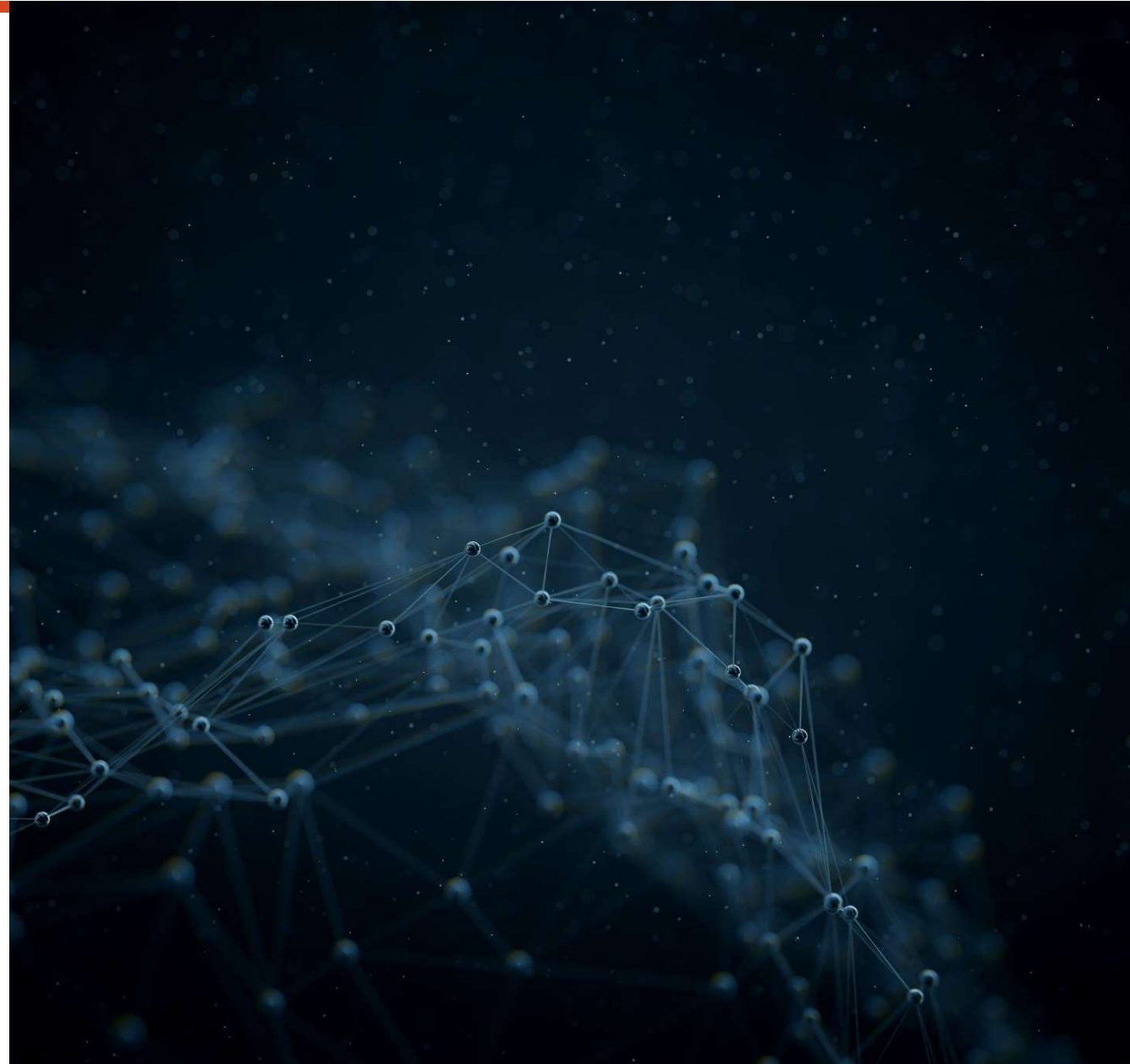


# Key SimIIR Functions: A Hands-On Introduction

Von: Andreas Kruff & Philipp Schaer

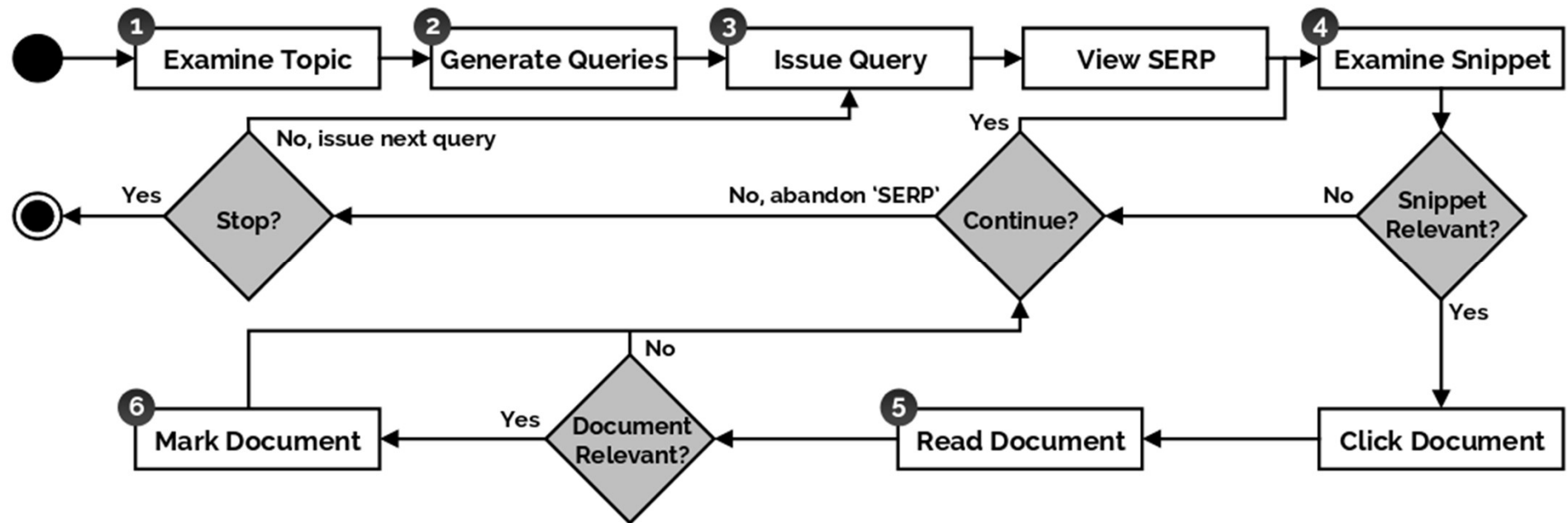
25-04-03 – Cologne, Germany  
<https://ir.web.th-koeln.de>

