | 148          |
|                            | 0.5       | 104                        | 107                 | 297                  | 180  | 166          | 17                          | 21                  | 625                  | 6428  | 4372         | 193                         | 190                 | 0                    | 117  | 131          |
| Kuhasalo King (test set-9) | 1         | 0                          | 0                   | 22                   | 0    | 0            | 0                           | 0                   | 0                    | 0     | 36           | 36                          | 14                  | 36                   | 36   |              |
|                            | 0.9       | 0                          | 0                   | 22                   | 0    | 19           | 0                           | 0                   | 0                    | 0     | 36           | 36                          | 14                  | 36                   | 17   |              |
|                            | 0.8       | 0                          | 0                   | 22                   | 20   | 27           | 0                           | 0                   | 0                    | 1     | 20           | 36                          | 36                  | 14                   | 16   | 9            |
|                            | 0.7       | 0                          | 0                   | 22                   | 28   | 27           | 0                           | 0                   | 111                  | 53    | 824          | 36                          | 36                  | 14                   | 8    | 9            |
|                            | 0.6       | 15                         | 15                  | 35                   | 31   | 31           | 0                           | 0                   | 162                  | 2295  | 5936         | 21                          | 21                  | 1                    | 5    | 5            |
|                            | 0.5       | 22                         | 22                  | 35                   | 31   | 31           | 3                           | 6                   | 379                  | 14523 | 19143        | 14                          | 14                  | 1                    | 5    | 5            |
| Kuntorastit (test set-10)  | 1         | 17                         | 17                  | 121                  | 17   | 17           | 0                           | 0                   | 1                    | 0     | 0            | 133                         | 133                 | 29                   | 133  | 133          |
|                            | 0.9       | 17                         | 17                  | 127                  | 17   | 50           | 0                           | 0                   | 1                    | 0     | 2            | 133                         | 133                 | 23                   | 133  | 100          |
|                            | 0.8       | 17                         | 17                  | 134                  | 92   | 112          | 0                           | 0                   | 6                    | 6     | 37           | 133                         | 133                 | 16                   | 58   | 38           |
|                            | 0.7       | 22                         | 24                  | 135                  | 120  | 118          | 2                           | 0                   | 19                   | 104   | 800          | 128                         | 126                 | 15                   | 30   | 32           |
|                            | 0.6       | 33                         | 36                  | 137                  | 130  | 123          | 3                           | 5                   | 129                  | 2105  | 5065         | 117                         | 114                 | 13                   | 20   | 27           |
|                            | 0.5       | 57                         | 78                  | 148                  | 144  | 135          | 20                          | 26                  | 1274                 | 14315 | 17402        | 93                          | 72                  | 2                    | 6    | 15           |
| Hölkä (test set-11)        | 1         | 0                          | 0                   | 57                   | 0    | 0            | 0                           | 0                   | 0                    | 0     | 130          | 130                         | 73                  | 130                  | 130  |              |
|                            | 0.9       | 0                          | 0                   | 57                   | 0    | 0            | 0                           | 0                   | 0                    | 0     | 130          | 130                         | 73                  | 130                  | 130  |              |
|                            | 0.8       | 0                          | 0                   | 82                   | 0    | 0            | 0                           | 0                   | 15                   | 9     | 1            | 130                         | 130                 | 48                   | 130  | 130          |
|                            | 0.7       | 0                          | 0                   | 82                   | 26   | 0            | 0                           | 10                  | 27                   | 44    | 49           | 130                         | 130                 | 48                   | 104  | 130          |
|                            | 0.6       | 0                          | 3                   | 82                   | 43   | 0            | 11                          | 26                  | 31                   | 344   | 437          | 130                         | 127                 | 48                   | 87   | 130          |
|                            | 0.5       | 16                         | 16                  | 84                   | 75   | 7            | 35                          | 216                 | 239                  | 2663  | 4095         | 114                         | 114                 | 46                   | 55   | 123          |
| Skiing (test set-12)       | 1         | 1                          | 1                   | 76                   | 1    | 1            | 0                           | 0                   | 0                    | 0     | 0            | 126                         | 126                 | 51                   | 126  | 126          |
|                            | 0.9       | 1                          | 1                   | 76                   | 1    | 17           | 0                           | 0                   | 0                    | 0     | 0            | 126                         | 126                 | 51                   | 126  | 110          |
