University of Stavanger

Faculty of Science and Technology

MASTER'S THESIS

Study program/ Specialization: COMPUTER SCIENCE	Spring semester, 2015 Open access
Writer: AMAN BERHANE GHIRMATSION	……………………………………… (Writer's signature)
Faculty supervisor: DR. KRISZTIAN BALOG External supervisor(s):	
Thesis title: PROBABILISTIC FIELD MAPPING FOR PRODUCT SEARCH	
Credits (ECTS): 30	
Key words: Product Search, Generative probabilistic retrieval,Term-specific mapping , Living Labs , online evaluation	Pages: 53 + enclosure: 1 Stavanger, 15th June, 2015

Probabilistic Field Mapping for Product Search

Aman Berhane

Department of Electrical Engineering and Computer Science

University of Stavanger

June 2015

Preface

This master thesis paper on 'Product Search' culuminates my Masters Studies in Computer Science at University of Stavanger.

I would like to express profound thanks to my supervisor Dr. Krisztian Balog for his invaluable assistance and support throughout the course of my thesis work .

I also would like to bestow my gratitude to my family and my girl friend for their limitless encouragement and blessings.

Aman Berhane
15 June, 2015

Contents

List of Tables

List of Figures

Abstract

Online shopping has shown a rapid growth in the last few years. Robust search systems are arguably fundamental to e-commerce sites. Most importantly, sites should have smart retrieval systems to present optimized results that could best satisfy customers purchase intent. To address the demand for such systems we adapted retrieval approaches based on a generative language modeling framework, representing products as semi-structured documents. We present and experimentally compare three alternative ranking functions which make use of different prior estimates. The first method is static field weighting approach relying on field's individual performance taking nDCG as an effectiveness measure. Two other methods dynamically assign term-field weights according to the distribution of terms in field's collection. These retrieval functions infers from user search keywords the most likely matching product property probabilistically. The methods differ as one of them considers a uniform field prior whereas the other utilizes performance based prior. The methods were evaluated in relatively new evaluation methodology that evaluated ranking systems when real customer were doing online shopping at toy webshop 'regiojatek.hu' : Living labs. In the experiment the lab present an interleaved result, based on Team draft interleaving, from production site and our experimental rankings to customers. The Lab employ an evaluation metric "outcome" and we applied outcome measure to compare our methods and to interpret our results. Our results show that both term-specific mapping methods outperformed the static weight assignment approach. In addition results also suggest that estimating field mapping priors based on historical clicks does not outperform the setting where the priors are uniformly distributed. Furthermore,we also discovered that a trec-style evaluation carried out deeming historical clicks as relevance indicators had ordered the methods inversely in relation to Living labs. This has possible implication that Living labs evaluation platform are essential in IR tasks.

1 Introduction

Effective retrieval systems are indispensable in e-commerce sites. Online shopping has become a popular phenomenon recently mainly due to its simplicity,large selection of products, convenience , easy price comparisons to mention some. Webshops need to tune their retrieval systems to enhance customers shopping experience by making available products easy to find. Product retrieval systems have to be able to present customers with products that match their needs precisely so that searching and finding a product wouldn't be time consuming. Apparently, there is no doubt that customer satisfaction influences the revenue positively. Typically, products can be searched using structured query language (SQL) statements from product database. Nonetheless, users hardly know how to formulate such structured queries to compose a well-formed information need neither do they have knowledge of the background data.Therefore retrieval system that can infer user information need from queries composed of handful keywords is vital. Thus we shall be considering customary retrieval system built on a "single box search" paradigm where users type in information need, product query, and the retrieval system presents a ranked list of matching products.

In information retrieval task, a well-founded document modeling approach has significant role in improving ranking of documents. General probabilistic retrieval models have been adopted in various ad-hoc retrieval tasks and have proven to be crucial owing to the flexibility to be adapted in various tasks and the simplicity they provide to incorporate document and query models into ranking system. Products data usually consists of some common properties that expresses their behaviour such as product's name, description about the product, product's brand information and its category. These inherent fielded organization in product data make them to qualify under the group of structured/semi-structured data.Optimizing the ranking of returned result in product search task hasn't got focus and little is being done to devise novel retrieval models. In [2] product retrieval model was proposed which followed generative probabilistic models approach to improve retrieval accuracy. Besides they have incorporated associated text such as user product reviews to enhance effectiveness and pointed out significant improvement on retrieval accuracy when compared to baseline Language model and variants. To address product search tasks, we propose retrieval functions based on conducive retrieval models applied to search tasks involving similar data organization. More specifically, we ask the question : Are term-specific mapping approaches better than static field perfomance based weighting in product search task ?

Fielded document representation allows to split document "bag of words" representation, adopted in classic retrieval systems, conceptually to smaller sections of a document that can be searched independently. For example, when users pose a keyword query, q_p, that reflects the desire of a product item/s instead of searching the query over single document model, retrieval can be performed in a product's multiple property representations (e.g. brand document model and product's name document model). Decomposing the problem into parts enables us to analyse the outcome of searching on properties and eliciting property's importance level that may have an impact in influencing the ranking systems. More specifically, for a given user query q_p query likelihood in each of those field document models is combined to form a mixture document language model. The ranking of product item can then be computed as a linear interpolation of query generation probabilities in mixture language model where the importance level of each field is attached as weight of each component in the interpolation. Our rationale in this thesis is to develop, implement, and evaluate methods that establish a mapping from individual query terms to specific product fields. We represent products as semi-structured documents and employ the Probabilistic Retrieval Model for Semistructured Data (PRMS) [3], a model that is known to perform well under conditions where the collection is homogeneous and fields have distinctive term distributions [4]. Given the inherent uncertainty, the term-field mapping is represented as probability distribution over fields. We decompose the estimation into term-field probability and prior field probability components and propose three specific instantiations of the PRMS model .

In addition to a robust retrieval system, evaluation in IR is highly significant. Logically, evaluation of product search system has to reflect the satisfaction of users with respect to the ranking of the products that gets presented as a result of product query. In other words, assuming users preference to scan retrieved documents in top-down fashion, evaluation should reproduce the simplicity to come across the needed product both in terms of time and utility. Evaluation of retrieval quality can be done either from experts relevance judgements or gathered historical implicit user interactions. Typical evaluation methodologies, Cranfield paradigm, rely on offline evaluation of systems on test collections which are comprised of documents, topic queries and experts relevance judgements.The outcome of such offline evaluation methodology are standard evaluation metrics(MAP, nDCG and MRR).The lack of user interaction in the retrieval evaluation process may lead to tentativeness or may miss to measure other essential user behaviours. Therefore, we will address the doubts by performing evaluation of our retrieval methods on real e-commerce site (`'regiojatek.hu'`) when real users are engaged on online shopping i.e in live environment (living labs). Does the evaluation of rankings when using the offline method, with judgements built from historical user interaction, correlate with experiments done in live settings?

The rest of the thesis paper is organized as follows: Section 2 discusses traditional evaluation methodologies, metrics and various information retrieval methods. More specific generative probabilistic retrieval models employed to product search task are explained meticulously in Section 3. The next Section 4 provides an insight to relatively new evaluation platform :Living labs and elucidates the evaluation procedure and data interaction among experimental and commercial site. Furthermore, the structure of data including queries, documents and historical feedbacks that form the dataset of the experiment are illustrated in that section. Most importantly Section 5 explains the three methods we proposed for product search task. Just before concluding the thesis paper with Section 8, we cover our experiment results in Section 6 followed by deeper analysis in Section 7.

2 Related Work

2.1 Evaluation Methodologies

Evaluation of retrieval systems has been playing a crucial role in the advancement and comparison of retrieval methods and models [5]. Cranfield paradigm [6] an old but a very popular and systematic way of evaluating information retrieval methods was introduced back(by Cleverdon) in 1952, with main goal of comparing between library indexing techniques. According to Cranfield methodology assessing the effectiveness of a retrieval model requires a test collection. A test collection is composed of a set of documents to be searched , set of topics or information need and a relevance judgement. A relevance judgement indicates whether a document is relevant to the topic searched (information need) or not. In graded relevance judgement a label that indicates the degree of relevance is assigned to document-query pairs. When the volume of the collection is big, relevance judgments are not only take up great deal of time but also are too expensive [7]. Consequently, a pooling technique is adopted in TREC [8] to judge only selected relevant documents from retrieval systems taking part on TREC. Specifically, for each topic a set of top k relevant documents(typically 100 documents) in the ranking of every participating retrieval systems is put together and gets assessed by human assessors.

Cranfield paradigm has some limitations in relation to the assumptions taken [9]. There is assumption that documents don't have any kind of influence to one another in terms of their relevance. Moreover, it not only assumes a relevance judgements would appropriately represent the judgement of generally of every user's intent at the time of information need but also the relevance assessments are complete that all the relevant documents to a given information need is judged. These disadvantages have strengthen the idea of devising better evaluation methodologies that better accommodate users behaviour.

Back in time, in [10] desktop search engine named "Staff I've Seen" was deployed at testing period with different interfaces to workers of an organization arbitrarily.Their objective was to comprehend how employees used the system based on logs of user interaction and also to observe the changes in user behaviour in relation to the interfaces.

Cooper [11] proposed a way of analyzing user behaviour by assigning users to perform search tasks in an old and new systems,without their knowledge, and investigate their activities live.Others have developed evaluation measure that closely reflects different user models so that to better determine systems utility to users [12].