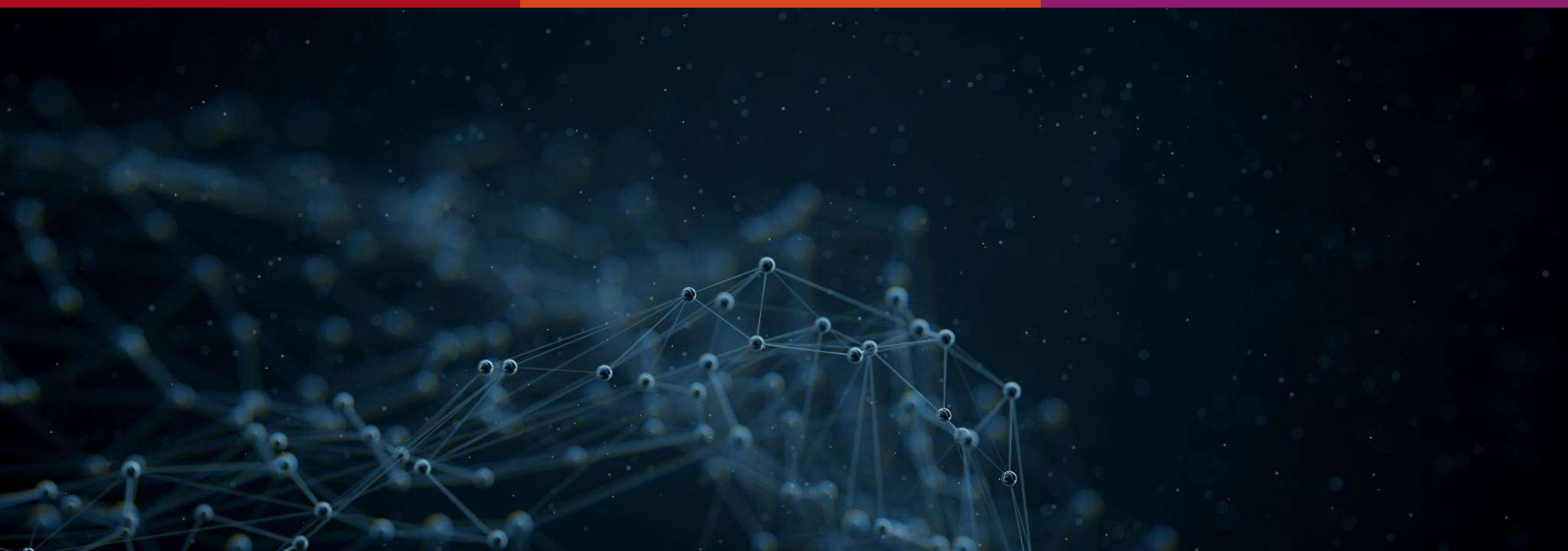


# SimIIR Framework: Konfiguration

- **Modulares Design:** Das Framework bietet ein modulares Design, das eine flexible Konfiguration des User Agents ermöglicht.
- **Benutzerdefinierte Interaktionen:** Jede Interaktion kann durch bereits implementierte Ansätze individuell konfiguriert werden.
- **Vorkonfigurierte Beispiele:** Vorhandene Konfigurationen sind als XML-Dateien unter ``example_sims/users/*.xml`` verfügbar und bieten eine gute Grundlage für Anpassungen.

# Structure of the SimIIR Framework





# Konfiguration eines User Agents

# Konfiguration eines User Agents

- **XML-Format:** Der Benutzeragent wird in einem XML-Format definiert, das eine strukturierte und leicht verständliche Konfiguration ermöglicht.
- **Root-Element:** Das Root-Element ``userConfiguration`` dient als übergeordnetes Element für alle benutzerspezifischen Einstellungen.
- **Benutzer-ID:** Innerhalb der ``userConfiguration`` wird ein eindeutiger Benutzer-ID (``userID``) festgelegt, um den Benutzeragenten zu identifizieren.
- **Benutzertyp:** Der ``User type`` wird ebenfalls im ``userConfiguration``-Element spezifiziert und definiert die Task des Nutzers (Interactive Retrieval vs. Conversational Search)

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
</userConfiguration>
```

# Konfiguration eines User Agents

- **XML-Format:** Der Benutzeragent wird in einem XML-Format definiert, das eine strukturierte und leicht verständliche Konfiguration ermöglicht.
- **Root-Element:** Das Root-Element ``userConfiguration`` dient als übergeordnetes Element für alle benutzerspezifischen Einstellungen.
- **Benutzer-ID:** Innerhalb der ``userConfiguration`` wird ein eindeutiger Benutzer-ID (``userID``) festgelegt, um den Benutzeragenten zu identifizieren.
- **Benutzertyp:** Der ``User type`` wird ebenfalls im ``userConfiguration``-Element spezifiziert und definiert die Task des Nutzers (Adhoc vs. Conversational Search)

Defines the approach

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
</userConfiguration>
```

# Konfiguration eines User Agents

- **XML-Format:** Der Benutzeragent wird in einem XML-Format definiert, das eine strukturierte und leicht verständliche Konfiguration ermöglicht.
- **Root-Element:** Das Root-Element ``userConfiguration`` dient als übergeordnetes Element für alle benutzerspezifischen Einstellungen.
- **Benutzer-ID:** Innerhalb der ``userConfiguration`` wird ein eindeutiger Benutzer-ID (``userID``) festgelegt, um den Benutzeragenten zu identifizieren.
- **Benutzertyp:** Der ``User type`` wird ebenfalls im ``userConfiguration``-Element spezifiziert und definiert die Task des Nutzers (Interactive Retrieval vs. Conversational Search)

Defines the approach

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
</userConfiguration>
```

Defines the stopping rule

# Konfiguration eines User Agents

- **XML-Format:** Der Benutzeragent wird in einem XML-Format definiert, das eine strukturierte und leicht verständliche Konfiguration ermöglicht.
- **Root-Element:** Das Root-Element ``userConfiguration`` dient als übergeordnetes Element für alle benutzerspezifischen Einstellungen.
- **Benutzer-ID:** Innerhalb der ``userConfiguration`` wird ein eindeutiger Benutzer-ID (``userID``) festgelegt, um den Benutzeragenten zu identifizieren.
- **Benutzertyp:** Der ``User type`` wird ebenfalls im ``userConfiguration``-Element spezifiziert und definiert die Task des Nutzers (Adhoc vs. Conversational Search)

Defines the approach

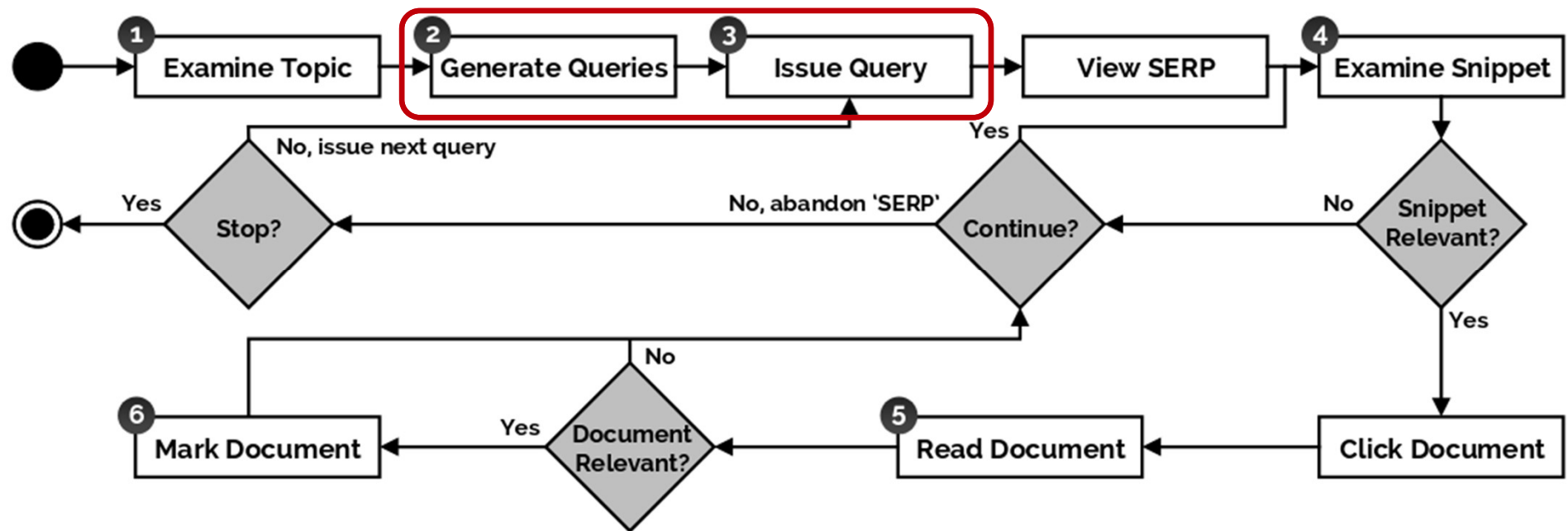
```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
</userConfiguration>
```

Defines the stopping rule

Important for the naming convention of the resulting log files



# Konfiguration der Query (Re)Formulierungsstrategie



**Wir fokussieren uns vorerst auf die Query (Re)Formulierung**

# Konfiguration der Query (Re)Formulierungsstrategie

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
  <queryGenerator class="TriTermQueryGenerator">  
    <attribute name="stopword_file" type="string" value="../example_data/terms/stopwords.txt" is_argument="true" />  
  </queryGenerator>  
</userConfiguration>
```

