Detecting and Characterizing User Sessions in the Context of a Search Engine for Legislative Contents

Pedro Miguel de Almeida Gomes

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Bruno Emanuel da Graça Martins
Doutor Luís Miranda da Cruz

Examination Committee

Chairperson: Prof. José Luís Brinquete Borbinha
Supervisor: Prof. Bruno Emanuel da Graça Martins
Members of the Committee: Doutor Daniel Gomes

November 2019

Acknowledgements

I would like to thank Professor Bruno Emanuel da Graça Martins and Doutor Luís Miranda da Cruz for their support, knowledge, and motivation, through which they contributed to the work and publications that we have together.

I would like to praise the support given by my family, in particular, my parents, godmother, cousins, and sister in this challenge and all my friends and colleagues from the DRE project for constant support during the biggest obstacle I faced in my life.

Finally, I would also want to thank Gayo-Avello for making available an annotated corpus and all the work developed by him reported in Gayo-Avello (2009), and Hagen for the amazing work, in which we also exchanged some ideas about segmenting user session and missions in search engine query logs in general. The previous methods were merged and extended to tanking into consideration different limitations, for instance, rely on an efficient cascade approach based on temporal, lexical and semantic heuristics.

Pedro Miguel de Almeida Gomes

For my parents and sister

Resumo

A segmentação das interações do utilizador conforme registradas nos registos de queries de um motor de busca, de acordo com as necessidades de informações subjacentes (por exemplo, delimitando as sessões do utilizador), é importante para perceber as necessidades de informação e avaliar como elas são satisfeitas, para melhorar a qualidade dos rankings do motor de busca e para melhor direcionar conteúdo para determinados utilizadores. A maioria dos métodos anteriores usa julgamentos humanos para informar algoritmos de aprendizagem supervisionada e/ou usam limites globais de proximidade temporal e métricas simples de similaridade lexical. Esta dissertação apresenta um método não supervisionado para segmentar sessões do utilizador que aprimora o atual estado da arte, aproveitando heurísticas adicionais e métricas de similaridade derivadas de *word embeddings*. Eu, mais concretamente, estendi uma abordagem anterior baseada na combinação de medidas de similaridade temporal e lexical, integrando componentes de similaridade semântica que usam *FastText embeddings* pré-treinados. Com base no método de segmentação de sessões, esta dissertação também avança uma abordagem não supervisionada para detectar missões (ou seja, conjuntos de queries, não necessariamente contínuas, referindo-se à mesma necessidade de informação dentro de um padrão de comportamento multitarefa e/ou considerando objetivos hierárquicos). Eu reporto experiências com dois subconjuntos diferentes do conhecido dataset do AOL, ambos usados em estudos anteriores. O primeiro subconjunto contém um total de 10.235 queries, com 4.253 sessões, 2,4 queries por sessão e 215 utilizadores únicos, sendo usado para avaliar a eficácia do algoritmo na detecção de sessões. O segundo subconjunto foi usado para avaliar a eficácia do algoritmo na detecção de sessões e missões, contendo um total de 8.840 queries, com 2.881 sessões, 1.378 missões, 3,1 queries por sessão, 6,42 queries por missão e 127 utilizadores únicos. Os resultados atestam a eficácia dos métodos propostos, que superam um grande conjunto de *baselines*, correspondendo também a técnicas não supervisionadas. No entanto, um desafio específico está relacionado com o facto de que geralmente os logs não apresentam identificadores únicos de utilizador. Nesses casos, os logs podem apresentar queries de diferentes utilizadores (e consequentemente também de diferentes sessões) que aparecem intercaladas em ordem cronológica, todas associadas ao mesmo endereço IP (por exemplo, de um proxy comum da Internet). Para resolver este problema, aproveitei as informações do *user-agent* juntamente com o endereço IP. Com base no método de segmentação

de sessões, realizamos um estudo de caracterização para inferir a satisfação do utilizador no contexto de um motor de busca de conteúdos legislativos.

Abstract

Segmenting user interactions as registered in search engine query logs, according to the underlying information needs (e.g., delimiting user sessions), is important to perceive information needs and assess how they are satisfied, to enhance the quality of search engine rankings, and to better direct content to certain users. Most previous methods use human judgments to inform supervised learning algorithms, and/or use global thresholds on temporal proximity and on simple lexical similarity metrics. This dissertation presents an unsupervised method for segmenting user sessions that improves on the current state-of-art, leveraging additional heuristics and similarity metrics derived from word embeddings. I specifically extend a previous approach based on combining temporal and lexical similarity measurements, integrating semantic similarity components that use pre-trained FastText embeddings. Building on the session segmentation method, the dissertation also advances an unsupervised approach for detecting search missions (i.e., sets of queries, not necessarily continuous, referring to the same information need inside a multitasking behavior pattern, and/or considering hierarchical goals). I report on experiments with two different subsets from the well-known AOL query dataset, both used in previous studies. The first subset contains a total of 10,235 queries, with 4,253 sessions, 2.4 queries per session, and 215 unique users, being used to evaluate the effectiveness of the algorithm for detecting sessions. The second subset was used to evaluate the effectiveness of the algorithm for detecting search sessions and missions, containing a total of 8,840 queries, with 2,881 sessions, 1,378 missions, 3.1 queries per session, 6.42 queries per mission, and 127 unique users. The results attest to the effectiveness of the proposed methods, which outperform a large set of baselines also corresponding to unsupervised techniques. However, one particular challenge relates to the fact that query logs often do not feature unique user identifiers. In these cases, the logs may feature queries from different users (and consequently also from different sessions) appearing interleaved in chronological order, all associated to the same IP address (e.g., from a common Internet proxy). To solve this problem, I leverage user-agent information together with IP address. Building on the session segmentation method, I made a characterization study for infer user satisfaction in the context of a search engine for legislative contents.

Palavras Chave
Keywords

Palavras Chave

Análise de registos de queries de motores de busca

Detecção de sessão do utilizador

Detecção de missão do utilizador

Medidas de semelhança de uma cadeia de caracteres

Word embeddings

Estudo de caracterização de um motor de busca português para conteúdos legislativos

Keywords

Analysis of search engine query logs

User session detection

User mission detection

String similarity metrics

Word embeddings

Characterization study from a portuguese search engine for legislative contents

Table of Contents

List of Figures

List of Tables

1 Introduction

In the context of user interactions with web search engines, the notion of session is critical to the study of user habits and intentions when using these systems. In brief, a session is a sequence of activities performed by one individual to satisfy an information need, regardless of the elapsed time, number of interactions with the system, or the existence of interruptions on these interactions. Identifying user sessions is important to understand a search engine's effectiveness in suggesting content pointers to user's searches, with several previous studies suggesting that by studying the properties of these sessions (e.g., clicks and dwell-time on search results) one can evaluate system quality and predict general user satisfaction (Feild et al., 2010; Hassan et al., 2013a; Jiang et al., 2015a; Kim et al., 2014a; Mehrotra et al., 2017a; Mayr and Kacem, 2017).

According to Hagen et al. (2011), there are two possible scenarios where the identification of user sessions can have a beneficial result: online, where it can help the search engine to present better results or to suggest queries that other users submitted in similar situations; and offline, where the identification of sessions within information collected from query logs gives information about the behavior of users, supporting the evaluation of their satisfaction. Typically, a user session ends when the user has satisfied his information need, or when the user decided to stop (i.e., the user leaves the system or proceeds to a new information need). The correct identification of user search sessions is still a challenging problem, which entails the following main issues:

- Session boundaries can be highly ambiguous. The most commonly used approach to identify user sessions involves grouping all interactions from the same user that happened within a constrained temporal interval. This approach is simple and efficient, having been reported to achieve a confidence level of approximately 70 percent on Web search logs (Jones and Klinkner, 2008). However, this approach also introduces noise, since it does not take into account activity breaks or sessions that are very long. One of the difficulties in using a global temporal threshold is that true session intervals usually have a smooth distribution, and it is almost guaranteed that longer sessions will be handled incorrectly by setting a temporal threshold. Nonetheless, previous proposals attempt to address this issue through variable thresholds depending on the user (Mehrzadi and Feitelson, 2012).

- It can be difficult to use query terms to infer if two queries from the same user belong to the same session, e.g. when they do not have search terms in common but correspond to the same underlying topic. For example, if one user searches for *iPhone* and then searches for *Apple*, there is no direct way to assess the underlying similarity between these query terms, although recent proposals have addressed the issue through semantic representations for the terms employed in user queries (e.g., through distributional semantic models and word embeddings methods (Bojanowski et al., 2017; Gomes et al., 2019)).

- Query logs often do not feature unique user identifiers, instead containing only cookie identifiers in a fraction of the records, plus information on source IP addresses and user-agents (i.e., identifiers for the type of Web browser in which the query was submitted). In these cases, the logs may feature queries from different users (and consequently also from different sessions) appearing interleaved in chronological order, all associated to the same IP address (e.g., from a common Internet proxy). Leveraging user-agent information together with IP addresses can partially address this problem, and approaches that extend the notion of session in order to better handle activity breaks can also be beneficial.

- It can be challenging to identify multitasking behavior or hierarchical sub-tasks with different purposes, where the related interactions will not appear as continuous in the search log. For instance, when planning vacations users will likely perform different queries corresponding to booking flights or hotels. In these cases, the queries can perhaps be grouped according to different sessions, even though they relate to the same underlying task or encompassing mission. The queries relating to a same sub-task may not appear continuous in the query log, and considering semantic similarity will not be enough to properly group queries according to search missions.

- Efficiency is a major concern, either in the context of timely online identification of user sessions, or in the processing of very large query logs for segmenting session and mission boundaries. Depending on the circumstances, there is usually a trade-off between algorithm accuracy and efficiency.

Tackling the aforementioned challenges usually involves a combination of different and carefully selected heuristics: temporal approaches (e.g., relying on a global threshold) have the advantage of simplicity and efficiency, although they also have problems with accuracy, whereas heuristics capturing lexical/topical similarity (e.g., methods based on distributional semantics) can have a high accuracy, although also a lower computational efficiency. By combining these general methods, I aim at achieving a useful trade-off between effectiveness and efficiency.

This dissertation also advances a characterization study based on the query log of a national search engine for legislative content, in the context of an ongoing technology transfer project. The study has the aim of understanding user behavior to improve the search and navigation in *Diário da República Eletrónico*[1] (DRE). *DRE* is a digital library for Portuguese laws, regulations and legal acts. To better understand the intentions of the users when using the system it is crucial to divide the logs into a group of interactions that correspond to the same session. Building on the session segmentation method, I present an analysis in the search engine query log from *DRE* to understand the search engine's effectiveness in suggesting content pointers for user searches. Thus, I was able to evaluate system quality and infer user satisfaction. However, these tasks are still a challenging problem, which entails the following main issues:

- The previous and more relevant studies only deal with small datasets (e.g., the first study made by Jansen et al. (1998) only had 51,473 queries). Therefore, with small a dataset, it is not possible to discover patterns of searches and seasonality. For instance, it would be interesting to see the most common queries at each season (e.g., summer, winter) and in a specific period in the legislative calendar.

- There are many studies focused on perceiving user behavior in certain areas: (i) log mining (e.g., Jansen et al. (1998)), and (ii) think-aloud protocols (e.g., Hölscher and Strube (2000)). However, several Web search engines are specialized in specific topics (e.g. legislative content), which can be highly ambiguous and I will never have the full picture of each user since different users can be exposed to a different source of bias (e.g., academic degree or area of expertise).

- For certain information needs, it is necessary to make the search more targeted to certain documents with more relevance. However, different search engine rankings order the results in a different way for a specific topic. For instance, if the user searches for *work law* the *DRE* system ranking will organize the results taking into account the type of document with more relevance. In this case, *decree-law* will be shown first. However, for some cases, if the user searches for a specific document such as *notice* and a document identifier, *DRE* search engine ranking, in some cases, will give more relevance to other types of documents (e.g., *decree-law*).

[1]`https://dre.pt`

- For specific topics or documents, to satisfy certain information needs it is necessary to do a long sequence of clicks. The main reason is because not all information that the user needs are in one specific document on which a user clicked. Hence, the user may need to click on other documents that are referenced in the first document. Thus, although a specific document clicked on can be relevant and the user finds the information that is looking for, the user commonly needs to click on another document in *DRE* to completely satisfy information needs. Such behavior can have an influence on the duration of the sessions (i.e., making a longer sequence of clicks will increase the number of interactions and the duration of the session) and user satisfaction since long sequence of clicks or different clicks for the same query may not always mean dissatisfaction.

- Due to the openness of web search engines and the potential profit inherent, many companies can use bots to manipulate the search result pages and/or extract documents and information. In brief, bots are programs that introduce queries and clicks automatically into the web search engines to extract relevant information or to consume resources, impacting the overall traffic for the search engines. It is crucial to detect and separate these interactions since it hinders (i) user experience (e.g., large traffic volume increase the response time of the search engine), and (ii) the analysis of user behavior (e.g., ranking algorithms learned through user clicks produce by bots).

The analyze presented in this dissertation will be useful to improve the ranking models and/or functions of the *DRE* search engine and understand user behavior when using the system. The data shown in this dissertation was collected from real users in real situations. However, in order to have a full picture of what the users do when they use the system, it is necessary to have a mechanism that captures what they really do (e.g., mouse movement). One possible solution could be running a set of tests over target users to better understand all the difficulties; however, these tests are expensive and can disturb the search process. Without this information, I will use the approach of analyzing the query log files of the search engines since it does not require any extra effort from the users and it is easy to collect. Thus, all the conclusions made in this dissertation are general, without the context of each user and not specific for an age group or academic degree.

1.1 Thesis Proposal

This work aims to investigate the use of unsupervised approaches to solving the session and mission detection problem, considering a trade-off between algorithm accuracy and computational efficiency. Previous state-of-the-art methods just consider temporal and lexical heuristics without considering semantic heuristics, and despite the interesting results, supervised approaches require training data, thus being harder to generalize to new application domains. One of the main goals of this thesis is to detect user sessions and search mission in the query logs of different systems, using suitable heuristics that operate directly on the logs, and avoiding the training of a classifier.

This work presents the results of a new unsupervised approach combining multiple heuristics, evaluating it against a set of baselines that covers the current state-of-the-art. An ablation analysis was also considered, checking the impact of temporal, lexical, and semantic similarity heuristics, in the overall method that integrates them. Taking inspiration on different studies (Gayo-Avello, 2009; Hagen et al., 2011, 2013), I extend the previous work by creating a new cascade approach (i.e., an incremental procedure based on a sequence of heuristics), extending the geometric method proposed by Gayo-Avello (2009) and also drawing on ideas from Hagen et al. (2011, 2013) by adding a new step to compute separate semantic similarity measurements, using the pre-trained FastText embeddings (Bojanowski et al., 2017).

Using the proposed methods, I also developed a preliminary characterization study that analyzed a large query log from a national search engine focused on legislative contents, to better understand user behavior on *Diário da República Eletrónico*. This work will be important to perceive what the users' information needs are and to assess whether the users are satisfied, in order to enhance the quality of search engine results, and to better understand a search engine's effectiveness in suggesting content pointers for user searches.

1.2 Contributions

In brief, the main contributions of this thesis are as follows:

- The proposal of a new unsupervised method for segmenting user sessions that improves upon a previous proposal by Gayo-Avello (2009), which is based on a geometric interpretation for how temporal and lexical similarity measurements can be effectively combined. For comparing consecutive queries from the same user, my approach uses a geometric method to decide if the more recent query belongs to the same session of the first, combining a user-specific temporal threshold with a lexical similarity value computed from n-gram overlaps. If a reliable decision cannot be made with this procedure, I compute separate semantic similarity measurements, e.g. with basis on pre-trained FastText embeddings (Bojanowski et al., 2017). The FastText approach represents tokens based on their character n-grams, and is capable of producing representations for query terms that were not seen during model training.

- Building on the session segmentation method, I also present an unsupervised approach for segmenting search missions, taking into consideration multitasking behavior and hierarchical goals for consecutive and non-consecutive user sessions. For efficiently assessing if sessions from the same user belong to the same underlying search mission, my approach first relies on an improved version of the geometric method, comparing the last query q of session s and the first query q' of session s'. If the decision is not reliable, I then use a semantic similarity step based on pre-trained FastText embeddings (Bojanowski et al., 2017). Additionally, I present new evaluation methodology to evaluate approaches that group user session into search missions take into account predicted sessions, using B^3 clustering metrics proposed by Amigó et al. (2009);

- An analysis on a large query log of a national search engine to understand user behavior and infer user satisfaction on a search engine whose focus is on legislative contents. The study confirmed the usefulness of the analyzes to improve the ranking system to better suggest content pointers for user searches. In particular, the results reported in this dissertation suggest that: (i) users that submit queries with numeric terms (denoting document identifiers) normally click in the first positions, and (ii) users that submit queries without numeric terms normally click after the first page.

Part of the work reported in this dissertation was also presented in an article entitled *Segmenting User Sessions in Search Engine Query Logs Leveraging Word Embeddings*, which was submitted on the 23rd International Conference on Theory and Practice of Digital Libraries (Gomes et al., 2019). In this paper, we propose a new unsupervised method for segmenting user sessions that extends the geometric method from Gayo-Avello (2009) and also draws on ideas from Hagen et al. (2011), which it is discussed in this dissertation and in the aforementioned article. As a side contribution, we deliver a reproducibility package[2] with all the scripts used in the evaluation experiments that are reported on this paper.

[2] `https://github.com/PedroG1515/Segmenting-User-Sessions`

Additionally, the article was nominated for the best paper award and we were invited to submit in an International Journal of Digital Libraries (IJDL) special issue associated to TPDL'19, in which is necessary to expand upon the description of the work by providing depth and detail of the approach and/or results. We presented an article entitled *Segmenting User Sessions and Missions in Search Engine Query Logs by Leveraging Word Embeddings* for this special issue. The new article describes research efforts towards the development of unsupervised approaches for segmenting user interactions, as registered in search engine logs, according to the underlying information needs. We extend on the approach and results presented in the TPDL'19 paper by considering (a) segmentation according to user sessions and also according to search missions, and (b) experiments with a second dataset.

Currently in preparation, we are considering publish soon a new article based on the analysis made over the *DRE* query log on the Journal of the Association for Information Science and Technology (JASIST).

1.3 Structure of the Document

The rest of this document is organized as follows: Chapter 2 presents the fundamental concepts and previous work, covering both session identification and user satisfaction prediction. Chapter 3 begins by describing the proposed approaches for segmenting search session and mission and an ablation analysis to check the impact of temporal, lexical, and semantic similarity heuristics, in the overall method that integrates them. This chapter ends with an experimental evaluation of the proposed methods, detailing the datasets, the evaluation methodologies, and the obtained results. Then, Chapter 4 presents a characterization study with the logs from an ongoing project related to a search engine for legislative contents. Finally, Chapter 5 concludes this document by summarizing the main conclusions and presents possible future research.

2 Concepts and Related Work

This chapter presents fundamental concepts and related work on user session identification and prediction of user satisfaction. First, a brief review of fundamental concepts is provided in Section 2.1. This review is divided into search log analysis in Section 2.1.1, and string similarity measurement in Section 2.1.2. Then, related work in the areas is presented in Section 2.2, beginning with a description of baseline methods for detecting user sessions in Section 2.2.1, and then with a description of methods for predicting user satisfaction in Section 2.2.2.

2.1 Fundamental Concepts

This section presents a brief review of fundamental concepts regarding search log analysis and string similarity measurement.

2.1.1 Analyzing User Sessions on Search Logs

This section presents several definitions to better understand the context of identifying user sessions. We start by defining a set of basic concepts.

First, we have that **searching** is a way to solve information needs, e.g. to reach a desired site or to obtain a different resource of information (Rose and Levinson, 2004). A **search query**, according to Swanson (1977) is "[...] *a guess about the attributes a desired document is expected to have* [...] *the response of the system is then used to correct the initial guess for another try*". There are three basic types of search queries: **navigational search queries**, which are submitted with the intent of finding a particular document or resource; **informational search queries**, which cover needs for information in a broad topic (i.e., with the intent of learning more about a particular topic); and **transactional search queries**, which is indicate, an intent to complete a transaction.

On the other hand, a **successive search** concern the process of refining either the query or its objectives (Spink et al., 1998). According to Jansen and Spink (2006), a **search episode** can be defined as a single query with other underlying actions (e.g., clicking on a result, requesting more results, or giving relevance judgments) and **searching episodes** are associated to the period from the first recorded date and time to the last recorded date and time on the search engine server.

To define the concept of **session** several references have been collected. Montgomery and Faloutsos (2001) defined session as "[...] *a period of sustained Internet usage; a new session begins when the user has not accessed the Web for more than two hours.*". Silverstein et al. (1999) said that a session is a group of queries made by a single user within a short time interval, capturing a single user's attempt to fill a single need for information. He and Göker (2000) instead defined a session as a group of user activities that are related to each other, not only through a need for information but also through proximity in time. In turn, Gayo-Avello (2009) defined sessions as short sequences of successive queries related to a single purpose or need for information.

2.1.2 String Similarity Metrics

An important concern in identifying sessions relates to inferring if two interactions belong to the same session through string similarity metrics. A problem that can occur when comparing two queries is commonly referred to as a **vocabulary gap**. This term is used to define the problem, when queries do not have exact terms in common, but yet belong to the same general topic, e.g. due to word synonymy or to morphological inflexions.

In order to provide an answer to the similarity issue mentioned above, several methods proposed on the literature will now be described, following the description from Santos et al. (2018b) and Cohen et al. (2003). These methods can be grouped according to three different approaches. **Character-based methods** are based on character edition operations (i.e., simple operations such as insertions, deletions, substitutions, and transpositions), whereas **vector-space based methods** compare vector representations constructed from the strings, in which each dimension corresponds to an element such as a word token or a character n-gram. **Hybrid methods** try to combine the aforementioned methods, in order to achieve better effectiveness.

Regarding character-based methods, the **Levenshtein edit distance** calculates the number of edits needed to transform a string a into a string b. These edits are insertions, deletions, or substitutions (e.g., the Levenshtein edit distance between the string ***fish*** and ***ifsh*** is 2, corresponding to a deletion of i and insertion of i after f).

The **Damerau-Levenshtein distance** (Damerau, 1964) is a modified version of the previous method, which also considers transpositions as editing operations (e.g., the Damerau-

Levenshtein distance between the string ***fish*** and ***ifsh*** is 1, as it corresponds to a transposition between f and i). This metric can be computed through dynamic programming, following the next definition:

$$d_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if min(i,j) = 0} \\ \min \begin{cases} d_{a,b}(i-1,j)+1 \\ d_{a,b}(i,j-1)+1 \\ d_{a,b}(i-1,j-1)+1_{a_i \neq b_i} \\ d_{a,b}(i-2,j-2)+1 \end{cases} & 100 \leq x \\ \min \begin{cases} d_{a,b}(i-1,j)+1 \\ d_{a,b}(i,j-1)+1 \\ d_{a,b}(i-1,j-1)+1_{a_i \neq b_i} \end{cases} & \text{otherwise} \end{cases} \tag{2.1}$$

The **Jaro similarity metric** (Jaro, 1989) takes into account the number of characters in common and their order, and is defined by:

$$s_{a,b} = \begin{cases} 0 & \text{if n = 0} \\ \frac{1}{3} \times \left(\frac{n}{|a|} + \frac{n}{|b|} + \frac{n-t}{|n|} \right) & \text{otherwise} \end{cases} \tag{2.2}$$

In the previous equation, n is the number of terms in common and t is half the number of character transpositions. This metric is useful to match strings, such as person names.

Winkler (1990) extended the original Jaro metric to support the idea that differences terms on the beginning of the string are more significant than differences near the end of the string. Given two strings a and b, we can define the **Jaro-Winkler similarity** by the following formula:

$$s_{a,b} = s'_{a,b} + (l \times p \times (1 - s'_{a,b})) \tag{2.3}$$

In the previous equation, $s'_{a,b}$ is the Jaro similarity between strings a and b, l is the length of the beginning of the strings in common and p is constant that shows how much the score is adjusted if the strings have a common beginning, usually set to $p = 0.1$. The main advantage of this method is the efficiency when comparing smaller strings, even though it does not consider characters out of order.

Christen (2006) proposed two alternative methods to improve the performance when matching person names, specifically, sorted Winkler and permuted Winkler. The first sorts the word tokens in the two strings before applying the similarity metric from Jaro-Winkler. The second calculates the Jaro-Winkler similarity over all possible permutations between word tokens, finally returning the maximum value.

Before describing vector-space methods, it is essential to describe the concept of ***n*-gram** (Wang et al., 2010) (i.e., a sequence of n consecutive words or letters). For instance, the set of 4-grams forming the word ***computer*** is {"comp", "ompu", "mput", "pute", "uter" }.

The **cosine similarity** method calculates the similarity between vectors encoding the n-grams thath form the string bwing compared. We can define this vector through n-grams of different length (normally n=2 and/or n=3) and considering a binary representation (i.e., 1 if the n-gram is in the string x or 0 otherwise). The metric is defined by:

$$s_{a,b} = \cos(\theta) = \frac{\mathrm{A} \cdot \mathrm{B}}{\mid\mid \mathrm{A} \mid\mid \times \mid\mid \mathrm{B} \mid\mid} = \frac{\sum_{i=1}^{|\Sigma|} a_i b_i}{\sqrt{\sum_{i=1}^{|\Sigma|} {a_i}^2} \times \sqrt{\sum_{i=1}^{|\Sigma|} {b_i}^2}} \tag{2.4}$$

In the previous equation, for the strings a and b, $A = \{a_1,...,a_{|\Sigma|}\}$ and $B = \{b_1,...,b_{|\Sigma|}\}$.

Instead of using a binary encoding for n-gram occurrence, another alternative is to use **Term Frequency times Inverse Document Frequency** (*tf.idf*). The *tf.idf* method is a term weighting heuristic from the area of Information Retrieval, which consists on the multiplication between *tf* (i.e., term frequency) and *idf* (i.e., inverse document frequency) component. The *tf* is a measure of term importance within an instance and is defined by division between the number of occurrences of term x in the instance d and the number of terms in instance d. The *idf* is a measure of term importance within a collection and is defined by the logarithm of the division between the total number of instances and the number of instances in which term x appears.