|                            | 0.8       | 1                          | 1                   | 76                   | 18   | 38           | 0                           | 0                   | 0                    | 7     | 20           | 126                         | 126                 | 51                   | 109  | 89           |
|                            | 0.7       | 1                          | 1                   | 76                   | 50   | 38           | 1                           | 1                   | 3                    | 259   | 380          | 126                         | 126                 | 51                   | 77   | 89           |
|                            | 0.6       | 1                          | 1                   | 76                   | 65   | 40           | 13                          | 17                  | 62                   | 1774  | 2384         | 126                         | 126                 | 51                   | 62   | 87           |
|                            | 0.5       | 22                         | 22                  | 78                   | 66   | 47           | 103                         | 109                 | 3072                 | 9612  | 9743         | 105                         | 105                 | 49                   | 61   | 80           |
| Barbeque (test set-13)     | 1         | 1                          | 1                   | 14                   | 1    | 1            | 0                           | 0                   | 1                    | 0     | 0            | 18                          | 18                  | 5                    | 18   | 18           |
|                            | 0.9       | 1                          | 1                   | 14                   | 1    | 7            | 0                           | 0                   | 1                    | 0     | 2            | 18                          | 18                  | 5                    | 18   | 12           |
|                            | 0.8       | 1                          | 1                   | 14                   | 7    | 9            | 0                           | 0                   | 2                    | 2     | 17           | 18                          | 18                  | 5                    | 12   | 10           |
|                            | 0.7       | 1                          | 1                   | 16                   | 14   | 10           | 0                           | 0                   | 4                    | 24    | 172          | 18                          | 18                  | 3                    | 5    | 9            |
|                            | 0.6       | 2                          | 2                   | 16                   | 16   | 12           | 1                           | 2                   | 43                   | 892   | 1526         | 17                          | 17                  | 3                    | 3    | 7            |
|                            | 0.5       | 8                          | 8                   | 18                   | 16   | 12           | 20                          | 23                  | 251                  | 7543  | 8478         | 11                          | 11                  | 1                    | 3    | 7            |
| Restaurant (test set-14)   | 1         | 13                         | 13                  | 474                  | 15   | 13           | 0                           | 0                   | 0                    | 0     | 0            | 760                         | 760                 | 299                  | 758  | 760          |
|                            | 0.9       | 22                         | 22                  | 477                  | 22   | 105          | 0                           | 0                   | 0                    | 0     | 0            | 751                         | 751                 | 296                  | 751  | 668          |
|                            | 0.8       | 23                         | 23                  | 496                  | 129  | 135          | 0                           | 0                   | 17                   | 1     | 54           | 750                         | 750                 | 277                  | 644  | 638          |
|                            | 0.7       | 25                         | 25                  | 499                  | 196  | 151          | 0                           | 0                   | 869                  | 150   | 649          | 748                         | 748                 | 274                  | 577  | 622          |
|                            | 0.6       | 55                         | 56                  | 502                  | 300  | 210          | 2                           | 2                   | 954                  | 2996  | 5929         | 718                         | 717                 | 271                  | 473  | 563          |
|                            | 0.5       | 189                        | 189                 | 506                  | 512  | 460          | 42                          | 54                  | 1498                 | 16283 | 17722        | 584                         | 584                 | 267                  | 261  | 313          |

| Keywords               | Threshold | Total true positive result |                     |                      |      |              | Total false positive result |                     |                      |       |              | Total false negative result |                     |                      |      |              |
|------------------------|-----------|----------------------------|---------------------|----------------------|------|--------------|-----------------------------|---------------------|----------------------|-------|--------------|-----------------------------|---------------------|----------------------|------|--------------|
|                        |           | Levenshtein                | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro  | Jaro-Winkler | Levenshtein                 | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Kahvila (test set-15)  | 1         | 4                          | 4                   | 171                  | 4    | 4            | 0                           | 0                   | 0                    | 0     | 0            | 586                         | 586                 | 419                  | 586  | 586          |
|                        | 0.9       | 4                          | 4                   | 175                  | 4    | 14           | 0                           | 0                   | 0                    | 0     | 0            | 586                         | 586                 | 415                  | 586  | 576          |
|                        | 0.8       | 4                          | 4                   | 183                  | 31   | 84           | 0                           | 0                   | 0                    | 0     | 25           | 586                         | 586                 | 407                  | 559  | 506          |
