RULK: A Framework for Representing User Knowledge in Search-as-Learning

Arthur Câmara[1,*,†], Dima El-Zein[2,*,†] and Célia da-Costa-Pereira[2]

[1] *Delft University of Technology, Delft, The Netherlands*

[2] *Université Côte d'Azur, Labortatoire I3S, CNRS, UMR 7271, France*

Abstract

When learning something new, users generally resort to search systems. By issuing queries and examining search results, users acquire their knowledge from the content of result pages - or documents - until they satisfy their information needs on a given subject. The search system, in return, should be able to *support* the user during this session by retrieving documents that are, at the same time, relevant to the user query and suitable according to what the user already knows about a topic. Especially for the latter, having a method that can accurately *estimate* user knowledge is crucial. To tackle that, we propose RULK, a framework for *representing* the user's knowledge throughout their search sessions. The intuition behind RULK is simple. By keeping an internal representation of the user's knowledge that gets updated as the user progresses on their search session, the framework estimates how much the user knows (or still does not know) about a given topic. We implement two variations of RULK, one based on keywords and one using large language models, and show that their estimations of user knowledge are correlated with actual user knowledge, as measured on a real learning search system. Therefore, RULK clears the path to future learning-focused search systems to provide an even better experience for users.

Keywords

Search-As-Learning, Interactive IR, Retrieval system, User Knowledge

1. Introduction and Related Work

In recent years, it has become increasingly common for users to use search systems, such as Web search engines, as learning tools. More often than not, a user with a *learning goal* (e.g., Learning about ethics) resorts to such systems for finding documents that will help them satisfy their learning goals [1, 2]. By submitting queries, evaluating returned rankings, and reading documents, users interactively explore documents retrieved by the system, acquiring knowledge until they become satisfied with their knowledge of that subject. This interactive process of search and knowledge acquisition is also known as *Search-As-Learning* (SAL) [3].

DESIRES 2022 – 3rd International Conference on Design of Experimental Search & Information REtrieval Systems, 30-31 August 2022, San Jose, CA, USA

[*] Corresponding author.

[†] These authors contributed equally.

A.BarbosaCamara@tudelft.nl (A. Câmara); elzein@i3s.unice.fr (D. El-Zein); Celia.DA-COSTA-PEREIRA@univ-cotedazur.fr (C. da-Costa-Pereira)

https://www.abcamara.com (A. Câmara); https://www.i3s.unice.fr/~elzein/ (D. El-Zein); https://www.i3s.unice.fr/~cpereira/ (C. da-Costa-Pereira)

0000-0003-1296-5531 (A. Câmara); 0000-0003-4156-1237 (D. El-Zein); 0000-0001-6278-7740 (C. da-Costa-Pereira)

CEUR Workshop Proceedings (CEUR-WS.org)

Many recent information retrieval systems consider these learning-oriented goals when retrieving documents [4, 5, 6, 7]. One of the main influences of these work comes from the concept of the "Anomalous State of Knowledge" introduced by Belkin et al. [8]. According to this concept, people have an internal knowledge model of the world. That model is constantly evolving and changing as they acquire more knowledge. Suppose they perceive an anomaly in that model (e.g., missing or contradictory information) during this process. In that case, they are compelled to seek information, perhaps using a search system, to solve such anomaly.

However, most current search systems are optimised to retrieve documents relevant to a single query rather than considering a more comprehensive view of the user session and knowledge [9]. Nevertheless, a user with a learning goal will likely need multiple rounds of interactions with the system, leading to considerably longer sessions. Moreover, as their session progresses, the users' cognitive state changes, impacting their behaviour [10, 11, 12].

Therefore, **representing** the users' cognitive (or knowledge) state and **updating** it during a search session is gaining attention from researchers. By estimating a user's knowledge gain, a search system can help a user reach their learning goals faster by, for instance, retrieving documents relevant not only to a single query but also to their current knowledge on the topic [13, 14].

In order to help future systems to solve that need, we propose a novel framework for **R**epresenting **U**ser **L**earning and **K**nowledge (RULK) to tackle this problem. We combine ideas already present in some SAL systems to represent and update a user's state [15, 16, 17, 18] and add a novel piece to these systems: An **Estimator**, capable of, given the user's state, predicting their current knowledge on the topic.

Usually, existing systems that implement some knowledge representation have (at least) two components: A **Feature Extractor** transforms clicked documents into features, and an **Updater** updates an internal representation of the users' knowledge state. These components are typically implemented by either extracting weighted keywords from documents [15, 16, 7] or embedding the documents into some semantic space [13, 18].

In RULK, we add a third component, the **Estimator**, that *estimates* the users' knowledge gains in their session (i.e., how much they learned). It does so by comparing the knowledge state maintained by the Updater to a "target" knowledge (e.g., a list of keyphrases relevant to a topic). While predicting users' knowledge is not new, our framework proposes not only to predict but also to track users' knowledge state throughout their sessions. While not considered here, we do encourage future instantiations to expand on RULK by including previous works on user knowledge prediction, like Yu et al. [19] and Liu et al. [20], that used users' behaviour features, and Yu et al. [21] that also included web features.