# Konfiguration der Query (Re)Formulierungsstrategie

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">  
  <queryGenerator class="TriTermQueryGenerator">  
    <attribute name="stopword_file" type="string" value="../example_data/terms/stopwords.txt" is_argument="true" />  
  </queryGenerator>  
</userConfiguration>
```

# Konfiguration der Query (Re)Formulierungsstrategie

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">

  <queryGenerator class="TriTermQueryGenerator">
    <attribute name="stopword_file" type="string" value="../example_data/terms/stopwords.txt" is_argument="true" />
  </queryGenerator>

</\dn6L06u6L9f0L>
```

```
class TriTermQueryGenerator(BaseQueryGenerator):
    """
    Implementing Strategy 3 from Heikki's 2009 paper, generating three-term queries.
    The first two terms are drawn from the topic, with the final and third term selected from the description - in some ranked order.
    """
    def __init__(self, stopwords_file, background_file=[]):
        super(TriTermQueryGenerator, self).__init__(stopwords_file, background_file=background_file)

    def generate_query_list(self, user_context):
        """
        Given a Topic object, produces a list of query terms that could be issued by the simulated agent.
        """
        self.__description_cutoff = 0

        topic = user_context.topic
        topic_title = topic.title
        topic_description = topic.content
        topic_language_model = self._generate_topic_language_model(user_context)


        topic_rankings = self._generate_topic_rankings(topic_language_model)
        topic_rankings = topic_rankings
        topic_rankings = topic_rankings
        topic_rankings = topic_rankings
```

# Konfiguration der Query (Re)Formulierungsstrategie

```
<userConfiguration id="query-rulebased-200td" type="SearchUser">

  <queryGenerator class="TriTermQueryGenerator">
    <attribute name="stopword_file" type="string" value="../example_data/terms/stopwords.txt" is_argument="true" />
  </queryGenerator>


</\dn61\06u619f01>
```



```
class TriTermQueryGenerator(BaseQueryGenerator):
    """
    Implementing Strategy 3 from Heikki's 2009 paper, generating the
    The first two terms are drawn from the topic, with the final and
    """
    def __init__(self, stopwords_file, background_file=[]):
        super(TriTermQueryGenerator, self).__init__(stopwords_file, b

    def generate_query_list(self, user_context):
        """
        Given a Topic object, produces a list of query terms that co
        """
        self.__description_cutoff = 0

        topic = user_context.topic
        topic_title = topic.title
        topic_description = topic.content
        topic_language_model = self._generate_topic_language_model(u
```



```
class BaseQueryGenerator(object):
    """
    The base query generator class.
    Generates 2-word queries from the topic title and description (content)
    ranked by the likelihood of producing that query given the topic.

    You can use this to inherit from to make your own query generator
    """
    def __init__(self, stopwords_file, background_file=None, allow_similar=False):
        self._stopword_file = stopwords_file
        self._background_file = background_file
        self.updating = False
        self.update_method = 1
        self._query_list = None
        self.background_language_model = None
        self._allow_similar = allow_similar

        if self._background_file:
            self.background_language_model = lm_methods.read_in_background(self._background_file)
```

# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with Safety Engineers

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating risks.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant.

**Topic 1**

# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with **Safety** Engineers

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating risks.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant.

**Topic 1**

# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with **Safety Engineers**

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating risks.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant.

**Topic 1**



# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with **Safety Engineers**

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating **risks**.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant. **Topic 1**

# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with **Safety Engineers**

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating **risks**.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant. **Topic 1**



Mögliche erste Query:

**Safety Engineers risks**

# Konfiguration der Query (Re)Formulierungsstrategie

## Funktionsweise des TriTermGenerators:

**Topic Title:** Role of Toxicologists in Collaboration with **Safety Engineers**

**Topic Description:** Documents discussing how toxicologists and safety engineers work together in assessing and mitigating **risks**.

**Topic Narrative:** Relevant documents should describe specific ways toxicologists and safety engineers collaborate, such as in workplace safety, regulatory compliance, or hazard assessments. Documents that discuss only one profession without reference to collaboration are not relevant. **Topic 1**



Mögliche erste Query:

**Safety Engineers risks**

**Welche Probleme könnten sich dabei ergeben?**

# Konfiguration der Query (Re)Formulierungsstrategie

- **Vielfältige regelbasierte Ansätze:** SimiIR enthält eine Vielzahl unterschiedlicher regelbasierter Ansätze zur Simulation von Benutzerinteraktionen.
- **Unterschiedlicher Komplexitätsgrad:** Einige Ansätze sind komplexer gestaltet und bieten eine breitere Palette an konfigurierbaren Parametern.
- **Speicherort:** Diese Ansätze sind unter ``simiir/user/query_generators`` zu finden, wo sie für Anpassungen und den Einsatz in Simulationen bereitstehen.

- query\_generators
  - > \_\_pycache\_\_
  - \_\_init\_\_.py
  - additional\_terms.py
  - base.py
  - basic\_langchain.py
  - bi\_term.py
  - dud\_smart.py
  - google\_suggest\_random.py
  - google\_suggest.py
  - langchain\_query\_expansion\_with\_rationales.py
  - predetermined\_query.py
  - qs34\_query.py
  - refining\_smarter.py
  - single\_reversed\_tri\_interleaved.py
  - single\_reversed\_tri\_reversed\_interleaved.py
  - single\_smarter\_interleaved.py
  - single\_term\_reversed.py
  - single\_term.py
  - single\_tri\_interleaved.py
  - smarter.py
  - trec\_topic\_alltext.py
  - trec\_topic.py
  - tri\_term\_reversed.py
  - tri\_term.py

# Konfiguration der Query (Re)Formulierungsstrategie

## Beispiel für Query Reformulierungen des CORE Log Files:

**"date": "2025-01-30 10:04:27"**, "search\_id": "65b4c65dc01fad71ddcd2b289586652c", "serp": [1176984, 3925832, 21747875, 3558453, 77564682, 84787823, 84344373, 66790505, 135543731, 6435020], **"query": "Endoscopic carpal tunnel release surgery surgical instruments used"**, "uid": "publicb2d130b0ab8bb28f9967f6104b813dc3c648db4dbb391424b4715486818a5cf9"}

**"date": "2025-01-30 10:04:32"**, "search\_id": "c671cfa84dc0b4d8a22a881addf650cf", "serp": [1176984, 3925832, 21747875, 3558453, 77564682, 84787823, 84344373, 66790505, 135543731, 6435020], **"query": "Endoscopic carpal tunnel release surgery surgical instruments used"**, "uid": "publicb2d130b0ab8bb28f9967f6104b813dc3c648db4dbb391424b4715486818a5cf9"}