Controlled experiments(A/B testing) ,that involve live users and studying their interaction, has been conducted to evaluate significance of new ideas in some popular commercial sites like Amazon ,Google and Microsoft [13]. During controlled experiments users are presented with two different versions: Control which is the original one and Treatment which is implemented applying new ideas. Thus to evaluate the new system user interaction and behaviours are observed and the difference among systems is determined statistically according to some metric of interest.
Recently, a relatively new evaluation methodology for information retrieval systems, similar notion to A/B testing methodology, was proposed by [14] that enables researchers to evaluate systems when real users (e.g customers in ecommerce webshop) carrying out real activities on genuine applications.[15] discussed thoroughly a practical living labs architecture design and associated evaluation platform ,that lays a groundwork for researchers and makes available needed data to conduct experiments.

Unlike in A/B testing which presents separate systems to the user,in [16] an interleaving technique was proposed to contrast different systems by merging the results of the two systems to one ranking and examining relative user inclination towards a system by the volume of clicks a system received. Balanced interleaving(BI) technique tries to avoid user result position biases by presenting an intermingled rank in balanced manner. Another method named Team-Draft interleaving introduced by [17] creates a ranked list picking top documents in randomized way from two comparing ranking systems. The method solves limitations of BI technique which is prone to bias on similar rankings.

2.2 Evaluation Metrics

Next we will discuss three standard evaluation metrics widely adopted in information retreival.

2.2.1 Normalized Discounted Cumulative Gain (nDCG)

Normalized Discounted Cumulative Gain is one of the popular evaluation metrics used in ranked retrieval [18]. nDCG manages a graded relevance assessments of document which make it quite different from former relevance metrics. For instance, document may be judged to be irrelevant , less relevant or very relevant each with decreasing relevance levels. Further a gain value is assigned to each relevance levels which of course reflects and agrees with its relevance. In simplest cases gain value can be considered to be equal to the relevance level value where relevance values start from 0(irrelevant) and increases monotonically to the most relevant document.

Originally nDCG measure was what followed a CG measure that didn't take into account the ranking position of documents during evaluation of a result-set. Formally CG is given by :

$$CG_p = \sum_{i=1}^{p} g_i$$

where g_i is gain value for document at rank i.
The fact that query results may vary in size makes it illogical to use DCG when making comparison among queries.Thus DCG has to be normalized to accomplish effectiveness comparison of retrieval system. In nDCG this is achieved by dividing the DCG measure at p by an ideal ranked list (scores highest possible DCG) that is generated by ordering documents in decreasing relevance level.

One way of calculating DCG according to [19] is :

$$DCG_p = g_1 + \sum_{i=2}^{p} \frac{g_i}{\log_2 i}$$

where p is called document cut-off.
Apparently, given a ranked result list of documents , a change in rank sequence where highly relevant document shifted from top to bottom wouldn't alter CG value. Thus more exact measure a discounted CG(DCG) fixes the problem by imposing penalties to very pertinent documents returned at bottom.

$$nDCG_p = \frac{DCG_p}{IDCG_p}$$

2.2.2 Mean Average Precision

The mean average precision(MAP) is another common evaluation metrics that is standard in TREC community. When compared relative to other effectiveness metrics MAP provides robust features such as good discrimination and stability [20]. For a single topic q , the average precision AP is computed by averaging precision measure at levels where a relevant document shows up on the result set of top k documents. Following, MAP measure for several topics in a run can be achieved by averaging the AP measures of each particular topics in the run.

Let I_k be a binary relevance of document at level k such that its 0 when document at k is non relevant and 1 if relevant. R be size of relevant documents of a topic $q \in Q$ $P_k$be precision at some level k- the number of pertinent documents in top k divided to k

$$P_k = \frac{\sum_{r=1}^{k} I_r}{k} \qquad\qquad AP = \frac{1}{R}\sum_{k=1}^{R} I_k P_k$$

And MAP is given by:

$$MAP(Q) = \frac{1}{|Q|}\sum_{q \in Q} AP$$

One of the advantages of MAP measure is that its delicacy to change of ranking position to relevant documents at the top. As a relevant document climbs some positions on result set (special at top) it contributes significantly to AP in contrast to position change at the bottom. For instance , let a document d was returned on 2 position in an updated system from 3rd place in the result sequence. Owing to this AP measure gains 0.17 (0.5 - 0.33) i.e from 0.33 when in third to 0.5 when in second.

Nevertheless , the absence of graded relevance in MAP metric leads to inability to discriminate retrieval systems which can produce ranking where highly relevant document returned before marginally relevant ones. The previously described metric nDCG can handle graded relevance judgements.

2.2.3 Mean Reciprocal Rank(MRR)

The above discussed metrics quantify the relevance based on the stipulation that users intent is to get sufficient relevant documents on the top of their search results. But there are case where the purpose of search is to find only one matching document as in known item search [19]. The Reciprocal Rank(RR) is appropriate and straightforward metric to assess relevance in such cases. Given a ranked list of documents , let r be the rank of needed document in that list with respect to a topic. Then RR is calculated by:

$$RR = \frac{1}{r}$$

The formula reflects the inverse proportionality of rank and the metric RR. If no relevant document is returned RR = 0 Thus Mean Reciprocal Rank averages RR measure over all the queries.

2.3 Information Retrieval Methods

The most important objective of information retrieval is : given an information need to return list of documents ranked according to their relevances where the most relevant document gets ranked at the top. To achieve this target optimal retrieval models are extremely important. Up to date although various retrieval methodologies has been adopted , but none has proved to be a superior [21].

Vector Space Model

In vector space model, documents are visualized as vectors consisting of term elements with corresponding weights that indicates the relevance of terms in the document $d = (w_{d,t_1}, w_{d,t_2} \dots w_{d,t_V})$ [22]. Equivalently, queries are represented as a vector of query terms and associated weights that shows importance of terms in the query. Typically term weights in document/query vector is determined by the product of term frequency(t_f) and Inverse document frequency(idf). Inverse document frequency component serves as a means of degrading abundant but irrelevant terms and is formally given by:

$$idf_t = \log \frac{N}{df_t} \qquad\qquad w_{d,t} = tf_{d,t} * idf_t \tag{1}$$

where N is number of documents in collection and df_t number of term t occurrences in all documents.

Now documents and queries being modelled as vector, a standard way of computing the similarity between those vectors is cosine similarity. Cosine similarity measure the angle difference among documents and the query when drawn in V-dimensional space (V is vocabulary size). Specifically,

$$sim(q,d) = \frac{\sum_t^V w_{q,t} * w_{d,t}}{\sqrt{\sum_t^V w_{q,t}^2 * \sum_t^V w_{d,t}^2}} \tag{2}$$

where $w_{q,t}$ denotes weight of term t in query q. The denominator normalizes the effect of document length in the scoring.

BM25

The BM25 model, initials represent for Best Match, is one of the popular classic retrieval models. The model is based on 2-Poisson model: we assume that documents constructed by inserting words , where each word has some probability of being selected from vocabulary (multinomial distributed) to be placed on certain position. As a result each word in a document has term frequency which is binomailly distributed [23]. The central idea the model addresses is to produce preferable term weighting functions based on the three important factors :(1) term frequency (2) Inverse document frequency and (3) document length normalization.

The document normalization depends upon the assumption of verbosity and scope. The former emphasizes documents may vary in word count even though they convey same information. In the other hand scope defines the depth to which extent documents explain contents. Based on these assumptions the normalization factor considers putting together the influences from both sides.

The document length is given by Eq.3 and it is independent of how the length is determined e.g instead of simply counting terms only the unique terms may get considered.

$$|d| = \sum n(t,d) \qquad\qquad avdl = \frac{\sum_d |d|}{N} \tag{3}$$

avdl - refers to average document length and a factor B is used to normalize length given by :

$$B = (1-b) + b\frac{|d|}{avdl}, 0 \leq b \leq 1 \tag{4}$$

then normalizing term frequency with length normalization factor B :

$$\begin{aligned} n(t,d)' &= \frac{n(t,d)}{B} \\ w_t^{BM25} &= \frac{n(t,d)'}{k_1 + n(t,d)'} * w_t^{RSJ} \\ w_t^{BM25} &= \frac{n(t,d)}{(1-b) + b\frac{|d|}{avdl} + n(t,d)} * w_t^{RSJ} \end{aligned} \tag{5}$$

The ultimate formula provides the term weighting for each of term t and therefore the relevance score of each document is computed by adding the term weights of component terms. The component w_i^{RSJ} represents weighting function based on the presence of relevance judgement (equals *idf* otherwise)and is given by the formula :

$$\log \frac{(r_t + 0.5)(N - R - n_t + r_t + 0.5)}{(n_t - r_t + 0.5)(R - r_i + 0.5)} \tag{6}$$

where
N : number of judged documents
n_t - number of documents in N where term t occurs
R: relevant documents
r_t: number of document in R where term i occurs

BM25f

The BM25F is weighted field variant of BM25 model. Analogous to field document representation model in language models the BM25 model has this field variant which reflect the concept that by applying ranking function to individual streams (fields) and then join together in some weight linear combination for computing the relevance of documents[23].

An easy form of BM25F takes into account a weighted variant of the total count of terms. Likewise, an alternative to total length of documents

, its weighted version is taken. The simple version can be viewed as if a search using BM25 was performed on documents where each field data in the document was replicated by a factor according to the weights assigned. The basic formula for BM25 remains same but the components change slightly as shown below:

Thus ,

$$\begin{aligned} n(t,d) &= \sum_{f \in F} \alpha_f \cdot n(t,d_f) \\ |d| &= \sum_{f \in F} |d_f| \qquad\qquad avdl = \frac{\sum |d|}{N} \end{aligned} \tag{7}$$

Language Models

Probabilistic language models are different from previously explained retrieval methods because documents are ranked based on probabilistic computations that depend on distribution of words on a document. Language models were successfully applied to other fields other than retrieval in the past such as speech recognition and machine translation systems[21]. A detailed explaination on various approaches of language model in information retrieval follows in the next section.