To demonstrate how one could implement RULK in SAL systems, we implement two variations of it: RULK_{KW}, using keywords representations inspired by El Zein and da Costa Pereira [15, 16], and RULK_{LM}, using embeddings generated by a large language model, inspired by Câmara et al. [18]. Using logs from real-world users' interaction with a learning-focused search system and associated user learning scores, we study how each implementation behaves, especially when estimating user knowledge. Therefore, we aim to answer the following research questions:

RQ1 Are the estimations produced by RULK correlated to actual users' knowledge?

RQ2 How do RULK_{KW} and RULK_{LM} behave as users' sessions grow longer?

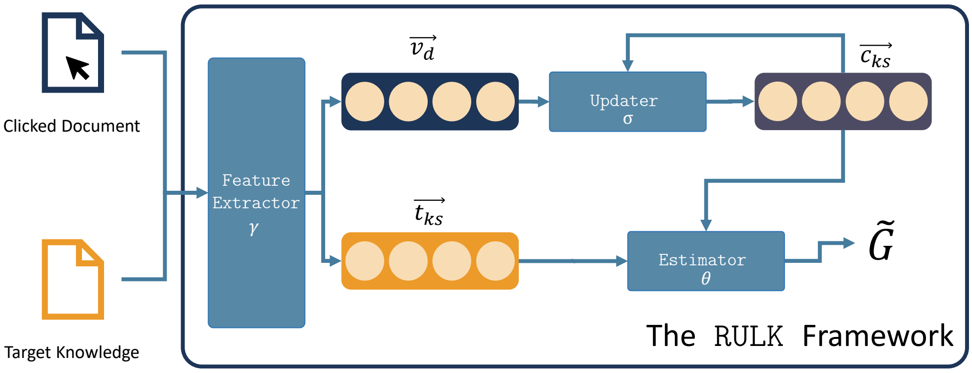

Figure 1: The RULK framework and its main components. First, a clicked document d is transformed into $\vec{v_d}$ by $\boldsymbol{\gamma}$. Next, $\boldsymbol{\sigma}$ updates the current state $\vec{c_{ks}}$ with $\vec{v_d}$. Finally, $\boldsymbol{\theta}$ compares $\vec{c_{ks}}$ to a target knowledge vector $\vec{t_{ks}}$ to get an estimation of the user's knowledge gain in the session ($\widetilde{G}$).

By answering these questions, we show that RULK can estimate a user's knowledge gain, with RULK_{KW} demonstrating a higher (6%) correlation with assessed learning gains. We also show how RULK_{KW} can better handle longer sessions, implying a more robust and stable option.

2. The RULK Framework

2.1. RULK Components

Our framework is composed of three main components: the **Feature Extractor** ($\boldsymbol{\gamma}$), the **Updater** ($\boldsymbol{\sigma}$) and the **Estimator** ($\boldsymbol{\theta}$). These components interact with each other in multiple situations. For instance, when estimating the user's current knowledge, $\boldsymbol{\theta}$ uses the user's state provided by $\boldsymbol{\sigma}$, which, in its turn, uses the documents' features provided by $\boldsymbol{\gamma}$ to keep the user's state.

Another way they interact is when estimating the user's knowledge gain. $\boldsymbol{\theta}$ requires reference knowledge to estimate how much the user's knowledge of a topic T has evolved throughout their session. This reference should represent the knowledge state at which we would consider a learning task as "achieved". We refer to this reference knowledge target as the "target knowledge state" ($\vec{t_{ks}}$), which is also generated by $\boldsymbol{\gamma}$ from a reference document that ideally matches the knowledge level the user wants to reach.

In this section, we set out each of the components that make up the RULK framework and leave to Section 2.2 the details of two possible instantiations of RULK, RULK_{KW} and RULK_{LM}, and how they implement each component. We also show an overview of RULK in Figure 1.

Feature Extractor ($\boldsymbol{\gamma}$) During users' interactions with search systems, the *content* of the pages they read is the main contributor to their knowledge and learning gains. These pages can be any content the user interacts with during their search session (e.g., Web pages, textbooks, videos, or courses). Without loss of generality, here we focus on the textual content of the pages read by the user, referring to it as a *document d*. Then, given d, a Feature Extractor $\boldsymbol{\gamma}$ encodes it into $\vec{v_d}$, a vector with fixed size m:

$$\vec{v_d} = \gamma(d). \tag{1}$$

By fixing the size m of the vectors and encoding all documents in the same embedding space (e.g., BERT or TF-IDF), $\boldsymbol{\gamma}$ allows RULK to compare and combine documents easily.

