

Information Retrieval

Exercise – Winter term 2025/2026

klara.gutekunst@uni-kassel.de

Agenda

1. Ranked Retrieval Measures
2. GitHub Repository
3. TIRA

Precision–Recall Curve

Exercise

Given the following two rankings:

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
D	t_2	👎	👎	👍	👍	👎	👎	👍	👍	👎	👍
E	t_1	👎	👍	👍	👎	👍	👍	👍	👎	👎	👎
E	t_2	👍	👎	👍	👍	👎	👎	👎	👍	👍	👎

Which system is better?

Precision–Recall Curve

Given the following two rankings:

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
D	t_2	👎	👎	👍	👍	👎	👎	👍	👍	👎	👍
E	t_1	👎	👍	👍	👎	👍	👍	👍	👎	👎	👎
E	t_2	👍	👎	👍	👍	👎	👎	👎	👍	👍	👎

Which system is better?

They achieve equal precision and recall for topics t_1 and t_2 .

System	Topic	Precision	Recall
D	t_1	0.5	1.0
D	t_2	0.5	1.0
E	t_1	0.5	1.0
E	t_2	0.5	1.0

Precision–Recall Curve

Given the following two rankings:

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
D	t_2	👎	👎	👍	👍	👎	👎	👍	👍	👎	👍
E	t_1	👎	👍	👍	👎	👍	👍	👍	👎	👎	👎
E	t_2	👍	👎	👍	👍	👎	👎	👎	👍	👍	👎

Which system is better?

Draw the precision–recall curves.

Precision–Recall Curve

Given the following two rankings:

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
D	t_2	👎	👎	👍	👍	👎	👎	👍	👍	👎	👍
E	t_1	👎	👍	👍	👎	👍	👍	👍	👎	👎	👎
E	t_2	👍	👎	👍	👍	👎	👎	👎	👍	👍	👎

Which system is better?

Draw the precision–recall curves.

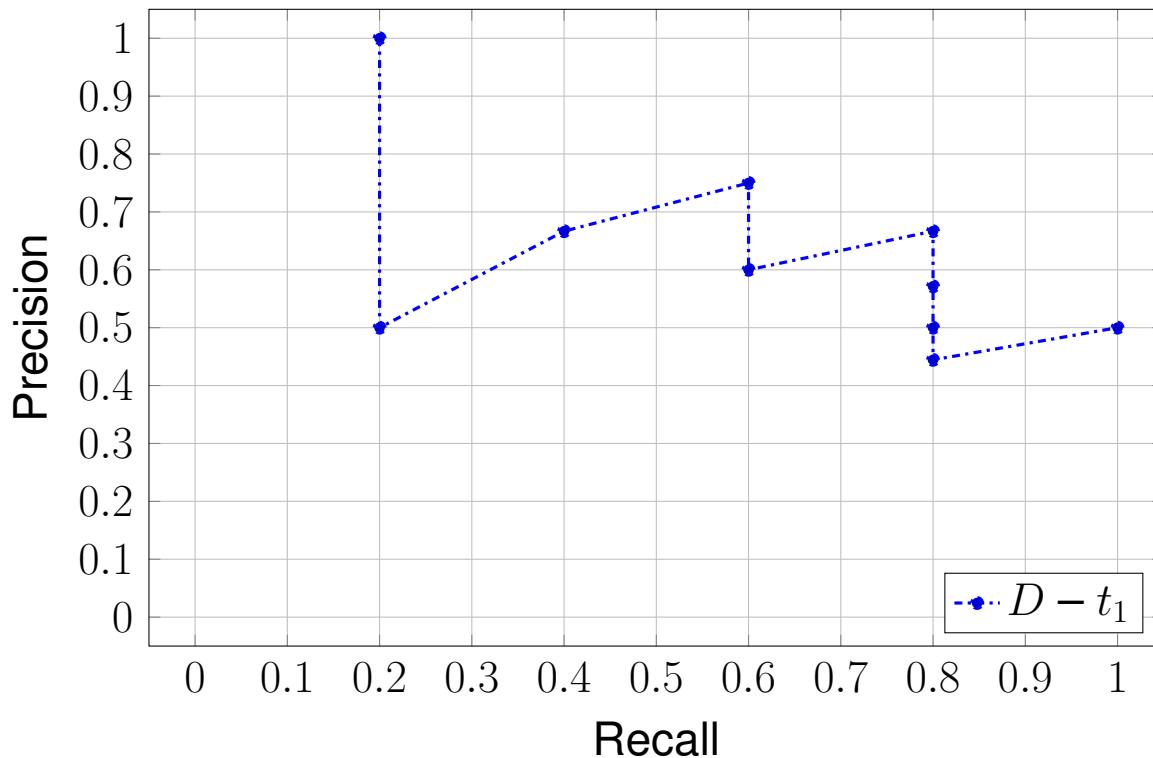
Compute precision and recall at rank k .