2.4 Product Retrieval

Lately [2] addressed the keyword queries search in product databases . They proposed a general probabilistic model for product entity retrieval which interprets the generative model as two step process . First a user interested in a product would draw specification (field) according to specification selection model $p(s|d)$ and then delve into the particular field and sample attribute related word based on conditional probability $P(w|s)$. The proposed methods were evaluated in two e-commerce related datasets.The model was taken further to manipulate product associated text such as user review data and log data into the retrieval model. Our work does resemble to this research in the sense that we employed probabilistic language models to product search too. But we adopt term-specific mapping probabilities with different versions instead of $P(s|d)$ which is uniform distributed among product possessing the specification. Besides we followed a different evaluation platform, Living Lab platform, to compare our methods. Their model has shown significant improvement over the baseline language model which gave us impetus to research on other version that hypothetically can have good potential to succeed.

3 Retrieval Framework & Baseline Methods

Our main aim in information retrieval,ad-hoc retrieval in this case, is retrieving relevant ranked documents that include the information need of users represented by keyword query. Thus we need models that reflect this core idea to satisfy information requests from users with best possible matching documents. A probabilistic language modelling approach was proposed by Ponte and Croft [24]. Their work brought to knowledge a novel idea of ranking document based on language models often referred as query likelihood method.The assumption taken is : the user searching has thought of archetypal document, is able to select terms which with a sound high degree of precision will be more probable to exist in that document and that query will distinguish it from the rest documents in the collection[20]. The approach can be described as process of generating a query from a previously constructed document language model and then ranking the documents according to the probability of the generation process. Document language model indicates to probability distribution of terms in a document. Intuitively, the relevance of document according to a given topic(query) is directly proportional to frequency of the query keywords existing on the document . This can be reasoned as : the probability of drawing a frequent query term as a sample from a document would be naturally high.

Language models have been applied to tasks other than information retrieval such as speech recognition and spelling modification.But these tasks employ bigram or trigram language models due to the importance of the order of words to accomplish the desired results. In information retrieval though, a unigram language model is most commonly adopted since the composition of the particular document in a collection not necessarily reflected by certain sequence of words.Generative language models provides flexibility to utilize document internal form such that it can be intergrated into retrieval process.

3.1 Query Likelihood Model

The main rationale being ranking documents according to their relevance to a query, the query likelihood model represents the problem in probabilistic terms as - How likely a document is pertinent to a given information need or query. Formally, given a query q and a document d the score of a document with regard to the query can be expressed as a conditional probability $p(d|q)$. Using Bayes' theorem $p(d|q)$ is formulated in Eq.8 below. Since the p(q) is identical for all candidating documents, it doesn't have any impact in the ranking formula. Thus the probability p(d|q) is given by the product of query generation probability and document prior probability p(d).

$$P(d|q) = \frac{P(q|d)P(d)}{P(q)} \approx P(q|d)P(d) \tag{8}$$

The document prior probability ,$p(d)$, can be taken as constant throughout whole collection and hence it hardly influences the calculation for $p(d|q)$. In other instances special characteristics such as the number of past visits to a document, the length of a document , popularity are adopted to determine the value [25]. As a result the ranking problem basically boils down to estimating $p(q|d)$ using unigram language model. Therefore the retrieval process is construed as a generative process of sampling a query from a document model. Previously both bernoulli process and multinomial process were adopted to generate a query from document model [24, 26].In our case, as stated previously, we use multinomial unigram model which is widely considered as standard query generation process of language models in information retrieval. In our subsequent discussions we will focus in estimating the conditional probability $p(q|d)$ using various document models.

3.2 Document Modeling

Based on the structure of documents , language models are tweaked to better serve the purpose of ranking. In this section two types of document orientations and corresponding generative process are discussed. First being the flat document representation which operates by ignoring the existence of structure while the second one considers structures in form of fields constructing the document.

3.2.1 Flat Document Representation

As the name self describes, flat document representation treats a document only as group of terms without taking in to account the existence of organization of any type internally. Unigram language model of such documents assign probability for each term based on how frequent they appear on a document. Apparently the more the term occurs in the document model , the more the document would match to the query composed of that term. The language model of this paper would for instance would get higher relevance score for query "probabilistic language models" and relatively much lower score to query "live sport events". An important assumption made in language models is the independence between distinct terms. This means that any term occurring in a document doesn't rely on any term appearing before or following it.

We shall refer to document model with symbol $\theta_d$and term probability with respect to the model as $P(t|\theta_d)$. It can be inferred from our formula Eq. 8 that it is one of the most important components that has to be estimated. According to maximum likelihood estimator $P(t|\theta_d)$ is a relative frequency of term t in document d.

$$P(t|\theta_d) = \frac{n(t,\theta_d)}{|d|} \tag{9}$$

where$|d|$ is the total number of terms in document d and $n(t,d)$ is the count of term t in document d. The formula indicates terms get assigned probabilities based on how frequent they occur in a document.

The unigram language model is the easiest way of estimating query probability. The model treats a query as group of independent terms(unigrams) where each term contributes to the probability without influence from other neighbouring ones. In other words the order of appearance of terms in the query is not essential.Thus as long as queries have same composition the probabilities would be equal.The probability of query constituted of sequence of terms becomes the product of the probabilities of individual terms.

$$P(q|\theta_d) = \prod_{t \in Q} P(t|\theta_d)^{n(t,q)} \tag{10}$$

Product name	Horse of the year
Category	Dolls
Description	Horse Of The Year : The Horse is a cross breed between Quarter Horse and Thoroughbred. Quarter horse is strong and fast riding horse, can compete in horse races, riding events, entertaining horse shows.

Table 1: A document :Horse toy

where $n(t, q)$ indicates how frequent term t appears in query q.

Let see a product search example that applies the discussed formulas. Assume a user searching for a horse doll in an online toyshop. The user type in a query `q="riding horse"`, and lets assume in our collection there exists a product document that looks like Table 1. Note that even though the table depicts a structure of the document , in flat document representation model the textual content is treated as if belonging to a field.

Using Eq.9 and Eq. 10:

$$P(\text{"}\texttt{horse}\text{"}|\theta_d) = 0.184$$

$$P(\text{"}\texttt{riding}\text{"}|\theta_d) = 0.052$$

$$P(\text{"}\texttt{riding horse}\text{"}|\theta_d) = 0.00968$$

Typically the term probabilities are small and their product fastly approaches to zero, thus multiplying both sides by logarithm is necessary. Therefore, we apply log in both side of the Eq. 10 to get :

$$\log P(q|\theta_d) = \sum_{t \in q} n(t, q) P(t|\theta_d) \tag{11}$$

This simple solution is susceptible to insufficient availability of text in documents. What happens if a term doesn't exist in a document or if term appears seldom? For non-existent terms apparently the term probability computed using Eq. 9 shall be zero. Consequently, the probability of query which holds a single nonexistent term gets a zero probability which shall significantly limit the ranking approach. In other words , the model is stern as a document must consist all the terms in the information need in order to be treated relevant which makes the ranking of documents illogical.In our example above if we search for `"horse pony"` then $P(\text{"horse pony"}|\theta_{\mathtt{d}}) = 0$ because term `pony` doesn't occur in the document. Besides terms that appear rarely and specially those that occurs only once, gets inflated probabilities.

We can come around this problem using smoothing.Smoothing is the adjustment of maximum likelihood estimator so that language modeling approach would be suitable and practical for retrieval purposes. Smoothing deals with the problem by diminishing the probabilities of words seen and compensating it by raising the probabilities of missing words, thus avoiding zero probabilities. Moreover, smoothing serves as term weighting factor that can enhance the precision of term probability. An easy but feasible way of smoothing is computed by combining document-specific multinomial distribution and a multinomial distribution estimated from entire collection. So when a term does not occur in a document, the probability of the term in the collection is taken as the term's probability according to document model.The method formulated below is called the Jelinek-Mercer method.

$$P(t|\theta_d)_{Jelinek} = (1-\lambda)P(t|d) + \lambda P(t|C) \tag{12}$$

where : $0 < \lambda < 1$ and P(t|C) denotes the probability of term based on entire collection's language model. In same manner to Eq .9 with ML the p(t|C) can be estimated by as. According [27] For keyword queries (short queries), highest retrieval effectiveness is attained for $\lambda = 0.1$, In the contrary for longer queries favorable choice for λ is larger which is around 0.7 . Thus more smoothing is applied to long queries than short ones.

$$P(t|C) = \frac{\sum_d n(t,d)}{|C|} \tag{13}$$

Another smoothing method known as Dirichlet smoothing [27] , unlike Jelinek-Mercer it depends on the length of documents. As the value of μ decreases the $P(t|\theta_d)$ is mainly influenced by number of matched keywords. :

$$\lambda = \frac{\mu}{|d| + \mu} \tag{14}$$

where μ denotes smoothing parameter. Finally putting it all together with Dirichlet smoothing we get :

$$P(t|\theta_d)_{dirichlet} = \frac{1}{|d| + \mu}\Big(n(t,d) + \frac{\mu \cdot \sum_d n(t,d)}{|C|}\Big) \tag{15}$$

3.2.2 Fielded Document Representation

A good candidate for representing documents can be based on the document's internal fields. This fielded document representation may seem similar to database way of organizing data in to fields. But unlike in database tables, the fields are not dictated by specified set of rules that define their composition. Instead contents of same field not only can possess any type of data but also may differ in their length. Besides fields don't necessarily hold data i.e they can be empty. Sections of a document data may also lack any kind of structure e.g. an image. A big fraction of documents in the web includes fields within them for instance research papers are composed of abstract , title ,introduction ,references etc fields.