The **Jaccard** (Jaccard, 1912) similarity coefficient consists in comparing members for two different sets to see which members are shared and which are distinct. The higher the value the greater the similarity between the two sets. There are several ways to define the sets, and one popular approach is to consider sets of character n-grams:

$$s_{a,b} = \frac{|\mathrm{A} \cap \mathrm{B}|}{|\mathrm{A} \cup \mathrm{B}|} \tag{2.5}$$

In the previous equation, for a strings a and b, $\mathrm{A} = a_1, ..., a_{|a|}$ and $\mathrm{B} = n_1, ..., b_{|b|}$. For example, if the user has submitted two queries: ***parking garage*** and ***park***, the queries do not have any terms in common. However, if we consider 4-grams, ***parking*** is replaced by: "park", "arki", "rkin" and "king" and then with Jaccard similarity coefficient will see one n-gram in common.

Instead of using regular n-grams, we can define a set of skip-grams consisting of bi-grams of non-adjacent letters. According to Keskustalo et al. (2003), an effective way to represent strings is through bi-grams with gaps of zero, one, or two characters. We can thus change the formula in Equation 2.5 into:

$$s_{a,b} = \frac{\sum_{Y \in \{\{0\},\{1,2\}\}} |\text{skipgrams}_Y(a) \cap \text{skipgrams}_Y(b)|}{\sum_{Y \in \{\{0\},\{1,2\}\}} |\text{skipgrams}_Y(a) \cup \text{skipgrams}_Y(b)|} \tag{2.6}$$

In the previous equation, Y is the set of skipped characters, and $\text{skipgrams}_Y(a)$ will return a set of skip-grams of string a, depending on the gap defined previously. In this formula it is possible to consider different gaps simultaneously.

In terms of hybrid methods, Monge et al. (1996) proposed the use of the Jaro-Winkler similarity to calculate the average similarity between the most similar pairs of word tokens in a string, as follows:

$$s_{a,b} = \frac{1}{|a|} \sum_{i=1}^{|a|} \max_{j=1}^{|b|}(s'_{a_i,b_j}) \tag{2.7}$$

In the previous equation, s'_{a_i,b_j} is the Jaro-Winkler similarity between string a and b (a_i is a token from a and b_j is a token from b). However, the major problem with this method is that it only works well for short strings, since this method punishes non-matching word more severely. Thus, Cohen et al. (2003) adapted the Jaccard coefficient explained previously to a method that instead of using n-grams uses sets of tokens and allows for small incompatibilities when aligning the tokens. So, through Soft-Jaccard metric and the Jaro-Winkler score defined, respectively by the following formulas:

$$s_{a,b} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{2.8}$$

$$s_{a,b} = \frac{1}{|a|} \sum_{i=1}^{|a|} \max_{j=1}^{|b|}(s'_{a_i,b_j}) \tag{2.9}$$

The authors defined the Soft-Jaccard metric through the Jaro-Winkler score as a secondary metric between the tokens:

$$s_{a,b} = \frac{Z}{|a| + |b| - Z}, \text{where} \tag{2.10}$$

$$Z = \frac{\sum_{i=1}^{|a|} \max_{j=1}^{|b|}(s'_{a_i,b_j}) + \sum_{j=1}^{|b|} \max_{j=1}^{|a|}(s'_{a_i,b_j})}{2} \tag{2.11}$$

In the previous equation, s'_{a_i,b_j} represent an inner similarity between string a and b (i.e., a_i is a token from a and b_j is a token from b) and will be 1 if the tokens are similar and 0 otherwise, and $|a|$ and $|b|$ consist in the number of tokens in string a and b, respectively. This way, Z consists in calculating the similarity between tokens from string a which match string b, and vice-versa.

Another different metric than the aforementioned is the phonetic encoding techniques Cohen et al. (2003) which consists in convert a string into a code, expressing how a name is pronounced. Two techniques widely used are the **Soundex** and **Double-Metaphone** (Philips, 2000). The soundex, through an encoding table, transforms a string into numbers, except the first letter of the string, while the double-metaphone is based on rules that takes into account the position within a name and the previous and following letters. The main drawbacks of these techniques are that they are language-dependent and, in general, they are designed based on the Latin alphabet and the English phonetic structure.

2.2 Related Work

This section begins with a description of baseline methods for detecting user sessions, afterwards complementing this description with more complex methods that have a better effectiveness and/or efficiency. This section ends with a description of methods for predicting user satisfaction, based on heuristics such as user clicks.

2.2.1 Detecting User Query Sessions

Most of the methods presented next are described on the survey from Gayo-Avello (2009), which covers the current state-of-the-art for detecting user sessions, we also complement the information on this survey with new approaches. This subsection will be split between methods that are based only on a temporal component (i.e., the time gap between queries), methods based only on a lexical component (i.e., similarity between queries, search patterns, etc.) and methods based on a combination of both these heuristics. The combined methods are in turn divided in to supervised methods (involving learning from labeled data so that a certain pattern or function can be deduced from that data) and unsupervised methods (identifying complex processes and patterns without human judgments to provide guidance along the way). Finally, in the end of this section it is presented an overview of all methods described.

2.2.1.1 Methods Using a Temporal Component

The most common methods to identify user sessions are based on a global temporal threshold, in which two consecutive queries belong to the same session if the elapsed time is less than a pre-defined threshold. Previous studies have considered limits of 5 (Downey et al., 2007), 10-15 (He and Göker, 2000), or 30 minutes (Radlinski and Joachims, 2005; Catledge and Pitkow, 1995). This approach is still widely used in practice (e.g., Google claims to apply a threshold of 30 minutes in their Web analytics application[1]), but an important limitation relates to the application of the same threshold for all contexts (e.g., different users may behave differently, and different query logs may reflect particular system and user characteristics that affect the typical duration of the sessions).

Wang et al. (2010) suggest considering the average between the time between clicks and the next query. They proposed a dynamic time model which considers the user's comprehension time in terms of queries and clicks, in which the main idea is to first calculate the most likely intent boundary, and then learn intent clusters (i.e., clusters of sessions) through the search query log to adjust this boundary. In order to reach better judgments the authors only considered sessions up to 60 min, with at least three distinct queries and using clicked URLs must present in the Open Directory Project[2] (i.e., a human-edited index of Web sites).

To explain this method the author first introduce some notation: $q \rightarrow q'$ represents two consecutive queries, $q \rightarrow u$ represents a query and a click, $u \rightarrow q$ represents a click leading to making another query, and $u \rightarrow u'$ represents a click leading to another click. Finally, $u \rightarrow q$ includes click u, search u, read u, and decide to do a new query q. With this notation, for example, $|q \rightarrow q'|$ is the time difference between queries. One problem is that $|u \rightarrow q|$ can depend on several factors (e.g., readability of page), and thus one alternative is to define the comprehension time of a specific session through:

[1] `https://support.google.com/analytics/answer/2731565`

[2] `http://odp.org/,lastaccessedJanuary6,2019.`

$$h = \frac{1}{n} \times \sum_{i=1}^{n} |u_i \rightarrow u_{i+1}| \tag{2.12}$$

In the previous equation, n is the number of $u \rightarrow u'$ transitions in a session. Finally, based on h, the authors defined static and dynamic comprehension time models. First, in the StaticCTime model, if $\mid u \rightarrow q \mid > \mid U \rightarrow Q \mid$ then the boundary is before q, where $\mid U \rightarrow Q \mid$ is the average time difference between click and query. Second, in the DynamicCTime model, if $\mid u \rightarrow q \mid -h > \mid U \rightarrow Q \mid - \mid U \rightarrow U \mid + S$ then the boundary is before q, where $\mid U \rightarrow U \mid$ is the average time difference between click and another click and S is the time deviation of $\mid U \rightarrow U \mid$.

The second step in the approach from Wang et al. (2010) is to find the intent clusters (i.e., groups of sessions of similar intents) using contextual information, since query and clicked URLs can be ambiguous. So, the authors defined the following features: query, URL, path, query+URL, and query+path over complete-link and average-link clustering algorithms, in order to build the intent clusters. In complete-link clustering, the similarity of two clusters is the similarity of their most dissimilar members (i.e., choose the cluster pair which the merge has the lowest diameter), being defined as the minimum similarity of pairs:

$$\text{sim}(c_i, c_j) = \min_{x \in c_i, y \in c_j} \text{sim}(x, y) \tag{2.13}$$

This way, the clusters will be more tighter which are typically preferable. After merging c_i and c_j the similarity with other cluster, c_k is defined by:

$$\text{sim}((c_i \cup c_j), c_j) = \min(\text{sim}(c_i, c_k), \text{sim}(c_j, c_k)) \tag{2.14}$$

In average-link clustering, the similarity between two clusters is defined as the average distance between each point in one cluster to every point in the other cluster. This way, the similarity between clusters r and s is the average length of each connecting points between the clusters defined by.

$$\text{sim}(r, s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} D(x_{r_i}, x_{s_j}) \tag{2.15}$$

In this equation, for cluster r, x_{r_i} is the set of points of cluster r. So, if the session has x queries then it can have x possible intent boundaries, as an URL may have more than one category in the Open Directory Project (ODP). Therefore, the authors first used the time model to find a possible boundary, and then the segment with highest similarity with the intent clusters is chosen. In order to refine this choice the boundary is moved to the left and to the right to see if the similarity improves or not.

To support their theory, the authors assume the following baselines: AvgTime (session defined through a threshold of 20 minutes); AvgQuery (session defined with a mean of 7 queries); StaticCTime Model; and DynamicCTime Model presented previously. Comparing the previous baselines with each others, the best result was obtained with the AvgTime with a precision of 0.6368, a recall of 0.6348 and a F-measure of 0.6355. This comparison was measured over Live search query log dataset (Craswell et al., 2009), where was selected 1,000 sessions randomly and labeled the intent boundaries in each session, resulting a total of 1,456 boundaries.

To improve over the baselines the authors introduce the intent clusters with DynamicCTime Model and then used five features (Query, URL, Path, Query + URL, and Query + Path) considering a complete-link clustering and a average-link clustering. The feature with the best result was the Query + Path considering a complete-link clustering with a precision of 0.6409, a recall of 0.6681 and an F-measure of 0.6543, proving that detecting intent boundaries in search query logs with intent clusters can be useful.

In turn, Murray et al. (2006) proposed a threshold based on user activity through a hierarchical agglomerative clustering approach. In this work, the authors used an AOL dataset (Pass et al., 2006) with 216,000 users from November and December of 2004 to January of 2005, with a total of 8,269,030 queries. The authors verified that their own algorithm achieves 98% of accuracy in identifying sessions, compared to human judgments.

Specifically, they defined three possible types of information regarding user search activity: the activity types in which one obtains the time that the users would need to perform different types of activities (e.g. number of clicks); the topical type, concerning user interests during the activity; and the semantic type, in which one obtains information on how users interact and express interest in a topic. By relating these three types it is possible to perceive the information needs and the changes of topic, being extremely useful to characterize the behavior of the user.

The method itself is based on Hierarchical Agglomeration Clustering (HAC), as proposed by Willett (1988), to detect session breaks for each user. The HAC method is a bottom-up approach for cluster analysis, which builds a hierarchy of clusters. Each observation starts as cluster and these clusters are then merged as one moves up the hierarchy. First, for each new query of the user, the authors calculated the gap interval through the difference between the current query time and the last query time. These gap intervals are then sorted from shortest to longest. Then, the method sets the variables $\max_{\delta:\sigma}$ as 0, and the $\text{argmax}(\delta : \sigma)$ and the *clustered_intervals* as *null*. In this case, argmax($\delta : \sigma$) represents the interval that had the most effect on the variance when smaller intervals were added. Third, for each sorted gap interval i, if the number of *clustered_intervals* is greater than 1, the authors calculated the μ (i.e., the mean of *clustered_intervals*), the σ (i.e., the standard deviation of *clustered_intervals*), and the δ_i (i.e., the difference between the sorted gap interval i and the μ). Then, if the division of δ_i and σ is greater than $\max_{\delta:\sigma}$, the $\max_{\delta:\sigma}$ is update to the division of δ_i and σ and the argmax($\delta : \sigma$) is update to the sorted gap interval i. The last operation in this cycle is to add the gap interval i to the set of *clustered_intervals*. Finally, for each query q the authors calculated the gap interval through the difference between the current query time and the last query time, and if this gap interval is less than argmax($\delta : \sigma$) they add q to the current cluster, otherwise they create a new cluster with q.

To evaluate the results, the authors randomly removed 500 users from the log, with less than 20 queries in 3 months, leading to 1,593 query pairs to be judged. With their method a total of 854 session breaks were generated, with a total of 263 missed session breaks. When comparing the results with human judgments, the authors compute 831 correct session breaks, with a recall of 76%. The average precision was 99% for each user, and the average recall was 88%. Finally, to show that the proposed method is better than a baseline using a fixed threshold, the authors tested the baseline with the same dataset. The fixed thresholds have a better recall 99%, when HAC achieve 76%, but the HAC achieves a higher precision 97%, when the fixed thresholds achieve 81%.

Another method based solely on a temporal component was proposed by Mehrzadi and Feitelson (2012). The authors tested each aforementioned method, showing the limitations of each one and proposing a method based on a threshold inferred from the user activity pattern, but with a relative better performance. There are different ways to assess the distribution of user behavior (e.g., the number of pages requested and the data transferred in each session). Bearing this in mind, the authors proposed an individual threshold based on the activity of each user, inferred through tests with the 2006 AOL dataset (Pass et al., 2006). In a first phase, the authors analyzed the distribution of intervals between successive queries from the 2006 AOL log and concluded the best intervals are between 20 and 30 minutes, although there is no indication that this is a natural single threshold. In order to prove that there is not natural single threshold, the authors created a histogram of session durations (i.e., the relation between the number of sessions and the session length), using a fixed threshold (i.e., 10 minutes, 20 minutes and 60 minutes) number of sessions, and concluding that:

- The number of sessions decreases when reaches the fixed threshold itself, thus supporting the conclusion that all non-trivial sessions are not correctly identified.
- The distribution for thresholds of 10 and 20 minutes are approximately the same.
- For the interval of 60 minutes, the number of sessions decreases to about a third (i.e., short sessions are joined with long sessions because the interval is too big), thus supporting the conclusion that long sessions are not correctly identified either.

In conclusion, there is not a natural single threshold because for some sessions a given value will be too short and for others it will be too long. As mentioned before, Murray et al. (2006) were the first to propose user-specific thresholds, but there are drawbacks to this approach. The maximum can be poorly defined (i.e., either too long or too short) and is based on the assumption that the distribution is bimodal (i.e., the intervals have little variability), which does not happen in practice as short thresholds can be very similar to each other, yet long limits can be totally different from user to user. So, Mehrzadi and Feitelson (2012) proposed an approach which deal with the aforementioned problem by using binning on a logarithmic scale with the aim of finding the global minimum, through the following steps:

- First, the authors create a histogram with bins based on powers of 2, to have an overview of the distribution of the thresholds without going into detail. This histogram shows the distribution for the number of bins.
- Secondly, they use bins from 512 seconds to 8,192 seconds to verify that the threshold is acceptable.
- Finally, the authors give a score to each bin. First, with the histogram the authors extract the bins with highest value, and the score is set as 0, then for each bin, the authors verify certain rules, in order to increment this score (i.e., plus one). The rules are: if the count of this bin is less than 2/3 of the left maximum histogram, and, if so, the authors increment the score; if it is less than half of the the left maximum, and, if so, the authors increment the score; if it is less than 1/3 of the left maximum, and, if so, the authors increment the score; if it is less than 2/3 of the left maximum, and, if so, the authors increment the score and the same increment to right maximum; Finally, if the count of this bin is equal to 0 the score will be set as 5. The bin with the highest score is chosen to be the threshold for this specified user. In case of a tie, the one closest to 1,200 is chosen as threshold.

In order to evaluate the method against the proposal from Murray et al. (2006), the authors used the same evaluation process this previous work, through the human judgment labeling over the AOL log (Pass et al., 2006). In the 4,992 intervals that were considered, there was consensus on 1,334 session breaks and 3,384 are intervals within a session, indicating a precision of 83%. In certain cases, the algorithm defines the threshold lower than the threshold defined by the human with thresholds between 60 and 120 minutes and with an opposite behavior compared to Murray et al. (2006). Although the most reliable solution is a threshold per user, and it should be noted that the boundaries depend on the context. For instance, most sites are expected to have very short sessions, while *Facebook* and *YouTube* can expect very long sessions, including long breaks.

2.2.1.2 Lexical Component

This section presents methods based only on a lexical component. In this case, the biggest issue in identifying user sessions is detecting query reformulations.

Jansen Bernard et al. (2007) proposed three different approaches to detect session boundaries, one of which takes into account the existence of reformulations. The authors retrieved the records of searcher-system interactions of a portion of the searches executed on Dogpile.com on May 6, 2005, with 2 million interactions from 500 thousand users.

The first approach consists in using only the IP and a cookie (i.e., a mechanism from the HTTP protocol to allow websites to remember stateful information often used to keep information on user identifiers) to define the initial query and the following ones forming the session. When they detect a change of IP or cookie, they consider this as a new session. The second approach consists in using the same idea from the first method, but using a temporal threshold of 30 minutes (i.e., still using only IPs and cookies to identify the user, but using a fixed threshold to define the length of the session). This threshold is based on the work of Catledge and Pitkow (1995) which showed that the typical Web session duration was 25.5 minutes on average. Finally, the third approach is also based on first method but, instead of using a fixed threshold, the authors used the content of the user queries (i.e., search patterns). These search patterns are: Assistance, which means queries generated through the Dogpile query reformulation assistance service (i.e., the system suggested different terms related to the submitted query); Content change, which occurs when the new query is equal but executed is in another content collection; Generalization, when both queries regard the same topic and the new query has more general information (e.g., containing only a subset of the terms); Reformulation, when the user adds or removes terms from the original query; and New, when the query is on a new topic (i.e., the queries do not have terms in common).

The authors concluded that, with Method 1, more than 79% of the sessions have three or fewer queries, while in Method 2 more than 97% of the sessions presented three or fewer queries, with Method 3, more than 93% of the sessions included three or fewer queries. Finally, in order to show the performance of each method, the authors measured the results according to five categories of errors: Vocabulary gap (i.e., errors in the new and the reformulation patterns), Cookie (i.e., the cookie is wrongly defined, leading into errors on the new pattern), special character change (i.e., when the original query has special characters, leading into errors on the new or the reformulation patterns) and time gap (i.e., when the duration of a session was too long to be considered a session, leading into errors on the new pattern). With these parameters the authors verified that the sum total of all misclassifications for Method 2 was 1.05% (i.e., 98.95% of accuracy) and for Method 3 was 4.45% (i.e., 95.55% of accuracy). Even though Method 2 showed better results, Method 3 entailed more categories of errors than Method 2, leading thus to the conclusion that Method 3 is better as it addresses the context of queries, unlike Method 2. Overall, Method 3 results in an 82% increase in the count of sessions, and in Method 2 the mean session duration was less than 30 minutes. Another interesting conclusion in the study from Jansen Bernard et al. (2007) is that user queries are most often modified by changing query terms (about 23%), in comparison to adding or deleting terms.

One of the biggest problems when assessing the similarity between queries is the fact that most search queries are very small (i.e., only two or three terms will influence the similarity). Hence, Sahami and Heilman (2006) developed a method aimed at determining if there is a high degree of semantic similarity between queries, leveraging text snippets associated to the results retrieved for the queries. This method can better handle problems such as acronyms (e.g., similarity between ***AI*** and ***Artificial Intelligence***), or queries with terms in common but used in different contexts (e.g., the term ***graphical***, in ***graphical models*** and in ***graphical interface***).

The authors proposed a new similarity function where for each short text snippet x, they create the query expansion of x, called QE(x), through the following steps: First, a vector representations are created for each of the n retrieved documents (i.e., $v_i = \{v_1, v_2, ..., v_n\}$) , with the $tf.idf$ score of each term, where the authors set the number retrieved documents n to 50, because this offered the best tradeoff between efficiency and robustness. Second, the authors truncate each vector v_i to include the 50 highest weighted terms. Third, a centroid is created of the normalized vector v_i with the following formula:

$$\mathrm{C}(x) = \frac{1}{n} \times \sum_{i=1}^{n} \frac{v_i}{\| v_i \|_2} \tag{2.16}$$

Finally, the centroid is normalized by the following formula:

$$\mathrm{QE}(x) = \frac{\mathrm{C}(x)}{\| \mathrm{C}(x) \|_2} \tag{2.17}$$

After experiments the authors concluded that 1,000 characters (in a token centered on the original query terms in the original text) are enough to achieve a high accuracy. With all this information, the authors defined the semantic similarity between query a and b through the inner product between QE(a) and QE(b).

In their study, from Sahami and Heilman (2006), the Google search engine was used as the mechanism for document retrieval. With the aforementioned function, the authors assessed the obtained results for tested different problems: acronyms, the interchangeable use of person names and their titles, and terms with ambiguity depending on the context. They observed that, regarding acronyms, ambiguous terms, and person names, the function is indeed very effective (e.g., the similarity between ***Microsoft CEO*** and ***Steve Ballmer*** is measured as 0.838).

2.2.1.3 Unsupervised Combination of Temporal and Lexical Component

This section provides an analysis of the state-of-the-art related to the methods using on a combination of temporal and lexical components. Each component alone presents distinct problems, and the combination aims at extracting the advantages of each one.

Hagen et al. (2011) proposed a cascade method (i.e., an incremental procedure) that relies firstly on more efficient features (i.e., search patterns), and then on features progressively with higher effectiveness but lower efficiency, until reaching more complex features, but only if necessary to obtain reliable results (i.e., verifying the similarity between documents returned as results). The authors based their study on the 2006 AOL dataset (Pass et al., 2006) and, in order to focus only on important interactions, the authors eliminated possible bots, through simple heuristics. First, they removed users with only one interaction, since it has no value to do session detection research. Second, they removed users with consecutive queries with a temporal distance of less than 10 seconds. Third, the users with queries whose average size exceeds 100 characters were also removed. After these eliminations, the method consisted of the following steps.

Search patterns are first used to detect similarity of queries, among which repetition, generalization and specialization, regardless of the time between queries. Although this step is very efficient, it shows low efficiency because it does not detect misspellings or vocabulary gaps.

The second step uses a previously proposed geometric method (Gayo-Avello, 2009) to solve the problems undetected in the first step. The geometric method is a very fast and efficient method that consists on the combination of two features (i.e., a temporal threshold and lexical similarity based on n-grams). The temporal component f_{time} is based on the following equation:

$$f_{time} = \max \left\{ 0, 1 - \frac{t'-t}{24h} \right\} \tag{2.18}$$

Assuming that we are comparing the consecutive queries q and q', the value t' corresponds to the time of the query q' and t corresponds to the time of the query q. If the result is close to 1 the two queries are temporarily close, otherwise the function will return 0. The lexical component f_{lex} is calculated from the cosine similarity between n-grams, more specifically 3-grams and 5-grams. Finally, q and q' belong to the same session if they respect the following equation:

$$\sqrt{f_{time}^2 + f_{lex}^2} \geq 1 \tag{2.19}$$

The strongest advantage of this second step is that it is able to deal with orthographic errors because it combines temporal proximity and n-grams. The disadvantage of this method consists in the lack of a mechanism to detect semantic similarities, between $f_{lex} < 0.4$ and $f_{time} < 0.8$ (i.e., when the queries are close temporally, but have a low similarity) and the cascade method adds additional steps to tackle this disadvantage. Therefore, these two steps are more efficient compared to the geometric method alone, but with the same accuracy and simplicity of implementation. The reason is that the first step can detect about 41% of cases, leaving only 59% of the cases to run in the second step.

The third step uses Explicit Semantic Analysis (ESA) (Gabrilovich and Markovitch, 2007), i.e. a method that builds vector representations of the textual units (i.e., words or a whole document) and uses an existing corpus as base knowledge (e.g., *Wikipedia*), being more efficient than n-gram overlap. Given a word, the method represents each term as a vector of weights (e.g., Mars $\rightarrow$(Planet, 0.9), (Solar System, 0.85), (Earth, 0.3). The algorithm starts by doing a pre-processing, creating an ESA array with basis on a $tf.idf$ weighted term-document-matrix of *Wikipedia* articles. In a second phase, the two textual units we want to compare are represented as vectors and are multiplied with the ESA matrix. Then, the cosine similarity of the resulting vectors, f_{esa}, is computed. The authors realized that the most reliable value when assessing if two queries belong to the same session is when $f_{esa} > 0.35$.

In the final step, the authors use Web search results to detect semantically similar queries, comparing the top-10 retrieved documents of queries q and q' through the Jaccard similarity. If at least one of the top-10 documents from q' is common to the results of query q, then they are in the same session. This step should only be done when the results of the previous step are not reliable, due to low efficiency.

The authors compared the cascade method with the geometric method (Gayo-Avello, 2009) and verified an improvement in F-measure and recall, although with lower precision. In terms of performance, the cascade method is also considerably better because in the first and second steps it showed a reliability of 40% and 35%, and a speed of 0.08ms and 0.20ms, respectively. On the other hand, the final step is the bottleneck in this algorithm, since it is necessary to be implemented over the commercial API of a search engine, which increases the time to 200ms for each pair of queries and adds only a reliability of 0.85%. The authors concluded that if the objective is a tradeoff between efficiency and effectiveness, it is preferable to consider this method until the third step. An additional advantage is its simplicity, because it does not need training data.

The last method corresponds to an improved version of the method mentioned in Section 2.2.1.1, corresponding to the use of one threshold per user (Mehrzadi and Feitelson, 2012). The same authors proposed an improvement, which combines the previous method with n-gram similarity between n-grams computed through the Jaccard coefficient. In order to see if this approach is valuable, the authors visualize successive queries using heatmaps (i.e., i and j in a heatmap represent similarity between the i-th query and the j-th query). With this methodology, the authors concluded that, in most cases, queries are very similar (i.e., the user continues to search similar things in the next session), although in some cases the users do several unrelated things in sequence. The results were quite satisfactory, with only 7% of sequences of related queries cut by a session boundary.