Precision–Recall Curve

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
	precision	1	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{3}{4}$	$\frac{3}{5}$	$\frac{4}{6}$	$\frac{4}{7}$	$\frac{4}{8}$	$\frac{4}{9}$	$\frac{5}{10}$
	recall	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{2}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{5}{5}$

Precision–Recall Curve

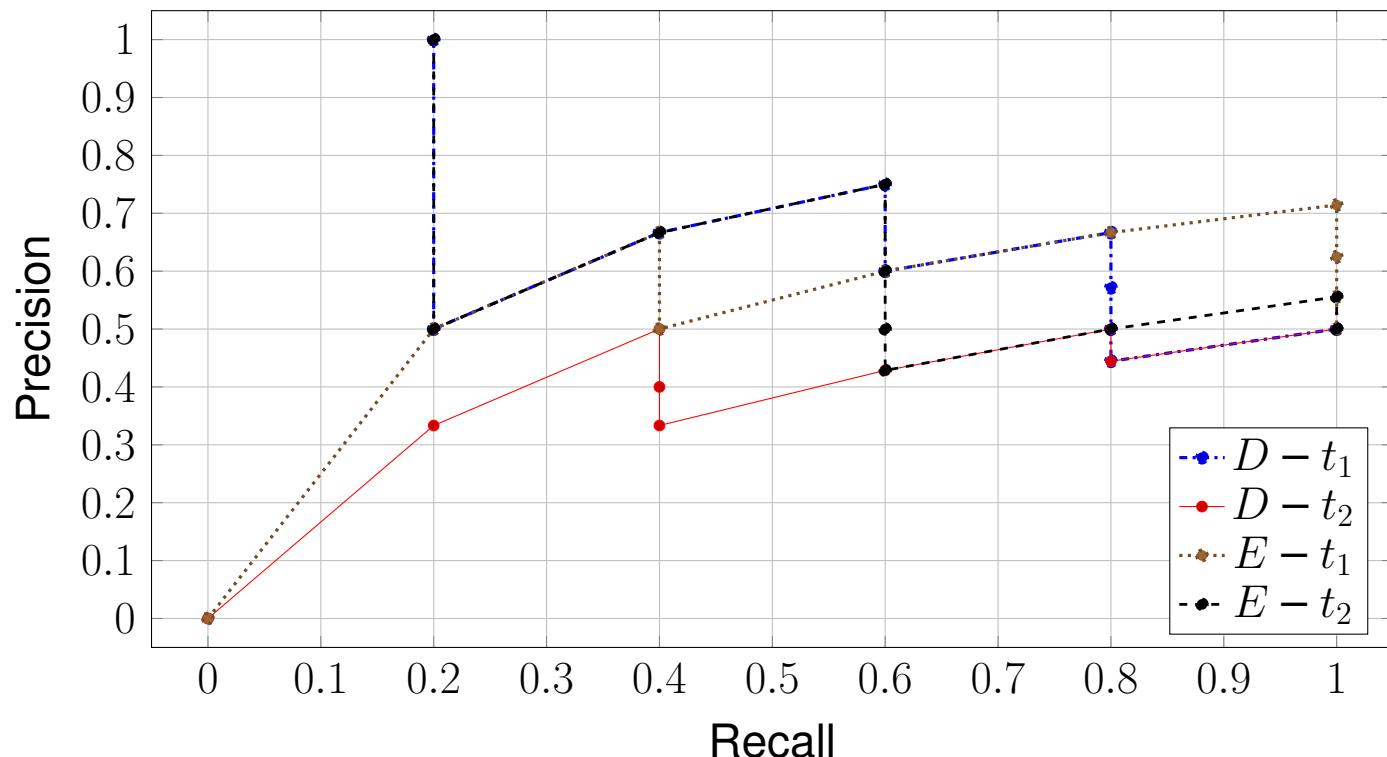
System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
	precision	1	$\frac{1}{2}$	$\frac{2}{3}$	$\frac{3}{4}$	$\frac{3}{5}$	$\frac{4}{6}$	$\frac{4}{7}$	$\frac{4}{8}$	$\frac{4}{9}$	$\frac{5}{10}$
	recall	$\frac{1}{5}$	$\frac{1}{5}$	$\frac{2}{5}$	$\frac{3}{5}$	$\frac{3}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{4}{5}$	$\frac{5}{5}$