|                        | 0.7       | 4                          | 4                   | 229                  | 99   | 88           | 0                           | 0                   | 2                    | 113   | 387          | 586                         | 586                 | 361                  | 491  | 502          |
|                        | 0.6       | 5                          | 6                   | 229                  | 145  | 111          | 2                           | 3                   | 29                   | 1663  | 2679         | 585                         | 584                 | 361                  | 445  | 479          |
|                        | 0.5       | 22                         | 23                  | 235                  | 276  | 233          | 18                          | 24                  | 505                  | 10707 | 10794        | 568                         | 567                 | 355                  | 314  | 357          |
| Sauna (test set-16)    | 1         | 23                         | 23                  | 129                  | 23   | 23           | 0                           | 0                   | 0                    | 0     | 0            | 117                         | 117                 | 11                   | 117  | 117          |
|                        | 0.9       | 23                         | 23                  | 129                  | 23   | 37           | 0                           | 0                   | 0                    | 0     | 0            | 117                         | 117                 | 11                   | 117  | 103          |
|                        | 0.8       | 23                         | 23                  | 133                  | 42   | 61           | 0                           | 0                   | 44                   | 19    | 46           | 117                         | 117                 | 7                    | 98   | 79           |
|                        | 0.7       | 23                         | 23                  | 134                  | 65   | 64           | 0                           | 0                   | 76                   | 534   | 656          | 117                         | 117                 | 6                    | 75   | 76           |
|                        | 0.6       | 29                         | 29                  | 136                  | 80   | 70           | 38                          | 39                  | 537                  | 3936  | 3198         | 111                         | 111                 | 4                    | 60   | 70           |
|                        | 0.5       | 42                         | 45                  | 136                  | 115  | 92           | 87                          | 94                  | 2419                 | 14978 | 11721        | 98                          | 95                  | 4                    | 25   | 48           |
| Hotelli (test set-17)  | 1         | 1                          | 1                   | 66                   | 1    | 1            | 0                           | 0                   | 1                    | 0     | 0            | 358                         | 358                 | 293                  | 358  | 358          |
|                        | 0.9       | 1                          | 1                   | 67                   | 7    | 15           | 0                           | 0                   | 3                    | 0     | 0            | 358                         | 358                 | 292                  | 352  | 344          |
|                        | 0.8       | 1                          | 1                   | 70                   | 32   | 159          | 0                           | 0                   | 11                   | 3     | 19           | 358                         | 358                 | 289                  | 327  | 200          |
|                        | 0.7       | 7                          | 7                   | 349                  | 136  | 162          | 0                           | 0                   | 36                   | 103   | 483          | 352                         | 352                 | 10                   | 223  | 197          |
|                        | 0.6       | 9                          | 9                   | 350                  | 184  | 164          | 1                           | 2                   | 82                   | 1605  | 3127         | 350                         | 350                 | 9                    | 175  | 195          |
|                        | 0.5       | 38                         | 39                  | 354                  | 234  | 190          | 19                          | 29                  | 1261                 | 10563 | 10922        | 321                         | 320                 | 5                    | 125  | 169          |
| Apteekki (test set-18) | 1         | 2                          | 2                   | 8                    | 2    | 2            | 0                           | 0                   | 0                    | 0     | 0            | 8                           | 8                   | 2                    | 8    | 8            |
|                        | 0.9       | 2                          | 2                   | 8                    | 2    | 3            | 0                           | 0                   | 0                    | 0     | 0            | 8                           | 8                   | 2                    | 8    | 7            |
|                        | 0.8       | 2                          | 2                   | 8                    | 3    | 4            | 0                           | 0                   | 0                    | 7     | 4            | 8                           | 8                   | 2                    | 7    | 6            |
|                        | 0.7       | 2                          | 2                   | 9                    | 3    | 4            | 0                           | 0                   | 0                    | 99    | 239          | 8                           | 8                   | 1                    | 7    | 6            |
|                        | 0.6       | 3                          | 3                   | 9                    | 4    | 4            | 7                           | 8                   | 63                   | 1542  | 3492         | 7                           | 7                   | 1                    | 6    | 6            |
|                        | 0.5       | 3                          | 3                   | 10                   | 4    | 4            | 32                          | 54                  | 1067                 | 11494 | 13269        | 7                           | 7                   | 0                    | 6    | 6            |

Table 2a. Calculation of recall, precision, and F-score for different test sets.