2.2.1.4 Supervised Combination of Temporal and Lexical Component

This section surveys methods previously described by Gayo-Avello (2009) based on a supervised combination of temporal and lexical components. For instance, Ozmutlu et al. (2008) proposed an automatic method for identifying topic changes based on the Multiple Linear Regression (MLR). The authors used the ANOVA principle to analyze the differences among groups in a sample and to examine the structure of the variance in topic changes, proving the similarity between non-semantic characters of web search queries and the existence of a topic change. The dataset used in the experiments was retrieved from the FAST search consisting of a random sample of 10,007 queries.

MLR considers more than one regression variable to understand and predict a dependent variable (i.e., in this case, existence or not of topic changes). This method is used to explain the relationship between one continuous dependent variable and two or more independent variables. The main idea of this model is to return a real number, but in this case, in order to identify the dependent variable, the authors trained the MLR model to return a value between 1 and 2, where 1 is a topic continuation and 2 a topic change (with a threshold value between 1.1 and 1.9). The independent variables are the search pattern of queries (SP), the time interval of queries (TI), and the order of the search query (QN). Previous experiments showed that considering only two variables is very effective, so the authors considered the SP-TI interaction, the SP-QN interaction, and the TI-QN interaction. To create a MLR the next formula was followed:

$$Y = \beta_0 + \sum_{i=1}^{3} \beta_i F_i \sum_{i=1}^{2} \sum_{j=i+1}^{3} \beta_n F_i F_j + \epsilon \tag{2.20}$$

In Equation 2.20, F_i refers to the variables, where $i = (1,2,3)$, $j = i+1, ..., 3$, $n = i+j+1$, and $\epsilon \sim iid\ N(\mu, \sigma^2)$ (i.e., a set of random variables $\{X_1,\ X_2,\ ...,\ X_n\}$ is independent and identically distributed (i.i.d.) if all the X_i are mutually independent, and they all have the same distribution).

The methodology proposed by the authors to identify topic changes consists of the following steps. First, a human expert will label the topic shift in the dataset. Next, the authors divide the existing data into two sets, one for training the MLR equation with 4,997 queries, and one for testing with 5,010 queries. In terms of the independent variables for the time interval, the authors considering the distribution of the queries with the 0-5, 5-10, 10-15, 15-20, 20-25, 25-30, 30+ minutes thresholds. The same search patterns used in other studies were considered: generalization; specialization; reformation; new, but defined as a query with no terms in common; relevance feedback, defined as query with no terms which is generated by the system; and other for the remaining cases. The authors see the position of each query in the session with the objective of ordering the queries within a session. Based on the third step, the MLR equation is determined with the first half of the dataset, and afterwards tested and compared with the labels made by human experts. Finally, the MLR equation is evaluated in terms of standard performance measures (i.e., precision, recall, F-measure). The following values for the parameters of MLR equation were obtained after training:

$$Y = 1.14 - 0.135TI - 0.0354SP - 0.000864QN + 0.0491TI*SP - 0.0163TI*QN + 0.00184SP*QN \tag{2.21}$$

Lastly, the authors applied ANOVA to show the significance of each regressor on the dependent variable, through the following hypothesis test: H_0 is $\beta_i = 0$ and H_1 is $\beta_i \neq 0$, with $i = 1, ..., 6$. So, if H_0 is rejected, that factor is irrelevant. They concluded through the previous MLR question that TI, SP and QN can be effective on topic shifts, but in the combination TI and QN is useless.

More recently, Zhang et al. (2013) described a better approach to solve the problem of the vocabulary gap, compared to the ESA method. Instead of enriching short queries with external information, from a corpus such as *Wikipedia*, these authors solved the problem by learning hidden topics from the query log, covering long tail queries. The main advantage of the method based on hidden topics is that *Wikipedia* only cares about popular terms, while hidden topics adjust to the dataset. This method, called SVMA, was tested in comparison to the geometric method, achieving better results in terms of performance, as well as in terms of accuracy, although the authors have not tested this, the method can also be included in the third step of the cascade method.

SVMA consists of two steps, involving search patterns and the use of hidden topic discoveries. In the first step, the authors used the aforementioned search patterns (i.e., repetition, generalization, specialization, reformulation and new). In the second step, to solve the problem of misspellings and the vocabulary gap, the method takes into account only the pattern new and considers several features, in order to disambiguate and better describe the topic change. The first feature is time gap, which is defined by the minimum of 1 and the difference between the time of q_i and q_j divided by the time limit defined as 24h. The second feature is the normalized Levenshtein edit distance. The third feature is clicked URL similarity (i.e., the similarity with the cosine metric between two queries that are first represented as bags of clicked URLs). The fourth feature is character n-grams similarity, equal to clicked URL similarity but with bags of character n-grams extract from queries. The fifth feature is ESA similarity (Gabrilovich and Markovitch, 2007). The last feature is the hidden topic similarity method proposed by the author, which is defined as follows.

To build hidden topics the authors consider the combination between queries and URLs, because they believe that if the user clicked on a URL then this is because it is relevant. They suggest viewing the URL as a document, since the URL may contain more terms that help define the border (e.g., the URL may contain a term that was not present in the corpus defined as base knowledge). If the user makes a new query with the term learned previously, it will better define the border.

The authors used the Gibbs Sampling algorithm, with 20 iterations and receiving a vector of words from the document and the respective topic. Gibbs Sampling is based on the Markov Chain Monte Carlo algorithm (Relaxation, 1984) for obtaining a sequence of observations which are approximated from a specified multivariate probability distribution, when direct sampling is difficult. For this topic sampling it is necessary to identify the topic that the term t belongs to, being thus necessary to calculate the following probability:

$$P(qz_i = k|\overrightarrow{qz_{\neg i}}, \overrightarrow{qw}, \overrightarrow{z}, \overrightarrow{w}) = \frac{n_k^{(n)} + qn_{k,\neg i}^{(t)} + \beta_t}{(\sum_{v=1}^{V} n_k^{(v)} + qn_{k,\neg i}^{(v)} + \beta_v) - 1} \times \frac{n_k^{(n)} + \alpha_k}{(\sum_{v=1}^{K} n_k^{(j)} + \alpha_j) - 1} \tag{2.22}$$

In the previous equation, $\overrightarrow{w}$ is the vector of all words in the document, while $\overrightarrow{z}$ is the respective topic. $\overrightarrow{q_w}$ represents all words of query q, and $\overrightarrow{q_z}$ represents all topics of query q. $qn_{k,\neg i}^{(t)}$ is the number of times that t belongs to topic k without counting current assignment, $n_{q,\neg i}^{(k)}$ is the number of times that t belongs to topic k without counting the current assignment, V is the total number of terms, K is the total number of hidden topics, and $\overrightarrow{\alpha}$ and $\overrightarrow{\beta}$ are parameters of a Dirichlet distribution. The parameters of a Dirichlet distribution are defined through the Latent Dirichlet allocation (LDA) algorithm (Blei et al., 2003), which consists of a generative probabilistic model that creates only topics for each document, believing that in a document there are few changes of topics or none. The authors fixed $K = 100$ (i.e., number of topics), $\alpha_k = 50/K$, with k between 1 and K and $\beta_j = 0.1$, with j between 1 and V (i.e., the total number of terms). Finally, to classify the pattern new the authors built a Support Vector Machine classifier using LIBSVM V3.12 [3] with the aforementioned features. The topic distribution of query q is $\overrightarrow{v_q} = \{v_{q,1}, ..., v_{q,K}\}$ obtained through the following formula:

$$v(q,k) = \frac{n_k^{(n)} + \alpha_k}{\sum_{j=1}^{K} n_q^{(j)} + \alpha_j} \tag{2.23}$$

[3] `https://bit.ly/2QA5owY,lastaccessedJanuary6,2019.`

In the previous equation, $n_q^{(k)}$ is the number of terms of the query q that belong to topic k. Finally, with topic distributions of q_i and q_j, respectively, represented as V_{qi} and V_{qj}, the authors compute topic similarity through the following formula:

$$\text{TopicSim}(q_i, q_j) = \frac{\mathrm{D}(\overrightarrow{v_{q_i}} \mid\mid \overrightarrow{v_{q_j}}) + \mathrm{D}(\overrightarrow{v_{q_j}} \mid\mid \overrightarrow{v_{q_i}})}{2} \tag{2.24}$$

In the previous equation, $\mathrm{D}(\overrightarrow{v_{q_i}} \mid\mid \overrightarrow{v_{q_j}})$ is the Kullback-Leibler (KL) divergence between V_{qi} and V_{qj}. The KL divergence is a measure of how one probability distribution is different from a second one, and for discrete probability distributions P and Q it is defined as follows:

$$\mathrm{D}(P \mid\mid Q) = \sum_i P(i) \log\left(\frac{Q(i)}{\mathrm{P}(i)}\right) \tag{2.25}$$

In their experiments, the authors used the 2006 AOL dataset (Pass et al., 2006), with 5-fold cross-validation, extracting 1,124,839 URL documents as the external information source. For their comparative the authors considered the following baselines: time gap (i.e., temporal threshold), where through tests they realized that the best threshold would be 20 minutes; Lexical Comparison (LC) (i.e., search patterns); The geometric method; SVMA (i.e., the proposed method) with all the features mentioned above to deal with the new pattern; SVM-topicsim, which is the same as SVMA but without the feature corresponding to hidden topic similarity; SVM-ESAsim, which is the same as SVMA but without the feature corresponding to ESA similarity; SVM-LC, which uses directly the SMV without using lexical patterns for border detection. The results were quite satisfactory and they concluded that the SVMA method was the one that obtained better results in accuracy, about 1.5% better than the geometric method. Another conclusion is that SVM-topicsim and SVM-ESAsim are worse than SVMA, proving that the problem of the vocabulary gap can be addressed with the feature corresponding topic similarity, better than with the ESA similarity. Finally, the authors concluded that using only lexical patterns results in a greater accuracy, about 90%, showing that most queries from the same session have words in common.

2.2.1.5 Overview

In conclusion, there are several methods that can be used to determine user sessions, previous methods evaluated on the 2006 AOL dataset (Pass et al., 2006) are summarized in Table 2.1, where the 30 min threshold was proposed by Jansen Bernard et al. (2007), the HAC was proposed by Murray et al. (2006), the threshold per user was proposed by Mehrzadi and Feitelson (2012),

Table 2.1: Summary of methods applied to the 2006 AOL dataset.

Component	**Method**	**Precision**	**Recall**	**F-Measure**
Temporal	30 min Threshold	0.88	0.74	0.80
	HAC	0.49	0.95	0.64
	Threshold Per User	0.83	0.99	0.90
Lexical	Method 3	0.81	0.94	0.87
	QueryExpanded	0.83	0.93	0.87
Temporal and Lexical	Geometric Method	0.87	0.94	0.92
	Cascade Method	0.86	0.97	0.93

the method 3 was proposed by Jansen Bernard et al. (2007), the QueryExpanded was proposed by Sahami and Heilman (2006), the geometric method was proposed by Gayo-Avello (2009), and the cascade method was proposed by Hagen et al. (2011).

Unfortunately, some of the methods that were described used different datasets or different evaluation metrics and, thus, there were not included in Table 2.1. More specifically, Wang et al. (2010) proposed the Dynamic Time Model, which used in the tests the Live search query log dataset (Craswell et al., 2009). In this case, the 30 min threshold method had 0.64 of precision, 0.63 of recall and 0.64 of F-measure, while the proposed method had 0.64 of precision, 0.66 of recall and 0.65 of F-measure. In conclusion, this method does not show great improvements, comparing with other methods, such as threshold per user method, because the average between a click and the next query can have very long intervals if there is break in the user activity.

The MLR method proposed by Ozmutlu et al. (2008) used a FAST search engine dataset with 1,257,891 queries, and this method was compared also with the 30 min threshold method. In this case, the basic fixed threshold method had 0.38 of precision, 0.74 of recall and 0.55 of F-measure, while the proposed method had 0.31 of precision, 0.94 of recall and 0.53 of F-measure. Although, later Gayo-Avello (2009) showed that the MLR method introduces extremely few topic shifts when compared with the other methods, being an unreliable method.

The hidden topics approach proposed by Zhang et al. (2013) used also the AOL dataset (Pass et al., 2006), but only the accuracy was used as the evaluation metric to compare the proposed method against the 30 min threshold method, the lexical search pattern method and the geometric method. In this case, the basic fixed threshold method had 0.8 of accuracy, the lexical search pattern method had 0.9 of accuracy, and the geometric method had 0.92 of accuracy, while the proposed method had 0.94 of accuracy. In conclusion, hidden topics approach presented better results than geometric method, which it is the best method to solve the vocabulary gap problem.

2.2.2 Predicting User Satisfaction and Modelling User Clicks

This section focuses on state-of-the-art methods for the prediction of user satisfaction. It first describe methods based only on clicks for trying to estimate relevance (i.e., click models), and then it describes methods for predicting user satisfaction, based on clicks and a combination between user behavior and attributes of pages.

2.2.2.1 Click Models

Click models attempt to predict relevant information through user activity and information about the user's needs. The main idea is to label a dataset of pairs of queries/documents with relevance judgments, directly from query logs and without human experts involved in the process. In particular, these models are used to predict a relevance of the search result and to predict click probability.

From previous surveys from Chuklin et al. (2015), Wang et al. (2016) and Grotov et al. (2015), it can be concluded that there are two types of click models, namely position models and cascade models. A position model assumes that a document is only clicked if it is examined and relevant (i.e., examination hypothesis) and each rank has a certain probability of being examined, depending on the rank itself (i.e., a lower rank will lead to a lower probability). Hence, if the user clicks on a document it means that the document was examined and considered relevant. The disadvantage of this model is the fact that it treats each document individually and does not take into account highly relevant documents in lower ranks of the list. The probability of the user clicking at position p for a certain document u is defined through:

$$P(C = 1|u, p) = P(C = 1|u, E = 1)P(E = 1|p) \tag{2.26}$$

This equation means that if the user did not examine the document u then there is no click, but if the document u is examined the probability of being clicked on depends only on the relevance of the document u, where $P(C = 1|u, E = 1)$ represents the relevance of the document u and $P(E = 1|p)$ represents the position effect.

An approach to estimate the position model probabilities based on the logs is through the maximum likelihood estimation. In order to capture the position effect it is necessary to observe the CTR of the same URL at different positions. Given the vector $P(E = 1|p)$, which will represent as β_p, the maximum likelihood equation for $P(C = 1|u, E = 1)$, which will represent as α_u is defined by:

$$\alpha_u = \arg\max_a \sum_{i=1}^{N} c_i \log(\alpha\beta_{p_i}) + (1 - c_i)\log(1 - \alpha\beta_{p_i}) \tag{2.27}$$

The vector β_p is estimated through an alternate maximization of the likelihood between α_u and β_p. However, one disadvantage of the previous approach is the fact that α_u can be greater than 1 in same cases. One solution is use the Expectation-Maximization (EM) algorithm (Dupret and Piwowarski, 2008), which E are the hidden variables.

The cascade model, in turn, makes an assumption that there is only one click per search and the user analyzes the results sequentially (top-down), with dependency between documents accessed in the same session. So, if the user clicks on the document u in position p we can conclude that the user skipped the ranks above position p and considered the document u relevant. The probability of clicking on the i-th document is defined through:

$$P(C_i = 1) = r_i \prod_{j=1}^{i-1}(1 - r_j) \tag{2.28}$$

In the previous equation, r_i is the probability that the i-th document is relevant and $(1 - r_j)$ is the probability that the j-th document is skipped. In the same way as defined in the position models is possible to find the probability of r_i and r_j through the maximum likelihood estimation (i.e., for all queries in each r_i is presented, how many time there was an click on r_i).

The Dynamic Bayesian Network Model (DBN) proposed by Chapelle and Zhang (2009) expands the idea of the cascade model, in order to consider the relevance of all documents. The model assumes that there is a click if the user examined and was attracted to the document:

$$A_i = 1, E_i = 1 \Leftrightarrow C_i = 1 \tag{2.29}$$

In the previous equation, for the i-th document, E_i represents if the user examined the document, A_i represents if the user was attracted to the document, and C_i represents if the user clicked in the document. The probability of the user finding the document interesting only depends on the document u:

$$P(A_i = 1) = a_u \tag{2.30}$$

If the user clicks on a certain i-th document there is the probability of the user being satisfied s_u:

$$P(S_i = 1|C_i = 1) = s_u \tag{2.31}$$

If the user does not click on a certain i-th document the user will be dissatisfied:

$$C_i = 0 \Rightarrow S_i = 0 \tag{2.32}$$

In the previous equation, S_i represents if the user was satisfied by the landing page. If the used was satisfied by the i-th document there is no other examination:

$$S_i = 0 \Rightarrow E_{i+1} = 0 \tag{2.33}$$

If the user is not satisfied, there is a probability of γ that he continues the search and he will leave with a probability of $1 - \gamma$:

$$P(E_{i+1}|E_i = 1, S_i = 0) = \gamma \tag{2.34}$$

If the user does not examine the i-th document he will not examine $i + 1$-th document:

$$E_i = 0 \Rightarrow E_{i+1} = 0 \tag{2.35}$$

Notice that, in the DBN model, there are two types of relevance. First, there is real relevance, which is the probability of the user being satisfied. The second one is the perceived relevance, which is the probability of the user click based on the document u. If $\gamma = 1$ we have a special case corresponding a simplified DBN model, that assumes that a user examines all documents until the last click and then abandons the search. Finally, in order to infer the variables a_u and s_u, the authors used the Expectaction-Maximization (EM) algorithm to find the maximum likelihood, and the forward-backward algorithm to compute the posterior probabilities of the hidden variables.

The Partially Sequential Click Model (PSCM) from Wang et al. (2015) further improves the aforementioned model, and is based on two hypotheses. The first is the locally unidirectional examination hypothesis, which consists in considering that users examine search results in a single direction without any changes, and this direction depends on the examination of the user (i.e., can be top-down or bottom-up). The second is the non first-order examination hypothesis, which consists in following the previous hypothesis but considering that there may be skips in results and the user may examine a result at a certain distance with a certain direction. More specifically, the Locally Unidirectional Examination Assumption and and the First-order Click Hypothesis are represented by the following equations, respectively:

$$P(C_t|C_{t-1}, ..., C_1) = P(C_t|C_{t-1}) \tag{2.36}$$

$$P(C_t = n|C_{t-1}, ..., C_1 = m) = P(\bar{C_m} = 1, ..., \bar{C_i} = 0, ..., \bar{C_n} = 1) \tag{2.37}$$

From Equations 2.37, the PSCM model considers that a user can click on a position and then click on an upper document and the documents between the two are also likely to be examined, through the following formula:

$$P(\bar{E_i} = 1|C_{t-1} = m, C_t = n) = \begin{cases} \gamma_{imn} & m \leq i \leq n \text{ or } n \leq i \leq m \\ 0 & other \end{cases} \tag{2.38}$$

In the previous equations, m and n represent the position the user may click on $(n > m)$, and α_{uq} represents the relevance of the i-th document u to the query q, through the following formula:

$$P(R_i = 1) = \alpha_{uq} \tag{2.39}$$

In order to estimate parameters the authors also used the EM algorithm, which estimates the parameters for a model when the data is incomplete or has unobserved latent variables.

To comparatively evaluate different click models, Wang et al. (2016) and Grotov et al. (2015) considered click prediction and relevance estimation tasks. In click prediction, there are three widely used metrics, namely Perplexity, Log-likelihood and Click-Through Rate (CTR). Perplexity consists in knowing how well a model can predict the clicks.

$$Perplexity_i = 2^{-\frac{1}{N}\sum_j^N C_i \log(p_i) + (1-C_i)\log(1-p_i)} \tag{2.40}$$

In the previous equation, C_i is the user click information at rank i, p_i is the predicted click probability at rank i, and N is the number of sessions. A lower perplexity score corresponds to a higher quality click model, being 1 the ideal case.

The log-likelihood evaluates how well a model approximates observed data (i.e., how well a click model approximates clicks of actual users) and is defined as follows:

$$\mathcal{LL}(M) = \sum_{s \in N} \log P_M(C_1, ..., C_n) \tag{2.41}$$

In the previous equation, P_M is the probability of observing a particular sequence of clicks, while M is the model which is being considered.

The CTR metric is defined as the ratio between the number of times that a particular document was clicked, and the number of times that it is showed. In Chapelle and Zhang (2009), the steps required to measure the quality of click models using CTR were described. First, it is necessary to consider a document u which appears in different query sessions, in the first and other positions. Second, all the sessions where document u appears are the test set and the others are the training set. Next, we need to train the click model M and predict the clicks on the test set (i.e., predicted CTR), so that we can then compute the CTR of the test set (i.e., actual CTR), and finally compute the Root-Mean-Square-Error (RMSE) between the predicted CTR and the actual CTR, through $\sqrt{(f-o)^2}$, where f are the unknown results and o the observed values.

In terms of relevance estimation there are two widely used metrics, namely the Mean Average Precision (MAP) and the Normalized Discounted Cumulative Gain (NDCG). In particular, Grotov et al. (2015) considered the NDCG@5, through the following steps. First, they retrieved all sessions that have complete judgments, afterwards sorting them by session ID. They then divided the data in training sessions (75%) and test sessions (25%). Next, the authors trained the model with the training sessions and predicted relevance for the test sessions. Finally, the authors sorted the documents taking into consideration the relevance, computing the NDCG@5 and then averaging the result over all sessions.

2.2.2.2 Predicting User Satisfaction

One problem of CTR analysis is that it may be inaccurate in some cases, due to caption bias (i.e., visual presentation of results) and/or to the order of results. The most used technique to solve this problem is to calculate the click dwell time, where a bigger dwell time on a clicked page indicates a satisfied user. Although click models are widely used, they are not enough to effectively predict user satisfaction since the user can be satisfied even if he does not click on any document, and be may click on a document and not be satisfied.

Hassan et al. (2013b) combined other signs besides clicks, to improve the effectiveness of the results (e.g., reformulations). The first base rule observed by the authors is if one finds a similar query (i.e., having at least one term in common) in an interval of less than 5 minutes, this implies dissatisfaction. The authors only use the queries immediately following the one that is being analyzed (i.e., satisfaction of the query will take into account a future action), and they also defined that the satisfaction at the level of clicks must be made through CTR-30 (i.e., at least one click with a dwell time of 30 seconds).

Hassan et al. (2013b) began by defining how to predict if a query was reformulated by another through the following steps. First, they normalized the queries (i.e., converting to lower case, normalize empty spaces, and remove any leading or trailing spaces). Then, they do query segmentation through the mutual information method (MI), based on computing the pointwise mutual information score (i.e., a correlation measure for two events) for each pair of consecutive words, with the following formula:

$$\mathrm{PMI}(x_i, x_{i+1}) = \log\left(\frac{\mathrm{p}(x_i, x_{i+1})}{\mathrm{p}(x_i) + \mathrm{p}(x_{i+1})}\right) \tag{2.42}$$

In the previous equation, $x = \{x_1, x_2, ..., x_n\}$, $\mathrm{p}(x_i, x_{i+1})$ is the probability of occurrence of the bi-gram (x_i, x_{i+1}), and $\mathrm{p}(x_i)$ and $\mathrm{p}(x_{i+1})$ are occurrence probabilities of individual tokens. The authors used the Microsoft Web n-gram service (Huang et al., 2010) to extract the n-grams and compute the probabilities. Finally, they created a set of rules for knowing whether two queries are semantically similar, defined from the most specific to the least specific: Exact Match (i.e., repetition); Approximate Match (i.e., solve the problem of misspellings using the Levenshtein edit distance with a threshold of 2); and Semantic Match, considering that two queries are semantically similar if the WordNet Wu Palmer measure (wup) is greater than 0.5. This measure considers WordNet synsets (i.e., sets of synonyms for each word) as a measure of similarity, with the following formula:

$$\mathrm{wup}(t_i, t_j) = \frac{2 \times \mathrm{depth}(LCS)}{\mathrm{depth}(t_i) + \mathrm{depth}(t_j)} \tag{2.43}$$

In the formula, LCS is least common subsumer (i.e., the LCS of concepts A and B is the most specific concept which is an ancestor of both A and B) and the depth of any synset is the depth of a node plus one. Finally, to compute the similarity between the two queries they used

Jaccard distance except that terms are considered identically if they can be matched using the Wu and Palmer measure.

Leveraging the previous ideas, the authors defined three main groups of features. The first are Textual Features, which consist on the following: if the normalized Levenshtein edit distance is greater than 2, return 1, otherwise return 0; the maximums number of characters in common starting from the left or starting on the right; the maximums number of words in common starting from the left or starting on the right; the number of words in common; and the Jaccard distance between the sets of words. Secondly, we have the following Keyword Features: the number of Exact Match keywords in common; the number of Approximate Match keywords in common; the number of Semantic Match keywords in common; the number of distinct keywords in the first query or in the second query; the number of keywords in Q1 but not in Q2, and vice-versa; and if the set of first query keywords contains all the second query's keywords and vice-versa. There are also Other Features, which consist on: the time between queries; the time between queries as a binary feature (i.e., if it is within 5 minutes, 30 minutes, 60 minutes, 120 minutes), and cosine distance between vectors derived from the top-10 search results for the query terms.

Finally, the authors presented five different systems based on the signals described on the previous paragraph. System 1 is based only on clicks (i.e., the query is satisfied if there is at least one click). System 2 is also based on clicks, using a combination of at least one click and 30 second dwell time. System 3 relies only on the rules of reformulation. System 4 is based on a combination between the reformulation rules and the information of the clicks, because some users could only make one query and multiple clicks and are dissatisfied (i.e., if there is no reformulation the authors use the information of the clicks). Finally, System 5 is based on System 4, but in this case the authors learn a classifier through the reformulation features and the click features to predict satisfaction.