An extended language modeling approach would blend the fielded document representations into one mixture language model that roughly calculates the query generation process, and then carry out retrieval using the mixture language model [28].

The issue of combining document representation to the retrieval model can be approached analogously to meta-search problem. In meta-search , the target is to put together the retrieved outcomes from different search engines in order to deliver a single ranked list that outperforms any single search engine. Thus searching on each field document representations independently and performing a meta-search would result final ranked list. Another option to this technique would be leveraging document representation to weight terms when searched inside the field representation. In our discussion we shall adopt the latter , where mixture language model (MLM) is estimated depending on the merge of language models emanating from the different document fields.

Thus, we need to define a formula that embraces the different field document representations to estimate the mixture of various language models. A simple technique of combining field language models would be a linear interpolation:

$$P(t|\theta_d) = \sum_{f \in F} \alpha_f P(t|\theta_{d_f}) \tag{16}$$

Field	**Content**
Product name	Lego Vulture Droid
Main category	Építőjáték, LEGO
Characters	Star Wars
Category	LEGO
Brand	LEGO

Table 2: Product example from site regiojatek.hu

where $P(t|\theta_{d_f})$ refers to field language model ,and F is the group of fields. Since the probability distribution to be valid the coefficients have to sum up to 1.

$$\sum_{f \in F} \alpha_f = 1$$

The estimation of $P(t|\theta_{d_f})$ can be done much same way to $P(t|\theta_d)$ in Eq. 12. We should consider the existence of term t within the field f instead of the whole document. Thus we replace $p(t|d)$ in the first component of Eq.12 by $p(t|d_f)$ and obviously the $p(t|C)$ is substituted by $p(t|C_f)$, where C_f denotes a collection language model with respect to the field.

$$P(t|\theta_{d_f}) = (1 - \lambda_f) P_{ML}(t|d_f) + \lambda_f P_{ML}(t|C_f) \tag{17}$$

where

$$P_{ML}(t|d_f) = \frac{n(t, d_f)}{|d_f|} \qquad P_{ML}(t|C_f) = \frac{\sum_d n(t, d_f)}{\sum_d |d_f|} \tag{18}$$

Here $|df|$ stands for length of the field and $n(t, d_f)$ represents the recurrence of t in field f .

Back to Eq.16 , a crucial component which we haven't set yet is weight (α_f) related to the fields in the document. One factor that influence the value of weight factor is how terms in a document are distributed among the fields. To be specific, in some cases the fielded document representations not unrelated in sense that they basically describe a document similarly but may vary in length.On the other hand, in semi structured data like XML each element(field) may discretely express a document.In the next section we will discuss each of the cases .

Disjoint field representation of Document

As stated above the assumption in disjoint field representation is that terms in each fields describe various aspects of a document i.e each field has discrete distribution of words. In XML retrieval According [3] a query may possess a characteristics of implicitly mapping to each XML element. A simple search for a movie in IMDB with query "denzel washington action" doesn't return relevant results in the top. Instead, if the query terms where split and searched through special advanced title search interface in corresponding fields intended to be meant i.e "action" in genre and "Denzel Washington" in cast , returns appropriate results with movie like "The Equalizer" [3]. This implies that the intended information need can be inferred from the query terms using flexible methods to map those terms to correlating fields.

Studying how terms are spread over several fields provides a good evidence to map query terms to intended fields. In previous example of searching in IMDB, the query term 'action' shall occur repeatedly under the genre field,on the other hand the terms 'Denzel' and 'Washington' appear mainly in the cast field. But still the assumption stated at the beginning is important in the sense that when query term match to different fields there is a possibility the retrieval system will not retrieve the expected results because of ambiguity.

Most importantly, the mapping relationship signifies the robustness of evidence each field gives for each query term. For instance in product search a query `"Horse doll"` may match several fields like product name ,category, description etc.But intuitively the user must have more probably needed to search for a product under category name `"doll"` and thus the category field gets assigned higher weight when the term is searched on the field.

Formally, applying bayes' theorem, the mapping probability of term to a field denoted by $p(f|t)$ can be estimated as a product of prior probability $p(f)$ and probability term field occurrence.This is given by:

$$P(f|t) = \frac{P(t|f)P(f)}{P(t)} \tag{19}$$

where

$$P(t) = \sum_{f' \in F} P(t|f')P(f') \tag{20}$$

$p(t)$ is substituted by applying the law of total probability. The value of prior probability $p(f)$,term field mapping before observing collection statistics,could be either taken uniform or assigned based on background knowledge. $p(t|f)$ is computed by dividing the number of times the term t is found in field f by total term count in the field f throughout the whole collection. Particularly, this can be seen as the probability of generating term t from a composition all terms appearing in the field f in the entire collection.

Subsequently, the estimated mapping probability is applied to weigh each field's contribution towards the score calculation in the probabilistic retrieval model for semistructured data(PRMS). Hence

$$P(t|\theta_d) = \sum_{f \in F} P(f|t)P(t|\theta_{d_f}) \tag{21}$$

Fields as alternative document representations

Some document's internal structure might not be distinctly spanned by the fields they are represented with. For instance document fields in may be composed of fields title , abstract, subject or heading fields in which some words occur both under title field and abstract, used with same interpretation. Therefore we need to have a sound way of field weighting in our retrieval model due the presence of query terms matching potentially in various fields. In presence of training data,the field weights(α_f) can be determined based upon individual field retrieval performance that is measured when retrieval is carried out on a field by itself [28]. Other relatively straightforward approaches are considering the field length across collection, total number of terms occurring in same field in entire collection, to dictate the weights. Much simpler technique can be to assign identical weight to each field i.e $\alpha_f = 1/|F|$, where F is number of fields.

4 Evaluation Methodology & Setup

Typical evaluation technique in information retrieval constitutes : a document collection, information needs represented by queries(topics) and relevance judgments which determines the presence or absence of relevance relationship between a document and a query pair [20]. Various standard test collections and evaluation methodologies such as CRANFIELD , TREC and INEX have been developed for ad-hoc information retrieval tasks. Due to access to common resources for performing retrieval evaluations, it is possible to set retrieval systems against each other and enhance retrieval algorithms. Although these methodologies have been popular for their effectiveness in evaluating retrieval system qualities , the process of collecting relevance judgement takes considerable amount of time [29]. Not only that its expensive but also presents some challenges to assessors to understand the query related information need. According to [30] this evaluation methods being laboratory experiments different choices for various variables are assumed to make an abstraction of the real world. This engenders unavoidable biases in what being examined. Another appealing and appropriate evaluation method is to perform evaluation with real users performing tasks using real-world applications [31].

4.1 Living Labs

“The basic idea of living labs for information retrieval is that rather than individual research groups independently developing experimental search infrastructures and gathering their own groups of test searchers for IR evaluations, a central and shared experimental environment is developed to facilitate the sharing of resources” [31]. They proposed a living laboratory ,in relation to challenge of Information-Seeking Support System(ISSS), which lays ground for researchers in form of both infrastructure and evaluation means to conduct experiments on user context.Similarly at [15] the notion of living labs is put as : Living labs have capability to supply a valuable user interaction data for researchers together with data repository to conduct worthwhile and appropriate research. Besides the lab serves as platform for conducting evaluation tasks for different participating retrieval methods and models.

4.2 Evaluation Platform

In this thesis a living labs evaluation platform that accommodate a benchmarking platform for participants to evaluate the effectiveness of retrieval systems where real users are engaged in search in a real world e-commerce site (a webshop). The lab serves as an intermediary in between the e-commerce site and our proposed retrieval methods. This means all the communications are guided through the living lab API , which feeds usage and interaction data from the commercial site to our retrieval model for training purposes. In addition the lab also make comparison among several participating experimental systems.

In order to accomplish a logical and acceptable comparisons among systems, systems are assigned similar set of queries and furthermore systems ranking gets presented to users relatively same number of times. Since the e-commerce site like an online toy shop attracts relatively fewer visitors than web search engines, the magnitude of queries requests might tend to be barrier as to how far the live environment would contribute to the research. Ideally, a living lab would be expected to act as a bridge between commercial site and experimental systems in the sense that all issued queries should be forwarded to participant's retrieval system in real time(during training phase). This lab solves these stated constraints by considering only head queries i.e queries which tend to be searched very often.

The decision of deliberately targeting head queries addresses most of the challenges stated in the living labs environment. Besides the communication between participant and living lab API is simplified because participating systems can prepare their runs (on head queries)offline and upload them to API. As a result these cuts the response time delay which the commercial site could be incurred to. Another advantage of eliciting head queries is the presence for a great deal of historical log data.