**"date": "2025-01-30 10:04:32"**, "search\_id": "9bfcbe2dfb8e172b2aff640df7343c2c", "serp": [70580693, 77550406, 84787823, 3925832, 99549973, 78360801, 70581657, 10915550, 3558453], **"query": "Endoscopic carpal tunnel release surgery and common causes of carpal tunnel syndrome"**, "uid": "publicb2d130b0ab8bb28f9967f6104b813dc3c648db4dbb391424b4715486818a5cf9"}

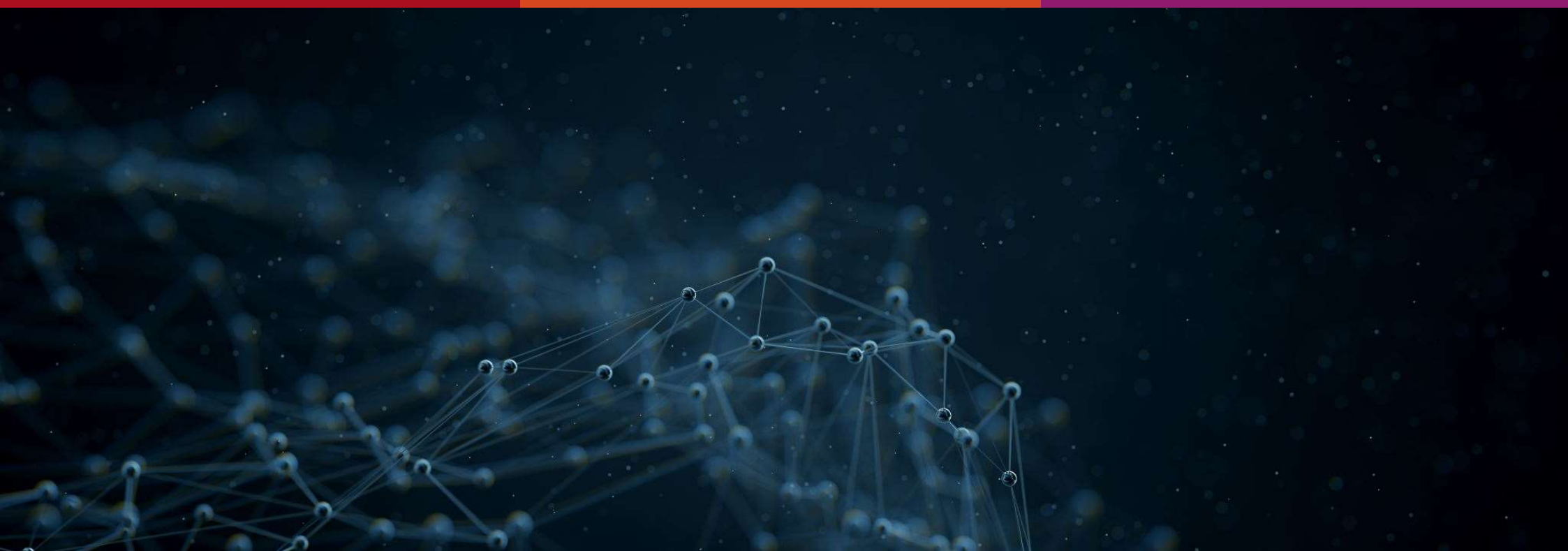
**"date": "2025-01-30 10:04:32"**, "search\_id": "fba84a2d0ab4405556a5045172869e3a", "serp": [1176984, 33611762, 143311744, 84787823, 156594179, 40189776, 26710320, 70580693, 59701085], **"query": "Cost-effectiveness of endoscopic carpal tunnel release surgery"**, "uid": "publicb2d130b0ab8bb28f9967f6104b813dc3c648db4dbb391424b4715486818a5cf9"}

# Zielsetzung

- **Sessions identifizieren:** Identifizierung von Sessions, die verschiedene Query Reformulierungen enthalten.
- **Query Reformulation Generator konfigurieren:** Konfigurierung einer Query Reformulierungsstrategie, die die User aus den Logs so gut wie möglich zu simulieren.
- **Ergebnisse evaluieren:** Evaluierung des Erfolgs der Query Reformulierungsstrategie







# Konfiguration der Simulationsexperimente

# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDirectory="../example_sims/output/"
    saveInteractionLog="true"
    saveRelevanceJudgments="true"
    trec_eval="false" />

  <topics>
    <!--
      These are the topics that each simulated user will complete
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>

  <users>

    <user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
    <user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

    <!--
      Add your newly created user configurations below. By adding:
    -->
    <user configurationFile="path/to/your/user_configuration.xml" />

  </users>

  <searchInterface class="PyTerrierSearchInterface">
    <!--
      This is the search interface that the simulated users will interact with
    -->
    <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
    <attribute name="text_field" type="string" value="text" is_argument="true" />
    <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
    <attribute name="memory" type="boolean" value="true" is_argument="false" />
  </searchInterface>

</simulationConfiguration>
```



# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDir=
    saveInte
    saveRela
    trec_eva

  <simulationConfiguration id="core-bm25">

  <topics>
    <!--
      These are the topics that each simulated user will complete
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>

  <users>

    <user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
    <user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

    <!--
      Add your newly created user configurations below. By adding:
    -->
    <user configurationFile="path/to/your/user_configuration.xml" />
  </users>

  <searchInterface class="PyTerrierSearchInterface">
    <!--
      This is the search interface that the simulated users will interact with
    -->
    <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
    <attribute name="text_field" type="string" value="text" is_argument="true" />
    <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
    <attribute name="memory" type="boolean" value="true" is_argument="false" />
  </searchInterface>

</simulationConfiguration>
```



**Relevant für die  
Namenskonvention  
des output files**

# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDirectory="../example_sims/output/"
    saveInteractionLog="true"
    saveRelevanceJudgments="true"
    trec_eval="false" />

  <topics>
    <!--
    These are the topics used in the experiment
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>

  <users>
    <user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
    <user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

    <!--
    Add your newly created user configurations below. By adding:
    <user configurationFile="path/to/your/user_configuration.xml" />
    -->
  </users>

  <searchInterface class="PyTerrierSearchInterface">
    <!--
    This is the search interface that the simulated users will interact with
    -->
    <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
    <attribute name="text_field" type="string" value="text" is_argument="true" />
    <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
    <attribute name="memory" type="boolean" value="true" is_argument="false" />
  </searchInterface>

</simulationConfiguration>
```

**Definiert den Pfad für  
die Output files & die  
zu speichernden Files**

# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDirectory="../example_sims/output/"
    saveInteractionLog="true"
    saveRelevanceJudgments="true"
    trec_eval="false" />

  <topics>
    <!--
      These are the topics that each simulated user will complete
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>

  <users>
    <!--
      Add your newly created user configurations below. By adding:
    -->
    <user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
    <user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

    <!--
      Add your newly created user configurations below. By adding:
    -->
    <user configurationFile="path/to/your/user_configuration.xml" />
  </users>

  <searchInterface class="PyTerrierSearchInterface">
    <!--
      This is the search interface that the simulated users will interact with
    -->
    <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
    <attribute name="text_field" type="string" value="text" is_argument="true" />
    <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
    <attribute name="memory" type="boolean" value="true" is_argument="false" />
  </searchInterface>

</simulationConfiguration>
```

**Definiert die zu  
verwenden Topics im  
Experiment**

# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDirectory="../example_sims/output/"
    saveInteractionLog="true"
    saveRelevanceJudgments="true"
    trec_eval="false" />

  <topics>
    <!--
    These are the topics that each simulated user will complete
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>
```

```
</users>

<user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
<user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

<!--
Add your newly created user configurations below. By adding:
<user configurationFile="path/to/your/user_configuration.xml" />
-->