As mentioned before, $\boldsymbol{\gamma}$ also plays a role when modelling the "target knowledge state" ($\vec{t_{ks}}$). However, selecting what the target knowledge is is not trivial. Therefore, for simplicity and discussion, we assume that our SAL system has access to a "reference document" that covers essential subtopics and themes for T (e.g., a textbook on T) used as the target knowledge. We can then generate $\vec{t_{ks}}$ by encoding this document according to Equation 1.

Updater (σ) RULK tracks the user's knowledge through an internal state represented by a vector $\vec{c_{ks}}$ having the same length m as the $\vec{v_d}$ embeddings produced by $\boldsymbol{\gamma}$. Following El Zein and da Costa Pereira [15], $\boldsymbol{\sigma}$ updates $\vec{c_{ks}}$ after the user reads a document, assuming that they were able to absorb the content of that document:

$$\vec{c'}_{ks} = \sigma(\vec{c_{ks}}, \vec{v_d}), \tag{2}$$

where $\vec{c'}_{ks}$ is the updated $\vec{c_{ks}}$ vector—or updated state of the user's knowledge—after reading a document d. The document is represented by $\vec{v_d}$, generated by Equation 1; and $\boldsymbol{\sigma}$ is a function that takes $\vec{c_{ks}}$ and $\vec{v_d}$ and combines them into an updated representation of the users' knowledge.

Estimator (θ) RULK then *estimates* the users' knowledge gain on the topic during the session by comparing the user's current knowledge state, $\vec{c_{ks}}$, to the target $\vec{t_{ks}}$:

$$\widetilde{G} = \theta(\vec{c_{ks}}, \vec{t_{ks}}), \tag{3}$$

Where $\widetilde{G}$ is an estimation of the user's knowledge gain in the session and $\boldsymbol{\theta}$ is implemented as a similarity function (e.g., cosine similarity). The intuition behind $\boldsymbol{\theta}$ is that the user, by progressing in their session, "moves" their knowledge state ($\vec{c_{ks}}$) towards the target ($\vec{t_{ks}}$). As both vectors are in the same embedding space, we interpret the similarity between $\vec{c_{ks}}$ and $\vec{t_{ks}}$ as an estimate of how close the user is to acquiring the knowledge contained in $\vec{t_{ks}}$. [1]

2.2. Implementing RULK

We show two possible implementations for deploying RULK in a search system. The first, called RULK_{KW}, relies on keyword-based feature extraction, while the second, RULK_{LM}, uses Large Language Models, like BERT. The main difference between the two implementations is in what semantic space they use when representing documents and the user's knowledge. Importantly, both share the same Estimator ($\boldsymbol{\theta}$), implemented as a cosine similarity between $\vec{t_{ks}}$ and $\vec{c_{ks}}$:

$$\widetilde{G} \approx \frac{\vec{c_{ks}} \cdot \vec{t_{ks}}}{|\vec{t_{ks}}|\,|\vec{c_{ks}}|}. \tag{4}$$

[1] Note that $\vec{t_{ks}}$ is a reference of the knowledge a user *can* acquire based on the target document chosen. While some users may not be interested in reaching all of the knowledge from the target, others may be interested in going beyond the knowledge contained in the target document.

RULK$_{\text{KW}}$: We adopt the user's learning model named *vocabulary learning* [22], which takes place at the lower level of Bloom's taxonomy [23]. This model considers that a user achieves their need to learn a topic T when they learn a set of related vocabulary keywords.

For a topic T, we define the target keyword set $K_T = \{tk_1, \ldots, tk_m\}$. The keywords are extracted from a reference document. Once the keywords to be learned are defined, the number of occurrences of every keyword the user has to read then must also be decided. To define the target knowledge state, $\boldsymbol{\gamma}$ embeds the reference document as an occurrence vector. The target knowledge state is then represented as $\vec{t_{ks}} = \{tko_1, \ldots, tko_m\}$, where tk_i is the number of occurrences of the keyword that the user must read before the framework considers the user to have learned it.

Similarly, a document d is embedded by $\boldsymbol{\gamma}$ by counting the occurrences of the keywords K_T in it, resulting in $\vec{v_d} = \{dko_1, \ldots, dko_m\}$, where dko_i is the number of occurrences of tk_i in d.

As for the Updater $\boldsymbol{\sigma}$, in line with [6, 16], we consider that the user's knowledge increases monotonically. That is, as they read documents d represented by $\vec{v_d}$, $\boldsymbol{\sigma}$ adds the count of the keywords K_T present in the document to the user's knowledge representation $\vec{c_{ks}}$.

RULK$_{\text{LM}}$ Many recent works have shown that Large Language Models (LLM) perform extraordinarily well in capturing the semantic meaning of texts [24]. Furthermore, in the Information Retrieval domain, transformers-based language models [25], mainly based on BERT [26], have been shown to excel in multiple tasks [27], even if not fine-tuned in that specific domain [28, 29].