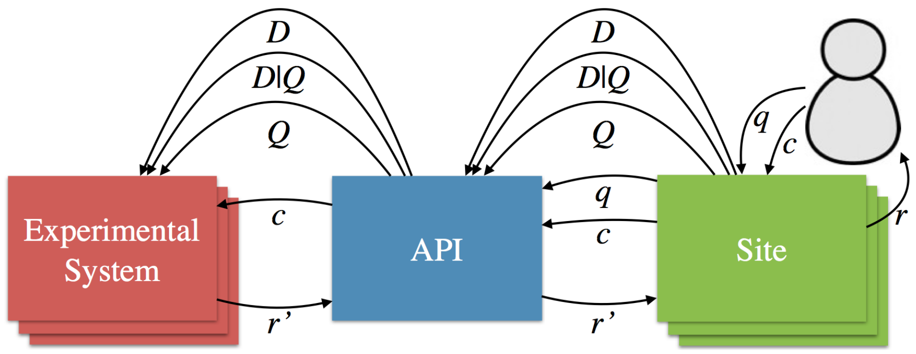


Figure 1: Living labs architecture[1]

4.3 Architecture

As shown in Figure-1 the search system is composed of an experimental system, a proxy API and commercial site. Initially, the commercial site makes available set of head queries Q , nominated document list matching each q in Q denoted by D|Q and respective document contents D. Users issue request query q to the site and receives a response ranked result set r. In the process the site saves the user interaction information c (click through data) of users. In particular when user submits query q that lies in set of head queries set Q, the site requests for ranking of experimental system r' through the API. Finally before presenting the ranking r' to the user, the site combines the rank with result from commercial site itself. In general,the API serves as data repository for previously upload runs from participants and makes sure information including queries, feedback and runs are relayed to and from experimental system to site.

4.4 Interleaving

Evaluation of retrieval quality of IR systems in living labs can be achieved based on implicit users feedback. This evaluation approach enables to compare systems against each other using relative or absolute metrics. Absolute metrics take into account a sole or handful noticeable user behaviours such as clicks ,time spent on examining result contents [32, 33] and session in-

formations to estimate user satisfaction [34]. In living labs, user interaction data is utilized to indicate relevances and standard IR metrics are computed based on the relevances.
On the other hand, a paired preference method (relative) in living labs elicits a superior system from two comparing system namely the commercial site and experimental system based on user behaviours (the count of clicks on system's ranking). In the latter case, one of the important design decisions living lab platform has taken is the interleaving method, that determines the way of presenting search results.

A balanced interleaving approach presents results by combining rankings of two system on an issued query and creates single ranking that represents both ranking systems approximately equally [16]. Consequently, the blended result would include top result from both systems on its top ranks. In special cases where the two ranking methods are roughly similar, balanced interleaving is susceptible to biased outcome as one method may get favored over the other.

Thus another interleaving method that solves this problem proposed by [17] is Team-Draft interleaving method based on the analogy of choosing teams for a friendly match. Here a combined rank is produced by selecting the best document ,which is not choosed before, from each ranking system but instead of each rank contributing turnwise , in each round one of the ranking system is given a chance randomly to pick a document. Due to Team Draft(TD) ability to address the previously stated problem in balanced interleaving its natural choice of interleaving method in living labs.

4.5 Evaluation Setup

The Lab arranged the evaluation in two phases namely the training and test phases.

Training Phase

The prime goal of this phase is to improve retrieval systems effectiveness through training data and user interaction data such as feedbacks. There are two means of training systems: TREC-style collections and living labs evaluation. The living labs training phases after uploading runs and feedback information can be retrieved from site so that to make modification on the ranking when appropriate.

In this thesis we have done the training based on TREC-style in which our collection is comprised of 50 frequent product queries, product documents and relevance judgements. The relevance judgement is prepared from historical click information of the training queries,historical feedback, of products i.e the fraction of clicks a product had when users search.

Test Phase

The Lab was setup as competing environment where different participant submit their rankings on the specific task (Living Labs challenge for product search under the name LL4IR CLEF 2015 Lab). Equivalent to training phase, test phase is undertaken with 50 product test queries, nominated documents and historical data (distribution of queries that led to a product). Unlike in training phase though, the ranking of products is never revised throughout the test phase period. We run the test phase for two weeks period.

4.6 Dataset & API

In this thesis we use the dataset from a toy store online ecommerce site in Hungary ,regiojatek.hu .The data generally includes the head queries of the site , list of document deemed to be relevant (i.e query and doclist pair) and each documents content. In the site, the percentage of queries issued by customers which lies under head queries make 25% of totally searched queries.On daily basis on average 1500 queries are submitted to the site out of which 380 of the queries are categorized under head queries.

All the communications between the ecommerce site and our client participant(experimental) system passes through the web API in living-labs [1]. The client fetches the queries, doclist and documents by calling corresponding API endpoints. Furthermore during training phase, after uploading runs through API a feedback can be retrieved back. Following a json format of each these data is shown.

4.6.1 Queries

Queries are defined by group of attributes (fields) name such as qstr (i.e query string) , qid - query identification , creation date and type that differentiate train query from test ones. Figure-2 below presents the format of queries.

```
{  "queries": [
         {
             "creation_time": "Thu, 27 Nov 2014 13:40:50 -0000",
             "qid": "R-q47",
             "qstr": "vonat",
             "type": "train"
         },
         {
             "creation_time": "Thu, 27 Nov 2014 13:40:50 -0000",
             "qid": "R-q1",
             "qstr": "monster high",
             "type": "test"
         }
     ]
}
```

Figure 2: json queries

4.6.2 DocList

A doclist represents an enumeration of candidate documents associated with a query. Below shows Figure-2 format (`UTF-8` encoded) of a doclist. The number of document list per query is not identical as it depends upon the number of products that the query may match to. Documents in the doclist have an id and title attributes. Our main objective is to rank these potentially matching product documents for each and every one of the query in test phase.

```
{  "doclist": [
        {"docid": "R-d903", "title": "Animagic \xfajsz\xfcl\xf6tt
        kedvencek: barna kutya"},

        {"docid": "R-d2217", "title": "YooHoo&Friens My YooHoo
        interakt\xedv figura "},

        {"docid": "R-d2136", "title": "Furreal Friends GoGo
         s\xe9t\xe1l\xf3 interakt\xedv kutyus"}

....}
```

Figure 3: Doclist

Since we are dealing with real commercial site , the product list apparently would be not static during the testing period. This means some product may go out of stock while other new ones arrive so during system comparison the commercial site may have an upper hand.

4.6.3 Documents

The API makes documents available by their unique identifier document ids. A document consists of many fields that describe it thoroughly. For retrieval purposes, we only discuss those fields that provide us rich textual content of products in Figure-5.

Field	**Description**
brand	Product's brand
category	Leaf level category
characters	Characters linked to product (e.g Barbie)
description	textual explanation of product
product_name	Name of product
queries	List of queries with probabilities through which the product got clicked
main_category	top level product category
short_description	concise product description

Figure 4: Fields description

```
{
   "content":
      {
   "category": "Bab\xe1k, Kellekek",
    "description":"" ,
    "main_category": "Baba, babakocsi",
    "brand": "Mattel",
    "queries": "barbie": "1.00000",
     "product_name":"Barbie Mariposa \xe9s a ...",
     "price": 4995.0,
    ... }
  "docid": "R-d1408",
  "creation_time": "Fri, 17 May 2015 04:55:54 -0000",
  "title": "Barbie Mariposa \xe9s a ..."
}
```

Figure 5: Document fields json

4.6.4 Runs & Feedbacks

Our client system uploads runs through living lab API . These contains a ranking of documents associated with query (ranked doclist). During the training phase, after uploading a run some amount of time is given for user to submit the query corresponding to the our run live on the commercial site. Then a feedback which indicates whether a document in the run was clicked by user or not. The format of runs and feedbacks are very alike except that feedback adds an indicator boolean attribute name clicked.

```
{
    "qid": "R-q47",
    "runid": "mlm"
    "creation_time": "Thu, 27 Nov 2014 13:40:50 -0000",
    "doclist": [
        {
            "docid": "R-d903"
        },
        {
            "docid": "R-d2217"
        }, ...
    ],
}
```

Figure 6: ranked documents : a run

4.6.5 Historical Feedback

Historical feedback data is related to each query and it provides a historical information of that query. Specifically, this information comes in the form of relevance between query and document pairs that is the fraction of clicks the product received for a given query. Based upon these data we generated a qrels file(relevance judgement) that enable us to train our system in TREC style. Below Fig.7 shows the format of a historical feedback for a query identified by the query id (qid).

```
{
    "feedback": [
        {"qid": "R-q1",
         "modified_time": "Sun, 27 Apr 2014 13:46:00 -0000",
         "type": "ctr",
         "doclist": [
             {"docid": "R-d903"
             "clicked": 0.6},
             {"docid": "R-d2217"
             "clicked": 0.3},
             ...
         ]},
         ...
}
```

Figure 7: Historical feedback

5 Retrieval Methods for Product Search

In this section we will be discussing three different methods employed for our product search task.The documents related to a product are fielded ones, in which each field provides an explanation of product's characterstics. For these reason, these fielded manner of organization enables us to use fielded document representations and take advantage of that to enhance the quality of our IR system. The methods discussed below are inspired by previously explained approaches in Sec.3 under disjoint field representation and alternative field representation , where the main difference among the methods is the way the relevance of fields is determined and incorporated to retrieval formulas.

As mentioned in previous section our client program retrieves queries, doclist and documents through Living Labs Restful API. First our client program requests for train queries using the appropriate API endpoints and thereafter for each product query a list of candidate product items is loaded. Then based on document ids (docid) , the contents of each site nominated product item associated to a query are fetched. A search index is prepared out of those documents to proceed with main task of searching. As part of a product document we have crucial query historical information containing distribution of query terms that led to that specific product item in the past. We have merged this information to our index of documents creating separate field by taking the proportion into account i.e mulitplying the relative frequencies of each query by 100 and concatenating the resulting number of terms.

The approaches adopted for product search are versions of mixture language modeling retrieval technique. We used Jelinek Mercer (JM) smoothing method to smooth our probabilistic estimations with a smoothing parameter $\lambda = 0.1$. Historical feedbacks are available for each query from which we created a relevance judgments to assist train our retrieval system offline. Below follows the explaination of the methods we proposed for product search.