Regarding evaluation experiments, the authors used a dataset from a commercial search engine with queries retrieval from a week in 2011. To evaluate the methods, the authors used the metrics of accuracy, precision, recall, and F-measure, in terms of satisfied and dissatisfied. System 1 presented the lowest accuracy and very low precision for satisfied queries, as well as a low recall for dissatisfied queries, being considered the worst system. System 2 showed satisfactory results, as all metrics are close to the 50%. System 3 had good results, with approximately 97% in terms of precision for dissatisfied queries, and recall for satisfied queries. System 4 showed no improvements when compared to System 3 and, on average, had worst results than System 3, although it had the highest recall for dissatisfied queries. Finally, System 5, had the best results on average (around 80%) compared to all the other systems, with the highest accuracy and F-measures.

Although System 5 had good results, as already mentioned in Section 2.2.1.1, a fixed threshold does not truly reflect each user's activity, because each user interacts differently with the system and takes a different time to do the tasks. Thus, Kim et al. (2014b) defined a method that produces a distribution of click dwell times depending on query-click attributes (e.g., topic, size and readability level). The model computes the dwell time distributions of satisfied and dissatisfied clicks, for different segments of clicks.

One problem that may occur is that some attributes (e.g., the type of query q being navigational) do not directly influence the dwell time of a given set of clicks, because it can be too general. To provide an answer to this problem, it is necessary to first associate different attributes to characterize a set of clicks (e.g., the query q having readability of 12 and the type being navigational), e.g. through a data mining approach, and then compute the distributions of satisfied and dissatisfied, with Maximum Likelihood Estimation (MLE), which find the parameter values that maximize a likelihood function given the observations.

To identify the dwell time distribution it is necessary to execute two steps. The first step is to generate a set of labels (i.e., satisfied or dissatisfied), through the association of the following attributes:

- **Query topic**, which consists of using Open Directory Project (ODP) to classify the topic of queries, using the clicked pages from a session (more details in Bennett et al. (2010)).
- **Query type**, which consists of inferring the need of the user, through training text classifiers to label the query-type categories of a query.
- **Page topic**, also based on ODP, but using the path of the URL to know which category it is (more details in White and Huang (2010)).
- **Reading level**, which consists in giving a classification between 1 and 12 (i.e., equal to American school grade levels). The author used a Multinomial Naïve Bayes classifier (more details in Collins-Thompson and Callan (2004)) to automatically identify the difficulty of a page through it is contents.

The C4.5 decision tree learning algorithm (Hall et al., 2009) is used to learn rules associating the different attributes by running the algorithm with different sets of features and resulting in rules like QueryTopic(Business) $\wedge$ RLV(12) (i.e., the clicks of the query are related to business and the difficulty is 12).

Next, the authors fit a regression model for the waiting time case using the Gamma distribution (Hogg et al., 2012). The main idea is to calculate the parameters of Gamma distribution that returns the ideal dwell time, where below the limit the query is satisfied and above the limit it is dissatisfied.

To cover all possible clicks, Kim et al. (2014b) defined an estimator to predict satisfied and dissatisfied dwell time distributions for new clicks (i.e., unseen), which identifies probabilistic distances of a click to satisfied and dissatisfied distributions. So, given a click segment, the authors collect satisfied and dissatisfied instances that belong to the segment, and create two lists of dwell times for each one. Then, for each list, the authors compute the values to fit the distribution with MLE through log-likelihood:

$$\ell(k,\theta) = (k-1)\sum_{i=1}^{n} \ln t_i - \frac{1}{\theta}\sum_{i=1}^{n} t_i - \ln\Gamma(k) - nk\ln\theta \tag{2.44}$$

In the previous equation, n is number of click segments, k is the shape and θ is the scale, which is given through the Gamma distribution used for model fitting. To find the maximum of θ the authors derivatives the Equation 2.44, resulting in the following formula:

$$\hat{\theta} = \frac{1}{kn}\sum_{i=1}^{n} t_i \tag{2.45}$$

Finally, to find the maximum of k, the authors substituted the Equation 2.45 into the Equation 2.44, in which the result is derivate, resulting in the following formula:

$$\ln k - \psi = \ln\left(\frac{1}{n}\sum_{i=1}^{n} t_i\right) - \frac{1}{n}\sum_{i=1}^{n} \ln t_i \tag{2.46}$$

In the previous equation, ψ is the diagamma function (i.e., the logarithmic derivative of the gamma function). To evaluate the proposed method, the authors used a human-labeled dataset with 7,500 queries, with 83% satisfied queries, proving that most queries are satisfied. Model training used a larger pseudo-labeled dataset, which was created with basis on following rules: if a click is followed by a reformulation then it is dissatisfied, otherwise it is satisfied, where the reformulation can be defined with features like edit distance or word overlap. This estimation depending only on reformation is not very reliable, but is it reliable enough in order to easily generate large datasets for model training.

The features of the page used for estimating the dwell time distribution are: PageSize (i.e., the size in bytes), PageLength (i.e., the number of the words), RenderTime (i.e., the time to render), HTML tag distribution, and the aforementioned QueryTopic, PageTopic and Readability features.

In their experiments, the authors defined three baselines with none or different sets of features. The first baseline was defined with only the dwell time of each click instance. The second baseline included all click attributes previously mentioned, as well as other search performance predictors as features. The third baseline was the combination of both. Subsequently, in each baseline, they added not only the dynamic dwell time features, but also features based in click segment rules. They analyzed the results and concluded that baseline 1 with dwell time and click segment rule features has an improvement of 26% in precision, comparatively to baseline 1 alone. The best results were obtained with baseline 2 plus the dwell time features, with 81% of accuracy.

Mehrotra et al. (2017b) proposed a method for predicting both query-level satisfaction and task-level satisfaction. According to Mehrotra et al. (2017b), one way to classify user satisfaction is through Search Engine Result Pages (SERPs), which reflect the interactions between the user and the system. Based on the sequences of interactions and with a deep sequential architecture, the authors predict the satisfaction of the query and the task.

The proposed method uses a combination of Recurrent Neural Networks (RNNs) (i.e., a type of neural networks designed for capturing information from sequences of data, allowing the classification of a sequence of actions according to a set of events occurred in the past) and parallel Convolutional Networks (CVNs). This combination is divided into two levels of abstraction. First, it is necessary to have an insight into how the user interacts with the system, and a deep sequential model is used to form a sequence of interactions. Second, the model uses signals as features (e.g., click and dwell time). To realize if the results are correct, crowdsourcing judgments are used as a form of evaluation.

The first step of this method is the creation of an action-LSTM, which takes as input a sequence of user actions ($x = (x_1, x_2, ..., x_T)$), calculating the hidden sequence ($h = (h_1, h_2, ..., h_T)$ and the output $y = (y_1, y_2, .., y_T)$, through the following formulas:

$$h_t = \mathcal{H}(W_{xh^{x_t}} + W_{hh^{h_{t-1}}} + b_h) \tag{2.47}$$

$$y_t = W_{hy^{h_t}} + b_y \tag{2.48}$$

In the equation, $t \in T$, where T is the total number of sequences, $W_{xh^{x_t}}$ represents the weight matrices between all input layers, $\mathcal{H}$ is the composite function, and b is a bias vector.

The action-LSTM is composed by action embeddings and Long Short-Term Memory (LSTM) sequence model (i.e., is a RNN capable of learning long-term dependencies). The Action Embeddings are used as inputs for the RNNs and are learned from the interaction sequence data, where given the set of action sequences, each action is insert into a continuous vector space using a skip-gram model, which is a method for learning distributed vector representations that capture a large number of precise syntactic and semantic word relationships proposed by Mikolov et al. (2013). The LSTM sequence model used by the authors is the bi-directional LSTM (BLSTM), which receive as input the action sequences describe before and takes into account past and future information, in order to do a retrospective offline satisfaction prediction. Firstly, a general LSTM model is composed by a cell state, which represents all the memory learned across time, a hidden layer representing the working memory, or current memory, a forget gate as a vector that represents what the LSTM should remember or not, and the input gate which determines how much input to go to cell state. The formulas of each state is:

$$i_t = \sigma(W_{xi^{x_t}} + W_{hi^{h_{t-1}}}) \tag{2.49}$$

$$f_t = \sigma(W_{xf^{x_t}} + W_{hf^{h_{t-1}}}) \tag{2.50}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc^{x_t}} + W_{hc^{h_{t-1}}}) \tag{2.51}$$

$$o_t = \sigma(W_{hx^{x_t}} + W_{ho^{h_{t-1}}} + W_{co^{c_t}}) \tag{2.52}$$

$$h_t = o_t \odot \tanh(c_t) \tag{2.53}$$

Where $\sigma(z) = (1 + e^{-z})^{-1}$ (i.e., sigmoid function), t is the current time step, i is the input gate, f is the forget gate, o is the output gate, c is the cell activation vector whose size is equal to hidden vector h and w is the weight matrices.

Secondly, in order to have a bi-directional LSTM Model it is necessary to define the forward hidden sequence $\overrightarrow{h}$ (i.e., to capture the past information) and backward sequence $\overleftarrow{h}$ (i.e., to capture the future information), and can be defined by the following formulas:

$$\overrightarrow{h}_t = \mathcal{H}(W_{x\overrightarrow{h}}x_t + W_{\overrightarrow{h}\overrightarrow{h}}\overrightarrow{h}_{t-1} + b_{\overrightarrow{h}}) \tag{2.54}$$

$$\overleftarrow{h}_t = \mathcal{H}(W_{x\overleftarrow{h}}x_t + W_{\overleftarrow{h}\overleftarrow{h}}\overleftarrow{h}_{t-1} + b_{\overleftarrow{h}}) \tag{2.55}$$

$$y_t = W_{\overrightarrow{h}y}\overrightarrow{h}_t + W_{\overleftarrow{h}y}\overleftarrow{h}_t + b_y \tag{2.56}$$

In the previous equation, $t \in T$, where T is the total number of sequences, and $\mathcal{H}$ is the composite function.

The second step, the authors proposed a coupled architecture composed of deep convolutional network and dense layers for modelling auxiliary features, consisting by the action-LSTM described before and a auxiliary feature module based on CVNs, which maps details of user interactions to their distributional vectors, in order to predict query-level user satisfaction. The CVNs receive as input the following features: Temporal signals: Page dwell time, Reading time per pixel, Viewport time per instance, Time to first pointer event, and Time to first scroll event; Click-based signals: Total click count, Algo click count, and Answer click count; Scroll & Pointer signals: Total scroll count Pointer horizontal distance, Pointer vertical distance, Pointer event count, Scroll Up count, Scroll down count, and Viewport direction changes.

In particularly, the convolution operator operates on sliding windows of signals, where if discrete input function $g(x) \in [1, l] \to R$ and discrete kernel function $f(x) \in [1, l] \to R$, then the convolution $h(x) \in [1, \lfloor (l-k)/d \rfloor + 1] \to R$ with stride d is defined through the following formula:

$$h(y) = \sum_{x=1}^{K} f(x) \cdot g(y \cdot d - x + c) \tag{2.57}$$

In the previous equation, c is an offset defined as $c = k - d + 1$. The convolutional layer passes its output to the pooling layer, which has the function of aggregating the information from the convolutional layer, where if discrete input function $g(x) \in [1, l] \to R$, the authors define a 1-D spatial max-pooling function $h(x) \in [1, \lfloor (l-k)/d \rfloor + 1] \to R$ through the following formula:

$$h(y) = max_{x=1}^{k} g(y \cdot d - x + c) \tag{2.58}$$

Next, in order to the network learn non-linear decision boundaries it is necessary to define a non-linear activation function, which will be applied to the output of the preceding layer. In particularly, the authors used the rectified linear units (ReLU) as activation function, which is defined by the following formula:

$$h(y) = max\{0, x\} \tag{2.59}$$

Additional, there is an extra hidden layer for modelling interactions with different views of user interactions defined by $\alpha(w_h \cdot x + b)$ where $\alpha()$ is the ReLU non-linearity function, and w_h is the weight vector.

Finally, the authors defined a Softmax Layer, which receive the output of the convolutional and pooling layers and calculates the probability distribution over the labels through:

$$p(y = j|x) = \frac{e^{x^T \theta_j}}{\sum_{k=1}^{K} e^{x^T \theta_k}} \tag{2.60}$$

where θ_k is a weight vector of the k-th class and x represents the final abstract representation of the input example resulting from a series of transformations.

Lastly, after propose a model which predicts query-level satisfaction, the authors defined four operations that based on the query can measure the satisfaction of the task: Maximum operation, which returns the satisfaction of a user depending on the best satisfaction given to a query of the same task, being a tolerant approach; Average operation, which consists of averaging the prediction of all the queries of the same task; Differentiation of weight operation, consisting in giving different weights to each query depending on whether or not a query has been reformulated (i.e., if a query has been reformulated it will have a higher weight than the original query); and Minimum operation consisting of extracting the minimum satisfaction between the queries of the same task, being a rigid approach. Although the minimum is more realistic than the maximum, since if a query was not satisfied then the task will not be either, it is the maximum that has better results. The subtasks estimation was based on a nested function composition which computed recursively the satisfaction of subtasks.

Through experiments, the authors concluded that the unified multi-view method presents the best results at query-level satisfaction, with about 5% over the best baseline (i.e., just considering the clicks and dwell time features), although the baseline had the highest precision. At task-level satisfaction, the authors concluded that the maximum operation combined with the unified multi-view method had the best results, concluding that knowing future actions helps to understand the user interaction.

3 Unsupervised Segmentation of Search Sessions and Search Missions

This chapter addresses the problem of segmenting user sessions and missions from user interactions registered in search engine query logs. I propose a new unsupervised approach combining multiple heuristics, evaluating it against a set of baselines that covers the current state-of-the-art. An ablation analysis was also considered, checking the impact of temporal, lexical, and semantic similarity heuristics, in the overall methods that integrate them. First, Section 3.1 describes the individual heuristics that were considered for the ablation tests, and which are also the main components of the complete method. Then, the complete method for segmenting user sessions, detailing some of the components that are involved (e.g., the use of word embeddings) is presented in Section 3.2. Section 3.3 describes an extension to previous unsupervised approach, which addresses the problem of identifying search missions. Next, Section 3.4.1 describes a statistical characterization of AOL datasets that supported the tests, together with the considered experimental methodology. Section 3.4.2 presents and discusses the obtained results. Finally, Section 3.5 overviews the contents of this chapter.

3.1 Individual Heuristics and Ablated Approaches for the Segmentation of User Sessions

In terms of temporal heuristics, the tests with baselines and ablated models for search session segmentation considered two different approaches. The first relies on a global temporal threshold, in which two consecutive queries belong to the same user session if the elapsed time is less than a pre-defined threshold. With basis on previous studies, I tested the standard values of 5 (Downey et al., 2007), 15 (He and Göker, 2000), and 30 minutes (Radlinski and Joachims, 2005). The second approach considers a user-specific temporal threshold, defined with basis on the user distribution of intervals between consecutive queries, as originally proposed by Mehrzadi and Feitelson (2012).

In terms of lexical and semantic heuristics, the ablation tests considered three different approaches. The first is based on the Jaccard similarity coefficient between the sets of 3- and

4-grams extracted from the new query, and from all the queries in the session of the previous query. The second is based on pre-trained FastText embeddings (Bojanowski et al., 2017), computing the cosine similarity between averaged embedding vectors for the words present in the consecutive queries. Finally, the third approach also relies on FastText embeddings, but in this case I use the word mover's distance (Kusner et al., 2015) to assess the similarity between sets of embeddings, respectively for words in the new query, and for words in all the queries in the session of the previous query. I used an existing implementation[1] for computing the earth mover's distance between word embeddings. In all three approaches, I tested different thresholds (i.e., between 0.1 and 0.9) in the obtained similarity value. The lexical similarity computations did not involve language-specific lists of stop-words or stemming algorithms, although I ignored punctuation symbols and sub-strings such as `www.` or `.com` (i.e., queries often contain URLs, and I ignored common URL tokens from the string similarity computations).

In terms of combinations for multiple heuristics, besides the complete approach, I also tested three different baseline methods. The first was the geometric method from Gayo-Avello (2009), with the same parameters proposed by the author. The second baseline corresponds to an improved version of the geometric method, which instead of using the overlap between character 3-grams, in the lexical component, uses the Jaccard similarity coefficient between sets of character n-grams of lengths 3 and 4, extracted from the last query and from all the queries in the session of the previous query. In the improved version of the geometric method, I also changed the temporal component given by $\max\left\{0, 1 - \frac{t_{i+1}-t_i}{24\text{ hours}}\right\}$, using a normalization constant equal to the minimum between 24 hours or twice the maximum time between consecutive queries for the user under analysis, instead of the fixed normalization constant of 24 hours. Finally, the third baseline method corresponds to a slightly different procedure from the complete method described in Section 3.2, using only the cosine similarity between averaged word embeddings for consecutive queries, instead of using the word mover's distance.

In the methods combining multiple heuristics that use a threshold over the Jaccard similarity coefficient between character n-grams (although not on the lexical baselines that use the Jaccard coefficient alone), I used a simple two-step approach to improve the computational performance, based on the intuition that a fast lower-bound for the similarity can be computed from the length of common prefixes and/or suffixes. First, notice that for a non-empty string of size k, the maximum number of distinct n-grams is given by $\max(1, k-(n-1))$. For two strings in which one is a prefix or a suffix of the other (i.e., strings resulting from a typical reformulation pattern,

[1]`https://radimrehurek.com/gensim/models/keyedvectors.html#gensim.models.keyedvectors.WordEmbeddingsKeyedVectors.wmdistance`

Algorithm 1 The Proposed Search Session Segmentation Method

```
1: Sort the log that registers user interactions using the userID as a first criterion, and then using the
   timestamp
2: Initialize each query in the log as belonging to a separate search session
3: for each user $u$ do
4:     $t_{\max_u}$ = Maximum time between consecutive queries for user $u$
5:     for each pair of consecutive sessions $i$ and $i+1$ from the same user $u$ do
6:         $f_t = \max\left\{0, 1 - \frac{t_{i+1}-t_i}{\min\{24 \text{ hours}, 2\times t_{\max_u}\}}\right\}$
7:         $f_{l_1}$ = Lower-bound on the Jaccard similarity coefficient
8:         if $f_{l_1} > \sqrt{1-f_t^2}$ then
9:             Merge the two consecutive sessions into a single one
10:        else
11:            $f_{l_2}$ = Jaccard similarity based on character $n$-grams with the parameter $n \in \{3,4\}$
12:            if $\sqrt{f_t^2 + f_{l_2}^2} > 1$ then
13:                Merge the two consecutive sessions into a single one
14:            else
15:                if $f_t > 0.7$ and $f_{l_2} < 0.5$ then
16:                    $f_{s_1}$ = Cosine similarity from averaged word embeddings
17:                    if $f_{s_1} > 0.5$ then
18:                        Merge the two consecutive sessions into a single one
19:                    else
20:                        $f_{s_2}$ = Word mover's distance (Kusner et al., 2015) from sets of word embeddings
21:                        if $f_{s_2} < 0.1$ then
22:                            Merge the two consecutive sessions into a single one
23:                        else
24:                            if $\sqrt{f_{s_1}^2 + (1.0 - f_{s_2}^2)} > 1$ then
25:                                $f_u$ = Similarity from largest common sub-strings in the URLs
26:                                if $f_u > 0.7$ then
27:                                    Merge the two consecutive sessions into a single one
```

corresponding to the addition or removal of terms from the search query), in which the size of the common sub-string is k_1 and the length of the longer string is k_2, the number of n-grams in common cannot be higher than $\max(1, k_1 - (n-1))$, and the number of distinct n-grams cannot be lower than the number of n-grams in common, or higher than $\max(1, k_2 - (n-1))$. The ratio between these two quantities gives an approximation on the Jaccard similarity coefficient, that I can use as a lower-bound. This procedure will, in some cases, lead to wrong lower-bound estimates when there are many n-grams appearing repeated in the strings. However, in such cases, the strings under comparison will still have a significant match in their contents, and I can use the estimate in a way that is similar to the query reformulation patterns in the cascade method (Hagen et al., 2011).

In the combined methods, when checking if the Jaccard similarity coefficient is above a given threshold, I first check if the lower-bound (computed with basis on the similarity towards the last query) is greater than the threshold, and only if this is not enough to reach a decision, the actual similarity coefficient is computed. Notice that $\sqrt{f_t^2 + f_l^2} \geq 1$ from the geometric method corresponds to a minimum threshold of $\sqrt{1 - f_t^2}$ on the (lower-bound to the) Jaccard similarity.

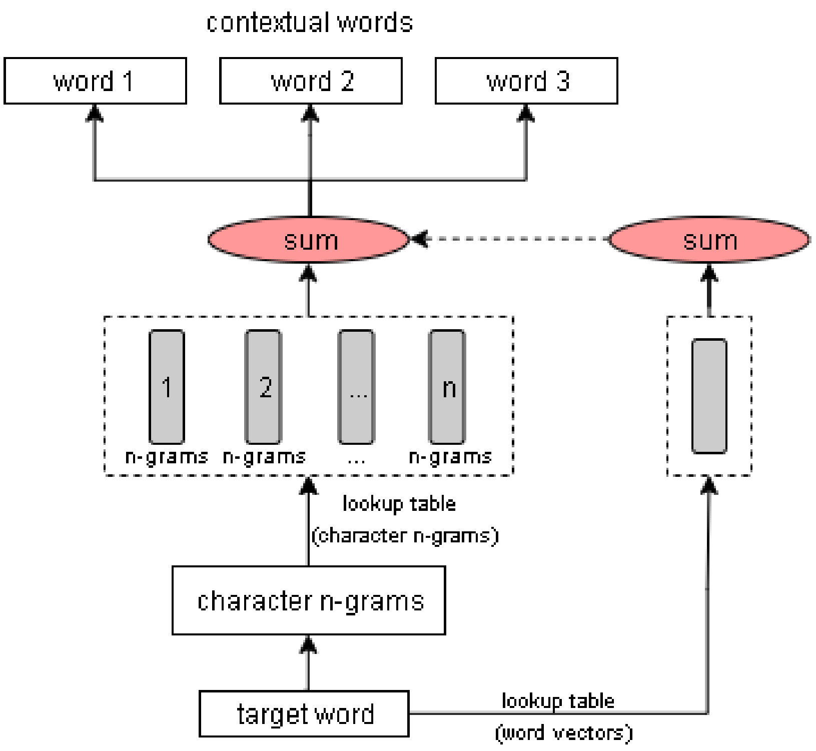


Figure 3.1: The training of FastText embeddings.

3.2 The Complete Approach for Session Segmentation

The complete method for segmenting user sessions corresponds to a cascade approach, extending the geometric method and the approaches from Hagen et al. (2011, 2013) through the use of FastText word embeddings. The complete method consists of the following three steps, and it can also be summarized through the pseudo-code that is shown in Algorithm 1.

First, I use the aforementioned improved version of the geometric method, relying on a per-user maximum threshold for the temporal component, using the fast lower-bound on the Jaccard similarity coefficient, and using character 3- and 4-grams in the lexical component. A new query will belong to the session of the previous query if an adapted version of $\sqrt{f_t^2 + f_l^2} \geq 1$ (i.e., using the condition greater than one, instead of greater or equal to one, thus ensuring that equal queries separated by very large time spans are not merged) is satisfied, and otherwise different sessions will be considered. Although effective on its own, this step fails at capturing semantic similarities (i.e., it incorrectly classifies many cases involving queries in close temporal proximity, but with a low lexical similarity). Adapting the cascade method from Hagen et al. (2011, 2013), if the temporal proximity is above the threshold of 0.7, and the lexical similarity is below the threshold of 0.5, I attempt to merge the queries into the same session according to the results of the second step.

In the second step, starting on Line 16 from Algorithm 1, I use pre-trained FastText embeddings (Bojanowski et al., 2017) to quickly assess the semantic similarity of two consecutive queries/sessions. The FastText approach, which involves training vector representations as illustrated in Figure 3.1, has been shown to perform well in representing words (e.g., it has been used on models for sentence classification as a down-stream task leveraging word embeddings), especially in the case of rare words, by making use of character level information. Each word is seen as a bag of character n-grams, in addition to the word itself. During model training, FastText learns weights for each of the n-grams, as well as for the entire word tokens. Rare words can be properly represented, since it is highly likely that some of their n-grams also appear in other words, and even out-of-vocabulary words can be represented, by taking the average of the embeddings for the corresponding n-grams.

I start by measuring the cosine similarity between averaged word embeddings for the consecutive queries. If the similarity is above the threshold of 0.5, then the queries are defined to belong to the same session. Otherwise, I compare the set of embeddings for the words in the query, against the set of embeddings corresponding to words in all the queries belonging to the session of the previous query. In this second case, a fast algorithm for computing the word mover's distance (Kusner et al., 2015) is used to compare the sets of embeddings and, if the resulting distance is lower than 0.1, I assume that the two consecutive queries belong to the same session (otherwise different sessions will be considered). The Word Mover's Distance (WMD) is a special case of the well-studied earth mover's distance transportation problem, measuring the dissimilarity between two sets of embeddings as the minimum amount of distance that the embeddings of one set need to travel to reach the embeddings of another set.

In the third step, if the decision remains unreliable (i.e., if the similarity scores from the previous step are within particular thresholds, also according to a geometric interpretation) I will compare clicked URLs through the longest sub-strings in common. I assume that the query log under analysis contains information on clicked URLs (i.e., for each query, if the user accessed one of the URLs in the search results, then the corresponding URL is registered on the log together with the query). I first normalize the URLs by removing redundant information, including prefixes corresponding to protocol specifications (e.g., `http://` or `https://`), sub-strings corresponding to top-level domain names (e.g., `.com`, `.org` or `.edu`), or suffixes corresponding to popular file extensions (e.g., `.html`, `.jsp` or `.php`). Then, I compute the longest common sub-strings between any of the normalized URLs associated with the new query, and any of the normalized URLs associated to queries in the same session as the previous query. If any of these longest common sub-strings has a length that is at least 70% of the length of one of

Table 3.1: Results for a sample of queries taken from the subset of the 2006 AOL query log hat was released by Gayo-Avello (2009).