</users>
```

```
<searchInterface>
  This is the search interface that the simulated users will interact with
  -->
  <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
  <attribute name="text_field" type="string" value="text" is_argument="true" />
  <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
  <attribute name="memory" type="boolean" value="true" is_argument="false" />
</searchInterface>

</simulationConfiguration>
```



**Definiert die zu  
verwenden User  
Agents im Experiment**



# Konfiguration der Simulationsexperimente

```
<simulationConfiguration id="core-bm25">

  <output baseDirectory="../example_sims/output/"
    saveInteractionLog="true"
    saveRelevanceJudgments="true"
    trec_eval="false" />

  <topics>
    <!--
    These are the topics that each simulated user will complete
    -->
    <topic id="1" filename="../example_data/CORE/topics/topic.1" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="2" filename="../example_data/CORE/topics/topic.2" qrelsFilename="../example_data/CORE/core.qrels" />
    <topic id="3" filename="../example_data/CORE/topics/topic.3" qrelsFilename="../example_data/CORE/core.qrels" />
  </topics>

  <users>

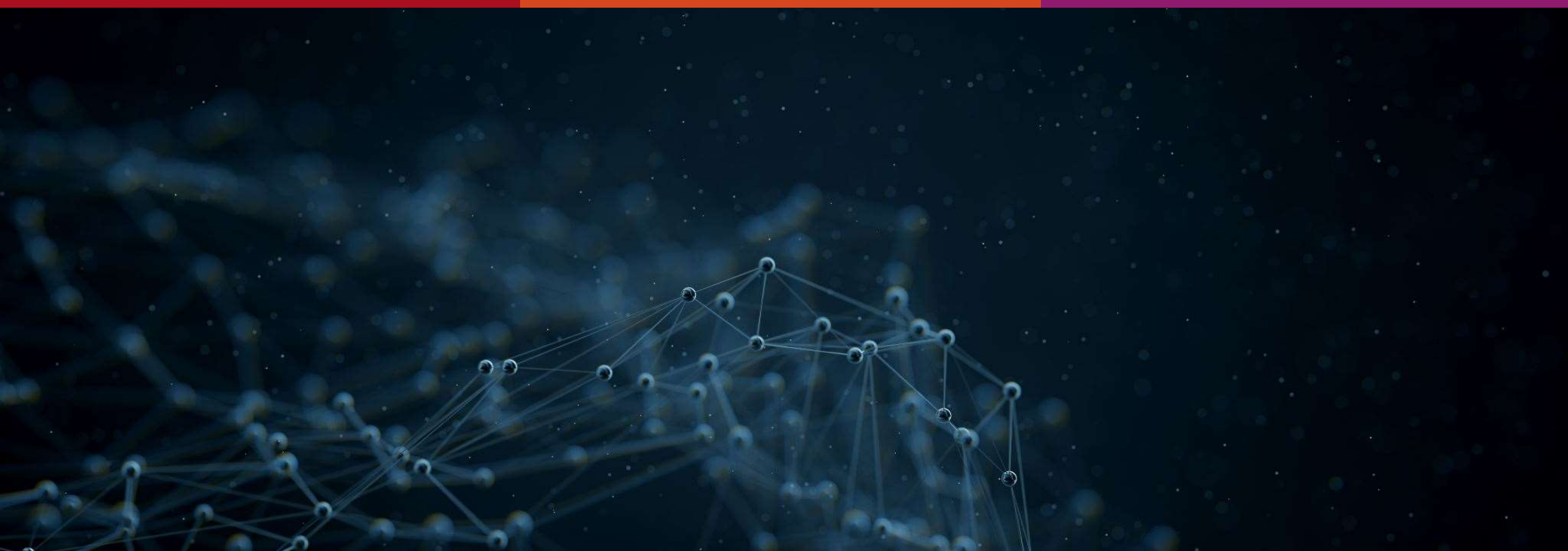
    <user configurationFile="../example_sims/users/CORE_tri_term_query_user_DIS22.xml" />
    <user configurationFile="../example_sims/users/CORE_predetermined_query_user_DIS22.xml" />

    <!--
    Add your newly created user configurations below. By adding:
    <user configurationFile="path/to/your/user_configuration.xml" />
    -->