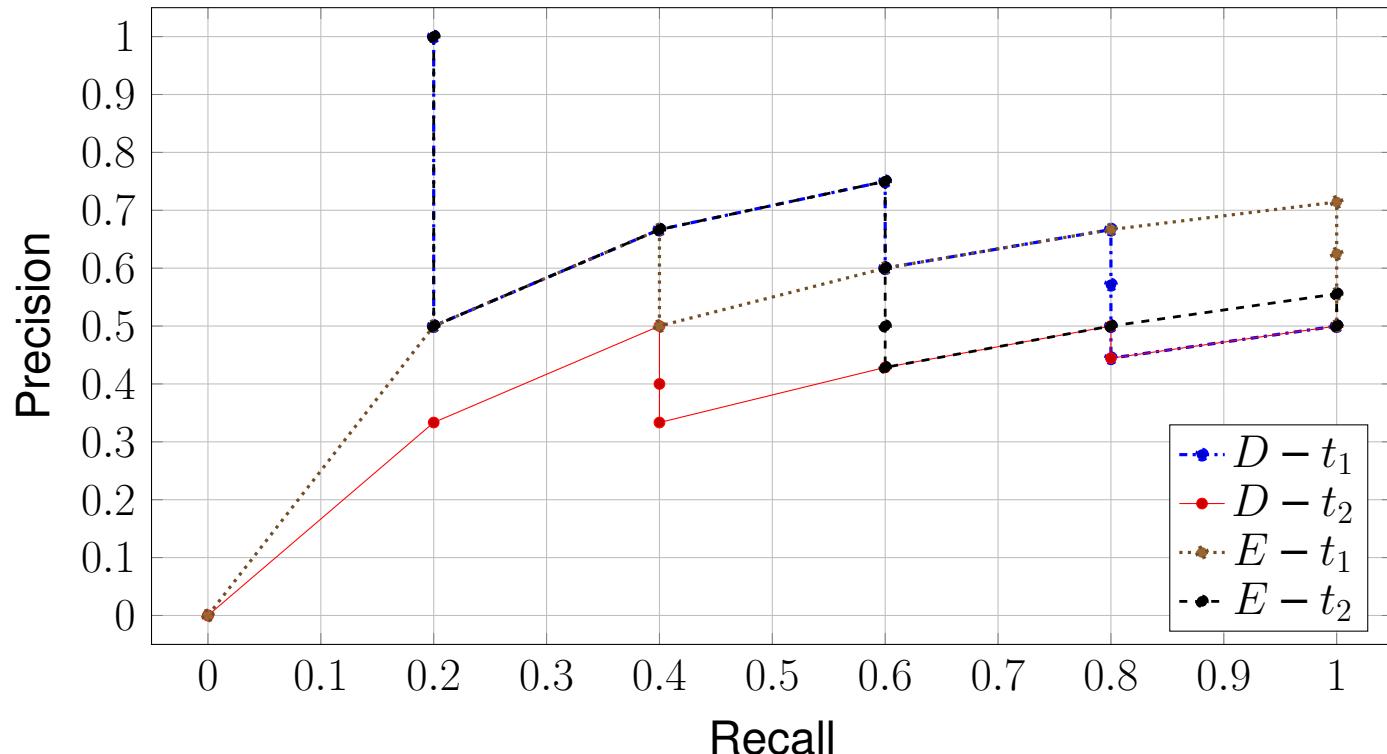
Precision–Recall Curve

System	Topic	Relevance at rank									
		1	2	3	4	5	6	7	8	9	10
D	t_1	👍	👎	👍	👍	👎	👍	👎	👎	👎	👍
	precision	1	1/2	2/3	3/4	3/4	4/6	4/7	4/8	4/9	5/10
	recall	1/5	1/5	2/5	3/5	3/5	4/5	4/5	4/5	4/5	5/5
	t_2	👎	👎	👍	👍	👎	👎	👍	👍	👎	👍
E	t_1	👎	👍	👍	👎	👍	👍	👍	👎	👎	👎
	precision	0	1/2	2/3	2/4	3/4	4/6	5/7	5/8	5/9	5/10
	recall	0	0	1/5	2/5	2/5	4/5	5/5	5/5	5/5	5/5
	t_2	👍	👎	👍	👍	👎	👎	👎	👍	👍	👎
E	precision	1	1/2	2/3	3/4	3/4	3/6	3/7	4/8	5/9	5/10
	recall	1/5	1/5	2/5	3/5	3/5	6/5	7/5	8/5	9/5	5/5

Precision–Recall Curves



Precision–Recall Curves



- Points between the original data points have no direct interpretation
- Best system can be quantified by the larger area under its curve
- Average precision $AP(q, R)$ estimates the area under the uninterpolated precision–recall curve for topic t and query $q \in Q$
- $MAP(Q)$: Average area under precision-recall curves for different topics
[“An Introduction to Information Retrieval” (Manning et al.), Section 8.4]

Next Steps

Assignment

- Exercise sheet on [temir.org](#)
- Download [Docker](#)
- Create a [GitHub](#) account for the next stage (i.e., building an IR system)
- Create [TIRA](#) account
- Inspiration

GitHub Repository

Version Control System VCS

We maintain all project work in a [monorepo](#) so that everything stays in one place.

1. Sign up for [GitHub](#) *.
2. [Fork](#) the [wows-code](#) repository †.
3. Clone the forked repository *.