Given their success, we assess whether such models can act as a Feature Extractor ($\boldsymbol{\gamma}$) for RULK. Thus, we implement a BERT-based variant of the framework, called RULK$_{\text{LM}}$, inspired by the method proposed by Câmara et al. [13] to track user exploration of a topic.

Both the target knowledge $\vec{t_{ks}}$ and clicked document's embedding $\vec{v_d}$ are represented by an embedding of fixed length m, as generated by the same language model. Given a document d (or, conversely, a reference document) with k sentences $\{s_1, s_2 \ldots s_k\}$, $\boldsymbol{\gamma}$ generates, for each sentence s_i, an embedding of size m given by:

$$\vec{v_{s_i}} = BERT([CLS]; s_{i\,:\,l}; [SEP]), \tag{5}$$

where ; is a concatenation, l the maximum input size of the model and $[CLS]$ and $[SEP]$ are special BERT tokens. $\vec{v_d}$ (conversely, $\vec{t_{ks}}$) is then given by an element-wise sum over all $\vec{v_{s_i}}$.

The Updater $\boldsymbol{\sigma}$ is then a simple element-wise sum over all elements of $\vec{v_d}$ and $\vec{c_{ks}}$. As $\vec{t_{ks}}$ and $\vec{c_{ks}}$ are vectors in the same embedding space, $\boldsymbol{\theta}$, similarly to RULK$_{\text{KW}}$, is also the cosine similarity between $\vec{t_{ks}}$ and $\vec{c_{ks}}$, as shown in Equation 4.[2]

3. Validating RULK

In this section, we discuss how we validated RULK by describing our dataset, metrics and implementation details for our two instantiations of RULK, RULK$_{\text{KW}}$ and RULK$_{\text{LM}}$ [3].

[2] We also experimented with other options for $\boldsymbol{\sigma}$ (e.g. averaging the embeddings instead of summing) and $\boldsymbol{\theta}$ (e.g. Using euclidean distance on a normalised vectors), and all options yielded very similar results.

[3] Our implementations can be found at https://github.com/ArthurCamara/RULK_SAL

	Total	Mean	Median
Number of users per topic	126	18.14 ± 2.79	19.0
Number of topics	7	-	-
Number of queries	1095	8.62 ± 6.47	7.0
Number of documents clicked	2116	16.66 ± 8.85	16.0
Number of snippets seen	15184	119.56 ± 72.43	105.0
Documents Clicked per query	-	2.78 ± 2.50	2.11
Session duration (minutes)	-	56.18 ± 14.58	54.05
Document dwell time (seconds)	-	79.94 ± 69.77	60.0
Pre-test scores (vks^{pre})	-	1.07 ± 1.60	0.00
Post-test scores (vks^{post})	-	6.21 ± 4.09	6.00
Actual Learning Gain (ALG)	-	0.53 ± 0.38	0.50
Realised Potential Learning (RPL)	-	0.28 ± 0.20	0.25

Table 1
Statistics, per user, extracted from the dataset used by Câmara et al. [13].

Using our proposed approaches, we apply the RULK framework to estimate the user's knowledge gain, $\widetilde{G}$. We analyse the logs from real users from a previous study to represent each user's current knowledge state $\vec{c_{ks}}$ as it is updated along their search session. We initiate $\vec{c_{ks}}$as a vector of zeros for all users. (c.f. Section 5 for a discussion on our assumptions and caveats).

Dataset To analyse how our implementations of RULK behave, especially when estimating users' knowledge in a search session, we test RULK_{KW} and RULK_{LM} on a publicly available dataset of SAL sessions built with the logs from the study by Câmara et al. [13] [4]. The authors collected this dataset with a search system implemented on top of SearchX [30], a framework for Interactive Information Retrieval research. We also show some statistics about the dataset in Table 1.

The dataset contains the interaction logs of 126 crowd-workers. The authors asked workers to search about a given topic for at least 45 minutes to collect it. The logged interactions contain behavioural features, issued queries, and clicked documents. At the start of the study, the system measured the previous knowledge of each participant on two randomly selected topics and selected the topic the user demonstrated a lower knowledge of as their target topic T. Finally, at the end of the search session, the user's knowledge was measured again, allowing us to calculate their actual learning knowledge during the session.

As the dataset does not contain the textual contents of the 1107 unique clicked documents, we used the Wayback Machine API [5] to fetch the documents when the study was conducted (August 2020). Of all the documents, 33 did not have a snapshot available and were discarded from our experiments (i.e., we do not consider their impact on user's knowledge).