A. Segmentation from Human Annotator

Query	URL	Time
teeth like god's shoeshine lyrics	www.selyrics.com	1142351220
grills lyrics		1142369580
grills lyrics nelly	www.lyrics07.com	1142369580
blink 182 lyrics	www.azlyrics.com	1142371620
edit the sad parts lyrics	www.selyrics.com	1142372820
my lips are cold the truth is told lyrics	www.lyricsdepot.com	1142375940
the authority song		1142449620
the authority song lyrics	www.selyrics.com	1142449680
black dresses by spill canvas	www.azlyrics.com	1142449980
playing for keeps lyrics	www.lyrics07.com	1142450040
rhyming dictionary	www.rhymer.com	1142452560
monstr in a wheelchair		1142465880

B. Segmentation Resulting from Step 1

Query	URL	Time
teeth like god's shoeshine lyrics	www.selyrics.com	1142351220
grills lyrics		1142369580
grills lyrics nelly	www.lyrics07.com	1142369580
blink 182 lyrics	www.azlyrics.com	1142371620
edit the sad parts lyrics	www.selyrics.com	1142372820
my lips are cold the truth is told lyrics	www.lyricsdepot.com	1142375940
the authority song		1142449620
the authority song lyrics	www.selyrics.com	1142449680
black dresses by spill canvas	www.azlyrics.com	1142449980
playing for keeps lyrics	www.lyrics07.com	1142450040
rhyming dictionary	www.rhymer.com	1142452560
monstr in a wheelchair		1142465880

C. Segmentation Resulting from Step 2

Query	URL	Time
teeth like god's shoeshine lyrics	www.selyrics.com	1142351220
grills lyrics		1142369580
grills lyrics nelly	www.lyrics07.com	1142369580
blink 182 lyrics	www.azlyrics.com	1142371620
edit the sad parts lyrics	www.selyrics.com	1142372820
my lips are cold the truth is told lyrics	www.lyricsdepot.com	1142375940
the authority song		1142449620
the authority song lyrics	www.selyrics.com	1142449680
black dresses by spill canvas	www.azlyrics.com	1142449980
playing for keeps lyrics	www.lyrics07.com	1142450040
rhyming dictionary	www.rhymer.com	1142452560
monstr in a wheelchair		1142465880

D. Segmentation Resulting from Step 3

Query	URL	Time
teeth like god's shoeshine lyrics	www.selyrics.com	1142351220
grills lyrics		1142369580
grills lyrics nelly	www.lyrics07.com	1142369580
blink 182 lyrics	www.azlyrics.com	1142371620
edit the sad parts lyrics	www.selyrics.com	1142372820
my lips are cold the truth is told lyrics	www.lyricsdepot.com	1142375940
the authority song		1142449620
the authority song lyrics	www.selyrics.com	1142449680
black dresses by spill canvas	www.azlyrics.com	1142449980
playing for keeps lyrics	www.lyrics07.com	1142450040
rhyming dictionary	www.rhymer.com	1142452560
monstr in a wheelchair		1142465880

the URLs for the new query, then the queries are considered to belong to the same user session, and otherwise a different user session will be considered for each of the search queries.

Tables 3.1.A to 3.1.D illustrate the obtained results at the different steps of the algorithm, for a sample of queries taken from the AOL query log (i.e., a sequence of queries for the same user, that made system interactions generally related to the topic of music). Table 3.1.A shows the segmentation boundaries made by a human annotator, whereas Tables 3.1.B to 3.1.D show the boundaries resulting from each step of the algorithm, progressively reconstructing the same decisions as the human annotator. For instance, Table 3.1.C shows that by considering semantic similarity based on word embeddings, one can almost reconstruct the first session from Table 3.1.A. Step 3 effectively refines the results by leveraging URLs, although the method still failed at joining the last two iterations on the table. In the example that is presented, the user clicked at most in one URL for each query.

3.3 The Proposed Approach for Grouping Queries According to Search Missions

In the context of query log analysis, and in addition to user sessions, considering search missions can provide a more general view of usage patterns. Search missions should group user sessions in a way that encodes (i) multitasking behavior (i.e., I should identify non continuous interactions that are broken across different sessions, but that refer to the same underlying task or more general information need) and (ii) hierarchical sub-tasks with different purposes, but that contribute to achieving the same more general goal. For instance, when booking a vacation, many people check out restaurants, flights, and hotels. All these different queries may be grouped in different user sessions, that nonetheless relate to the same underlying mission.

To identify user search missions, I propose an extension to the unsupervised method from the previous section, in which Algorithm 1 is considered as the first step. Thus, assuming that the queries are already grouped into user sessions, through Algorithm 1, I will leverage a separate and similar cascade approach to merge user sessions into search missions.

Contrary to what has been reported by other studies, I consider that the temporal component is a valid feature for identifying search missions, since similar goals, even if not continuous, will likely be close in time. Nevertheless, the application of the same threshold for all contexts can be noisy, since different users may behave differently in each situation.

The approach analyzes pairs of sessions s and s' belonging to the same user, not necessarily contiguous, using an adapted version of the method to identify user sessions, that in the first steps only considers the last query q of s and the first query q' of a subsequent session s'. A perhaps more reasonable approach would involve always comparing q' against all queries from s. However, the proposed approach offers a better comprise between algorithm accuracy and computational efficiency.

As a first step for grouping search sessions into missions, I use the improved version of the geometric method, considering only the last query q of s and the first query q' of s' on the lexical component, and last timestamp t of s together with the first timestamp t' of s' on the temporal component. However, instead of using a global threshold of 24 hours, I propose to use a more relaxed threshold of 48 hours.

In the second step, to capture semantic similarity, I adapted the cascade method from Hagen et al. (2011, 2013). If the temporal similarity is greater than 0.5 and the lexical similarity is less than 0.7, I calculate the cosine similarity between average embedding vectors for queries q and

q'. If the returned value from the cosine similarly is greater than 0.5, sessions s and s' will be grouped in the same search mission. If the decision is still unreliable (i.e., for similarities below 0.5), I calculate the word mover's distance between sets of embedding vectors for all queries in session s, and all queries from session s'. If the value is less than 0.3, sessions s and s' will be grouped in the same search mission.

Finally, in a third step, I compare the clicked URLs associated to queries q and q', through the longest sub-strings in common, as described in the previous section. If the longest common sub-string has a length that is at least 70% of the length of one of the URLs for the more recent query q', then the sessions s and s' will be grouped in the same mission. Notice that, instead of using all URLs from s', I am only considering URLs from query q', which can nonetheless be more than one.

In the example from Table 3.1.A, there are two sessions and only one search mission, since all interactions can be said to correspond to the same encompassing information need related to music lyrics. The proposed method would correctly identify the single search mission that is present in the example.

3.4 Experimental Evaluation

This section describes the experimental evaluation of the proposed methods. I first present a statistical characterization of AOL datasets that supported the tests, together with the considered experimental methodology. Then, Subsection 3.4.2 presents and discusses the obtained results for both approaches.

3.4.1 Datasets and Experimental Methodology

The datasets used in the experiments correspond to two subsets of the AOL query log released on August 2006 (Pass et al., 2006). To ensure a meaningful comparison against previously published results for session segmentation, I used the subsets of the AOL query log, with ground-truth annotations regarding sessions, that were respectively made available by Gayo-Avello (2009) and by Hagen et al. (2013). Both datasets were used in several previous studies in the area. The subset from Gayo-Avello (2009) has a total of 10,235 queries from 215 unique users, which are divided into 4,253 sessions with an average of 2.4 queries per session. The subset made available by Hagen et al. (2013) also contains ground-truth annotations for user missions, and it contains a total of 8,840 queries, with 2,881 sessions, 1,378 missions, 3.1 queries per session, 6.42 queries per mission, and 127 unique users. This second subset was used in the experiments to evaluate both session and mission segmentation.

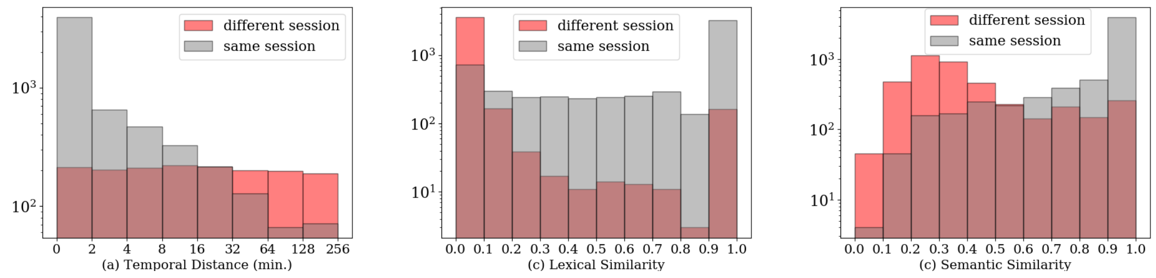
In both datasets, each record contains the following attributes: (i) userID (i.e., a unique user identifier); (ii) query text (i.e., the set of keywords submitted by the user); (iii) URL (i.e., the URL that the user clicked after receiving the results for the query, or empty if no clicks were made); (iv) timestamp (i.e., the instant when the user submitted the query); and (v) session boundary (i.e., a Boolean indicator for whether the query marks the beginning of a new session, according to the ground-truth annotations). The dataset made available by Hagen et al. (2013) has the following additional attributes: (i) click rank (i.e., the position in the list of search results for the document being clicked), and (ii) mission boundary (i.e., a unique search mission identifier). In both methods, the records (i.e., the user queries) are first sorted according to userID (i.e., joining together queries from the same user), and then sorted according to the timestamp, prior to analysis.

To better understand each dataset, I first looked at consecutive queries from the same users, judged by the human annotators as belonging or not to the same search session or mission. The distributions for several characteristics associated to these consecutive queries are depicted in Figure 3.2, which shows side-by-side the distribution for (a) the temporal proximity in minutes, (b) the lexical similarity according to the Jaccard coefficient between character n-grams (i.e., 3- and 4-grams) from consecutive queries, and (c) the semantic similarity according to the cosine similarity between averaged word embeddings of consecutive queries. The figure shows that all three heuristics have different distributions for the consecutive queries in each of the four classes (i.e., same versus different sessions, and same versus different missions, respectively), although the three heuristics seem to capture different cases. Through the experiments, I attempted to assess the contribution of each heuristic.

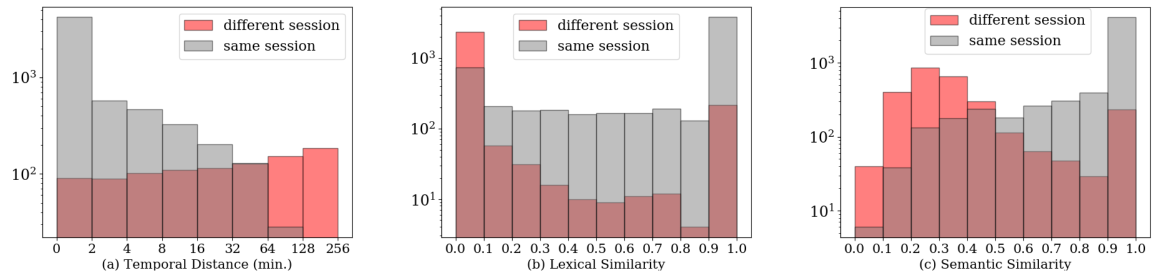
To evaluate the proposed approach for session segmentation, I relied on the same evaluation methodology and metrics considered by Gayo-Avello (2009), corresponding to notions of precision and recall. Precision is defined as the ratio between the number of consecutive queries for which there is a change of session where the algorithm has agreed with the ground-truth, and the number of consecutive queries for which the algorithm predicted a change of session. Recall, on the other hand, is defined as the ratio between the number of consecutive queries for which there is a change of session where the algorithm agreed with the ground-truth, and the number of consecutive queries corresponding to a session change in the ground truth.

On the other hand, to evaluate the proposed approach for identifying search missions, I used

Distributions for the number of queries according to search sessions in the dataset from Gayo-Avello (2009).



Distributions for the number of queries according to search sessions in the dataset from Hagen et al. (2013).



Distributions for the number of queries according to search missions in the dataset from Hagen et al. (2013).

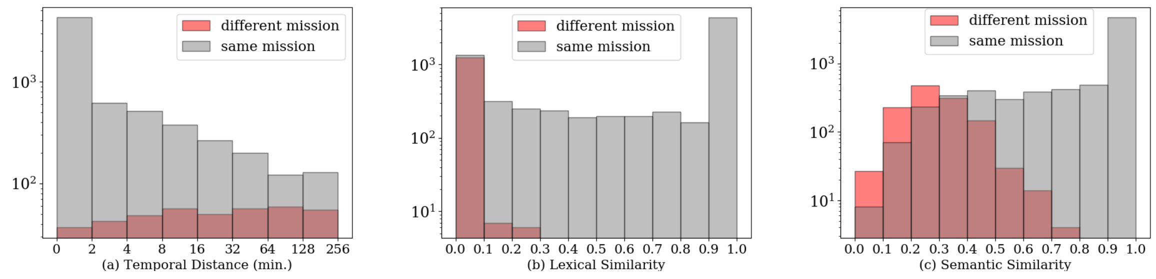


Figure 3.2: Comparison of the distribution for the number of consecutive queries, according to (a) temporal proximity/distance, (b) lexical similarity, and (c) semantic similarity. The top row corresponds to search sessions in the subset that was released by Gayo-Avello (2009), the middle row corresponds to search sessions in the subset that was released by Hagen et al. (2013), and the bottom row corresponds to the search missions also in the subset released by Hagen et al. (2013)

the B^3 clustering metrics discussed by Amigó et al. (2009). The mission groupings generated by the proposed algorithm are seen as clusters of queries, which I compare against ground-truth clusterings corresponding to the missions that are present in the annotated dataset.

Being $\mathrm{R}(q)$ and $\mathrm{P}(q)$ respectively the real and predicted mission groups for a query q, I can define the correctness of the relation between queries q and q' as:

$$\text{Correctness}(q, q') = \begin{cases} 1 & \text{if } \mathrm{R}(q) = \mathrm{R}(q') \longleftrightarrow \mathrm{P}(q) = \mathrm{P}(q') \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

The B^3 precision of a query is the proportion of correctly related queries in the predicted mission group for the query (including the query itself), and the overall B^3 precision is the averaged precision of all queries. The B^3 recall is analogous, replacing the predicted mission group by the true mission group. An F1-Score can also be computed as usual, i.e. by taking the harmonic mean of B^3 precision and recall.

Besides assessing the quality of the predictions, I also measured the time involved in processing the entire subsets of the AOL query log. The measurements for the different methods were all made in a standard PC with an Intel Core i7 8700K (3.7 GHz) CPU, an SSD drive where the log file was stored, and 64 Gb of RAM.

3.4.2 Experimental Results

I compared the different baseline approaches for session segmentation that are listed in Section 3.1, against the complete method given in Section 3.2. Table 3.2 presents the obtained results for segmenting sessions, both over (a) the subset of the AOL query log from Gayo-Avello (2009), and (b) the subset of the 2006 AOL query log from Hagen et al. (2013). The table shows that the complete procedure outperforms all the considered baselines in terms of the F1-score, although also with a higher computation time.

The results illustrate that methods based on a global temporal threshold already achieve a very satisfactory performance for session segmentation (i.e., F1-scores of 82.40 and of 87.81, in the datasets from Gayo-Avello (2009) and Hagen et al. (2013), when using thresholds of 15 and 30 minutes, respectively), at the same time also being faster. Relying on user-specific temporal thresholds is not much slower, although it failed to outperform the results obtained with a global threshold. When using a lexical similarity heuristic alone, the results over the dataset from Gayo-Avello (2009) are slightly better than those obtained with a temporal threshold, although the

Table 3.2: Performance metrics for different user session segmentation methods (a) over the subset of the 2006 AOL query log that was released by Gayo-Avello (2009), and (b) over the subset of the 2006 AOL query log that was released by Hagen et al. (2013).

Component	Method	Limit	Gayo-Avello (2009)			Hagen et al. (2013)			Execution Time (m.sec)
			Precision	Recall	F1-Score	Precision	Recall	F1-Score	
Temporal	Global Threshold	$T = 5$	77.00	87.54	81.93	75.21	91.91	82.72	1615
		$T = 15$	84.91	80.04	82.40	87.40	86.43	86.91	
		$T = 30$	89.00	75.12	81.47	93.70	82.61	87.81	
	Threshold per User		90.68	71.15	79.74	**96.37**	79.35	87.04	1975
Lexical or Semantic	Jaccard Coefficient	>= 0.1	83.67	92.50	87.86	77.24	87.16	81.90	3090
		>= 0.3	75.42	94.59	83.93	69.38	90.35	78.49	
		>= 0.5	69.49	95.18	80.33	63.94	91.15	75.16	
		>= 0.7	64.26	95.67	76.88	59.34	91.84	72.10	
		>= 0.9	60.62	96.10	74.34	55.56	92.40	69.40	
	Word Embeddings Cosine	>= 0.1	**95.87**	7.64	14.16	95.29	6.32	11.85	4090
		>= 0.3	90.58	54.03	67.69	88.91	49.81	63.85	
		>= 0.5	84.82	88.53	86.63	80.07	82.99	81.51	
		>= 0.7	76.05	93.98	84.07	71.22	89.17	79.19	
		>= 0.9	65.16	95.65	77.52	60.32	91.81	72.80	
	Word Embeddings WMD	<= 0.1	61.49	95.79	74.90	56.04	92.29	69.74	6049
		<= 0.3	64.82	92.23	76.47	59.03	91.46	71.75	
		<= 0.5	69.45	92.55	79.35	63.80	90.91	74.98	
		<= 0.7	74.42	90.24	81.57	68.76	89.55	77.79	
		<= 0.9	79.25	87.11	83.00	73.72	86.78	79.72	
Temporal + Lexical and Semantic	Geometric Method (GM)		88.24	88.90	88.57	83.16	87.75	85.39	3542
	Improved GM		83.93	**97.60**	90.25	79.38	**95.28**	86.61	2287
	Proposed Method		88.06	95.20	91.49	84.37	92.36	88.19	4983
	Proposed Method (WMD)		88.19	95.13	**91.53**	84.49	92.29	**88.22**	5771

semantic similarity heuristics (i.e., both methods relying on word embeddings) alone perform slightly worse. On the dataset from Hagen et al. (2013), the results with the lexical and semantic heuristics alone are clearly worse than those achieved with temporal thresholds. Moreover, the lexical and semantic approaches are also slower compared to temporal approaches.

In terms of the combined methods, the proposed approach outperforms all individual baselines in both subsets, the simpler geometric method, and the variants that were considered, although at the cost of higher computation time. The improved geometric method, and a variation of the proposed method that does not use the word mover's distance, both offer a good compromise between result quality and computation time.

Table 3.3 shows the results for mission identification over the subset of the 2006 AOL query log proposed by Hagen et al. (2013), confirming that the complete approach to identify search missions also outperforms the baselines in terms of the F1-score. The different rows in Table 3.3 correspond to alternative heuristics (or sequences of heuristics) for detecting search missions, in all cases relying on the best approach (i.e., the complete method that achieved the higher F1-score) for identifying search sessions. The column with the execution time refers to the time involved in segmenting search sessions, together with the time involved in grouping search sessions into missions. On average, more than 80% of the overall execution time relates to grouping search sessions into missions.

Methods based on a global temporal threshold already achieve good results (i.e., an F1-score

Table 3.3: Performance metrics for different user mission segmentation methods, as obtained over a subset of the 2006 AOL query log released by Hagen et al. (2013).

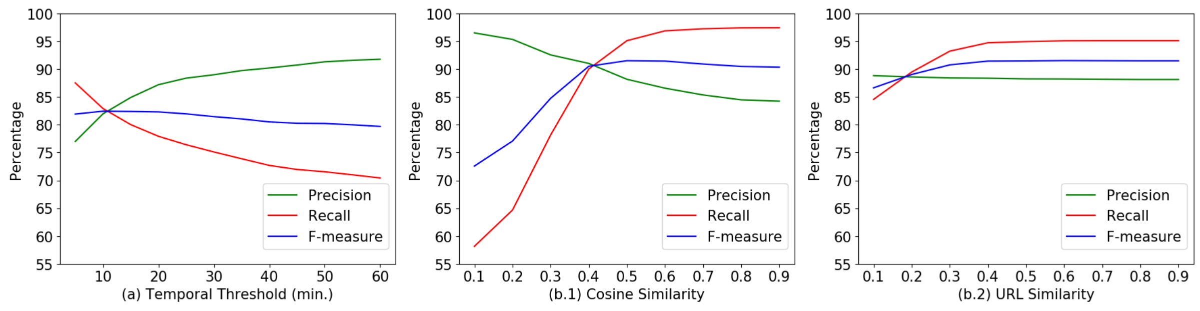
Component	Method	Limit	Identified Sessions			Execution
			B^3Precision	B^3Recall	B^3F1-Score	Time (m.sec)
	No Heuristics (Use the Sessions)		83.49	65.41	73.35	5798
Temporal	Global Threshold	$T = 5$	81.03	68.12	74.01	6987
		$T = 15$	77.96	70.21	73.88	
		$T = 30$	75.86	70.69	73.19	
	Threshold per User		76.44	71.52	73.90	8024
Lexical or Semantic	Jaccard Coefficient	>= 0.1	72.92	72.81	72.86	9058
		>= 0.3	75.20	70.10	72.80	
		>= 0.5	78.00	70.16	73.87	
		>= 0.7	80.14	69.90	74.67	
		>= 0.9	79.14	69.01	73.73	
	Word Embeddings Cosine	>= 0.1	58.16	**79.03**	67.01	11587
		>= 0.3	67.61	74.69	70.97	
		>= 0.5	74.49	70.80	72.59	
		>= 0.7	76.27	70.80	73.44	
		>= 0.9	77.86	69.69	73.55	
	Word Embeddings WMD	<= 0.1	77.27	69.99	73.45	13587
		<= 0.3	77.27	69.45	73.15	
		<= 0.5	76.80	69.67	73.06	
		<= 0.7	76.16	69.93	72.91	
		<= 0.9	73.93	70.99	72.43	
Temporal + Lexical and Semantic	Geometric Method (GM)		80.94	68.89	74.43	9568
	Improved GM		**83.52**	65.35	73.32	8679
	Proposed Method		80.15	70.74	**75.15**	12693
	Proposed Method (All Queries)		77.51	68.72	72.85	19023

of 74.01, when using a global threshold of 5 minutes), being also faster. Relying on user-specific temporal thresholds is not much slower, although this failed to outperform the results obtained with a global threshold. On the other hand, depending on the threshold, lexical heuristics alone can perform slightly better than the results obtained with temporal thresholds, while semantic heuristics alone are slightly worse. The proposed combination method achieves a higher accuracy than competing unsupervised approaches, while not significantly expanding the computational effort.

Unexpectedly, the improved version of the geometric method does not achieve good results on its own. In the last method that is shown in Table 3.3, in each step of the complete approach, I compare all the queries from session s against all queries from session s', confirming that comparing all queries brings noise and is much slower than previous methods. Finally, the geometric method offers a good compromise between result quality and computation time.

Figure 3.3 further details the results for session segmentation in both datasets, plotting the variation on precision, recall, and the F1-score (a) for a baseline corresponding to a global temporal threshold, as a function of that threshold, and (b) for the complete method, as a function

Variation in performance for session segmentation in the data subset from Gayo-Avello (2009).



Variation in performance for session segmentation in the data subset from Hagen et al. (2013).

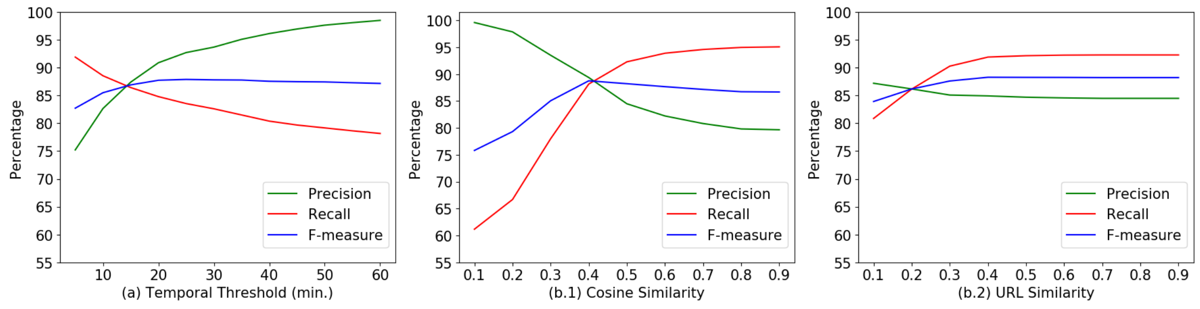


Figure 3.3: Variation in precision, recall, and F1-scores for session segmentation, as a function of thresholds in both datasets.

of the threshold (b.1) on the cosine similarity between sets of word embeddings, or (b.2) on the similarity between URLs in both approaches (i.e., user sessions and missions segmentation).

The results on the subset that was released by Gayo-Avello (2009), shown in chart (a), confirm that a temporal threshold of approximately 15 minutes corresponds to the best trade-off in terms of the F1-score, while Charts (b.1) and (b.2) show that the thresholds that were given in Section 3.2, in connection to the proposed method, are also adequate. On the other hand, the results on the subset that was released by Hagen et al. (2013) illustrate that each threshold could be better adjusted to this dataset. For instance, if I set the cosine similarity threshold to 0.4 I would achieve an F1-score of 88.76 for session segmentation.