  </users>

  <searchInterface class="PyTerrierSearchInterface">
    <!--
    This is the search interface that the simulated users will interact with
    -->
    <attribute name="index_or_dir" type="string" value="../example_data/index_CORE/" is_argument="true" />
    <attribute name="text_field" type="string" value="text" is_argument="true" />
    <attribute name="wmodel" type="string" value="BM25" is_argument="true" />
    <attribute name="memory" type="boolean" value="true" is_argument="false" />
  </searchInterface>
</simulationConfiguration>
```

**Definiert den Index  
und die  
Rangfolgefunktion  
usw., die verwendet  
werden sollen.**



# Erste Schritte

# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können

Um Zugriff auf GitHub Education zu erhalten, folgen Sie den Anweisungen hier:

[https://education.github.com/discount\\_requests/application](https://education.github.com/discount_requests/application)

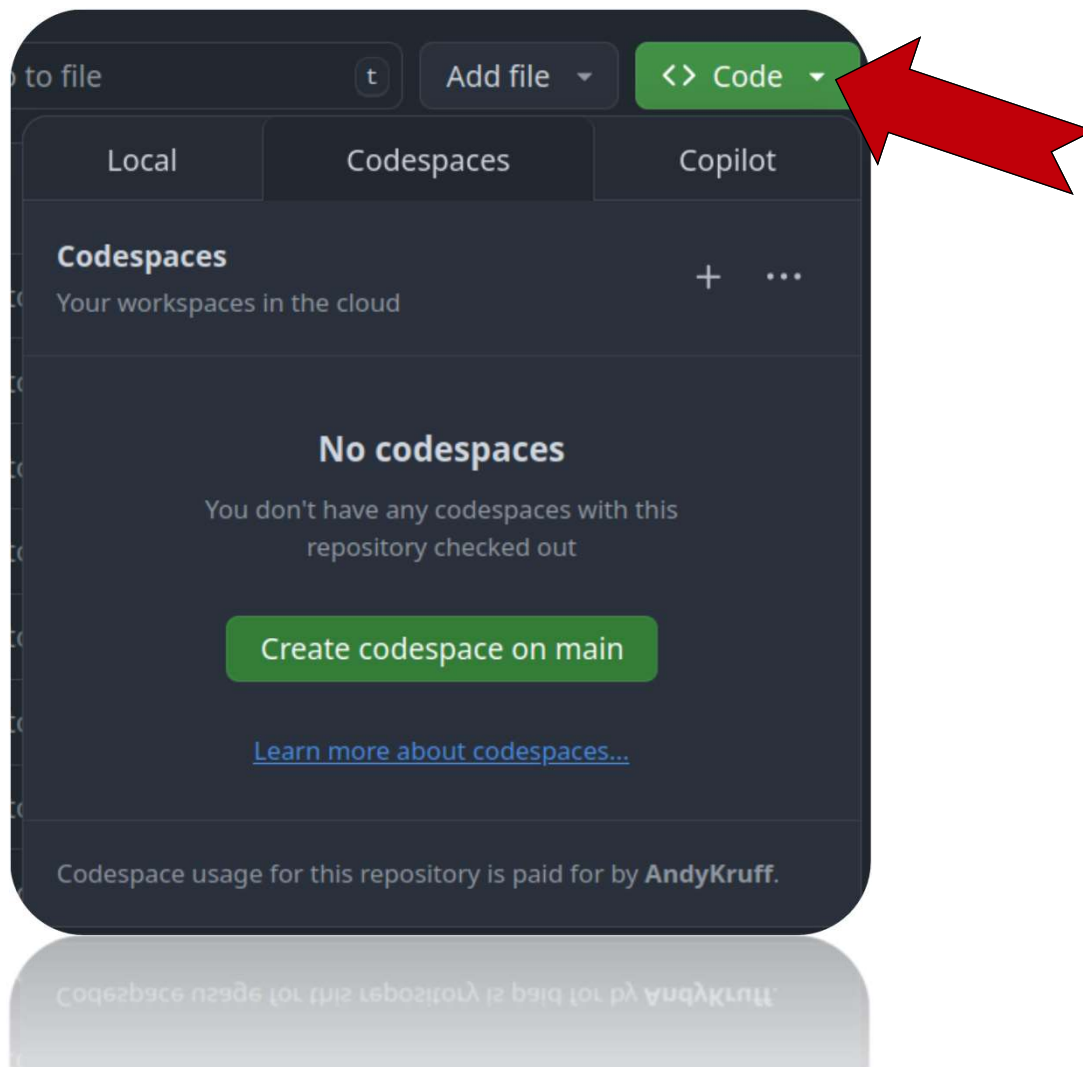
# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces

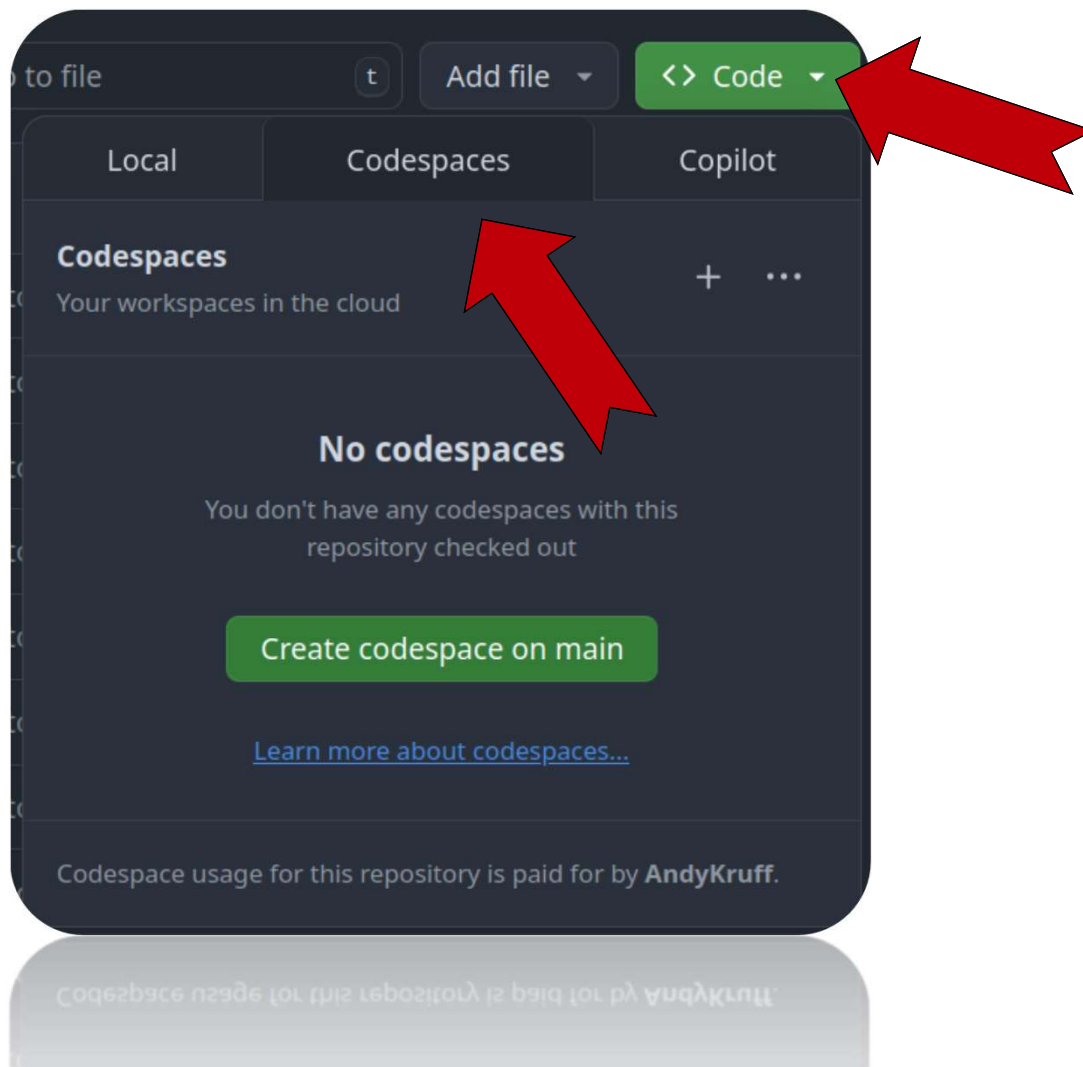
Link zum Repository: [https://github.com/irgroup/DIS22\\_Sim4IA\\_Prototype](https://github.com/irgroup/DIS22_Sim4IA_Prototype)



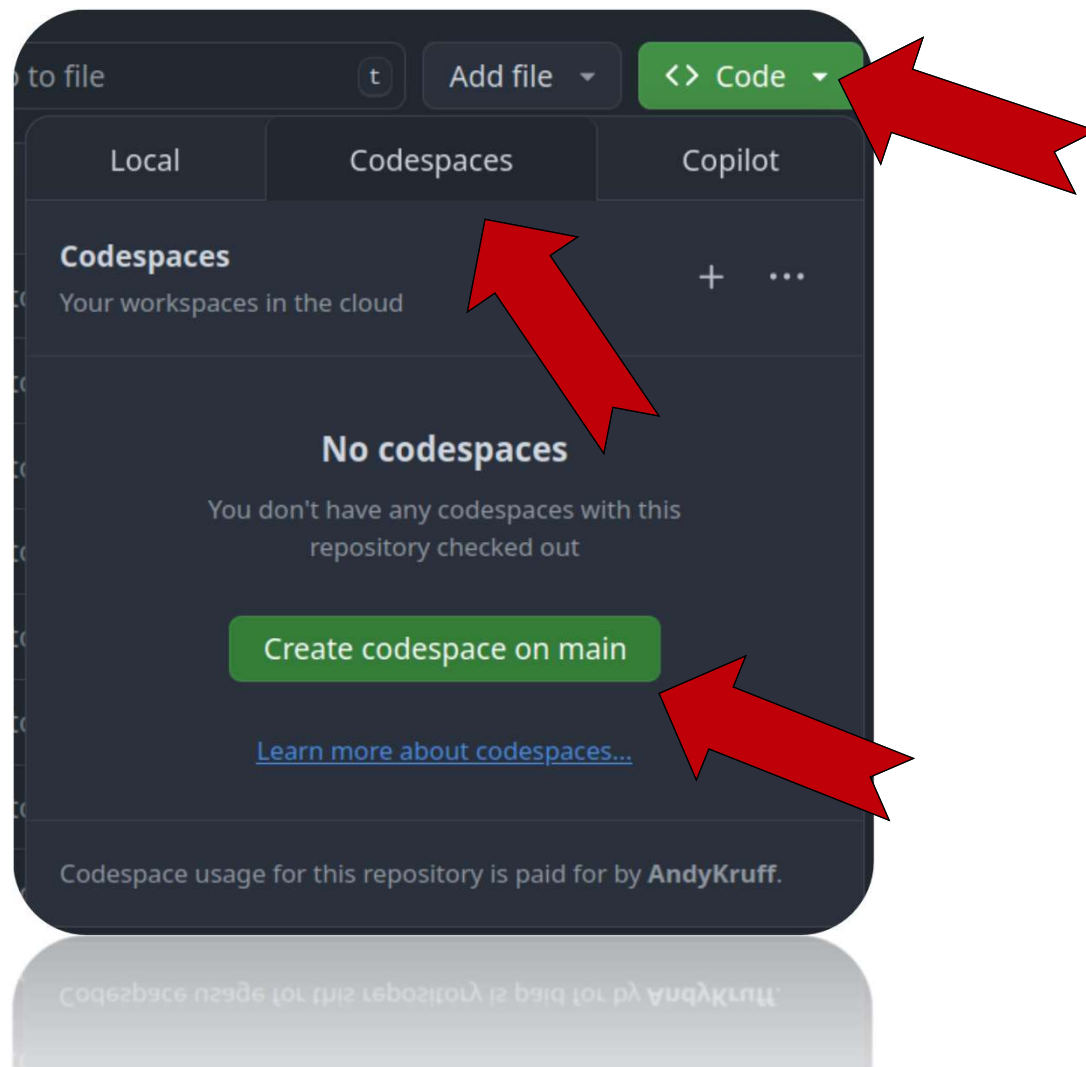
# Erste Schritte



# Erste Schritte



# Erste Schritte



# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces
3. Herunterladen des vorläufigen Index des CORE Datensatz von Sciebo