Actual Knowledge Gain Measurement The dataset contains self-reported users' knowledge before and after the search session, vks^{pre} and vks^{post} respectively. The authors measured

[4] The data is available at https://github.com/ArthurCamara/CHIIR21-SAL-Scaffolding
[5] https://web.archive.org/

these values with a *Vocabulary Knowledge Scale (VKS)* test [31, 32], a commonly used method to measure user knowledge [33, 34, 35, 7]. For that, the researchers presented the users with a 4-point scale questionnaire, asking about their familiarity with ten keywords selected by the authors from the Wikipedia article related to the topic.

We can measure a user's learning during their session by computing the difference between vks^{pre} and vks^{post}. We follow the authors of the original study by using the *Realised Potential Learning (RPL)* as the primary metric to measure user knowledge gain. It is defined as follows:

$$\begin{aligned} ALG &= \frac{1}{10}\sum_{i=1}^{10} max(0, vks^{post}(v_i) - vks^{pre}(v_i)) \\ MLG &= \frac{1}{10}\sum_{i=1}^{10} 2 - vks^{pre}(v_i) \\ RPL &= \frac{ALG}{MLG} \end{aligned} \tag{6}$$

where ALG is the *Absolute Learning Gain* of a user, MLG the *Maximum Learning Gain* (i.e., the maximum amount of *new* knowledge a user can acquire, given what they already know), and $vks(v_i)$ the score of the user for the i-th term.

The score of a given term t ($vks(v_t)$) ranges from 0 to 2. It is measured by asking the user to rate their familiarity with t on a 4-point scale. In this scale, selecting values 1 or 2 mean that the user does not know the term ($vks(v_i) = 0$). Finally, selecting value 3 means they partially know it ($vks(v_i) = 1$) and, by selecting 4, the user indicates that they fully understand what the term means ($vks(v_i) = 2$). Therefore, RPL represents the fraction of knowledge the user acquired from the total knowledge they could obtain in their session. Here we use RPL and *reported knowledge gain* interchangeably.

Consider a user unfamiliar with all terms from the topic that, after their session, indicates that they partially learned all terms (i.e., $vks^{pre} = 0$ and $vks^{post} = 1$ for all terms). In this case, their Absolute Learning Gain ALG (i.e. their average added knowledge per term) is 1, while their Maximum Learning Gain MLG is 2 (i.e. they could raise their knowledge by 2 points in all terms). Therefore, their Realised Potential Learning RPL is given by $RPL = \frac{ALG}{MLG} = 0.5$.

Target knowledge The topics used in the original user study came from the list of topics used in the CAR track from TREC 2018 [36]. In that track, each topic is the title of a Wikipedia article from a 2018 dump. Therefore, both of our implementations use these Wikipedia texts, from the same 2018 dump as the original paper, as "reference documents" for generating $\vec{t_{ks}}$. Furthermore, we use the same dump as in the original paper. It's also worth mentioning that, in the user study which originated this dataset, the authors filtered Wikipedia and clones of Wikipedia from the search results during the user study. Therefore, no document proposed to the user in the page results of the experiment came from Wikipedia or a similar page.

RULK$_{KW}$ implementation We implement the RULK$_{KW}$'s model by extracting the keywords of $\vec{t_{ks}}$ using the *Yet Another Keyword Extraction (YAKE)* method [37] on the Wikipedia texts. This method is a lightweight unsupervised automatic keyword extraction method that relies on statistical features extracted from documents to select the most important keywords of a text.

We set the maximum n-gram size at 3; we noticed, however, that all the keyphrases extracted by *YAKE*, for all topics, were 1-gram. We also choose a value of $m = 10$ as the size of $\vec{t_{ks}}$, $\vec{c_{ks}}$, and $\vec{v_d}$; hence the top 10 keywords in the Wikipedia pages are considered for the vectors. To avoid keywords that are too similar, we stem the keywords after being extracted by *YAKE* (using the *Porter Stemmer* of the *NLTK* Python library [38]). As this can lead to duplicate keywords (e.g., water and waters have the same stem, "water"), we remove duplicated stems and add the next more relevant keyword until ten keywords per topic are left. The same stemming process was also performed on the clicked documents when generating their embeddings $\vec{v_d}$.

RULK_{LM} implementation We implement RULK_{LM}'s $\boldsymbol{\gamma}$ as a BERT-based Language Model. Specifically, we use a MiniLM [39] model with 6 layers and a hidden layer's dimension of 384. The model was also fine-tuned on the MsMarco dataset [40], as made available in the SBERT framework [29] [6]. The embeddings $\vec{v_d}$ and $\vec{t_{ks}}$ have a fixed length of $m = 384$. We split the documents into sentences using the *NLTK*'s implementation of the *Punkt Sentence Tokenizer*, feed each sentence individually into the $\boldsymbol{\gamma}$ and sum their respective embeddings [7].

