ABSTRACT
Simulated user–retrieval system interactions enable studies with controlled user behavior. To this end, the SimIIR framework offers static, rule-based methods. We present an extended SimIIR 2.0 version with new components for dynamic user type-specific Markov model-based interactions and more realistic query generation. A flexible modularization ensures that the SimIIR 2.0 framework can serve as a platform to implement, combine, and run the growing number of proposed search behavior and query simulation ideas.

CCS CONCEPTS
• Information systems → Users and interactive retrieval; • Computing methodologies → Modeling and simulation.

KEYWORDS
Simulation, Search Behavior, User Modeling, Software Framework

ACM Reference Format:

1 INTRODUCTION
Behavior analyses are key to understand how searchers interact with a retrieval system and to assess whether changes to the interface or the retrieval model help to improve the user experience. Still, traditional academic retrieval evaluation often follows the Cranfield paradigm [9] with static test collections (documents, queries, relevance judgments) to ensure a controlled and reusable setup. But the Cranfield paradigm does not really cover dynamic interactions (e.g., query reformulation [7]) or the evaluation of user interface variants (e.g., with or without facets and filters). Such more realistic evaluations today are mostly conducted via large-scale A/B tests [28] or via smaller controlled user studies [11, 24]. However, user studies are costly and hard to reproduce, while A/B testing requires a large enough user base to draw meaningful conclusions.

For evaluation scenarios with a smaller number of users (e.g., in digital libraries), simulation offers an alternative beyond the Cranfield paradigm or controlled user studies. In a way, simulation also allows to ”A/B-test” different back end configurations or interface variants by monitoring interactions of parameterized user types. Clearly, results from such artificial A/B-tests strongly depend on the realism and representativeness of the simulated behavior.

The open-source SimIIR framework [31] supports repeatable simulated retrieval experiments with static interaction modules for user behavior within the Complex Searcher Model. In this paper, we present an extended SimIIR 2.0 framework with dynamic and user type-specific simulation components. We include improved query formulation approaches and Markov modeling for global and search-type specific behavior. From the simulated interactions, various metrics can be computed that indicate how well a system assists users in completing their tasks. When comparing simulations from the existing framework to our extended version, one can observe that the new dynamic components support more diverse developments of users’ information needs during sessions.

Our extended SimIIR 2.0 framework is updated to the latest Python version, comes with additional dynamic user and query modules, and—like the original SimIIR framework—is available as an open-source resource with a permissive license that allows others to easily contribute further components, modules, or adjustments. ¹

SimIIR 2.0 can thus serve as a modern platform to implement, configure, combine, run, and compare the growing number of user models and query simulation ideas from the literature.

¹GitHub: https://github.com/padre-lab-eu/simiir-2
2 RELATED WORK

Evaluation has always been a core IR topic; Harman [19], Kelly [24], and Sanderson [36] nicely cover the history. Today, the Cranfield paradigm from the 1960s [9] is still often used even though this usually means to evaluate systems with static queries and no user interactions. Some studies thus employed simulation but mostly for single aspects like click behavior [8], query (re-)formulation [2, 6, 23, 42], relevance feedback [20, 25], or stopping [32, 41].

Later, the importance of simulating the search process as a whole has been emphasized [31, 48] even though Cole [10] had collected several challenges when developing realistic simulations against “real” retrieval systems [10]. Cole’s challenges are based on Boring’s five step operationalist approach [3] and essentially state that simulations need to be aligned with real user behavior.

The SimIIR framework [31] provides tools to simulate user-system interactions as a whole (queries, clicks, stopping, etc.) for different configurations of simulation components, experimental conditions, and retrieval systems. However, the simulation components originally implemented in SimIIR produce rather static behavior sequences. We thus extend the SimIIR framework by including more dynamic components and allowing the simulation modules to influence each other (e.g., to take the interaction history into account for the next simulation steps).

To align simulations with real user behavior, SimIIR 2.0 contains components to train Markov models on real log data to simulate global or user type-specific behavior. Markov models are based on a well-established theory, are rather simple and compact, and have been used for simulations in various areas: first-order or higher-order Markov models [37], partially observable Markov decision processes (POMDPs) [40], and hidden Markov models (HMMs) [12]. For now, we rely on first-order Markov models in SimIIR 2.0 due to their simplicity but other variants can later be included.

3 THE EXISTING SIMIIR FRAMEWORK

SimIIR [31] is a Python-based framework for simulating search sessions following the Complex Searcher Model (CSM). The CSM has components for the decision points and activities in search sessions (cf. Figure 1; from formulating a query on a topic over examining some documents to stopping the search).