3.5 Overview

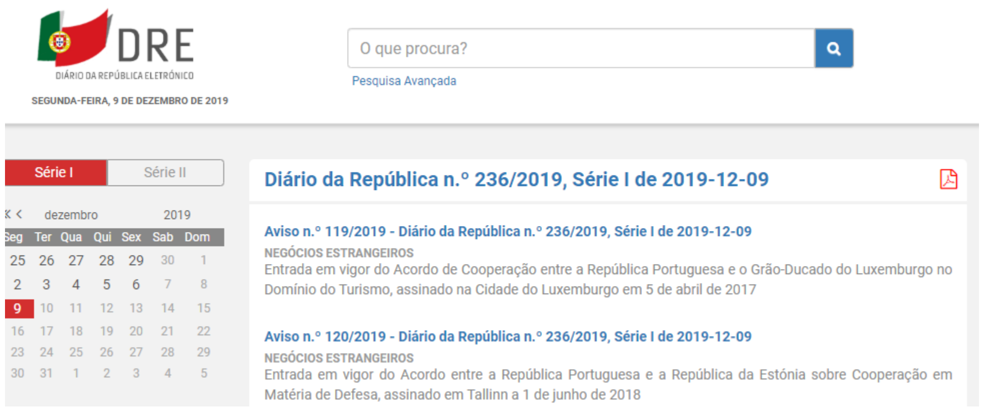
This chapter presented new unsupervised procedures for session and mission segmentation improving upon current state-of-the-art methods Gayo-Avello (2009); Hagen et al. (2011, 2013) through the usage of pre-trained word embeddings. First, I described the individual heuristics that were considered for the ablation tests to check the impact of each heuristic, in the overall methods that integrate them. Then, I describe the complete method for segmenting user sessions and an extension to addresses the problem of identifying search missions. In sum, the results confirmed the effectiveness of the proposed methods, which achieve a higher segmentation accuracy than competing unsupervised approaches, while not significantly expanding on the computational effort. In fact, the proposed unsupervised approaches can be used across search domains since not require training data or significant parameter tuning.

4 An Analysis on a Large Query Log from a Search Engine for Legislative Contents

This chapter presents a characterization study of a search engine query log from a Portuguese web search engine focused on legislative documents. One of the goals of this dissertation is to study the searching behavior of the users in search engines whose focus is on legislative contents, which can be crucial to improving the ranking functions. The chapter starts with a description of *DRE* search engine is presented in Section 4.1. Then, Section 4.2 describes a simple characterization statistics of the queries and a temporal, spatial and thematic distribution of user queries. Section 4.3 presents a standard characterization of users based on the sessions and a set of statistics to infer user satisfaction. Finally, Section 4.4 overviews the contents of this chapter.

4.1 DRE Search Engine

DRE stands for *Diário da República Electrónico* and is a digital library and search engine for Portuguese laws, regulations and legal acts. The service was initially provided with a fee until 2017, when it became free and accessible for everyone. More recently, *DRE* also operates through a mobile application. At the moment, it is possible to access the service from external sources (e.g., *Google*) to retrieve a particular document. Such cases will not be considered in this study since I will only be considering queries submitted in the search box and the clicks originated from these queries (i.e., intern search system) as presented in Fig 4.1.



Diário da República n.º 236/2019, Série I de 2019-12-09

Aviso n.º 119/2019 - Diário da República n.º 236/2019, Série I de 2019-12-09
NEGÓCIOS ESTRANGEIROS
Entrada em vigor do Acordo de Cooperação entre a República Portuguesa e o Grão-Ducado do Luxemburgo no Domínio do Turismo, assinado na Cidade do Luxemburgo em 5 de abril de 2017

Aviso n.º 120/2019 - Diário da República n.º 236/2019, Série I de 2019-12-09
NEGÓCIOS ESTRANGEIROS
Entrada em vigor do Acordo entre a República Portuguesa e a República da Estónia sobre Cooperação em Matéria de Defesa, assinado em Tallinn a 1 de junho de 2018

Figure 4.1: *DRE* interface for simple queries.

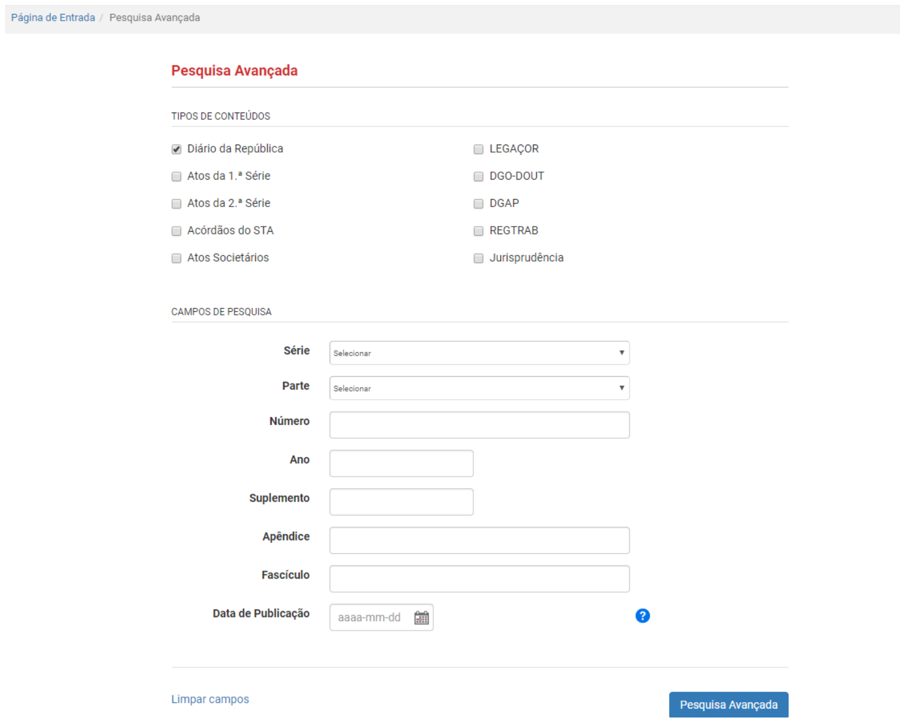
To cover the limitations mentioned in Section 1, I will analyze the query logs from *DRE*, covering two years, between 2017 and 2018, which correspond to the same timeline when *DRE* was available for all users. Retrieving logs from a large time range can bring several advantages into the analysis of the logs: (i) it is possible to see the evolution of users (ii) it allows to see if there is a particular seasonal search pattern in the logs (e.g., when the President of the Republic approves the state budget); and (iii) the analysis will be less affected by bias.

The queries logs retrieved from *DRE* have the Apache Common Log Format, in which each record in the log is an interaction between a user and the search engine through an HTTP request. Then, after a preprocessing step, each entry has the following attributes: (i) userID (i.e., a unique identifier that consists in the combination between IP address, cookie (if it exists) and hash-md5 of the user agent); (ii) timestamp (i.e., the instant when the user submitted the query); (iii) query text (i.e., the set of keywords submitted by the user); (iv) documentID (i.e., the ID of the document that the user clicked on after receiving the results for the query, or empty if no clicks were made); (v) click rank (i.e., the absolute position of the document clicked by the user amongst the list of documents retrieved in a given search query); (vi) title (i.e., the title of each document clicked); (vii) sessionID (i.e., a unique search session identifier); and (viii) mode (i.e., a Boolean indicator which indicates if the query has made on a browser or mobile).

One particular challenge relates to the fact that query logs often do not feature unique user identifiers, instead containing only cookie identifiers in a fraction of the records, plus information on source IP addresses and user-agents (i.e., identifiers for the type of Web browser in which the query was submitted). In these cases, the logs may feature queries from different users (and consequently also from different sessions) appearing interleaved in chronological order, all associated to the same IP address (e.g., from a common Internet proxy). For instance, the users that work in the same government institution will have the same IP address.

In the context of interactions with search engines, the notion of a session is critical to describe user tastes, habits, and their intentions when using these systems. I use the term session to define a sequence of activities executed by one individual to satisfy an information need. Due to the previous limitation and the fact that there is no mechanism to identify user sessions in *DRE*, I adapted/extended the unsupervised method for segmenting user sessions proposed in Section 3.2 by adding a new feature. First, I sort the dataset according to the combination of user cookie, IP address, and user-agent (instead of sorting according to userID only). Then, I considered a post-processing step to merge non-consecutive sessions (perhaps interleaved with interactions from different users sharing the same IP address and user agent) that are temporally and thematically coherent. In this new step, after processing each user (detecting user sessions) I will see whether two non-consecutive sessions correspond to the same session, based on the geometric method proposed by Gayo-Avello (2009). Sessions are the first step to understand how effective the search engine is in suggesting content pointers for user searches, allowing to evaluate the quality of the system.

Another peculiarity of *DRE* is the fact that users can do advanced searches as presented in Fig 4.2. In the advanced search, some parameters are selected to carry out a more refined search. This type of search targets scenarios in which: (i) the user does not have the necessary knowledge to make a simple query (i.e., does not really know what he is looking for and looks for a specific publication date); or (ii) the user knows exactly what he is looking for and selects a set of parameters in order to quickly retrieve a certain document, similar to the idea implemented in *Google* (e.g., the user can submit a query with a specific date range).


Página de Entrada / Pesquisa Avançada

Pesquisa Avançada

TIPOS DE CONTEÚDOS

✓ Diário da República	☐ LEGAÇOR
☐ Atos da 1.ª Série	☐ DGO-DOUT
☐ Atos da 2.ª Série	☐ DGAP
☐ Acórdãos do STA	☐ REGTRAB
☐ Atos Societários	☐ Jurisprudência

CAMPOS DE PESQUISA

Série	Selecionar
Parte	Selecionar
Número	
Ano	
Suplemento	
Apêndice	
Fascículo	
Data de Publicação	aaaa-mm-dd

Limpar campos

Pesquisa Avançada

Figure 4.2: *DRE* interface for advanced queries.

Finally, in all the statistics made in this study, language-specific lists of stop-words or stemming algorithms were not applied, and all queries were put in lowercase to better show how real users search in *DRE* system, which includes punctuation and misspelling. To minimize the possibility of bots in the logs, I discard (i) all interactions and accesses to documents coming from external accesses (e.g., *Google*), and (ii) all possible bots and/or crawlers identified trough the flowing implementation[1].

[1]`https://pypi.org/project/crawlerdetect/`

4.2 The Search Log Supporting the Characterization Study

In this section, I first describe generally the characteristics of the *DRE* search engine with a simple characterization statistics of the dataset, and a description of the dataset through a temporal, spatial and thematic distribution of user queries.

4.2.1 Simple Characterization Statistics

The dataset used in the experiments contains a total of 14,841,198 records (i.e., the number of queries and clicks), divided into 8,618,233 queries and 6,222,965 clicks. In more detail, I identify over 1,113,967 unique users with 3,185,973 sessions. In these two years, on average, each user submitted 13.54 records, 8.88 queries, 9.06 clicks and had 2.86 sessions. These suggest that the user needs to do at least one click to achieve a particular information need.

Table 4.1: General analysis of queries

A. Simple Queries

Description	Total	Browser	Mobile
Number of simple queries	5,469,271	4,919,663	549,608
Number of unique simple queries	1,708,425	1,575,525	184,698
Mean number of clicks originated by simple query	0.30	0.32	0.16
Median number of clicks originated by simple query	0	0	0
Number of unique terms in all simple queries	227,229	209,989	51,203
Mean number of terms per query in all simple queries	4.24	4.35	3.19
Median number of terms per query in all simple queries	3	4	3

B. Advanced Queries

Description	Total	Browser	Mobile
Number of advanced queries	3,148,962	3,080,075	68,887
Number of unique advanced queries	800,195	790,801	18,111
Mean number of clicks originated by advanced query	0.31	0.32	0.17
Median number of clicks originated by advanced query	0	0	0
Number of unique terms in all advanced queries	65,647	64,910	4,336
Mean number of parameters per query in all advanced queries	5.36	5.36	5.26
Median number of parameters per query in all advanced queries	5	5	5

Table 4.1 focuses on simple and advanced queries, and I divided the analysis in (i) Total (i.e., queries submitted in the browser and the mobile), (ii) Browser (i.e., queries only submitted in the browser) and (iii) Mobile (i.e., queries submitted only in the mobile). The results demonstrate

that there is no discrimination between simple and advanced queries since 63% of all queries are simple queries, and the number of clicks originated through simple queries is 60%.

On the other hand, about 90% of the simple queries were submitted in the browser while about 98% of the advanced queries were submitted in the browser, which shows the disproportion between both platforms. Considering all queries, there are a few unique terms, which suggests that some of the users probably look for the same type of information (e.g., looking for contests). However, the mean number of terms per simple query is much higher than in previous studies (i.e., 4.26 versus 3.380 reported by Chau et al. (2007)). The reason is that there are users that submit URLs or large queries originated from other sources (e.g., copy-paste terms from *Google*).

Interestingly, about 53% of the simple queries have numeric characters, which indicate two types of search, (i) queries more targeted to specific terms (e.g., year and law number), and (ii) more general queries (e.g., health). These cases I will analyze in more detail later in the article. However, in the mobile, the percentage of the number of queries with numeric characters decreased to 24%, which suggests that the users use the app for more generic information needs (e.g., contests).

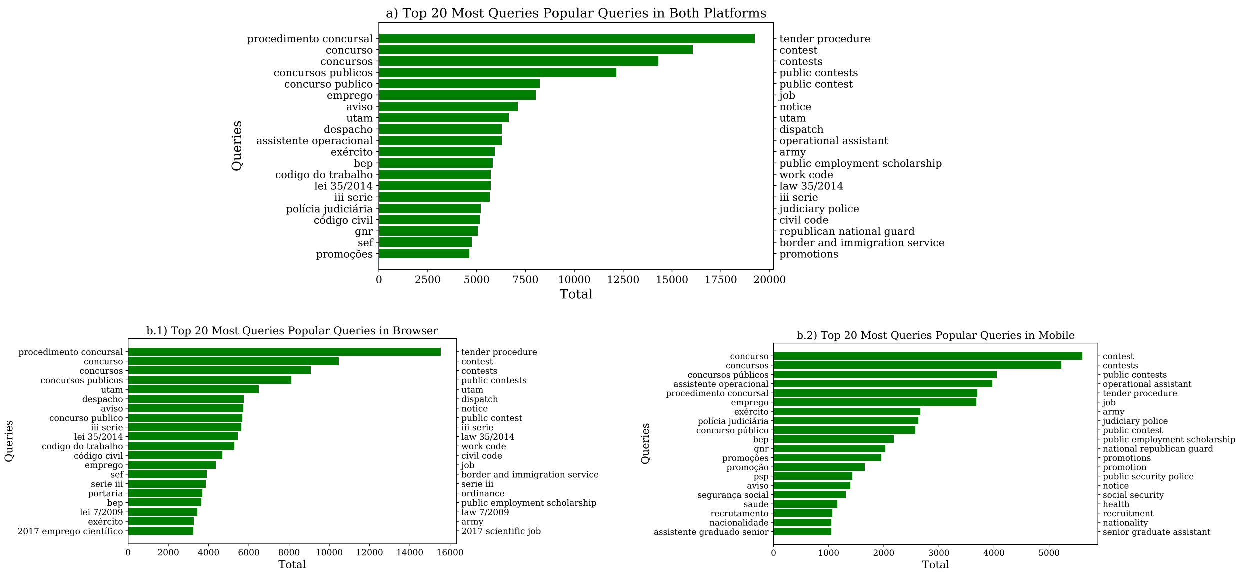


Figure 4.3: Most popular queries in *DRE*.

Figure 4.3 lists the 20 most popular queries searched: (a) in both platforms (i.e., browser and mobile), (b.1) in the browser and (b.2) in the mobile. *Procedimento concursal* (i.e., tender procedure) is the most searched query, while *concurso* (i.e., contest) is the second. Then, I extract the top 50 most popular queries to do a more detailed analysis, and I verify that 30% of all simple queries are related with contests, which reflects one of the main purposes of this search engine, 15% are related with a specific type of documents in the *DRE* (e.g., decree-law), 13% are related with service and public institutions, 11% are related with army and police, and 8% are related with jobs. Although the main focus continues to be related to contests, in the browser there are more top queries related to user job (i.e., the user searches for a specific type of documents or specific law). On the other hand, in the mobile, top queries are more related to general information (e.g., police or recruitment).

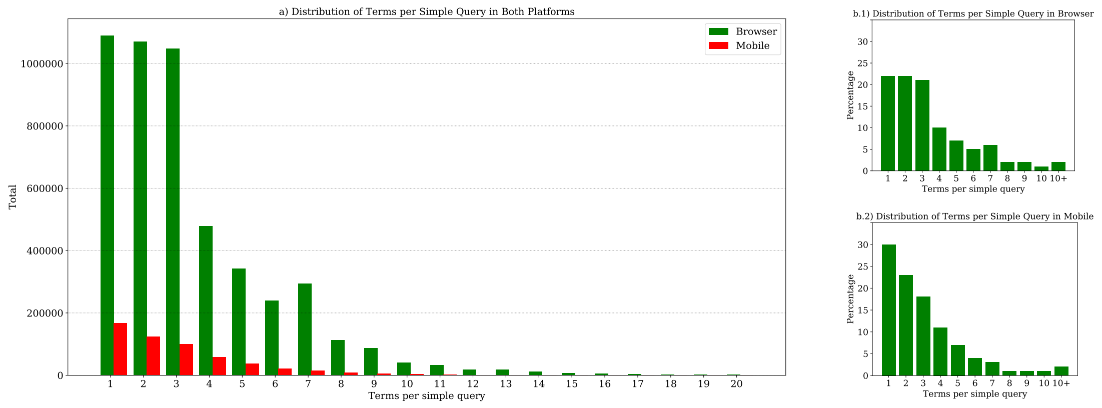


Figure 4.4: Distribution of terms per simple query.

Figure 4.4 illustrates the distribution of the number of terms per simple query, where (a) is the number of terms per query in both platforms, (b.1) is the percentage of the number of terms in the browser, (b.2) is the percentage of the number of terms in the mobile. In the chart (a) it is possible to observe the difference between the number of queries in the browser and the mobile. As reported by several studies, most of the queries have between one and three terms. However, considering queries with six and seven terms it is possible to see a not normal pattern since it would be logical that the number of queries with seven terms decreases. However, such does not happen since some user submits long queries through copy-pasting from external sources. In charts (b.1), 65% of the queries have fewer than or equal to 3 terms, while in charts (b.2) it is about 70%, which is an expected result that can be caused by different reasons, showing no considerable difference between platforms.

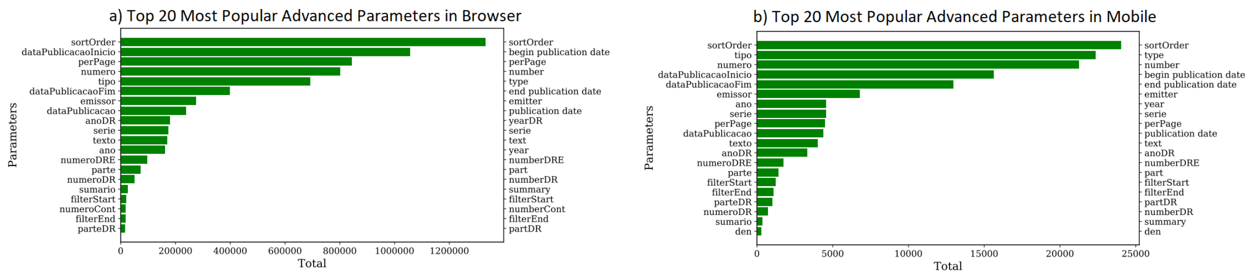


Figure 4.5: Most popular advanced parameters in *DRE*.

In addition, it is possible to do advanced queries with or without any query terms, and each user can use different parameters to target the search (e.g., year, publication date, and document identifier). Figure 4.5 lists the 20 most used parameters in advanced queries, which are not automatically introduced by the system. Since 98% of the advanced queries are submitted through the browser, in this case, it is not interesting to show the top 20 parameters in both platforms. The plot (a) lists the top most used parameters in the browser and (b) on the mobile. *SortOrder* is the most used parameter in both whereas *dataPublicacaoInicio* (i.e., first publication date) and *tipo* (i.e., type) are the second, respectively.

The main parameters in the top 20 are related with restricting the timeline (i.e., restricting the year or concrete publication date of a specific document), for instance, *first publication date*, *last publication date*, *publication date*, *yearDR*, and *year*, reflecting one of the main reasons why users use advanced queries. However, there are also other parameters with a lot of influence, which restrict the order (e.g., *sortOrder*), the document (e.g., *type*), issuing entity (e.g., *emitter*) and the number of results per page (e.g., *perPage*).

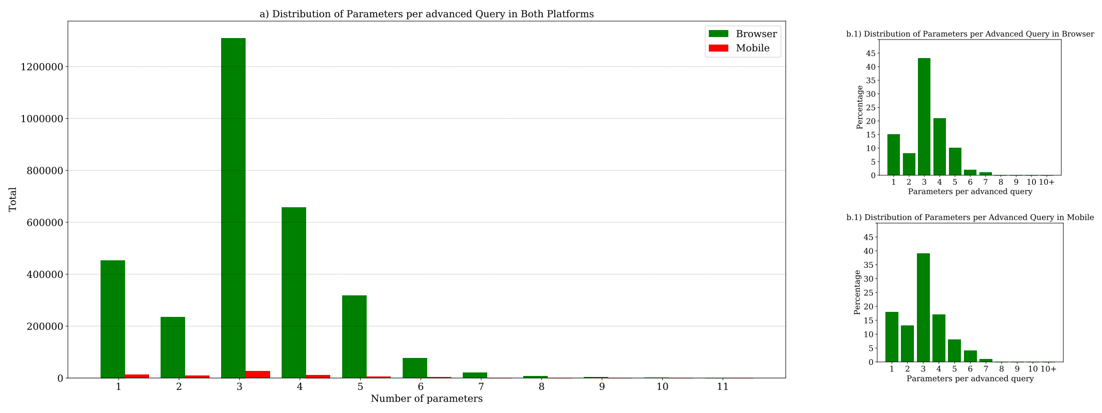


Figure 4.6: Number of Advanced Search Parameters.

Figure 4.6 shows the number of parameters used in advanced queries and the respective distribution. As expected, the number of parameters used in advanced queries is small, as most queries have three parameters either in the browser and on the mobile. In more detail, considering only queries with one parameter I conclude that in 90% of the cases the queries are related with the parameter *type* (e.g., *type=SERIE I*, *type=SERIE II* and *type=SERIE I, SERIE II*), although another parameter used is *emitter* but with less impact.

4.2.2 Temporal, Spatial and Thematic Distribution of User Queries

In terms of the temporal, spatial and thematic distribution of user queries, Figure 4.7 illustrates the frequency of queries for each day in the two years studied. First, the number of queries increases between 2017 and 2018, except at the beginning of 2017, since was when

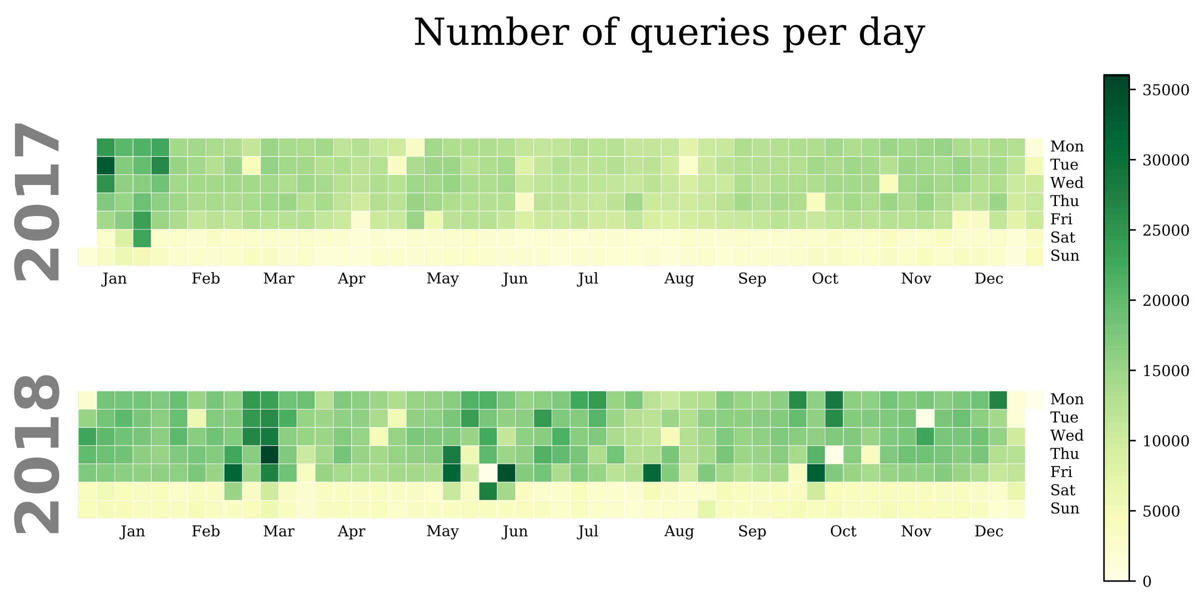


Figure 4.7: Number of queries per day.

the system became a public service. Second, the number of queries is lower on weekends and holidays (e.g., January, 1st) compared with the number of queries on working days, reinforcing the idea that this service is more targeted and used at work. Third, most of the peaks in the figure are directly related to a large number of advanced queries. On the other hand, the number of simple queries is uniform over time, with a small increase in the average number of queries per day between 2017 and 2018.

In more detail, March 8th, 2018, June 8th, 2018, and November 20th, 2018 are the days with more queries per day with a major difference compared with other days. To better understand this event I did a more thorough analysis and noticed that there is a particular IP address with most queries on that day. This IP address belongs to the *Priberam.pt* domain, which is an institution whose main focus is to collect documents from the *DRE*. Furthermore, all the queries submitted on that day were advanced queries, probably due to a crawler looking for a particular set of documents. To better show search patterns I removed theses days from the analysis. 9 June 2018 is also irregular since the number of queries on a Saturday is similar to the other days of the week. This is due to the fact that the crawler continues to look for a particular set of documents given that 86% of the queries submitted had the IP address belonging to *Priberam.pt* domain. As expected, the days with fewer queries are holidays (e.g., December 24th of 2017).