```
curl -L -o index_CORE.zip "https://th-koeln.sciebo.de/s/F9AEa1CXyk2RTpf/download"  
unzip index_CORE.zip -d ./example_data/
```

# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces
3. Herunterladen des vorläufigen Index des CORE Datensatz von Sciebo
4. Erstellen des Docker Containers und Zugriff auf den Container über das Terminal

```
docker-compose up -d --build
```

```
docker exec -it SIM4IA_DIS22_container bash
```

# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces
3. Herunterladen des vorläufigen Index des CORE Datensatz von Sciebo
4. Erstellen des Docker Containers und Zugriff auf den Container über das Terminal
5. **Aufgabe:** Konfiguration eines eigenen User Agents mit eigener Query Reformulierungsstrategie. Ausprobieren von bestehenden Query Reformulierungsstrategien und die dazugehörigen Parameter. Hinzufügen zum Simulationsexperiment (``example_sims/core_bm25_DIS22_Sim4IA.xml``)

# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces
3. Herunterladen des vorläufigen Index des CORE Datensatz von Sciebo
4. Erstellen des Docker Containers und Zugriff auf den Container über das Terminal
5. **Aufgabe:** Konfiguration eines eigenen User Agents mit eigener Query Reformulierungsstrategie. Ausprobieren von bestehenden Query Reformulierungsstrategien und die dazugehörigen Parameter. Hinzufügen zum Simulationsexperiment (`example\_sims/core\_bm25\_DIS22\_Sim4IA.xml`)
6. Ausführen der Experimente

```
cd simiir
```

```
python run_simiir.py ../example_sims/core_bm25_DIS22_Sim4IA.xml
```

# Erste Schritte

1. Beantragen von GitHub Education, um Codespaces nutzen zu können
2. Starten des Prototyp-Repositorys in Codespaces
3. Herunterladen des vorläufigen Index des CORE Datensatz von Sciebo
4. Erstellen des Docker Containers und Zugriff auf den Container über das Terminal
5. **Aufgabe:** Konfiguration eines eigenen User Agents mit eigener Query Reformulierungsstrategie. Ausprobieren von bestehenden Query Reformulierungsstrategien und die dazugehörigen Parameter. Hinzufügen zum Simulationsexperiment (``example_sims/core_bm25_DIS22_Sim4IA.xml``)
6. Ausführen der Experimente
7. Evaluierung der Results (Results in ``example_sims/output/``)
  1. Qualitative Analyse der gefundenen Dokumente und der Query Variants



# Qualitative Analyse der Query Reformulierungen

**REMINDER:** core-bm25-1-query-predetermined-200td.log

# Qualitative Analyse der Query Reformulierungen

**REMINDER:** core-bm25-1-query-predetermined-200td.log

SimulationConfiguration ID

# Qualitative Analyse der Query Reformulierungen

**REMINDER:** core-bm25-1-query-predetermined-200td.log

UserConfiguration ID

# Qualitative Analyse der Query Reformulierungen

**REMINDER:** core-bm25-1-query-predetermined-200td.log

Topic ID

# Qualitative Analyse der Query Reformulierungen

**REMINDER:** core-bm25-1-query-predetermined-200td.log

  
Topic ID

## Auswertung der Log Files:

```
ACTION START 600 0 START
ACTION QUERY 600 10 carpal tunnel instruments
ACTION SERP 600 15 EXAMINE_SERP
ACTION SNIPPET 600 18 SNIPPET_RELEVANT 3774654
ACTION DOC 600 38 EXAMINING_DOCUMENT 3774654
ACTION MARK 600 41 CONSIDERED_RELEVANT 3774654
ACTION SNIPPET 600 44 SNIPPET_RELEVANT 8834257
```

```
ACTION SNIPPET 600 44 SNIPPET_RELEVANT 8834257
ACTION MARK 600 41 CONSIDERED_RELEVANT 3774654
```

# Qualitative Analyse der Query Reformulierungen


**REMINDER:** core-bm25-1-query-predetermined-200td.log

  
Topic ID

## Auswertung der Log Files:

```
ACTION START 600 0 START
ACTION QUERY 600 10 carpal tunnel instruments
ACTION SERP 600 15 EXAMINE_SERP
ACTION SNIPPET 600 18 SNIPPET_RELEVANT 3774654
ACTION DOC 600 38 EXAMINING_DOCUMENT 3774654
ACTION MARK 600 41 CONSIDERED_RELEVANT 3774654
ACTION SNIPPET 600 44 SNIPPET_RELEVANT 8834257
```