To run a SimIIR simulation, the following four main elements must be configured. (1) Topics represent the simulated users’ information needs and consist of a title and a description. In SimIIR, the standard topics come from TREC tracks (e.g., the TREC 2005 Robust track). (2) A retrieval system that returns a ranked list of documents with snippets for a query. In SimIIR, Whoosh is used as the standard retrieval system. (3) An output controller that generates output files for a simulation run compatible with evaluation programs like trec_eval. Finally, a simulation requires (4) a series of simulated users, each possibly with differently configured but still rather static characteristics for the decision points and activities in the CSM. During the simulation, the users attempt to complete a session on a given topic while interacting with the retrieval system.

4 SIMIIR 2.0 EXTENSION

After describing our conceptual CSM extensions, we give details on the newly added query generation and Markov model components.

4.1 Extended Complex Searcher Model

We add two novel elements to the CSM to improve the realism of the simulated sessions: advanced query generation and user type-specific Markov model-based stopping (green blocks in Figure 1).

Query generation. In the original CSM [31], a pool of queries is generated once at the start of a simulated session. From this static pool, a query is selected whenever the simulated user decides to submit a new query. However, in real sessions, the seen results will often influence subsequent queries (e.g., a user may acquire new vocabulary from a read document) [18, 21, 38, 43]. In the extended CSM, we thus enable the query generation to access the session history and to dynamically generate new query candidates based on this information. When a ‘dynamic’ simulated user wants to submit a new query, it is selected from an updated pool of candidates.

User types. The original CSM does not include a possibility to group different simulated users as kind of a user type with possibly specific search behavior. For instance, ‘exploratory searchers’ will explore a search result list more exhaustively than ‘lookup
searchers’ who will only investigate the first few results and then rephrase their queries rather quickly [1, 45]. We thus include user types in the extended CSM in the sense that the components of the CSM can be initialized with user type-specific characteristics to support the simulation of user type-specific sessions.

Markov models. In our extended CSM, the stopping decisions on the SERP and session level are made by user type-specific Markov models instead of the original stochastic heuristics with stopping threshold variables. To this end, we categorize users into different types and simulate their search process using specific Markov models. At the stopping decisions, these models also predict a user’s next likely step by taking the session history into account.

Besides stopping, we also employ Markov model-based decisions for query generation to let later queries in a session depend on the content of previously viewed search results. User type-specific Markov models predict the next likely query ‘change’ direction (e.g., generalization or specialization) based on how a particular user group reformulates their queries and the predicted direction is then used to select an appropriate query from the (updated) pool.

4.2 Realization and Implementation

Query generation. We add several types of query generation approaches. The first type uses an actual search engine’s API to obtain query suggestions—a technique earlier demonstrated to yield realistic sessions [16]. Our second type extends the original SimIIR query generation approaches—that determine query terms from the static topic information—by additionally giving them access to the session history in form of examined snippets and documents as a resource for new query terms. The third type of approaches implements Markov model-based query change prediction (e.g., did a generalization or specialization happen before) as this was demonstrated to reliably simulate specific querying behavior [44]. The model then guides the selection of a next query from the (possibly dynamic) query pool based on a user’s previous changes.

User types. In order to simulate user type-specific behavior, we categorize users into different groups based on their search and stopping behavior. So far, we use the previously introduced contextual search types [1, 46] of ‘exploratory searchers’ who tend to fully explore a search result list and extensively use potentially available results, and ‘lookup searchers’ who only investigate the first few results and quickly rephrase their queries. But also other user types like ‘fast and liberal’ vs. ‘slow and picky’ users [27, 39] or ‘build up’ vs. ‘boil down’ behavior [35] could be the basis.

Markov models. For using a Markov model in a simulation, one simply specifies files with the respective states and transition probabilities. SimIIR 2.0 offers the possibility to derive the probabilities from some existing search logs but they can also be set manually.

Extended configuration. Listing 1 shows an example of how the configuration attributes for SimIIR 2.0 have been extended. The file extends the SimIIR tree_user4 with a new section behaviorModeler for the user type-specific behavior. The user_type attribute indicates to use the exploratory Markov model with the states and transition_matrix.

Listing 1: Configuration file markov_google_trec_user.xml with the new options behaviorModeler and GoogleSuggestGenerator.

```xml
<userConfiguration id="markovgoogletrecuser">
  <behaviorModeler class="Markov">
    <attribute name="user_type" value="exploratory"/>
    <attribute name="states" value="<..>/states.data"/>
    <attribute name="transition_matrix" value="<..>/transition.data"/>
  </behaviorModeler>
  <queryGenerator class="GoogleSuggestGenerator">
    <attribute name="stopword_file" value="<..>/stopwords.txt"/>
    <attribute name="max_depth" type="integer" value="5"/>
  </queryGenerator>
  <relevanceAssessor class="TrecAssessor"/>
  <snippetAssessor class="TrecAssessor"/>
  <documentAssessor class="TrecAssessor"/>
  <stoppingDecisionMaker class="FixedDepthDecisionMaker">
    <attribute name="depth" value="10"/>
  </stoppingDecisionMaker>
  <costCalculator class="FixedCostCalculator">
    <attribute name="query_cost" value="10"/>
    <attribute name="document_cost" value="28"/>
    <attribute name="snippet_cost" value="3"/>
    <attribute name="serp_results_cost" value="5"/>
    <attribute name="mark_document_cost" value="3"/>
  </costCalculator>
  <searchContext class="SearchContext">
    <attribute name="relevance_revision" value="1"/>
  </searchContext>
  <serpImpression class="SimpleSerpImpression">
    <attribute name="qrel_file" value="<..>/trec2005.qrels.all"/>
  </serpImpression>
</userConfiguration>
```

The value of user_type can also be set to none to resemble original SimIIR configurations without the behavior section.