| Keywords                           | Threshold | Recall      |                     |                      |      |              | Precision   |                     |                      |      |              | F-score     |                     |                      |      |              |
|------------------------------------|-----------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|
|                                    |           | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Chess (test set-1)                 | 1         | 1%          | 1%                  | 79%                  | 1%   | 1%           | 100%        | 100%                | 100%                 | 100% | 100%         | 2%          | 2%                  | 88%                  | 2%   | 2%           |
|                                    | 0.9       | 1%          | 1%                  | 79%                  | 1%   | 2%           | 100%        | 100%                | 99%                  | 100% | 67%          | 2%          | 2%                  | 88%                  | 2%   | 4%           |
|                                    | 0.8       | 1%          | 1%                  | 79%                  | 5%   | 9%           | 100%        | 100%                | 81%                  | 83%  | 31%          | 2%          | 2%                  | 80%                  | 9%   | 14%          |
|                                    | 0.7       | 1%          | 1%                  | 79%                  | 18%  | 11%          | 100%        | 100%                | 59%                  | 10%  | 1%           | 2%          | 2%                  | 67%                  | 12%  | 2%           |
|                                    | 0.6       | 1%          | 1%                  | 79%                  | 37%  | 11%          | 7%          | 100%                | 13%                  | 2%   | 0%           | 2%          | 2%                  | 22%                  | 3%   | 1%           |
|                                    | 0.5       | 3%          | 3%                  | 80%                  | 65%  | 20%          | 7%          | 6%                  | 5%                   | 1%   | 0%           | 4%          | 4%                  | 9%                   | 1%   | 1%           |
| Lenkkireitin maisemia (test set-2) | 1         | 72%         | 72%                 | 75%                  | 74%  | 74%          | 100%        | 100%                | 95%                  | 100% | 100%         | 83%         | 83%                 | 84%                  | 85%  | 85%          |
|                                    | 0.9       | 74%         | 74%                 | 75%                  | 75%  | 75%          | 100%        | 100%                | 92%                  | 100% | 78%          | 85%         | 85%                 | 83%                  | 86%  | 76%          |
|                                    | 0.8       | 74%         | 74%                 | 75%                  | 75%  | 75%          | 100%        | 100%                | 72%                  | 77%  | 68%          | 85%         | 85%                 | 74%                  | 76%  | 71%          |
|                                    | 0.7       | 75%         | 75%                 | 75%                  | 75%  | 75%          | 98%         | 98%                 | 67%                  | 62%  | 18%          | 85%         | 85%                 | 71%                  | 68%  | 30%          |
|                                    | 0.6       | 76%         | 75%                 | 75%                  | 75%  | 75%          | 91%         | 91%                 | 52%                  | 13%  | 3%           | 83%         | 83%                 | 61%                  | 22%  | 6%           |
|                                    | 0.5       | 94%         | 94%                 | 75%                  | 75%  | 75%          | 78%         | 77%                 | 30%                  | 2%   | 2%           | 85%         | 85%                 | 43%                  | 4%   | 3%           |
| Lenkkireitti (test set-3)          | 1         | 17%         | 17%                 | 19%                  | 17%  | 17%          | 100%        | 100%                | 100%                 | 100% | 100%         | 29%         | 29%                 | 31%                  | 29%  | 29%          |
|                                    | 0.9       | 17%         | 17%                 | 19%                  | 17%  | 92%          | 100%        | 100%                | 100%                 | 100% | 29%          | 29%         | 31%                 | 29%                  | 96%  |              |
|                                    | 0.8       | 17%         | 17%                 | 94%                  | 93%  | 94%          | 100%        | 100%                | 98%                  | 100% | 79%          | 29%         | 29%                 | 96%                  | 96%  | 86%          |
|                                    | 0.7       | 17%         | 18%                 | 94%                  | 94%  | 94%          | 97%         | 100%                | 96%                  | 75%  | 38%          | 29%         | 30%                 | 95%                  | 83%  | 54%          |
|                                    | 0.6       | 17%         | 18%                 | 94%                  | 94%  | 94%          | 68%         | 70%                 | 79%                  | 20%  | 9%           | 27%         | 29%                 | 86%                  | 32%  | 17%          |
|                                    | 0.5       | 93%         | 93%                 | 94%                  | 94%  | 94%          | 80%         | 80%                 | 45%                  | 3%   | 2%           | 86%         | 86%                 | 61%                  | 6%   | 5%           |
| Pizza (test set-4)                 | 1         | 3%          | 3%                  | 60%                  | 3%   | 3%           | 100%        | 100%                | 100%                 | 100% | 100%         | 6%          | 6%                  | 75%                  | 6%   | 6%           |
|                                    | 0.9       | 3%          | 3%                  | 60%                  | 3%   | 9%           | 100%        | 100%                | 63%                  | 100% | 90%          | 6%          | 6%                  | 61%                  | 6%   | 17%          |
|                                    | 0.8       | 3%          | 3%                  | 100%                 | 13%  | 37%          | 75%         | 75%                 | 72%                  | 93%  | 54%          | 6%          | 6%                  | 84%                  | 23%  | 44%          |
|                                    | 0.7       | 3%          | 3%                  | 100%                 | 41%  | 40%          | 75%         | 75%                 | 72%                  | 45%  | 20%          | 6%          | 6%                  | 83%                  | 43%  | 26%          |
|                                    | 0.6       | 3%          | 3%                  | 100%                 | 56%  | 40%          | 10%         | 10%                 | 65%                  | 9%   | 4%           | 5%          | 5%                  | 79%                  | 15%  | 7%           |