Mixing RULK_{KW} and RULK_{LM} RULK can be easily extended. To show that, and that our proposed implementations are not the only ones possible, but rather examples, we propose to *combine* both methods. The intuition is simple. While RULK_{KW} may be useful for capturing some characteristics of the read documents, RULK_{LM} can be useful for other types of characteristics. Therefore, we implement an interpolated estimator $\boldsymbol{\theta}$, parameterised by α, defined as:

$$\theta_{\text{RULK}_{\text{LM+KW}}} = (\alpha)\widetilde{G}_{\text{RULK}_{\text{LM}}} + (1-\alpha)\widetilde{G}_{\text{RULK}_{\text{KW}}}, \tag{7}$$

where $\widetilde{G}_{\text{RULK}}$ is the estimated knowledge gain of the respective RULK implementation. Experimentally, we found that $\alpha = 0.38$ yields the better results.

4. Results

Recall that we proposed two research questions, measuring different aspects of RULK:

RQ1: Are the estimations produced by RULK correlated to actual users' knowledge? To answer our first research question, we test the validity of the RULK by computing the Pearson's correlation between the *actual* knowledge gains of a user and the *estimated* knowledge gain $\widetilde{G}$, as measured by each implementation's $\boldsymbol{\theta}$. As our main goal in this paper is to propose RULK, this section is mainly descriptive of observed results rather than an in-depth analysis. Therefore, we leave it to future work the *why* of these results and *how* to improve them.

As shown in Table 2, on the set of all users, RULK_{KW} has a considerable correlation with both RPL and ALG. Additionally, as shown in the last three columns, the estimated knowledge gain generated by all our implementations are highly correlated among themselves. This implies

[6] https://huggingface.co/sentence-transformers/msmarco-MiniLM-L6-cos-v5

[7] Other common approaches, such as truncating the document, averaging the sentences, and only using the sentence most similar to the user's query (MaxP [41]), but they all resulted to worst or similar performance

Method	ALG	RPL	RULK$_{\text{KW}}$	RULK$_{\text{LM}}$	RULK$_{\text{KW+LM}}$
RULK$_{\text{KW}}$	0.3022	0.3086	-	0.7831	0.9662
RULK$_{\text{LM}}$	0.2955	0.2923	0.7831	-	0.9169
RULK$_{\text{KW+LM}}$	**0.3164**	**0.3192**	0.9662	0.9169	-

Table 2
Pearson's correlation between a given implementation of the framework and reported user's learning and other implementations. **bold** values indicate the best correlation against a learning metric.

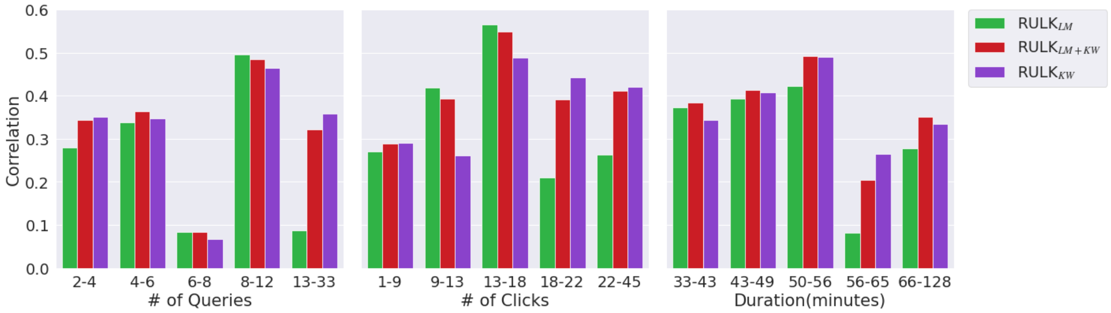

Figure 2: Pearson's correlations between estimated and measured (RPL) knowledge gains by quartile.

that, regardless of implementation, RULK is a viable option for representing user knowledge. It is also interesting to note that this repeats for both ALG and RPL. Finally, as a testament to the flexibility of RULK, and reinforcing the idea that LM and KW models capture different characteristics of texts, RULK$_{\text{KW+LM}}$ shows an even higher correlation to actual learning gains when compared to either implementation in isolation.

RQ2: How does RULK$_{\text{KW}}$ and RULK$_{\text{LM}}$ behave as users' sessions grow longer? To assess the second research question, we split users into quartiles based on the length of their sessions as given by three measurements: Number of queries issued, number of documents clicked, and session duration in minutes. We show the results of this analysis in Figure 2.

We can better understand when each implementation fails by measuring how RULK$_{\text{KW}}$ and RULK$_{\text{LM}}$ behave for different users. From Figure 2, we can see that RULK$_{\text{KW}}$ and RULK$_{\text{LM}}$ perform differently for different types of users. The most interesting result, however, is that the knowledge estimations of RULK$_{\text{LM}}$ for users with the highest number of interactions are generally considerably worst in the language-model implementation. For instance, considering the number of queries, RULK$_{\text{KW}}$ outperforms RULK$_{\text{LM}}$ by 75% in the highest split. We also observe this advantage for RULK$_{\text{KW}}$ when considering the number of documents clicked (37% of difference) and session duration (70% in the second-to-last split and 17% in the last). It indicates that, as the sessions grow longer and the users click on more documents, RULK$_{\text{KW}}$ can better handle the increasing magnitude of $\vec{c_{ks}}$. However, the gap diminished greatly by employing our proposed RULK$_{\text{KW+LM}}$ method. It shows that, while one approach is better in some scenarios, an approach that considers multiple features leads to a more robust implementation of RULK.