5.1 Method 1

As stated previously, our methods are founded in mixture language model shown in Eq.22. This method assigns field weights on the basis of how significant a particular field is in the retrieval process. We assume that the performance of individual field is proportional to field's significance and thus our measure for importance is done by evaluating each field's quality when searched upon separately. Using TREC-style evaluation , since we have relevance judgement derived from historical data, the relevance metric nDCG is considered as a measure of each field's individual effectiveness. Then the weight of each field is taken as ratio of the field's nDCG result to the sum of all nDCG results from the other fields. This guarantees that all field weights add up to 1 as stated in Eq.22 Here we are repeating those formulas discussed in previous section for sake of simplicity.

$$P(t|\theta_d) = \sum_{f \in F} \alpha_f P(t|\theta_{d_f}) \qquad \sum_{f \in F} \alpha_f = 1 \tag{22}$$

where ,

$$\alpha_f = \frac{nDCG_f}{\sum_{f' \in F} nDCG_{f'}} \tag{23}$$

The Figure-3 displays the list product fields that have rich textual content and are important for retrieval purposes. It also shows the nDCG performance measure for every individual field together with the corresponding relative weights computed as per Eq.23. Note that the Contents field a catch all field, combination of all field contents, included while indexing. Note that the "contents" field embeds more fields than those listed such as "queries" field.

Field Name	**nDCG**	**Field weight (α_f)**
Brand	0.0684	0.024
Product Name	0.5632	0.1989
Characters	0.3792	0.1339
Category	0.4305	0.152
Description	0.3919	0.1384
Short description	0.2986	0.1054
Contents	0.6987	0.2468

Table 3: Performance based field weights

5.2 Method 2

The second method we considered treats field document representation as an exclusive document model and aims in singling out fields matching to the query components to enhances the robustness of IR system.The mapping probability of a field to query term P($f|t$) is computed as stated in Eq.19 and Eq.20. As such instead of using field weights derived from individual field performance as in method 1, the mapping probabilities obtained are used.Apparently, the computation of mapping probability depends on prior field probabilities P(f) component. In this method we take those probabilities to be uniform across fields, thus the mapping probability P($f|t$) simplifies to Eq.25 in which term mapping depends only on field collections. A field collection can be seen as a virtual document made concatenating all the data at that particular field from all documents in collection.

$$P(f|t) = \frac{P(t|f)P(f)}{P(t)} \qquad P(t) = \sum_{f' \in F} P(t|f')P(f') \tag{24}$$

$$P(f|t) = \frac{P(t|C_f)}{\sum_{f' \in F} P(t|C_{f'})} \tag{25}$$

where F denotes set of all fields

5.3 Method 3

Third and the last method proposed is mixture of those two methods explained above. In method 3, the prior field probabilities which were assumed to be identical thoughout the fields, while calculating the mapping probability in method 2, are replaced by individual field performance results from method 1.

thus method 2 can be rewritten as :

$$P(f|t) = \frac{P(t|C_f) \times \alpha_f}{\sum_{f' \in F} P(t|C_{f'}) \times \alpha_{f'}}$$

6 Experimental Results

This section describes the experimental results achieved for our proposed retrieval functions. We explain those results in two parts. First part represents results obtained performing offline evaluation based on traditional TREC-style on training queries. The second part elucidates online results from Living Lab experiment on the test queries.

6.1 TREC-style offline evaluation

Table 4 shows evaluation results achieved by preparing historical relevance judgement out of historical click distributions. In other words, graded relevance of each query product-pairs is determined by the fraction of clicks the products received (historical CTR) extracted from search log. For the binary relevance metrics (MAP and MRR) we consider historical CTR above 0.001 as relevant. Important thing to point out is that the collected historical CTR is probably biased by the site's ranking system. The evaluation results indicates effectiveness of the three methods based on retrieval metrics MAP, MRR,nDCG , nDCG@5 ,nDCG@10.

We can clearly see from the table that method 1 is superior to method 2 and method 3 in all the evaluation metrics except in MRR where all methods achieved same outcome. In addition, method 3 follows to be a runner up recording marginal difference from method 2 but still better in all measures. The best measures obtained in overall the metrics are highlighted in bold face.

6.2 Living Labs online evaluation

Our rankings run in Living labs test phase from 1st of May and continued until the 16th of May for during of two weeks. Table 5 depicts the result of the experiment conducted. Living Lab evaluation environment registers the number of wins, losses and ties for our experimental ranking methods against production ranking. When the product items in the experimental system receive more clicks than production system, a win is counted. Impressions refer to the number of times an interleaved ranking ,made up of a method ranking and production's ranking, is being presented to users. The main evaluation metrics is the outcome. Outcome is fraction number of wins of methods to total impressions without the ties given by:

$$outcome = \frac{\#wins}{\#wins + \#losses}$$

According to Living labs result shown in Table 5 , All methods received about the same number of impressions, the relative difference between them is within 10%. In over 70% of the cases there is a tie between the experimental and production rankings; this is the same for all three methods.

In terms of outcome metric method-2 outperforms both methods. This disagrees with results obtained based on historical relevance judgements. Method-2 has strong outcome metric with significant margin to method-1. In accordance with the outcome measure, method-2 has also registered more wins per impression against the production system. Method-3 holds second place but has achieved the lowest number of losses per impression. Overall, method-1 ‘s performance is degraded when compared to its effectiveness on historical judgements , dropping from first to third. Since method-1 is baseline PRMS which compute the mapping probability based on fields instead of query terms themselves , the relative performance recorded is not unanticipated. One important observation we made is that term-specific mapping is beneficial as reflected from both method-2 and method-3 prevalence over method-1.

Methods	**MAP**	**MRR**	**nDCG**	**nDCG@5**	**nDCG@10**
Method-1	**0.8118**	**0.9948**	**0.7045**	**0.5136**	0.5709
Method-2	0.7916	**0.9948**	0.7012	0.5121	0.5703
Method-3	0.7997	**0.9948**	0.7026	0.5108	**0.5710**

Table 4: Offline Evaluation for training queries , Best results show in boldface

Methods	**Outcome**	**Wins**	**Losses**	**Ties**	**Impressions**
Method-1	0.2827	54	137	508	699
Method-2	**0.3413**	**71**	137	517	725
Method-3	0.3277	58	**119**	488	665

Table 5: LivingLabs platform results

7 Analysis

In analysis section we begin our analysis on the presented methods both quantitatively and qualitatively. The variation among the three ranking proposed methods in product search is determined using correlation coefficient ,τ in Table 6. From the table we observe that the ranking of method-2 and method-3 are highly correlated and method-1 has relatively lower similarity to both of the methods. Despite the ranking resemblances, Table 7 below shows varying assignment of weights across the different fields of sample keyword queries searched in the experiment. The sample queries are composed of single term queries and a multiple term query. In method-1 the weights are constant across fields owing to the dependence solely on fields performance. While in method-2 and method-3, the weights alternate according to the mapping probability of terms to fields.

method	Method-2	Method-3
Method-1	0.867	0.864
Method-2	X	0.95

Table 6: Kendal correlation

In Figures (8,9,10) the wins method-X has recorded against the production system are represented by upper bars, where as the red bars correspond to losses per each and every query (both test and train queries). We can infer from the graphs that all the methods has registered losses specially in leftmost part of the plot. This shows beating the production systems was not a trivial task. Generally the shape of the graphs are comparable owing to the high correlation among the rankings. The deciding metric taken into account to contrast the systems is the outcome metric , and according to outcome metric method 2 is the best. The marginal difference in outcome metric in between method 2 and method 3 conform to the rankings similitude according to τ measure .However, due to the minimal outcome metric gap between the method 2 and method 3, together with the favorable result method 3 has scored in terms of # losses we would like to run the rankings for more impressions to decide the winner with high degree of confidence.

Query	**query terms**	**Field Name**	**M-1**	**M-2**	**M-3**
baba	baba	product_name	0.1989	0.2509	0.2713
		contents	0.2468	0.2592	0.3509
		category	0.152	0.3723	0.3024
		short_desc	0.1054	0.0658	0.0356
		description	0.1384	0.0516	0.0391
pötyi	pötyi	product_name	0.1989	0.1067	0.0877
		contents	0.2468	0.882	0.906
		description	0.1384	0.0109	0.006
minnie	minnie	product_name	0.1989	0.086	0.1156
		contents	0.2468	0.086	0.1479
		characters	0.1339	0.8102	0.7214
		short_desc	0.1054	0.0117	0.008
		description	0.1384	0.0072	0.0068
bogyó és babóca	bogyó	product_name	0.1989	0.061	0.0861
		contents	0.2468	0.0633	0.1118
		description	0.1384	0.0034	0.0034
		short_desc	0.1054	0.0062	0.0044
		characters	0.1339	0.8657	0.7941
	és	product_name	0.1989	0.0874	0.1218
		contents	0.2468	0.0984	0.1714
		description	0.1384	0.2578	0.2514
		short_desc	0.1054	0.2316	0.1613
		characters	0.1339	0.3246	0.2939
	babóca	product_name	0.1989	0.0594	0.0823
		contents	0.2468	0.0875	0.1514
		description	0.1384	0.0034	0.0033
		short_desc	0.1054	0.0061	0.0042
		characters	0.1339	0.8434	0.7587