|                                    | 0.5       | 11%         | 11%                 | 100%                 | 67%  | 41%          | 23%         | 21%                 | 43%                  | 1%   | 1%           | 15%         | 15%                 | 60%                  | 2%   | 2%           |
| Lounas (test set-5)                | 1         | 3%          | 3%                  | 86%                  | 3%   | 3%           | 100%        | 100%                | 100%                 | 100% | 100%         | 7%          | 7%                  | 93%                  | 7%   | 7%           |
|                                    | 0.9       | 3%          | 3%                  | 86%                  | 3%   | 4%           | 100%        | 100%                | 100%                 | 100% | 100%         | 7%          | 7%                  | 93%                  | 7%   | 8%           |
|                                    | 0.8       | 3%          | 3%                  | 89%                  | 6%   | 30%          | 100%        | 100%                | 96%                  | 58%  | 63%          | 7%          | 7%                  | 92%                  | 11%  | 40%          |
|                                    | 0.7       | 3%          | 3%                  | 89%                  | 39%  | 37%          | 100%        | 100%                | 96%                  | 12%  | 7%           | 7%          | 7%                  | 92%                  | 19%  | 12%          |
|                                    | 0.6       | 3%          | 3%                  | 89%                  | 58%  | 47%          | 57%         | 50%                 | 50%                  | 3%   | 2%           | 7%          | 7%                  | 64%                  | 5%   | 4%           |
|                                    | 0.5       | 12%         | 12%                 | 100%                 | 96%  | 63%          | 8%          | 8%                  | 10%                  | 1%   | 1%           | 10%         | 9%                  | 18%                  | 2%   | 1%           |
| Marathon (test set-6)              | 1         | 0%          | 0%                  | 62%                  | 0%   | 0%           | null        | null                | 100%                 | null | null         | null        | null                | 76%                  | null | null         |
|                                    | 0.9       | 0%          | 0%                  | 62%                  | 0%   | 8%           | null        | null                | 100%                 | null | 100%         | null        | null                | 76%                  | null | 15%          |
|                                    | 0.8       | 0%          | 0%                  | 62%                  | 10%  | 12%          | null        | null                | 62%                  | 83%  | 36%          | null        | null                | 62%                  | 17%  | 18%          |
|                                    | 0.7       | 0%          | 0%                  | 62%                  | 18%  | 12%          | null        | null                | 59%                  | 20%  | 6%           | null        | null                | 60%                  | 19%  | 8%           |
|                                    | 0.6       | 5%          | 5%                  | 62%                  | 34%  | 13%          | 87%         | 88%                 | 49%                  | 4%   | 1%           | 9%          | 10%                 | 55%                  | 8%   | 2%           |
|                                    | 0.5       | 18%         | 18%                 | 62%                  | 37%  | 24%          | 46%         | 44%                 | 16%                  | 1%   | 0%           | 26%         | 25%                 | 26%                  | 2%   | 1%           |
| Swimming (test set-7)              | 1         | 1%          | 1%                  | 57%                  | 1%   | 1%           | 100%        | 100%                | 100%                 | 100% | 100%         | 2%          | 2%                  | 73%                  | 2%   | 2%           |
|                                    | 0.9       | 1%          | 1%                  | 57%                  | 2%   | 12%          | 100%        | 100%                | 100%                 | 100% | 100%         | 2%          | 2%                  | 73%                  | 3%   | 22%          |
|                                    | 0.8       | 1%          | 1%                  | 57%                  | 14%  | 22%          | 100%        | 100%                | 99%                  | 90%  | 100%         | 2%          | 2%                  | 72%                  | 24%  | 36%          |
|                                    | 0.7       | 2%          | 2%                  | 59%                  | 28%  | 46%          | 100%        | 100%                | 91%                  | 28%  | 30%          | 3%          | 3%                  | 71%                  | 28%  | 36%          |
|                                    | 0.6       | 15%         | 15%                 | 61%                  | 43%  | 50%          | 87%         | 83%                 | 82%                  | 6%   | 5%           | 26%         | 26%                 | 70%                  | 10%  | 9%           |
|                                    | 0.5       | 27%         | 27%                 | 100%                 | 69%  | 60%          | 28%         | 28%                 | 42%                  | 1%   | 1%           | 28%         | 28%                 | 59%                  | 3%   | 2%           |

| Keywords                   | Threshold | Recall      |                     |                      |      |              | Precision   |                     |                      |      |              | F-score     |                     |                      |      |              |     |
|----------------------------|-----------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|-----|
|                            |           | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |     |
| Beach (test set-8)         | 1         | 27%         | 27%                 | 100%                 | 28%  | 27%          | 100%        | 100%                | 100%                 | 100% | 100%         | 100%        | 42%                 | 42%                  | 100% | 44%          | 42% |
|                            | 0.9       | 27%         | 27%                 | 100%                 | 28%  | 33%          | 100%        | 100%                | 100%                 | 100% | 100%         | 42%         | 42%                 | 100%                 | 44%  | 50%          |     |
|                            | 0.8       | 28%         | 28%                 | 100%                 | 36%  | 49%          | 93%         | 93%                 | 95%                  | 94%  | 90%          | 43%         | 43%                 | 97%                  | 52%  | 64%          |     |
|                            | 0.7       | 28%         | 28%                 | 100%                 | 52%  | 49%          | 93%         | 93%                 | 90%                  | 75%  | 57%          | 43%         | 43%                 | 95%                  | 61%  | 53%          |     |