5. Conclusions and Future Work

This paper introduced RULK, a framework for **R**epresenting **U**ser **L**earning and **K**nowledge, containing three main components: The Feature Extractor $\boldsymbol{\gamma}$ generates embeddings from the documents clicked by users. The Updater $\boldsymbol{\sigma}$ maintains a vector representing the user's knowledge, and the Estimator $\boldsymbol{\theta}$ estimates how much knowledge the user gained during their search session. We hope that RULK can pave the way for applications that benefit from this information, like ranking functions that can better find documents according to the user's current knowledge.

This work used simplified assumptions regarding the user's knowledge acquisition and learning progress. We want to acknowledge these so that future work can use these as a starting point. First, we assumed that users can assimilate *all* of the knowledge encoded in $\vec{v_d}$ and that no forgetting takes place. However, factors like the user's learning rate, familiarity with the subject, or the time spent reading the content can impact how well the user absorbs knowledge from documents. Future work can incorporate these factors into RULK by changing Equation 2 to consider these factors. Another simplifying assumption we made is to consider that the user has no previous knowledge about the topic (i.e. we initialise $\vec{c_{ks}}$ as a vector of 0s), although it has been observed that users have at least some knowledge about the topic they are searching for [16, 13, 34, 42].

To demonstrate how other works can implement RULK, we implemented two variants ourselves: RULK_{KW} and RULK_{LM}, each of them relying on a different method for knowledge representation. While the former uses extracted keywords, the latter uses a transformers-based language model to generate semantic representations of texts. Through our experiments, we show that both implementations can, to a certain degree, estimate the actual user knowledge gains, with RULK_{KW} leading to a slightly higher correlation with self-reported knowledge gains. We hope that our framework can be helpful for researchers aiming to incorporate users' knowledge in search systems, primarily when focusing on learning (Search-as-Learning) and that our work can spark discussion on how to estimate and track user knowledge.

References

[1] N. Selwyn, An investigation of differences in undergraduates' academic use of the internet, Active learning in higher education 9 (2008) 11–22.

[2] J. Biddix, C. Chung, H. Park, Convenience or credibility? a study of college student online research behaviors, The Internet & Higher Education 14 (2011) 175–182.

[3] K. Collins-Thompson, P. Hansen, C. Hauff, Search as learning, Dagstuhl Reports 7 (2017) 135–162.

[4] K. Byström, K. Järvelin, Task complexity affects information seeking and use, Inf. Process. Manag. 31 (1995) 191–213.

[5] L. Harbarth, S. Delsing, F. Richtscheid, V. Yücepur, F. Feldmann, M. Akhavanfarm, S. Manske, J. Othlinghaus, H. U. Hoppe, Learning by tagging - supporting constructive learning in video-based environments, in: DeLFI, volume P-284 of *Lni*, Gesellschaft für Informatik e.V., 2018, pp. 105–116.

[6] R. Syed, K. Collins-Thompson, Retrieval algorithms optimized for human learning, in:

Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval, 2017, pp. 555–564.

[7] R. Syed, K. Collins-Thompson, Optimizing search results for human learning goals, Information Retrieval Journal 20 (2017) 506–523.

[8] N. J. Belkin, R. N. Oddy, H. M. Brooks, Ask for information retrieval: Part i. background and theory, J. Documentation 38 (1982) 61–71.

[9] A. Hassan Awadallah, R. White, P. Pantel, S. Dumais, Y.-M. Wang, Supporting complex search tasks, in: Proc. 23rd ACM CIKM, 2014, pp. 829–838.

[10] N. J. Belkin, The cognitive viewpoint in information science, Journal of information science 16 (1990) 11–15.

[11] M. J. Bates, The design of browsing and berrypicking techniques for the online search interface, Online review (1989).

[12] P. Ingwersen, Information retrieval interaction, volume 246, Taylor Graham London, 1992.

[13] A. Câmara, N. Roy, D. Maxwell, C. Hauff, Searching to learn with instructional scaffolding, Proceedings of the 2021 Conference on Human Information Interaction and Retrieval (2021).

[14] N. Roy, A. Câmara, D. Maxwell, C. Hauff, Incorporating widget positioning in interaction models of search behaviour, Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval (2021).

[15] D. El Zein, C. da Costa Pereira, A cognitive agent framework in information retrieval: Using user beliefs to customize results, in: International Conference on Principles and Practice of Multi-Agent Systems, Springer, 2020, pp. 325–333.