5 SIMIIR 2.0 IN ACTION

The transition probabilities of our new Markov model-based components can be instantiated manually or be derived from search logs. For our experiments, we use the Sowiport User Search Session dataset (SUSS) [33] that includes 558,008 sessions with about 8 million interactions (179.7% of them queries) collected from April 2014 to April 2015 from users of the Sowiport digital library search system. Within the sessions, 58 different actions were logged while users interacted with the system (e.g., formulating a query or clicking on a document). Following Zerhoudi et al. [45], we split the SUSS data into exploratory and lookup subsets to train respective individual models. Of course, also any other search log or splitting strategy could be used to train the Markov models.

The simulation process in the original SimIIR framework is triggered by a single XML file.3 It defines the output options, topics (i.e., titles and descriptions available to the query generation strategies), the search interface, and the configuration files of the simulated users. Each individual simulated user can mimic an individual participant of a user study with a specific static query generation strategy, document/snippet relevance assessment method, and stopping criterion. Examples are fixed depth users (stopping at a certain

---

4https://github.com/leifos/simiir/blob/master/example_sims/users/trec_user.xml

Figure 2: Excerpt of simulated sessions for the topic extinction wildlife generated by (left) the standard SimIIR 2.0 user, (middle) an exploratory SimIIR 2.0 user, and (right) a lookup SimIIR 2.0 user. A session includes the actions of the simulated user (e.g., QUERY, SERP, SNIPPET), the session’s time limit (600 seconds), the cumulated elapsed time (e.g., 10, 15, 18 seconds), and an action’s metadata (e.g., the query string in green or the relevance assessment for some snippet or document ID).

The simulation process in SimIIR 2.0 allows for more complex experimental settings. Simulated users are defined by an elaborated search behavior like the user type-specific Markov models for exploratory and lookup users. These models can determine the stopping behavior instead of the original threshold-based strategies.6 In the new SimIIR 2.0 setup, Markov model-based decisions can also be combined with the stopping strategies of the original SimIIR framework. For instance, while predicting the next actions of a simulated exploratory user using the respective Markov model, the search result examination can be stopped when the gained knowledge drops below a user’s average gain rate.

Figure 2 (left) shows an excerpt of a session generated by a basic simulation configuration of the original SimIIR framework.6 Given the topic extinction wildlife and its description, the simulated fixed-depth smart user starts by submitting the topic title as the first query, examines some snippets and a document before submitting a second query, inspecting further snippets, etc. The simulated session in Figure 2 (middle) is generated by a new exploratory user,7 while the session in Figure 2 (right) comes from a new lookup user.8 Both use the Google suggest API to select a next query from the up to ten suggestions. Just like in the example, we observed that simulated exploratory users tend to more exhaustively explore the search result list and reformulate the query as they learn more about the topic while lookup users only investigate the first few results and quickly rephrase their queries.

6 CONCLUSION
We have presented SimIIR 2.0: an extended and updated version of the SimIIR search behavior simulation framework. Since the rather static components of the original framework do not take session history into account, we add this ability to the components for query formulation and stopping decisions—also including Markov modeling abilities to reflect different dynamic user types.

In future work, we plan to enable more influence between the different components of the extended Complex Searcher Model for more realistic simulated sessions. We also experiment with other non-Markov models for interaction simulation [15] and plan to include respective components in upcoming SimIIR 2.0 versions. Furthermore, so far, only single sessions are simulated but also cross-session search [29] or search missions [17, 22], as well as cross-device search [14] could be interesting simulation targets.

With the SimIIR 2.0 framework open-sourced under a permissive license, others can also easily contribute further simulation components. SimIIR 2.0 thus can become a platform for accessible and reproducible retrieval simulation. Ideally, it can directly support or otherwise quickly include components for new simulation ideas in “classic” search box-based but also in conversational scenarios; for instance, the various simulation-based studies published recently at ECIR 2022 [4, 5, 30, 34] or at SIGIR 2022 [13, 26, 47].

ACKNOWLEDGMENTS
This work has been partially supported by the DFG (German Research Foundation) through the project 408022002 “SINIR – Simulating INteractive Information Retrieval”.

REFERENCES