|                            | 0.6       | 28%         | 28%                 | 100%                 | 56%  | 50%          | 88%         | 88%                 | 61%                  | 18%  | 17%          | 43%         | 43%                 | 76%                  | 27%  | 25%          |     |
|                            | 0.5       | 35%         | 36%                 | 100%                 | 61%  | 56%          | 86%         | 84%                 | 32%                  | 3%   | 4%           | 50%         | 50%                 | 49%                  | 5%   | 7%           |     |
| Kuhasaló King (test set-9) | 1         | 0%          | 0%                  | 61%                  | 0%   | 0%           | null        | null                | 100%                 | null | null         | null        | null                | 76%                  | null | null         |     |
|                            | 0.9       | 0%          | 0%                  | 61%                  | 0%   | 53%          | null        | null                | 100%                 | null | 100%         | null        | null                | 76%                  | null | 69%          |     |
|                            | 0.8       | 0%          | 0%                  | 61%                  | 56%  | 75%          | null        | null                | 100%                 | 95%  | 57%          | null        | null                | 76%                  | 70%  | 65%          |     |
|                            | 0.7       | 0%          | 0%                  | 61%                  | 78%  | 75%          | null        | null                | 17%                  | 35%  | 3%           | null        | null                | 26%                  | 48%  | 6%           |     |
|                            | 0.6       | 42%         | 42%                 | 97%                  | 86%  | 86%          | 100%        | 100%                | 18%                  | 1%   | 1%           | 59%         | 59%                 | 30%                  | 3%   | 1%           |     |
|                            | 0.5       | 61%         | 61%                 | 97%                  | 86%  | 86%          | 88%         | 79%                 | 8%                   | 0%   | 0%           | 72%         | 69%                 | 16%                  | 0%   | 0%           |     |
| Kuntorasit (test set-10)   | 1         | 11%         | 11%                 | 81%                  | 11%  | 11%          | 100%        | 100%                | 99%                  | 100% | 100%         | 20%         | 20%                 | 89%                  | 20%  | 20%          |     |
|                            | 0.9       | 11%         | 11%                 | 85%                  | 11%  | 33%          | 100%        | 100%                | 99%                  | 100% | 96%          | 20%         | 20%                 | 91%                  | 20%  | 50%          |     |
|                            | 0.8       | 11%         | 11%                 | 89%                  | 61%  | 75%          | 100%        | 100%                | 96%                  | 94%  | 75%          | 20%         | 20%                 | 92%                  | 74%  | 75%          |     |
|                            | 0.7       | 15%         | 16%                 | 90%                  | 80%  | 79%          | 92%         | 100%                | 88%                  | 54%  | 13%          | 25%         | 28%                 | 89%                  | 64%  | 22%          |     |
|                            | 0.6       | 22%         | 24%                 | 91%                  | 87%  | 82%          | 92%         | 88%                 | 52%                  | 6%   | 2%           | 35%         | 38%                 | 66%                  | 11%  | 5%           |     |
|                            | 0.5       | 38%         | 52%                 | 99%                  | 96%  | 90%          | 74%         | 75%                 | 10%                  | 1%   | 1%           | 50%         | 61%                 | 19%                  | 2%   | 2%           |     |
| Hölkä (test set-11)        | 1         | 0%          | 0%                  | 44%                  | 0%   | 0%           | null        | null                | 100%                 | null | null         | null        | null                | 61%                  | null | null         |     |
|                            | 0.9       | 0%          | 0%                  | 44%                  | 0%   | 0%           | null        | null                | 100%                 | null | null         | null        | null                | 61%                  | null | null         |     |
|                            | 0.8       | 0%          | 0%                  | 63%                  | 0%   | 0%           | null        | null                | 85%                  | 0%   | 0%           | null        | null                | 72%                  | null | null         |     |
|                            | 0.7       | 0%          | 0%                  | 63%                  | 20%  | 0%           | null        | 0%                  | 75%                  | 37%  | 0%           | null        | null                | 69%                  | 26%  | null         |     |
|                            | 0.6       | 0%          | 2%                  | 63%                  | 33%  | 0%           | 0%          | 10%                 | 73%                  | 11%  | 0%           | null        | 4%                  | 67%                  | 17%  | null         |     |
|                            | 0.5       | 12%         | 12%                 | 65%                  | 58%  | 5%           | 31%         | 7%                  | 26%                  | 3%   | 0%           | 18%         | 9%                  | 37%                  | 5%   | 0%           |     |
| Skiing (test set-12)       | 1         | 1%          | 1%                  | 60%                  | 1%   | 1%           | 100%        | 100%                | 100%                 | 100% | 100%         | 2%          | 2%                  | 75%                  | 2%   | 2%           |     |
|                            | 0.9       | 1%          | 1%                  | 60%                  | 1%   | 13%          | 100%        | 100%                | 100%                 | 100% | 100%         | 2%          | 2%                  | 75%                  | 2%   | 24%          |     |
|                            | 0.8       | 1%          | 1%                  | 60%                  | 14%  | 30%          | 100%        | 100%                | 100%                 | 72%  | 66%          | 2%          | 2%                  | 75%                  | 24%  | 41%          |     |
|                            | 0.7       | 1%          | 1%                  | 60%                  | 39%  | 30%          | 50%         | 50%                 | 96%                  | 16%  | 9%           | 2%          | 2%                  | 74%                  | 23%  | 14%          |     |