[16] D. El Zein, C. da Costa Pereira, User's knowledge and information needs in information retrieval evaluation, in: Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization, 2022, pp. 325–333.

[17] R. Syed, K. Collins-Thompson, Exploring document retrieval features associated with improved short-and long-term vocabulary learning outcomes, in: Proceedings of the 2018 conference on human information interaction & retrieval, 2018, pp. 191–200.

[18] A. Câmara, D. Maxwell, C. Hauff, Searching, learning, and subtopic ordering: A simulation-based analysis, in: Ecir, 2022, pp. 142–156.

[19] R. Yu, U. Gadiraju, P. Holtz, M. Rokicki, P. Kemkes, S. Dietze, Predicting user knowledge gain in informational search sessions, The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (2018).

[20] J. Liu, C. Liu, N. J. Belkin, Predicting information searchers' topic knowledge at different search stages, J. Assoc. Inf. Sci. Technol. 67 (2016) 2652–2666.

[21] R. Yu, R. Tang, M. Rokicki, U. Gadiraju, S. Dietze, Topic-independent modeling of user knowledge in informational search sessions, Inf. Retr. J. 24 (2021) 240–268.

[22] P. Bailey, L. Jiang, User task understanding: a web search engine perspective, 2012. URL: https://www.microsoft.com/en-us/research/publication/user-task-understanding-a-web-search-engine-perspective/, presentation delivered at the NII Shonan: Whole-Session Evaluation of Interactive Information Retrieval Systems workshop. 8-11 October 2012, Shonan, Japan.

[23] B. S. Bloom, Taxonomy of educational objectives: The classification of educational goals, Cognitive domain (1956).

[24] A. Tamkin, M. Brundage, J. Clark, D. Ganguli, Understanding the capabilities, limitations, and societal impact of large language models, CoRR abs/2102.02503 (2021).

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: Nips, 2017, pp. 5998–6008.

[26] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, ArXiv abs/1810.04805 (2019).

[27] J. Lin, R. Nogueira, A. Yates, Pretrained transformers for text ranking: BERT and beyond, CoRR abs/2010.06467 (2020).

[28] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, I. Gurevych, BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models, in: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2), 2021.

[29] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Emnlp/ijcnlp (1), Association for Computational Linguistics, 2019, pp. 3980–3990.

[30] S. R. Putra, F. Moraes, C. Hauff, Searchx: Empowering collaborative search research, in: Sigir, Acm, 2018, pp. 1265–1268.

[31] M. B. Wesche, T. S. Paribakht, Assessing second language vocabulary knowledge: Depth versus breadth., Canadian Modern Language Review-revue Canadienne Des Langues Vivantes 53 (1996) 13–40.

[32] K. A. D. Stahl, M. A. Bravo, Contemporary classroom / vocabulary assessment / for content areas, The Reading Teacher 63 (2010) 566–578.

[33] S. Salimzadeh, U. Gadiraju, C. Hauff, A. van Deursen, Exploring the feasibility of crowd-powered decomposition of complex user questions in text-to-sql tasks, in: Ht, Acm, 2022, pp. 154–165.

[34] N. Roy, F. Moraes, C. Hauff, Exploring users' learning gains within search sessions, Proceedings of the 2020 Conference on Human Information Interaction and Retrieval (2020).

[35] H. L. O'Brien, A. Kampen, A. W. Cole, K. Brennan, The role of domain knowledge in search as learning, in: Chiir, Acm, 2020, pp. 313–317.

[36] L. Dietz, B. Gamari, J. Dalton, N. Craswell, TREC complex answer retrieval overview, in: Trec, volume 500-331 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 2018.

[37] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, A. Jatowt, Yake! collection-independent automatic keyword extractor, in: European Conference on Information Retrieval, Springer, 2018, pp. 806–810.

[38] S. Bird, E. Klein, E. Loper, Natural language processing with Python: analyzing text with the natural language toolkit, " O'Reilly Media, Inc.", 2009.

[39] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, M. Zhou, Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, in: Advances in Neural Information Processing Systems, volume 33, 2020, pp. 5776–5788.

[40] N. Craswell, B. Mitra, E. Yilmaz, D. Campos, J. Lin, MS MARCO: benchmarking ranking models in the large-data regime, in: Sigir, Acm, 2021, pp. 1566–1576.

[41] Z. A. Yilmaz, S. Wang, W. Yang, H. Zhang, J. Lin, Applying BERT to document retrieval with birch, in: Emnlp/ijcnlp (3), Association for Computational Linguistics, 2019, pp.

19–24.

[42] Gadiraju, R. Yu, S. Dietze, P. Holtz, Analyzing knowledge gain of users in informational search sessions on the web, Proceedings of the 2018 Conference on Human Information Interaction & Retrieval (2018).