```
ACTION SNIPPET 600 44 SNIPPET_RELEVANT 8834257
ACTION MARK 600 41 CONSIDERED_RELEVANT 3774654
```

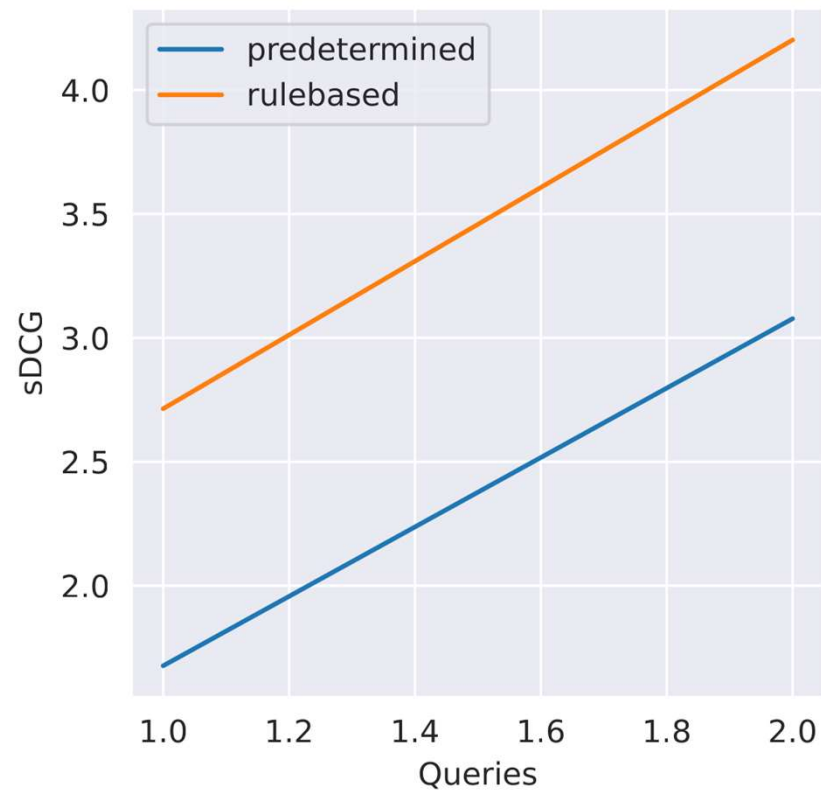
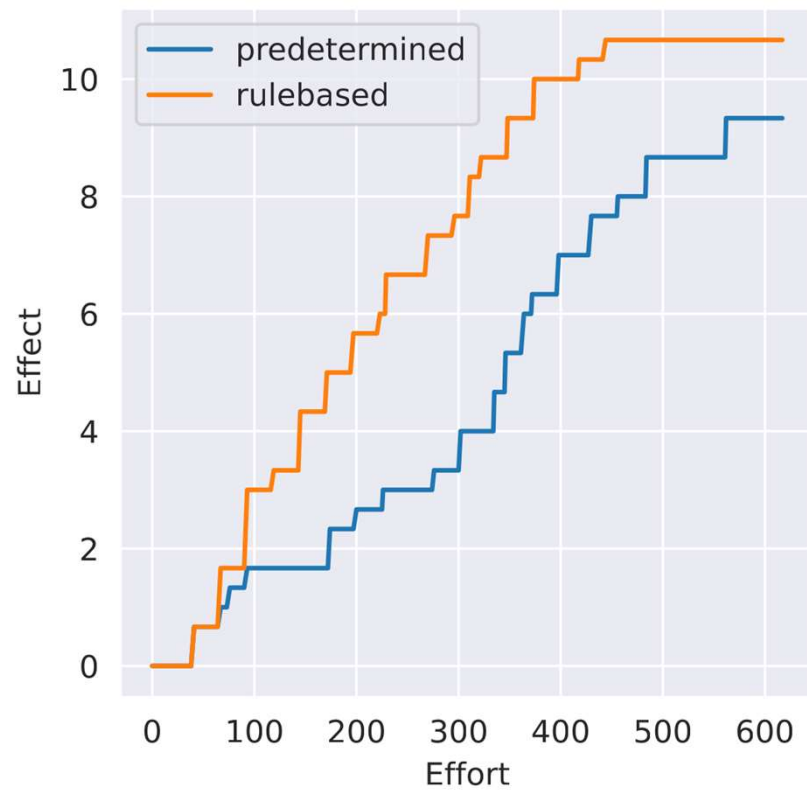


```
index_ref = "3774654"
index = pt.IndexFactory.of("./index_CORE")
meta_index = index.getMetaIndex()
doc_id = "3774654"
idx = meta_index.getDocument("docno", doc_id)
text_field = "text"
content = meta_index.getItem(text_field, int(idx))
```



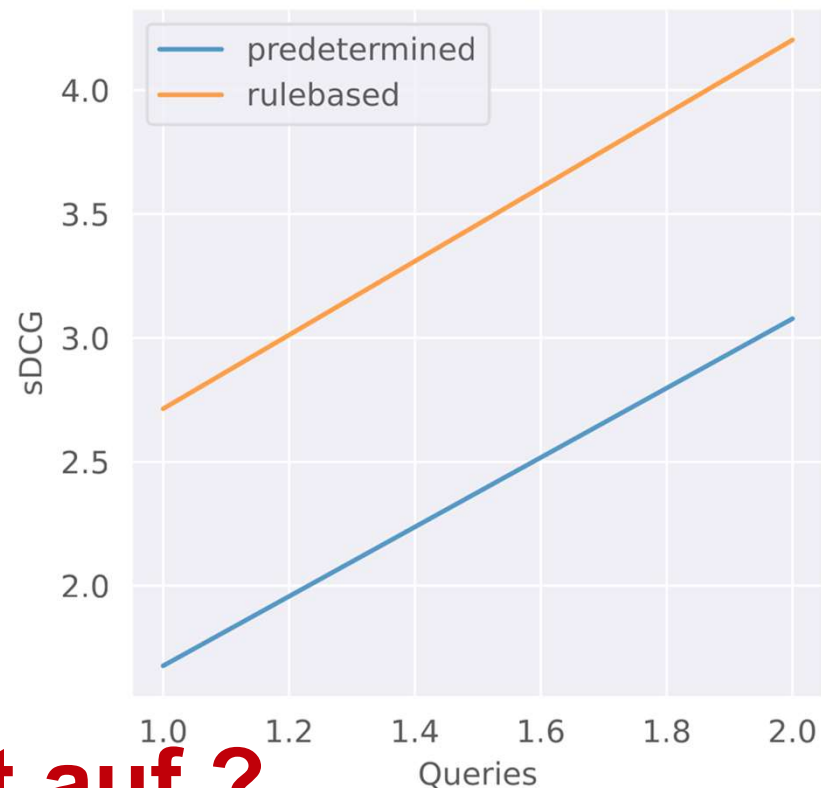
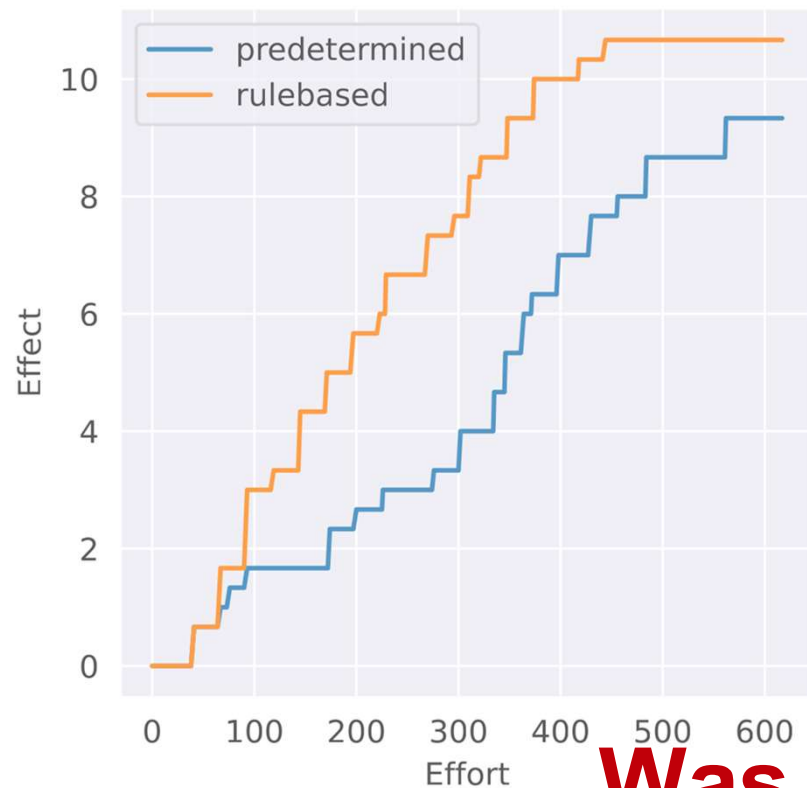
# Ausblick

# Weitere Evaluationsmöglichkeiten





# Weitere Evaluationsmöglichkeiten



**Was fällt auf ?**

## Derzeitige Limitationen des Prototypen

**Ergebnisse der Evaluation nur begrenzt aussagekräftig, da...**

- **Unzureichende Themenbeschreibung**
- **Fehlende Verallgemeinbarkeit durch zu wenige Sessions & Topics**
- **Nur wenige Dokumente wurden auf Relevanz bewertet.**