|                            | 0.6       | 1%          | 1%                  | 60%                  | 51%  | 31%          | 7%          | 6%                  | 55%                  | 4%   | 2%           | 1%          | 1%                  | 57%                  | 7%   | 3%           |     |
|                            | 0.5       | 17%         | 17%                 | 61%                  | 52%  | 37%          | 18%         | 17%                 | 2%                   | 1%   | 0%           | 17%         | 17%                 | 5%                   | 1%   | 1%           |     |
| Barbeque (test set-13)     | 1         | 5%          | 5%                  | 74%                  | 5%   | 5%           | 100%        | 100%                | 93%                  | 100% | 100%         | 10%         | 10%                 | 82%                  | 10%  | 10%          |     |
|                            | 0.9       | 5%          | 5%                  | 74%                  | 5%   | 37%          | 100%        | 100%                | 93%                  | 100% | 78%          | 10%         | 10%                 | 82%                  | 10%  | 50%          |     |
|                            | 0.8       | 5%          | 5%                  | 74%                  | 37%  | 47%          | 100%        | 100%                | 88%                  | 78%  | 35%          | 10%         | 10%                 | 80%                  | 50%  | 40%          |     |
|                            | 0.7       | 5%          | 5%                  | 84%                  | 74%  | 53%          | 100%        | 100%                | 80%                  | 37%  | 5%           | 10%         | 10%                 | 82%                  | 49%  | 10%          |     |
|                            | 0.6       | 11%         | 11%                 | 84%                  | 84%  | 63%          | 67%         | 50%                 | 27%                  | 2%   | 1%           | 18%         | 17%                 | 41%                  | 3%   | 2%           |     |
|                            | 0.5       | 42%         | 42%                 | 95%                  | 84%  | 63%          | 29%         | 26%                 | 7%                   | 0%   | 0%           | 34%         | 32%                 | 13%                  | 0%   | 0%           |     |
| Restaurant (test set-14)   | 1         | 2%          | 2%                  | 61%                  | 2%   | 2%           | 100%        | 100%                | 100%                 | 100% | 100%         | 3%          | 3%                  | 76%                  | 4%   | 3%           |     |
|                            | 0.9       | 3%          | 3%                  | 62%                  | 3%   | 14%          | 100%        | 100%                | 100%                 | 100% | 100%         | 6%          | 6%                  | 76%                  | 6%   | 24%          |     |
|                            | 0.8       | 3%          | 3%                  | 64%                  | 17%  | 17%          | 100%        | 100%                | 97%                  | 99%  | 71%          | 6%          | 6%                  | 77%                  | 29%  | 28%          |     |
|                            | 0.7       | 3%          | 3%                  | 65%                  | 25%  | 20%          | 100%        | 100%                | 36%                  | 57%  | 19%          | 6%          | 6%                  | 47%                  | 35%  | 19%          |     |
|                            | 0.6       | 7%          | 7%                  | 65%                  | 39%  | 27%          | 96%         | 97%                 | 34%                  | 9%   | 3%           | 13%         | 13%                 | 45%                  | 15%  | 6%           |     |
|                            | 0.5       | 24%         | 24%                 | 65%                  | 66%  | 60%          | 82%         | 78%                 | 25%                  | 3%   | 3%           | 38%         | 37%                 | 36%                  | 6%   | 5%           |     |



| Keywords               | Threshold | Recall      |                     |                      |      |              | Precision   |                     |                      |      |              | F-score     |                     |                      |      |              |
|------------------------|-----------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|-------------|---------------------|----------------------|------|--------------|
|                        |           | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler | Levenshtein | Damerau-Levenshtein | Smith Waterman-Gotoh | Jaro | Jaro-Winkler |
| Kahvila (test set-15)  | 1         | 1%          | 1%                  | 29%                  | 1%   | 1%           | 100%        | 100%                | 100%                 | 100% | 100%         | 1%          | 1%                  | 45%                  | 1%   | 1%           |
|                        | 0.9       | 1%          | 1%                  | 30%                  | 1%   | 2%           | 100%        | 100%                | 100%                 | 100% | 100%         | 1%          | 1%                  | 46%                  | 1%   | 5%           |
|                        | 0.8       | 1%          | 1%                  | 31%                  | 5%   | 14%          | 100%        | 100%                | 100%                 | 100% | 77%          | 1%          | 1%                  | 47%                  | 10%  | 24%          |
|                        | 0.7       | 1%          | 1%                  | 39%                  | 17%  | 15%          | 100%        | 100%                | 99%                  | 47%  | 19%          | 1%          | 1%                  | 56%                  | 25%  | 17%          |
|                        | 0.6       | 1%          | 1%                  | 39%                  | 25%  | 19%          | 71%         | 67%                 | 89%                  | 8%   | 4%           | 2%          | 2%                  | 54%                  | 12%  | 7%           |
|                        | 0.5       | 4%          | 4%                  | 40%                  | 47%  | 39%          | 55%         | 49%                 | 32%                  | 3%   | 2%           | 7%          | 7%                  | 35%                  | 5%   | 4%           |
| Sauna (test set-16)    | 1         | 16%         | 16%                 | 92%                  | 16%  | 16%          | 100%        | 100%                | 100%                 | 100% | 100%         | 28%         | 28%                 | 96%                  | 28%  | 28%          |
|                        | 0.9       | 16%         | 16%                 | 92%                  | 16%  | 26%          | 100%        | 100%                | 100%                 | 100% | 100%         | 28%         | 28%                 | 96%                  | 28%  | 42%          |
|                        | 0.8       | 16%         | 16%                 | 95%                  | 30%  | 44%          | 100%        | 100%                | 75%                  | 69%  | 57%          | 28%         | 28%                 | 84%                  | 42%  | 49%          |