In terms of the difference between the distribution of the number of queries per day in the browser and the mobile, about 90% of the queries were submitted in the browser, so the distribution will be equal as shown in Figure 4.7. However, the distribution of the number of queries in the mobile follows the same distribution described in the browser on weekends and holidays. There are also a few days out of the ordinary which related to the fact that most people on these days are on vacations, so it is probably easier to access *DRE* through the app (e.g., January 3rd, 2018). Interestingly, starting in August, the number of queries per user in the app increases and stabilizes, verifying that more and more users are using mobile devices.

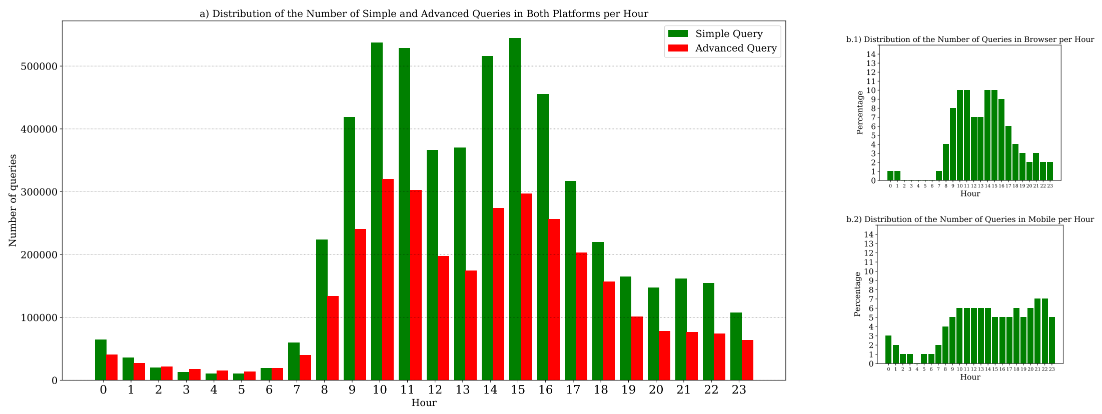


Figure 4.8: Distribution of the number queries per hour.

Figure 4.8 presents the distribution of the number of simple and advanced queries per hour of the day, where (a) considers both types of searches (i.e., advanced and simple queries), while (b.1) and (b.2) illustrates the distribution of the number of queries in the browser and the mobile, respectively. The plots confirm the previous conclusions that most of the queries submitted on the browser were done during working hours, while in the app the distribution of queries per hour is more consistent. As shown, there is no difference between the number of simple and advanced queries since both types of searches are reasonable ways to get the user's information need. However, during the night period, there is a non normal pattern on mobile which confirms the presence of crawlers, although this case needs to be analysed in more detail.

In terms of spatial statics, I address this issue focusing solely on mainland Portugal (i.e., without considering *Madeira* and *Açores*). Figure 4.9 shows four different views of the data related to the frequency of queries by the district of origin, according to the IP address. This was carried out based on the number of queries per person and district, taking into consideration the *PORDATA*[2], which it is a certified statistics base for Portugal divided in several themes like population, education, health, and others. Chart a) corresponds to a general view of the number of all queries per person. *Lisboa* and *Leiria* have 2 queries per resident while other districts have only one query per person. Chart b) and c) show the number of simple and advanced queries per person, respectively. Only *Lisboa* has 2 simple queries per person, whereas all other districts and also all districts from advanced queries map only have one query per person. Finally, in

[2]https://www.pordata.pt/

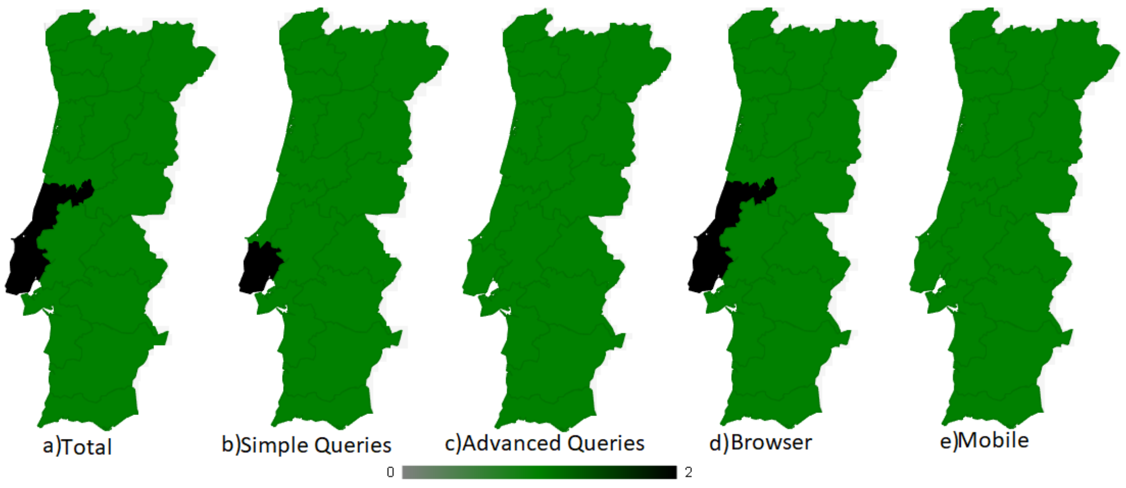


Figure 4.9: Number of queries per person per district.

chart d) and e) the difference between the number of records per person in the browser and the mobile can be observed, showing the same distribution, and it is on the coast where exists more people access the *DRE*.

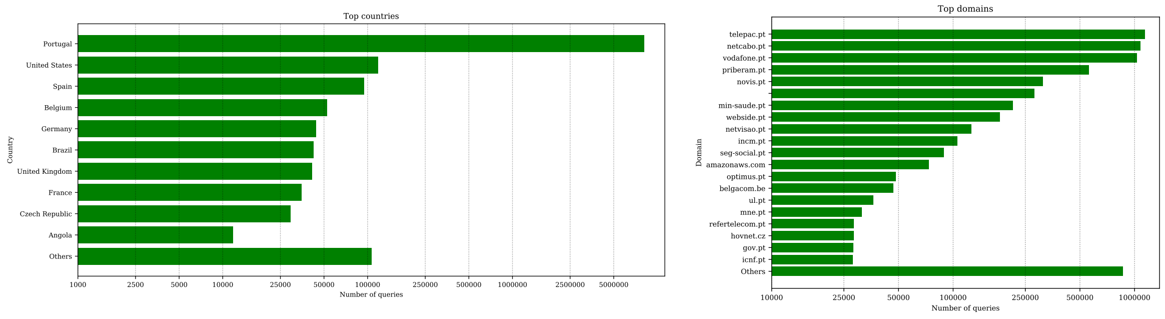


Figure 4.10: Top countries with more queries and number of queries in each domain from all countries.

Finally, with Figure 4.10 it is possible to see the top 10 countries with more accesses in *DRE* on the top row, and the top 20 Web domains with more queries on the bottom row. The top 3 countries with more queries are *Portugal*, *United States* and *Spain*. Most of the countries are former Portuguese colonies (e.g., *Brazil* and *Angola*) or countries with a considerable number of Portuguese immigrants (e.g., *Belgium*, *Germany*, *United Kingdom*, and *France*). However, there are unexpected countries in the top 10, for instance, *Czech Republic*, which may suggest possible bots or crawlers. If I take into consideration the data from 2018, *Czech Republic* is in the top 4 countries with more records. Interestingly, there is still a large percentage of other countries, around 180 different countries, showing that *DRE* is a multi-platform and can be useful in different countries across the world.

On the other hand, the bottom row shows that most of the domains are from *Portugal* (i.e., ending in *.pt*), but an interesting observation is the fact that there are several entries with an *unknown* domain. Analyzing only the Portuguese domains, as expected, the domains with more queries are from private companies which provide Internet to consumers (e.g., the company *Altice* has the domain *telepac.pt*). However, in *Portugal*, each ministry has it is own domain, and 7 out of the top 20 domains are from government institutions. In more detail, the government domains in the top 20 are from *Lisbon* and are related with health (i.e., *min-saude.pt*), *INCM* which is the organization responsible for *DRE* (i.e., *incm.pt*), social security (i.e., *seg-social.pt*), university (i.e., *ul.pt*), ministry of foreign affairs (i.e., *mne.pt*), finance (i.e., *gov.pt*), and institute for conservation of nature and forestry (i.e., *incf.pt*).

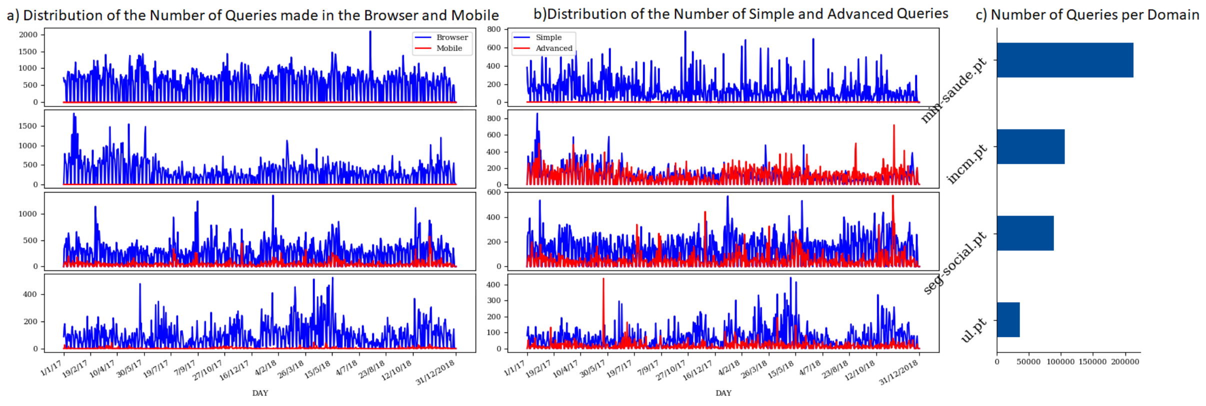


Figure 4.11: Number of simple and advanced queries and the number of queries in the browser and mobile for the 4 domains with most queries in the log from government institutions.

Finally, in Figure 4.11 I present a characterization of the users from the top 4 government domains (i.e., *mini-saude.pt*, *incm.pt*, *seg-social.pt* and *ul.pt*) respectively, with more query logs, combining multiple statistics. In plot (a) it is possible to observe the distribution of the number of queries made in the browser and in the mobile on each day, chart (b) shows the distribution of the number of simple and advanced queries, and in chart (c) the number of queries per domain in all dataset.

In charts (a) and (b), it is possible to conclude two different situations. First, chart (a) shows a major difference between the number of interactions in the browser and the mobile. The reason is that, normally, when the user is in the workplace it is preferable to use the browser since it is easier to do complex tasks. However, it would be interesting to see if the user continues to look for the same information (i.e., the same search mission) after leaving the workplace. For instance, the user continues to search for the same information need but with a different IP address (i.e., a different domain from a private company), or through the app. Second, chart (b) reinforces the idea that for some domains, when the user needs to search for a specific topic there is no discrimination between types of search. However, unexpected, *min-saude.pt* do not have any advanced queries or mobile interactions. Also, the distribution on each day follows the same format as before, concerning the difference between the record number at the weekend and on days of the week.

However, the main focus of this figure is to show search patterns (e.g., contests) in each government institution. Regarding *min-saude.pt* domain, it is possible to see the same distribution over all days, but on 26 July 2018, the number of queries increased. To better understand this phenomenon, I retrieve the top 10 most popular queries on 26 July 2018 for *min-saude.pt* domain and I verified that this increase is due to a dispatch being published on that day identifying the most deprived geographical areas, as defined by the health establishment and medical specialty to incentives for the mobility of medical workers. Second, *incm.pt* domain is the institution responsible for *DRE* and it is possible to see more queries at the beginning of 2017. One of the reasons for this increase is the fact that *DRE* became a public service on 1 January 2017. So, probably there was a need to do extra testing and changes. Third, *seg-social.pt* domain is a service-related with social security, and most of the days the queries are related to new legislation, as well as searching for final deliberations regarding the employment status of each employee. In terms *ul.pt* domain, it is possible to see that there are fewer interactions during university holidays months.

4.3 Patterns in User Sessions

This subsection will be split between a standard characterization of users based on the sessions identified by an improved version of the method proposed in Section 3.2 and a characterization of user satisfaction based on a set of unsupervised heuristics.

4.3.1 Simple Session Characterization Statistics

I was able to retrieve more than 3 million sessions, with an average of 2.86 sessions per user, 2.95 queries per session, and 2.64 clicks per session with an average duration of 1,879 seconds.

Figure 4.12 present the number of sessions per day, in which the number of sessions follows the same distribution concerning the difference between the number of queries in 2017 and 2018 reported in Figure 4.7, and the difference between the number queries on weekends/holidays and working days. However, there are more days out of the ordinary. First, without considering the previously reported outliers (i.e., March 8th, 2018, June 8th of 2018, and November 20th 2018),

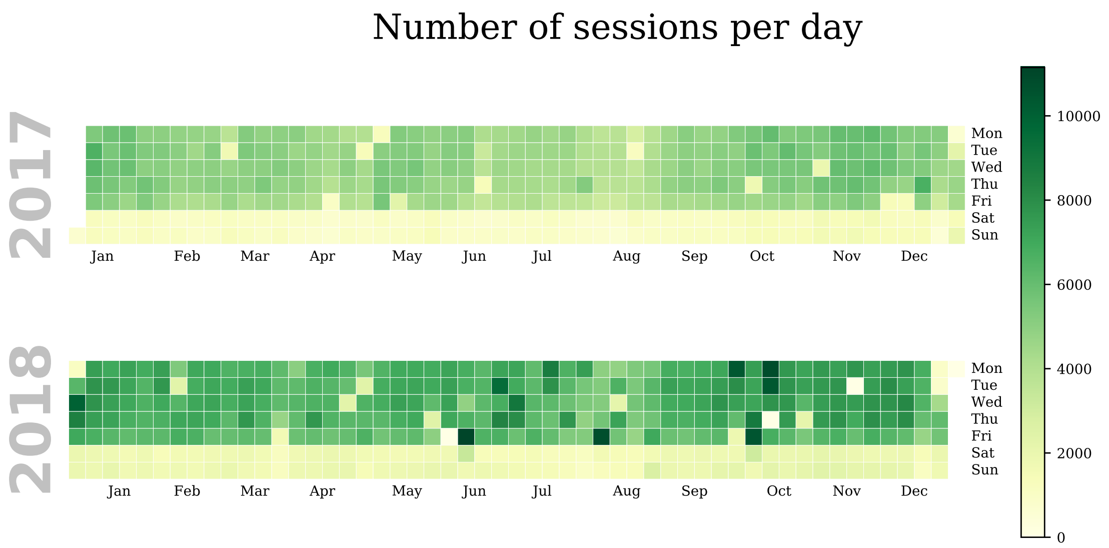


Figure 4.12: Number of sessions per day.

it is possible to see that the number of sessions on the following day (i.e., June 9th of 2018) did not follow the same distribution seen before (i.e., the correlation between the number of queries and the number of sessions per day). As explained previously, this is due to the fact that most of the queries submitted on June 9th of 2018 are from a bot (i.e., from *Priberam.pt* domain). Contrary, in Figure 4.7, March 15th, 2018 is the second day with more queries. However, Figure 4.12 does not show the same correlation between the number of queries and the number of sessions, implying that on this day the number of queries per session is much higher compared to other days.

Figure 4.13 produce an overview of the distribution of the number of queries and clicks per session. First, most of the sessions have one query, regardless the type of search, which can mean two different situations, (i) most of the interactions are from a sporadic users, or (ii) most of the users only need to do one query to satisfy their information need, which will be directly related to user satisfaction.

In terms of the number of clicks per session, on average there is between one and two in each type of search. Nevertheless, for simple queries, around 45% of the sessions have only one click, whereas, for advanced queries, only 34% of the sessions have only one click, suggesting that the user is more likely to do an extra query when using advanced queries in order to satisfy his information need than when using a simple query. The reasons for this event could be the fact that (i) most users that use advanced queries are expert users, so it is more likely that when they search for a specific topic they need to understand a set of documents, or (ii) there are issues in search engine rankings. Finally, based on these distributions, I can conclude that 93%

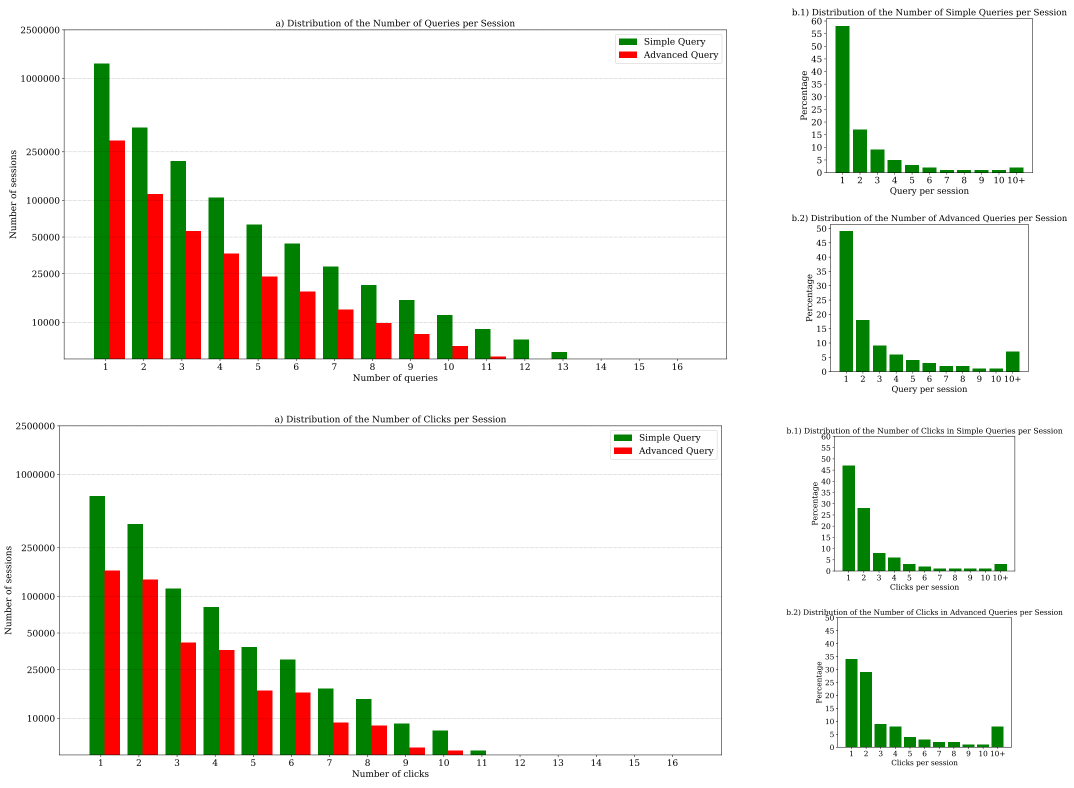


Figure 4.13: Number of queries and clicks per session

of the users have at most 3 sessions, 67% of the sessions have at most 3 queries, and 73% of the sessions have at most 3 clicks, which will be directly related to search engine quality and user satisfaction.

4.3.2 User Satisfaction

Another point that will be addressed is inferring user satisfaction. Su (2003) referred that user satisfaction is directly related to the user's judgment of the overall success of a search engine in answering a specific information need. Hence, the quality of a search engine is determined by the user satisfaction with the searching methodology, quality of information, and the ranking functions of each search engine (Liaw and Huang, 2003).

The goal of this section is to assess whether each user was satisfied in order to estimate an overview of retrieval performance. User satisfaction can be described at a query-level or a task-level. Whereas satisfaction at a query-level can be described as the satisfaction of the user after only one interaction with the system, task-level satisfaction corresponds to the satisfaction of the user concerning a certain need for information (i.e., search sessions). Nowadays, users increasingly feel the need to perform complex tasks, which can mean that they have hierarchical goals within the same task or a set of related tasks, each of which is recursively complex. The approaches that will be addressed focus on inferring query-level satisfaction through log analysis (i.e., offline).

There are several ways to assess user satisfaction in query log analysis. In this context this assessment can be performed through a set of unsupervised heuristics: (i) dwell time, which is the time gap between submitting the query and interact with the system (e.g., clicking on a result or rephrase the query) is a strong signal for satisfaction since it is directly correlated with result-level satisfaction or document relevance (Su et al., 2018; Jiang et al., 2015b; Kim et al., 2014b). For instance, Morita and Shinoda (1994) discovered the correlation between user satisfaction and the reading time since users tend to spend more time on interesting documents; (ii) the time needed to achieve a particular information need (i.e., the duration of each session). For instance, Ageev et al. (2011) described that successful sessions have an average duration of 182 seconds, while unsuccessful sessions have an average duration of 276 seconds; (iii) the percentage of the number of reformulated queries in a session; (iv) the number of interactions between the user and the system; (v) the distribution of the position of the document clicked has also been widely used as implicit feedback to measure user satisfaction for Web search engines. However, previous studies proved that click-through can be noisy since the user may click on some result due to the attractiveness and/or position of the search result rather than its relevance (Joachims et al., 2005). Nonetheless, these methods can be very powerful as a first analysis.

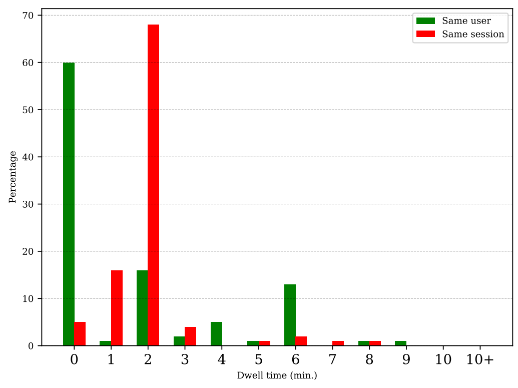


Figure 4.14: Distribution of the time gap between consecutive queries in user and sessions.

Figure 4.14 shows the distribution of the dwell time for interaction of the same user and session. As expected, most interactions of the same user have the dwell time less than one minute since the time between a query and the first click is always short as reported in other studies. Interestingly, there is a difference between the dwell time for the same user and the same session. Most of the interactions, in the same session, have a dwell time of two minutes (i.e., considering only the dwell time between click and other interaction), showing that most of the users take some time to analyze each document (i.e., the correlation between dwell time and reading time) which can be a positive signal for satisfaction (i.e., normally small dwell time indicate that the user realized that the clicked document not proper for his information need).

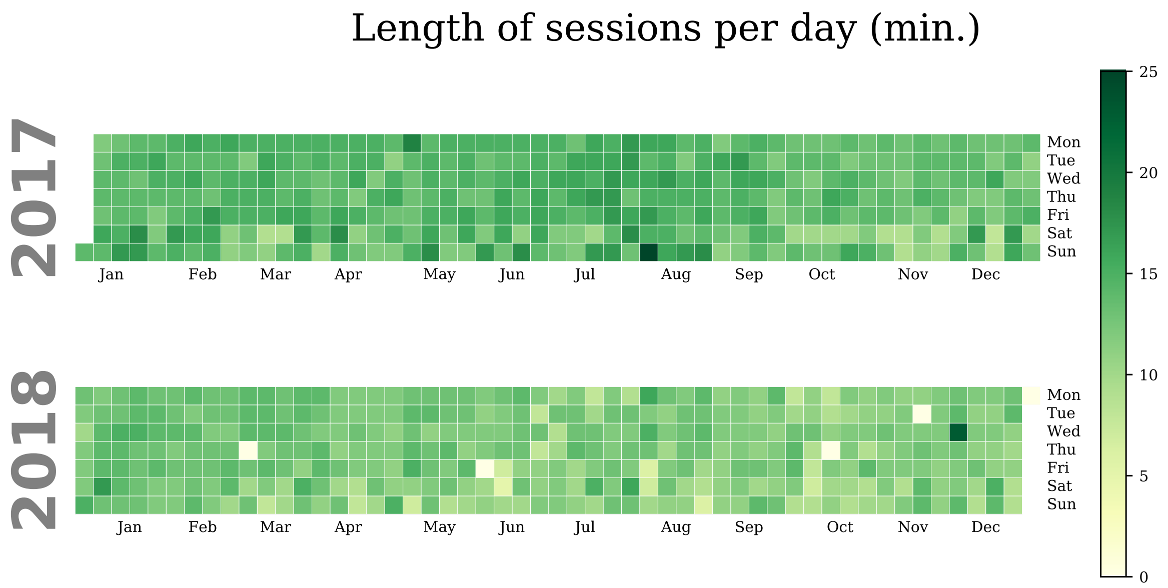


Figure 4.15: Distribution of the length of sessions per day in minutes.

As explained, the number of sessions per day increases between 2017 and 2018 and one of the reasons is the fact that the number of queries and the number of new users also increase every day. Nevertheless, another possibility can be due to residual changes to the system. Thus, in Figure 4.15 it is possible to observe the distribution of the length of sessions per day in minutes over the two years.

As expected, most of the user are experts since, on average, the length of the sessions are between 10 and 20 minutes. However, this analysis was very important since I was able to identify bots, in which for a specif group of IP address there were sessions with more than 130 minutes, and all related with a similar advanced queries looking for a group of documents (i.e., crawler to extract a set of documents). In more detail, surprisingly, on Christmas holidays, the average sessions length of December 24th of 2018 and December 25th of 2018 was 150 and 175 minutes, respectively, and about 90% of all queries were advanced queries. Thus, similar advanced queries will be joined, which will increase the length of the sessions on these days. Finally, without considering these outliers (i.e., removing the bots as explained in the begin of this chapter), the average length of sessions on December 6th of 2018 it is interestingly high. The reason is that the top queries in that day are related with *promotions* and *contest*, in which despite the number of queries is not high, elapsed time it is bigger. Interestingly, the length of a session decreases from an average of 14 minutes with a median of 14 minutes in 2017 to an average of 11.47 minutes with a median of 11 minutes in 2018, showing that the system is improving and focusing more and more on user needs, which also indicate satisfaction.