Table 7: Field weight assignment according to methods

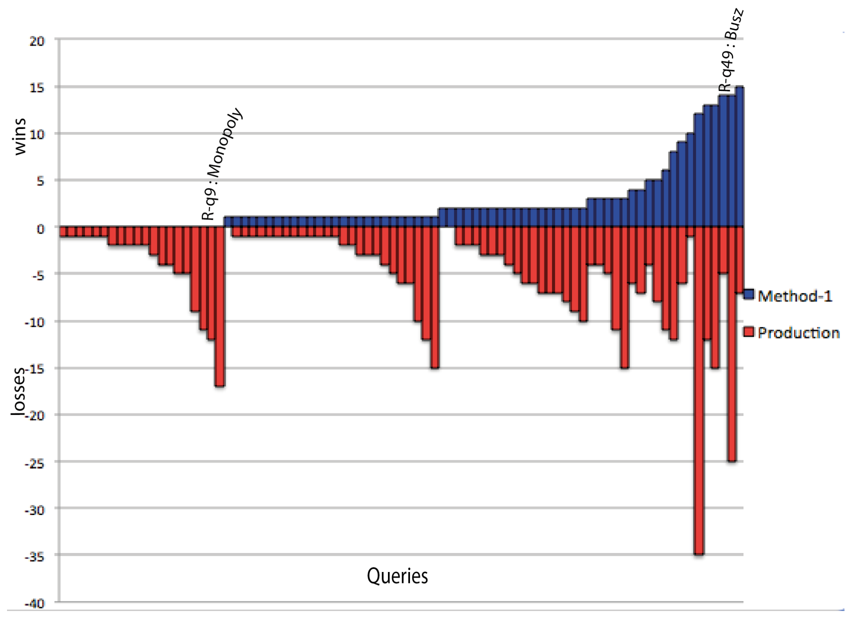


Figure 8: Method-1 : # wins and # losses against production system ordered by # wins

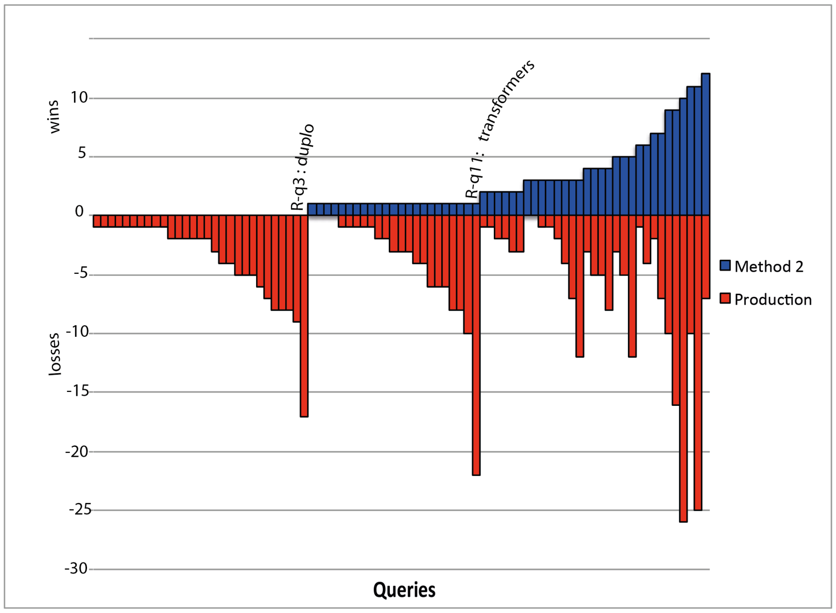


Figure 9: Method-2 : # wins and # losses against production system ordered by # wins

We found that this was in accordance to the hypothesis we made that term-specific mapping in product search (method 2 and method 3) has accomplished significant improvement over static weighting (method 1) which is based solely on field's performance. Method2 assumes equal field prior for the all the fields whereas method3 deployed field effectiveness measures from method1 for field prior probabilities. From this we can infer that estimating field mapping priors based on historical clicks hardly performs better than the setting where the priors are uniformly distributed.

The result of experiments in Living Labs has shown to be inversely related to historical relevance judgements results. This result possibly implies the relevance of Living lab platforms for information retrieval task.

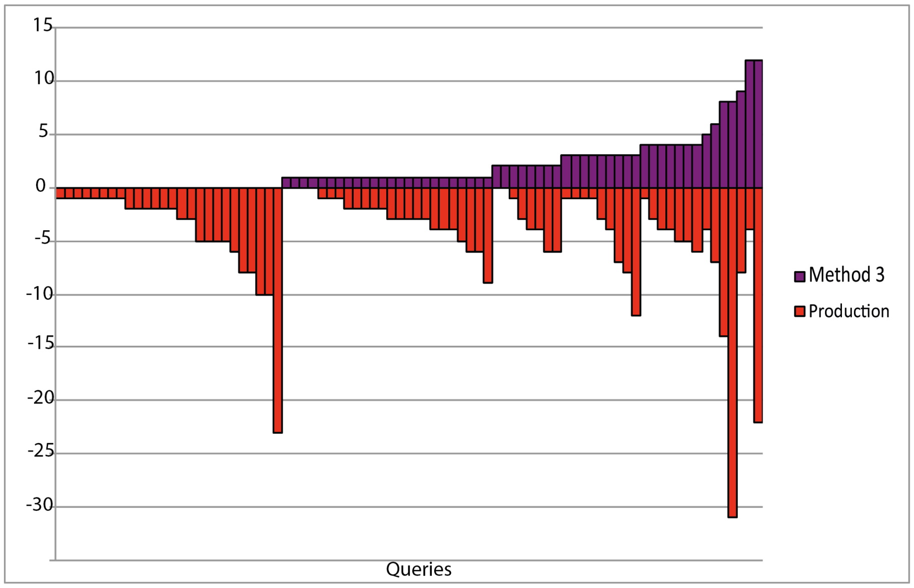


Figure 10: Method-3 : # wins and # losses against production system ordered by # wins

Query Analysis

The following analysis takes a deeper look at queries marked on previous method 1 plot in based on aggregated number of clicks products received during the training phase. For each query two plots are shown : the former portrays the number of clicks of products received in descending order as a result of a query search(unranked product-click plot). The second plot orders the products according to one of the methods' rankings against the number of clicks(ranked product-click plot). Besides some of the plots show a secondary vertical axis indicating the price of products.

Logically a ranked product-click plot that has equivalent shape to the unranked product-click plot would be considered acceptable ranking. For query `"monopoly"` , this is not the case and it has lost 13 times and won only once against production system. If we look closely at the ranked product-click plot, products returned at ranking positions 2-4 have got relatively low number of clicks. One likely reason can be attributed to the price level of the products. As shown in the line graph at the ranked product-click plot, products at top ranks in particular those at position 2 and 3 are very expensive compared to other products ranking at top(almost double to the average price).The result implies that in product search one crucial factor that we must take in consideration is the price of the products. We can infer that including some price related characterstics in order to supplement product data may be beneficial in improving the retrieval accuracy.

Another query `"busz"` has recorded satisfactory result against production system winning 14 and lost only 5 times. Unlike to the ranked product-query plot of query `"monopoly"`, this has roughly similar shape to the unranked product-query plot. The unranked product-query plot has multiple products which likely are relevant and the ranking has achieved more click for the top ranked products.

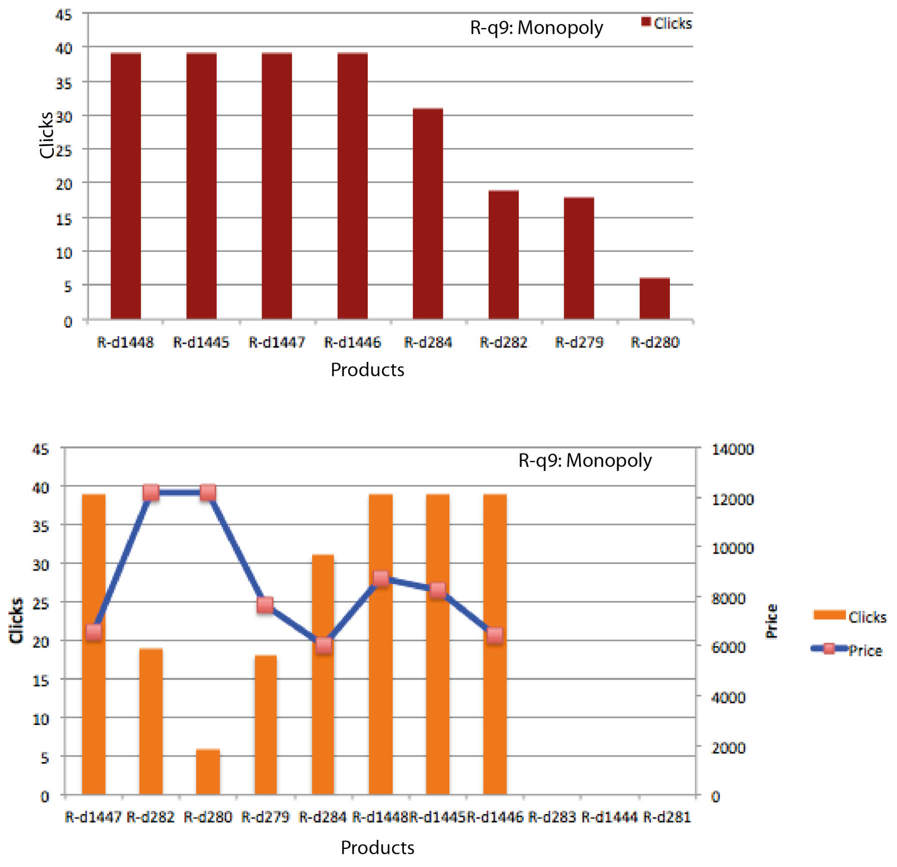
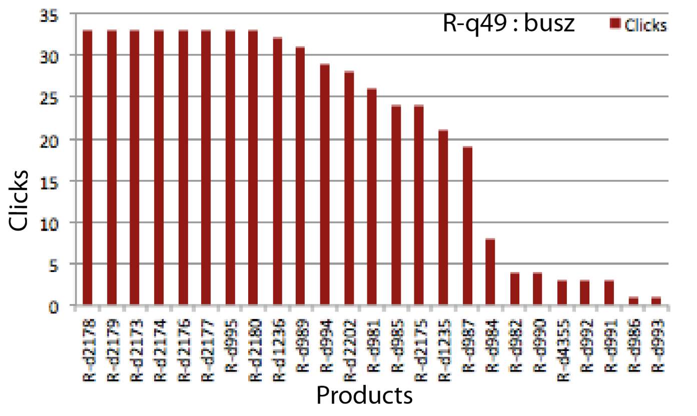