|                        | 0.7       | 16%         | 16%                 | 96%                  | 46%  | 46%          | 100%        | 100%                | 64%                  | 11%  | 9%           | 28%         | 28%                 | 77%                  | 18%  | 15%          |
|                        | 0.6       | 21%         | 21%                 | 97%                  | 57%  | 50%          | 43%         | 43%                 | 20%                  | 2%   | 2%           | 28%         | 28%                 | 33%                  | 4%   | 4%           |
|                        | 0.5       | 30%         | 32%                 | 97%                  | 82%  | 66%          | 33%         | 32%                 | 5%                   | 1%   | 1%           | 31%         | 32%                 | 10%                  | 2%   | 2%           |
| Hotelli (test set-17)  | 1         | 0%          | 0%                  | 18%                  | 0%   | 0%           | 100%        | 100%                | 99%                  | 100% | 100%         | 1%          | 1%                  | 31%                  | 1%   | 1%           |
|                        | 0.9       | 0%          | 0%                  | 19%                  | 2%   | 4%           | 100%        | 100%                | 96%                  | 100% | 100%         | 1%          | 1%                  | 31%                  | 4%   | 8%           |
|                        | 0.8       | 0%          | 0%                  | 19%                  | 9%   | 44%          | 100%        | 100%                | 86%                  | 91%  | 89%          | 1%          | 1%                  | 32%                  | 16%  | 59%          |
|                        | 0.7       | 2%          | 2%                  | 97%                  | 38%  | 45%          | 100%        | 100%                | 91%                  | 57%  | 25%          | 4%          | 4%                  | 94%                  | 45%  | 32%          |
|                        | 0.6       | 3%          | 3%                  | 97%                  | 51%  | 46%          | 90%         | 82%                 | 81%                  | 10%  | 5%           | 5%          | 5%                  | 88%                  | 17%  | 9%           |
|                        | 0.5       | 11%         | 11%                 | 99%                  | 65%  | 53%          | 67%         | 57%                 | 22%                  | 2%   | 2%           | 18%         | 18%                 | 36%                  | 4%   | 3%           |
| Apteekki (test set-18) | 1         | 20%         | 20%                 | 80%                  | 20%  | 20%          | 100%        | 100%                | 100%                 | 100% | 100%         | 33%         | 33%                 | 89%                  | 33%  | 33%          |
|                        | 0.9       | 20%         | 20%                 | 80%                  | 20%  | 30%          | 100%        | 100%                | 100%                 | 100% | 100%         | 33%         | 33%                 | 89%                  | 33%  | 46%          |
|                        | 0.8       | 20%         | 20%                 | 80%                  | 30%  | 40%          | 100%        | 100%                | 100%                 | 30%  | 50%          | 33%         | 33%                 | 89%                  | 30%  | 44%          |
|                        | 0.7       | 20%         | 20%                 | 90%                  | 30%  | 40%          | 100%        | 100%                | 100%                 | 3%   | 2%           | 33%         | 33%                 | 95%                  | 5%   | 3%           |
|                        | 0.6       | 30%         | 30%                 | 90%                  | 40%  | 40%          | 30%         | 27%                 | 13%                  | 0%   | 0%           | 30%         | 29%                 | 22%                  | 1%   | 0%           |
|                        | 0.5       | 30%         | 30%                 | 100%                 | 40%  | 40%          | 9%          | 5%                  | 1%                   | 0%   | 0%           | 13%         | 9%                  | 2%                   | 0%   | 0%           |

Table 3a. Inclusion searching for all keywords.

| Method    | Keyword               | Golden standard | True positive | False positive | False negative | Recall | Precision | F-score |
|-----------|-----------------------|-----------------|---------------|----------------|----------------|--------|-----------|---------|
| Inclusion | Chess                 | 108             | 85            | 0              | 23             | 78.7%  | 100.0%    | 88.1%   |
|           | Lenkkireitin maisemia | 391             | 293           | 0              | 98             | 74.9%  | 100.0%    | 85.7%   |
|           | Lenkkireitti          | 391             | 73            | 0              | 73             | 50.0%  | 100.0%    | 66.7%   |
|           | Pizza                 | 98              | 59            | 0              | 39             | 60.2%  | 100.0%    | 75.2%   |
|           | Lounas                | 115             | 99            | 0              | 16             | 86.1%  | 100.0%    | 92.5%   |
|           | Marathon              | 262             | 162           | 0              | 100            | 61.8%  | 100.0%    | 76.4%   |
|           | Swimming              | 131             | 72            | 0              | 59             | 55.0%  | 100.0%    | 70.9%   |
|           | Beach                 | 297             | 296           | 0              | 1              | 99.7%  | 100.0%    | 99.8%   |
|           | Kuhasalo King         | 36              | 22            | 0              | 14             | 61.1%  | 100.0%    | 75.9%   |
|           | Kuntorastit           | 150             | 121           | 0              | 29             | 80.7%  | 100.0%    | 89.3%   |
|           | Hölkä                 | 130             | 57            | 0              | 73             | 43.8%  | 100.0%    | 61.0%   |
|           | Skiing                | 127             | 76            | 0              | 51             | 59.8%  | 100.0%    | 74.9%   |
|           | Barbeque              | 19              | 14            | 0              | 5              | 73.7%  | 100.0%    | 84.8%   |
|           | Restaurant            | 733             | 473           | 0              | 260            | 64.5%  | 100.0%    | 78.4%   |
|           | Kahvila               | 590             | 171           | 0              | 419            | 29.0%  | 100.0%    | 44.9%   |
|           | Sauna                 | 140             | 129           | 0              | 11             | 92.1%  | 100.0%    | 95.9%   |
|           | Hotelli               | 359             | 62            | 0              | 297            | 17.3%  | 100.0%    | 29.5%   |
| Apteekki  | 10                    | 8               | 0             | 2              | 80.0%          | 100.0% | 88.9%     |         |