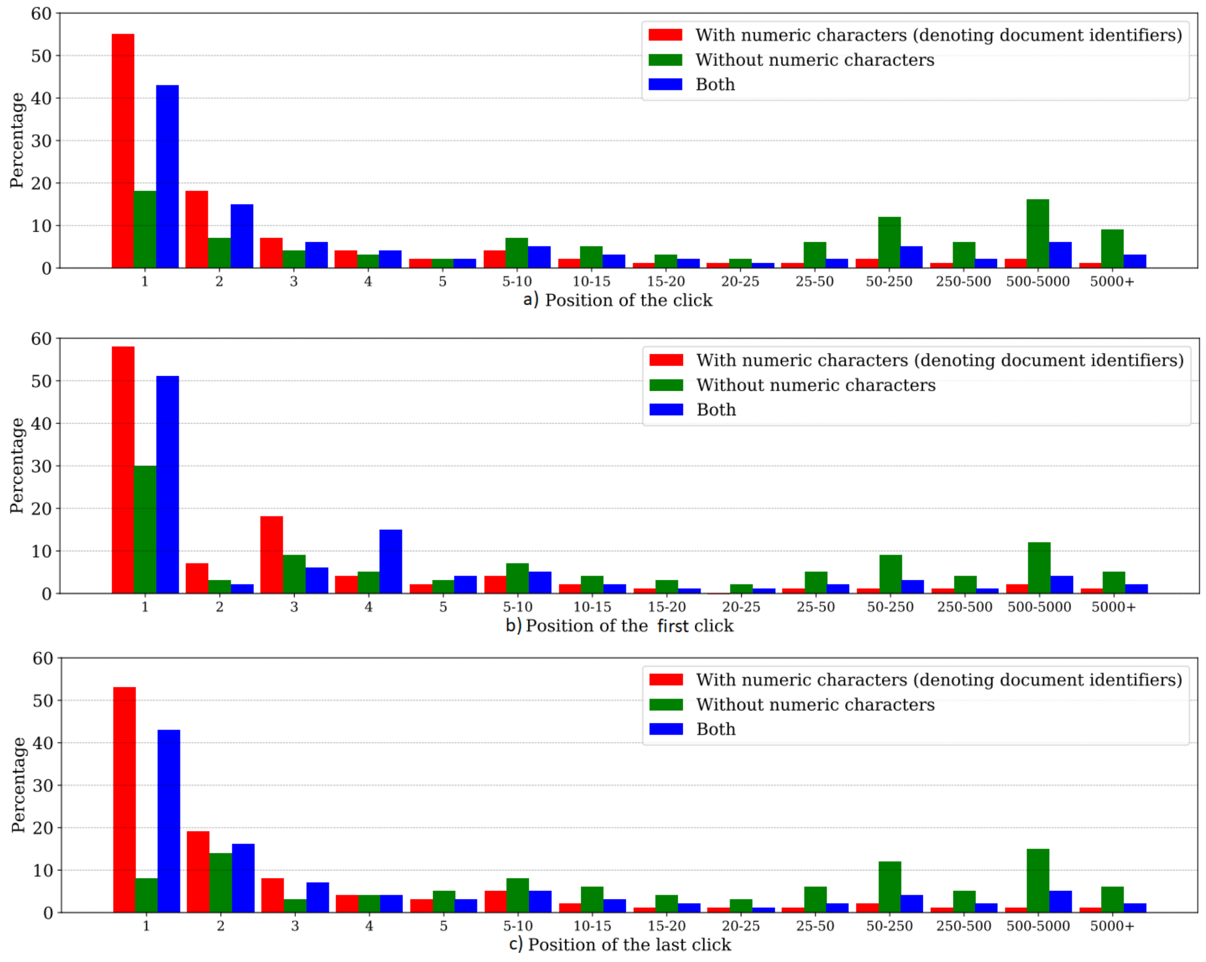


Figure 4.16: Distribution of the position of the click

In Figure 4.16 the distribution of the absolute position of each click originated from a simple query is represented, considering (i) queries with numeric terms (i.e., documents identifier), (ii) queries without numeric terms, and (iii) queries with both characteristics. Chart (a) shows the distribution of the position of the document clicked by the user for each simple query, while charts (b) and (c) show the distribution of the position of the first and last click by the user for a given simple query, respectively.

On the one hand, in queries with numeric terms, the users normally click in the first positions, since queries with numeric terms are more targeted to specific documents (e.g., *decree law 15/2018* or *work law 2018*)). Furthermore, 7% of the clicks are after position 25 (i.e., before the first page considering 25 results per page). The reason is due to the fact that, for some cases, *DRE* system uses a specific balancing which places specific documents first regardless of the query submitted by the user.

There are two different examples of this phenomenon. First, if the user submits the query *notice nr 1606/2018* the system only finds one result and that document is from a completely different topic from the user intentions. The main reason is the sub-string *nr* since if the user submits the query *notice n° 1606/2018*, 62 results are found. However, the ranking system does not place the type of document referred to on the query in the first position (i.e., the type *notice*) only appears in the sixth position whereas the type of documents like *decree law* or *law* are given more priority by the ranking function. The reason is the fact that in the *DRE* there is documents more relevant than others. Thus, despite the query submitted, for some cases the ranking function gives different boosts to a specific documents (e.g., normally *decree law* appears first than *notice*). Finally, if the user refines the query by making it more specific with a date (e.g., *notice n° 1606/2018 from 5/2/2018* or *notice n° 1606/2018 from 2018-02-05*) the ranking system changes the order of the results, but the type of document *notice* continues in the sixth position.

The second case is when the user searches for *decree law n° 66 from 1992* and then the user clicks in the first position, but to satisfy his information needs he will need to look for different documents in the same search. For instance, after analyzing the first position the user feels the need to click on the tenth position which is the type of document *Ordinance n.° 212/92*. In sum, these cases also justify the percentage verified in charts (b) and (c) related to first and last clicks for each query.

On the other hand, if considering only queries without numeric terms it is possible to observe a different behavior since most of the users only click after the 25 position and to explain this phenomenon I analyzed this case more thoroughly. First, most of the queries are related to general information (e.g., the top 4 are *tender procedure*, *diario da republica*, *securities* (i.e., actions), and *procedure notice*). Second, I analyzed the top 10 domains with more queries where the top 4 are *unknown*, *telepac.pt*, *netcabo.pt*, and *vodafone.pt*, which the last three domains correspond to a private Internet providers. Furthermore, if I consider only the queries from the *unknown* domain the top 3 queries are *energ** (i.e., related with energy), *cresap* (i.e., Recruitment and Selection Committee for Public Administration), and *securities*. However, I manually identified that most queries are from the *ceger.pt* domain (i.e., Government Computer Network Management Center), *sgmai.pt* domain (i.e., General Secretariat Ministry of Internal Affairs), and *meo.pt* (i.e., private Internet provider). Third, the queries where the clicked document is on the farthest position are (i) *Diario da Republica* in the position 4,553,452 with the General Secretariat Ministry of Internal Affairs domain, (ii) *iii* in the page 3,396,771 with *vodafone.pt* domain, (iii) *serie iii* with different interactions around the page 3,264,213 from different domains (*cmah.pt*, *telepac.pt*, *webside.pt*, *vodafone.pt*, and *telepac.pt*).

4.4 Overview

This chapter presented the first characterization study of a search engine query log from a Portuguese web search engine focused on legislative documents. The chapter provides a set of statistics to build a full characterization of user behaviors, in terms of queries, terms, and sessions, and the differences between browsers and mobile interactions. In sum, I concluded that it is possible to distinguish two types of users: (i) users that submit queries with numeric terms (denoting document identifiers) normally click in the first positions, due to the fact that queries with numeric terms are more targeted to specific documents, and (ii) users that submit queries without numeric terms normally click after the first page since most of the queries are related with general information. This analysis will help to improve the ranking system to better suggest content pointers for user searches. For instance, rethink the boosts given to a specific documents in the search engine rankings.

5 Conclusions and Future Work

Segmenting user sessions and missions in search engine query logs are important tasks in the context of several applications. Both these tasks are nonetheless quite challenging, for instance involving the detection of query reformulations that employ a variety of factors: correcting misspellings (e.g., *gogle* versus *google*), using co-referential expressions (e.g., *CEO Facebook* versus *Mark Zuckerberg*), acronyms (e.g., *John Fitzgerald Kennedy* versus *JFK*), generalizations and specializations (e.g., *Jaws* versus *Jaws the movie*), etc.

This dissertation presented new unsupervised procedures for session and mission segmentation, improving upon current state-of-the-art methods (Gayo-Avello, 2009; Hagen et al., 2011, 2013) through the usage of pre-trained word embeddings. The experiments confirmed the effectiveness of the proposed methods, which achieve a higher segmentation accuracy than competing unsupervised approaches, while not significantly expanding on the computational effort. The performance of the proposed algorithms, together with the fact that they do not require training data or significant parameter tuning, makes them ideal for processing very large query logs from real-world search systems, independently of the domain. Additionally, I extend the notion of session to better handle activity breaks by adding a post-processing step that groups consecutive and non-consecutive sessions that are temporally and thematically coherent. Building on the session segmentation method, I made a characterization study to analyze a query log of a national search engine for legislative contents, in the context of an ongoing technology transfer project. Despite the wide range of studies reported, this task is particularly challenging, since it involves analyzing a specific search engine in a different context, inferring user satisfaction based on general unsupervised procedures. The experiments confirmed the following conclusions:

- As expected, there are very few interactions on weekends and holidays, showing that *DRE* search engine is more targeted to users whose work depends on it (e.g., users that need to look for laws and legislation's), particularly given that more than 70% of the queries were made during work time.
- On average, a user types four terms per query, which is much higher than the previous studies (e.g., the number of terms in queries reported in *Excite* is 2.16). However, most

recent studies show an increase in the number of terms per queries (e.g., the number of terms reported in Chinese queries is 3.380). This phenomenon happens due to two reasons: (i) some users submit URLs or large queries originated from other sources (e.g., copy-paste terms from *Google* results), and (ii) users normally made queries more targeted to a specific document, using document identifiers (e.g., *notice n° 1606/2018 from 2018-02-05*)).

- The results also demonstrated that it is most likely that the users feel the need to do more than one click per query. The main reason is because of not all information that the user needs is in one specific document on which a user clicked. Thus, although a specific document clicked can be relevant the user needs to click on another document to completely satisfy the information need since legislative documents are inter-related typically. In particular, for some types documents there is multiple reference to previous/related legislations that the current document updates. *DRE* is not a general web search engine, but a search engine for a specific usage context. However, another reason is the fact that the search engine also can have issues in the ranking function since most of the clicks resulting for queries without numeric terms (i.e., generic topics) are after the first page, while users that submit queries with numeric terms (denoting document identifiers) normally click in the first positions.

The results obtained when conducting the experiments presented in this dissertation revealed to be worthy contributions to perceive the information needs of each user in a legislative context and assess how the users are satisfied, to enhance the quality of search engine rankings, and to better direct content to certain users. Nevertheless, many questions can be made when comparing *DRE* search engine with other commercial search engines (e.g., *Google*). Based on the article published by Rand Fishkin[1] it is not possible to do a full comparison between these two search engines since *Google* has 40 to 60 billion searches a month which is a completely different reality compared to digital library *DRE* which has around 400,000 queries per month. Second, in *Google*, a session lasts on average less than 1 minute, which is an indicator of the effectiveness in suggesting content pointers for user searches, whereas in *DRE* it lasts about 30 minutes on average, which is due to the fact that the number of clicks per session is about 2.64 on average and the time gap between queries and click or between clicks and clicks is also high. Finally, in *Google*, the percentage of distinct *Google* searches resulting in one or more clicks is about 66% while in *DRE* it is about 60%. Some information needs can be answered by the SERP without occurring any click since It depends on the interface of each search engine. For

[1]https://moz.com/blog/state-of-searcher-behavior-revealed

instance, most search engines have snippets where the query terms appears in the document or if a user search for *wheater in lisbon* in the google there is no need to do any click.

5.1 Overview on the Contributions

The solutions presented in this dissertation revealed to be valuable contributions. The most important contributions of my M.Sc. thesis are as follows:

- **A new unsupervised method for segmenting user sessions**: This dissertation describes an effective and efficient unsupervised method, leveraging word embeddings and a cascade of temporal and lexical heuristics (Gomes et al., 2019), which can be used across search domains. The proposed method was evaluated in two different subsets from the well-known AOL query dataset (Pass et al., 2006), which has frequently supported studies on user session identification. Experiments confirmed that the proposed method achieves a higher segmentation accuracy than competing unsupervised approaches, while not significantly expanding the computational effort.

- **A new unsupervised method for segmenting search missions**: I describe an adaptation/extension of the proposed method for segmenting user sessions, taking into consideration multitasking behavior and hierarchical goals for consecutive and non-consecutive user sessions (i.e., search missions). To identify user search missions, I considered the method for segmenting user sessions as the first step. Then, assuming that the queries are already grouped into user sessions, I leverage a separate and similar cascade approach to merge user sessions into search missions.

- **A analysis of a large query log from a search engine for legislative content**: A preliminary analysis on a large query log to better understand user behavior and improve the search and navigation in *Diário da República Eletrónico* (DRE). The results reported in this dissertation suggest that there are two types of users: (i) expert users that do more target search (i.e., using document identifiers) and (ii) users that looking for more general information (i.e., queries without numeric characters). This analysis will be helpful to improve the effectiveness and efficiency of *DRE* search engine, being the first time this dataset was evaluated for this task.

5.2 Future Work

Despite the interesting results, there are also many possibilities for improvement in future work. For instance, in terms of segmenting user sessions and missions, instead of relying on traditional string similarity metrics (e.g., the Jaccard similarity coefficient between character n-grams), I can perhaps experiment with the use of learned similarity functions (Santos et al., 2018b,a; Gan et al., 2017), which have been shown to achieve significantly better results in other types of string matching problems. One such similarity function could be pre-trained on general data from other domains besides query logs (e.g., on large collections of alternative names for Wikipedia entities), and then used in the segmentation procedure. Another idea relates to the inclusion of additional steps in the proposed procedure, improving on the computational performance by first relying on simple string similarity metrics with a very permissive threshold, and later using increasingly more reliable, although more compute-intensive, string similarity functions with more restrictive thresholds.

In terms of log analysis, one particular challenge that I plan to tackle in future work is to leverage the session identification method for building user click models from large datasets. Taking this into account, It will be interesting to extend the study to click-through data to better understand the pages that the users visited after submitting a query, using this information to build click models to better infer user satisfaction. In this case, one possible strategy is to see the difference of the organic CTR (i.e., the percentage of searchers that click on a search engine result, which depends on (i) ranking position, (ii) document title, (iii) description, (iv) URL, and (v) rich snippets) on mobile compared to desktop.

Additionally, previous studies have defined a classifier only considering the CTR information and the next query submitted by the user as the main signals for predicting query success (Hassan et al., 2013b). However, CTR information can be noisy due to snippet bias and result presentation bias. Another idea relates to the inclusion of an additional step, considering the distribution of click dwell time, taking into consideration the characteristics of the clicked document: (i) size of the document, (ii) topic of the document through the Open Directory Project (ODP)[2] which has been widely used for topical classification of query text and web pages (Bennett et al., 2010; Xue et al., 2008), and (iii) readability of the document to identify if a particular document is difficult to read and understand (Collins-Thompson and Callan, 2004). Finally, one possible evaluation could be using the dataset proposed by Ageev et al. (2011).

[2] `http://odp.org/`

Bibliography

Ageev, M., Guo, Q., Lagun, D., and Agichtein, E. (2011). Find it if you can: a game for modeling different types of web search success using interaction data. In *SIGIR*.

Amigó, E., Gonzalo, J., Artiles, J., and Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4).

Bennett, P. N., Svore, K., and Dumais, S. T. (2010). Classification-enhanced ranking. In *Proceedings of the International Conference on World Wide Web.*

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan).

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5.

Catledge, L. and Pitkow, J. (1995). Characterizing browsing strategies in the world wide web. *Computer Network and ISDN Systems.*

Chapelle, O. and Zhang, Y. (2009). A dynamic Bayesian network click model for web search ranking. In *Proceedings of the International Conference on World Wide Web.*

Chau, M., Fang, X., and Yang, C. C. (2007). Web searching in chinese: A study of a search engine in hong kong. *Journal of the American Society for Information Science and Technology*, 58(7).

Christen, P. (2006). A comparison of personal name matching: Techniques and practical issues. In *Proceeding of the Workshop of the IEEE International Conference on Data Mining.*

Chuklin, A., Markov, I., and Rijke, M. d. (2015). Click models for web search. *Morgan & Claypool Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3).

Cohen, W., Ravikumar, P., and Fienberg, S. (2003). A comparison of string metrics for matching names and records. In *Proceedings of the SIGKDD Workshop on Data Cleaning and Object Consolidation.*

Collins-Thompson, K. and Callan, J. P. (2004). A language modeling approach to predicting reading difficulty. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics*.

Craswell, N., Jones, R. R., Dupret, G., and Viegas, E. (2009). Proceedings of the 2009 workshop on web search click data. In *WSCD@WSDM*.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3).

Downey, D., Dumais, S. T., and Horvitz, E. (2007). Models of searching and browsing: Languages, studies, and application. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Dupret, G. E. and Piwowarski, B. (2008). A user browsing model to predict search engine click data from past observations. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Feild, H., Allan, J., and Jones, R. (2010). Predicting searcher frustration. In *Proceedings of the ACM Conference on Research and Development in Information Retrieval*.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

Gan, Z., Singh, P. D., Joshi, A., He, X., Chen, J., Gao, J., and Deng, L. (2017). Character-level deep conflation for business data analytics. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*.

Gayo-Avello, D. (2009). A survey on session detection methods in query logs and a proposal for future evaluation. *Information Sciences*, 179(12).

Gomes, P., Martins, B., and Cruz, L. (2019). Segmenting user sessions in search engine query logs leveraging word embeddings. In *International Conference on Theory and Practice of Digital Libraries*.

Grotov, A., Chuklin, A., Markov, I., Stout, L., Xumara, F., and de Rijke, M. (2015). A comparative study of click models for web search. In *Proceedings of the International Conference of the Cross-Language Evaluation Forum for European Languages*.

Hagen, M., Gomoll, J., Beyer, A., and Stein, B. (2013). From search session detection to search mission detection. In *Proceedings of the Conference on Open Research Areas in Information Retrieval.*

Hagen, M., Stein, B., and Rüb, T. (2011). Query session detection as a cascade. In *Proceedings of the ACM International Conference on Information and Knowledge Management.*

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1).

Hassan, A., Shi, X., Craswell, N., and Ramsey, B. (2013a). Beyond clicks: Query reformulation as a predictor of search satisfaction. In *Proceedings of the ACM Conference on Information and Knowledge Management.*

Hassan, A., Shi, X., Craswell, N., and Ramsey, B. (2013b). Beyond clicks: query reformulation as a predictor of search satisfaction. In *Proceedings of the ACM international Conference on Information and Knowledge Management.*

He, D. and Göker, A. (2000). Detecting session boundaries from web user logs. In *Proceedings of the BCS-IRSG Annual Colloquium on Information Retrieval Research.*

Hogg, R., McKean, J., and Craig, A. (2012). Introduction to mathematical statistics, Pearson.

Hölscher, C. and Strube, G. (2000). Web search behavior of internet experts and newbies. *Computer Networks*, 33(1-6).

Huang, J., Gao, J., Miao, J., Li, X., Wang, K., Behr, F., and Giles, C. L. (2010). Exploring web scale language models for search query processing. In *Proceedings of the International Conference on World Wide Web.*

Jaccard, P. (1912). The distribution of the flora in the alpine zone. *New Phytologist*, 11(2).

Jansen, B. J. and Spink, A. (2006). How are we searching the world wide web? a comparison of nine search engine transaction logs. *Information Processing and Management*, 42(1).

Jansen, J., Spink, A., Bateman, J., and Saracevic, T. (1998). Real life information retrieval: A study of user queries on the web. *SIGIR Forum*, 32.

Jansen Bernard, J., Spink, A., Blakely, C., and Koshman, S. (2007). Defining a session on web search engines. *Journal of the American Society for Information Science and Technology*, 58(6).

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*, 84(406).

Jiang, J., Hassan Awadallah, A., Shi, X., and White, R. W. (2015a). Understanding and predicting graded search satisfaction. In *Proceedings of the ACM Conference on Web Search and Data Mining*.

Jiang, J., Hassan Awadallah, A., Shi, X., and White, R. W. (2015b). Understanding and predicting graded search satisfaction. In *International Conference on Web Search and Data Mining*.

Joachims, T., Granka, L. A., Pan, B., Hembrooke, H., and Gay, G. (2005). Accurately interpreting clickthrough data as implicit feedback. In *SIGIR*.

Jones, R. and Klinkner, K. L. (2008). Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proceedings of the ACM Conference on Information and Knowledge Management*.

Keskustalo, H., Pirkola, A., Visala, K., Leppänen, E., and Järvelin, K. (2003). Non-adjacent digrams improve matching of cross-lingual spelling variants. In *Proceedings of the International Symposium on String Processing and Information Retrieval*.

Kim, Y., Hassan, A., White, R. W., and Zitouni, I. (2014a). Modeling dwell time to predict click-level satisfaction. In *Proceedings of the ACM Conference on Web Search and Data Mining*.

Kim, Y., Hassan, A., White, R. W., and Zitouni, I. (2014b). Modeling dwell time to predict click-level satisfaction. In *Proceedings of the ACM International Conference on Web Search and Data Mining*.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *Proceedings of the International Conference on Machine Learning*.

Liaw, S.-S. and Huang, H.-M. (2003). An investigation of user attitudes toward search engines as an information retrieval tool. *Computers in Human Behavior*, 19(6).

Mayr, P. and Kacem, A. (2017). A complete year of user retrieval sessions in a social sciences academic search engine. In *Proceedings of the International Conference on Theory and Practice of Digital Libraries*.

Mehrotra, R., Awadallah, A. H., Shokouhi, M., Yilmaz, E., Zitouni, I., El Kholy, A., and Khabsa, M. (2017a). Deep sequential models for task satisfaction prediction. In *Proceedings of the ACM on Conference on Information and Knowledge Management.*

Mehrotra, R., Awadallah, A. H., Shokouhi, M., Yilmaz, E., Zitouni, I., El Kholy, A., and Khabsa, M. (2017b). Deep sequential models for task satisfaction prediction. In *Proceedings of the ACM Conference on Information and Knowledge Management.*

Mehrzadi, D. and Feitelson, D. G. (2012). On extracting session data from activity logs. In *Proceedings of the Annual International Systems and Storage Conference.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems.*

Monge, A. E., Elkan, C., et al. (1996). The field matching problem: Algorithms and applications. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Montgomery, A. L. and Faloutsos, C. (2001). Identifying web browsing trends and patterns. *Computer*, 34(7).

Morita, M. and Shinoda, Y. (1994). Information filtering based on user behavior analysis and best match text retrieval. In *SIGIR.*

Murray, G. C., Lin, J., and Chowdhury, A. (2006). Identification of user sessions with hierarchical agglomerative clustering. *Proceedings of the American Society for Information Science and Technology*, 43(1).

Ozmutlu, S., Ozmutlu, H. C., and Spink, A. (2008). Automatic new topic identification in search engine transaction logs? using multiple linear regression. In *Proceedings of the Hawaii International Conference on System Sciences.*

Pass, G., Chowdhury, A., and Torgeson, C. (2006). A picture of search. In *Proceedings of the International Conference on Scalable Information Systems.*

Philips, L. (2000). The double metaphone search algorithm. *C/C++ Users Journal*, 18(6).

Radlinski, F. and Joachims, T. (2005). Query chains: learning to rank from implicit feedback. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining.*

Relaxation, S. (1984). Gibbs distributions, and the Bayesian restoration of images. s. geman and d. geman. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6).

Rose, D. E. and Levinson, D. (2004). Understanding user goals in web search. In *Proceedings of the International Conference on World Wide Web*.

Sahami, M. and Heilman, T. D. (2006). A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the International Conference on World Wide Web*.

Santos, R., Murrieta-Flores, P., Calado, P., and Martins, B. (2018a). Toponym matching through deep neural networks. *International Journal of Geographical Information Science*, 32(2).

Santos, R., Murrieta-Flores, P., and Martins, B. (2018b). Learning to combine multiple string similarity metrics for effective toponym matching. *International Journal of Digital Earth*, 11(9).

Silverstein, C., Marais, H., Henzinger, M., and Moricz, M. (1999). Analysis of a very large web search engine query log. *ACM SIGIR Forum*, 33(1).

Spink, A., Wilson, T., Ellis, D., and Ford, N. (1998). Modeling user's successive searches in digital environments. *D-lib Magazine*, 4(4).

Su, L. T. (2003). A comprehensive and systematic model of user evaluation of web search engines: Ii. an evaluation by undergraduates. *Journal of the American Society for Information Science and Technology*, 54(13).

Su, N., He, J., Liu, Y., Zhang, M., and Ma, S. (2018). User intent, behaviour, and perceived satisfaction in product search. In *International Conference on Web Search and Data Mining*.

Swanson, D. R. (1977). Information retrieval as a trial-and-error process. *The Library Quarterly*, 47(2).

Wang, C., Liu, Y., and Ma, S. (2016). Building a click model: From idea to practice. *CAAI Transactions on Intelligence Technology*, 1(4).

Wang, C., Liu, Y., Wang, M., Zhou, K., Nie, J.-y., and Ma, S. (2015). Incorporating non-sequential behavior into click models. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Wang, C.-J., Lin, K. H.-Y., and Chen, H.-H. (2010). Intent boundary detection in search query logs. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*.

White, R. W. and Huang, J. (2010). Assessing the scenic route: measuring the value of search trails in web logs. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Willett, P. (1988). Recent trends in hierarchic document clustering: a critical review. *Information Processing and Management*, 24(5).

Winkler, W. E. (1990). String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage. *Proceedings of the Section on Survey Research Methods.*

Xue, G.-R., Xing, D., Yang, Q., and Yu, Y. (2008). Deep classification in large-scale text hierarchies. In *SIGIR.*

Zhang, Z., Sun, L., and Han, X. (2013). Learning to detect task boundaries of query session. In *Proceedings of the ACM International Conference on Information and Knowledge Management.*