Figure 11: click distribution for query Monopoly



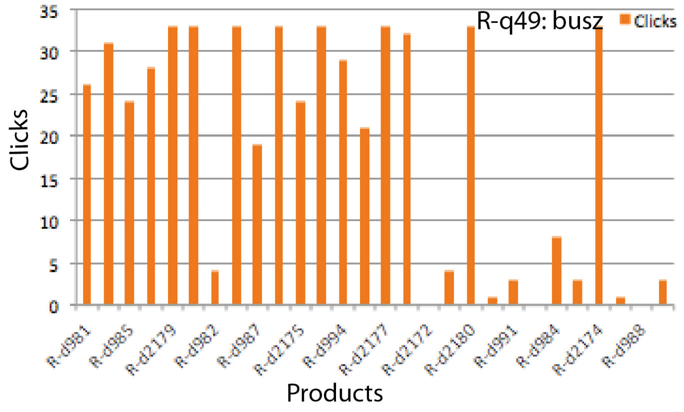


Figure 12: click distribution for query Busz

8 Conclusion

In this thesis we studied, adapted and evaluated well-established retrieval models and analyzed how they fit to product search task. We applied generative mixture language models approach based on fielded document models. Principally, we proposed functions to estimate query terms-field mapping probability component. We estimated field weights depending on exclusive field efficiency determined by making use of evaluation metric nDCG computed from historical CTR relevance judgement. Altenative methods estimated the mapping probability based on query term abundance in field Collection. The retrieval methods were evaluated both in traditional Trec-style employing historical CTR relevance judgement and Living Lab platform where the rankings were presented to real customers to an e-commerce site `'regiojatek.hu'`. The comparison among the methods was carried out by taking production system as a reference.

We discovered , according to Living Labs evaluation metric 'outcome', methods that are based on term-field mapping were superior to the static field performance weighting approach. The evidence we presented shows that term specific mapping has a positive effect on retrieval performance in product search task.

We also found out that the accuracy measure of methods performed according to historical CTR has given different results from Living labs results evaluation. This supports the notion of the need for Living lab platform in information retrieval tasks. To make more concrete conclusion about the significance of Living labs platform though, it would have been interesting to compare evaluation results based on expert relevance judgements ,which we don't have access, from the e-commerce site against result from Living labs instead of relevance judgements built of historical CTR which only can barely provide hints.

The price of products in the e-commerce site was dynamic ,thus we believe factors such as product's price level and price adjustments e.g. in form of sales has an influence on which products got clicked during our online experiment. In future it would be appealing to study those factors and devise retrieval models that manipulate such data and supplement it in the ranking of the product in order to attain high user utility.

Another question that could be interesting to answer in the future is :If product retrieval accuracy can be boosted by incorporating product popularity information learned out of sale records.

References

[1] K. Balog, L. Kelly, and A. Schuth, “Head first: Living labs for ad-hoc search evaluation,” in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, ser. CIKM ’14. ACM, 2014, pp. 1815–1818.

[2] H. Duan, C. Zhai, J. Cheng, and A. Gattani, “Supporting keyword search in product database: A probabilistic approach,” *Proc. VLDB Endow.*, pp. 1786–1797, 2013.

[3] J. Kim, X. Xue, and W. B. Croft, “A probabilistic retrieval model for semistructured data,” in *Proceedings of the 31th European Conference on IR Research on Advances in Information Retrieval*, ser. ECIR ’09. Springer-Verlag, 2009, pp. 228–239.

[4] K. Balog, “Semistructured data search,” in *Bridging Between Information Retrieval and Databases*, ser. Lecture Notes in Computer Science, N. Ferro, Ed., 2014, pp. 74–96.

[5] E. M. Voorhees and D. K. Harman, *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[6] C. Cleverdon, “Readings in information retrieval.” Morgan Kaufmann Publishers Inc., 1997, ch. The Cranfield Tests on Index Language Devices, pp. 47–59.

[7] J. Allan, B. Carterette, J. A. Aslam, V. Pavlu, B. Dachev, and E. Kanoulas, “Million query track 2007 overview,” DTIC Document, Tech. Rep., 2007.

[8] E. M. Voorhees and H. Donna, “Overview of the seventh text retrieval conference(trec-7),” in *Proceedings of the Seventh Text REtrieval Conference(TREC-7)*, vol. 500, no. 242, 1999, pp. 1–23.

[9] R. A. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., 1999.

[10] S. Dumais, E. Cutrell, J. Cadiz, G. Jancke, R. Sarin, and D. C. Robbins, "Stuff i've seen: A system for personal information retrieval and re-use," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval.* ACM, 2003, pp. 72–79.

[11] W. S. Cooper, "On selecting a measure of retrieval effectiveness," *Journal of the American Society for Information Science*, vol. 24, no. 2, pp. 87–100, 1973.

[12] B. Carterette, "System effectiveness, user models, and user utility: A conceptual framework for investigation," in *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '11, 2011, pp. 903–912.

[13] R. Kohavi, R. Longbotham, D. Sommerfield, and R. M. Henne, "Controlled experiments on the web: Survey and practical guide," *Data Min. Knowl. Discov.*, pp. 140–181, 2009.

[14] P. Bailey, A. P. De Vries, N. Craswell, and I. Soboroff, "Overview of the trec 2007 enterprise track." in *TREC.* Citeseer, 2007.

[15] L. Azzopardi and K. Balog, "Towards a living lab for information retrieval research and development: A proposal for a living lab for product search tasks," in *Proceedings of the Second International Conference on Multilingual and Multimodal Information Access Evaluation*, ser. CLEF'11. Springer-Verlag, 2011, pp. 26–37.

[16] T. Joachims *et al.*, "Evaluating retrieval performance using clickthrough data," 2003.

[17] F. Radlinski, M. Kurup, and T. Joachims, "How does clickthrough data reflect retrieval quality?" in *Proceedings of the 17th ACM Conference on Information and Knowledge Management.* ACM, 2008, pp. 43–52.

[18] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Trans. Inf. Syst.*, vol. 20, pp. 422–446, 2002.

[19] P. B. Kantor and E. M. Voorhees, "The trec-5 confusion track: Comparing retrieval methods for scanned text," *Inf. Retr.*, pp. 165–176, 2000.

[20] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval.* New York, NY, USA: Cambridge University Press, 2008.

[21] C. Zhai, "Statistical language models for information retrieval a critical review," *Found. Trends Inf. Retr.*, pp. 137–213, 2008.

[22] D. L. Lee, H. Chuang, and K. Seamons, "Document ranking and the vector-space model," *IEEE Softw.*, vol. 14, no. 2, pp. 67–75, 1997.

[23] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Found. Trends Inf. Retr.*, pp. 333–389, 2009.

[24] J. M. Ponte and W. B. Croft, "A language modeling approach to information retrieval," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 1998, pp. 275–281.

[25] D. R. H. Miller, T. Leek, and R. M. Schwartz, "A hidden markov model information retrieval system," in *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 1999, pp. 214–221.

[26] D. Hiemstra, "A linguistically motivated probabilistic model of information retrieval," in *Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries.* Springer-Verlag, 1998, pp. 569–584.

[27] C. Zhai and J. Lafferty, "A study of smoothing methods for language models applied to ad hoc information retrieval," in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 2001, pp. 334–342.

[28] P. Ogilvie and J. Callan, "Combining document representations for known-item search," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval.* ACM, 2003, pp. 143–150.

[29] B. Carterette, P. N. Bennett, D. M. Chickering, and S. T. Dumais, "Here or there: Preference judgments for relevance," in *Proceedings of ECIR 2008.* Springer, 2008.

[30] S. Robertson, "On the history of evaluation in ir," *J. Inf. Sci.*, vol. 34, pp. 439–456, 2008.

[31] D. Kelly, S. Dumais, and J. O. Pedersen, "Evaluation challenges and directions for information-seeking support systems," *Computer*, vol. 42, pp. 60–66, 2009.

[32] R. White, I. Ruthven, and J. M. Jose, "The use of implicit evidence for relevance feedback in web retrieval," in *Proceedings of the 24th BCS-IRSG European Colloquium on IR Research: Advances in Information Retrieval*, 2002, pp. 93–109.

[33] Y. Liu, Y. Fu, M. Zhang, S. Ma, and L. Ru, "Automatic search engine performance evaluation with click-through data analysis," in *Proceedings of the 16th International Conference on World Wide Web*, ser. WWW '07, 2007, pp. 1133–1134.

[34] S. Fox, K. Karnawat, M. Mydland, S. Dumais, and T. White, "Evaluating implicit measures to improve web search," *ACM Trans. Inf. Syst.*, pp. 147–168, 2005.

Appendix

Attachments

- Source Code : Retrieval package and Client code for Living labs API