4. Create a directory to store your team's work and push your changes †. Use the format `ks-<TEAM-NAME>`.
5. Start developing with [dev containers](#) *.
6. Once your approach is “finished enough”, create a [pull request](#) to the original repository *.

For detailed instructions, see the repository's [ECIR26 README](#) file.

To contribute multiple approaches, keep each in its own directory, e.g.,

`ks-<TEAM-NAME>-01`, `ks-<TEAM-NAME>-02`,

* Required for **every** team member

† Required for only **one** team member

Get Started

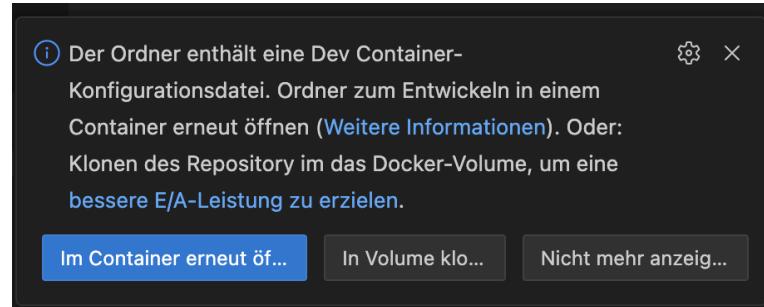
Development Container

- ❑ Dev containers allow using container as a full-features development environment
- ❑ Container environments should be easy to use, create, and recreate
- ❑ `Dockerfile` builds environment (e.g., base image, package installation)
- ❑ `devcontainer.json` configures how VS Code uses environment (e.g., VS Code extensions)
- ❑ For further details on Docker and dev containers, refer to these [slides](#)

Get Started

Development Container

1. Ensure that **Docker** is running.
2. Open a directory that contains **exactly one** dev container configuration in its root.
 - ❑ Open your cloned fork of the [wows-code](#) repository.
 - ❑ Use the CLI to navigate to the directory containing the dev container configuration (e.g., `cd ecir26/template-new-approach`).
 - ❑ Run `code .` to open the directory in a dedicated VS Code instance.
3. When prompted, select [Reopen in Container](#) in VS Code, or invoke it manually from the [Command Palette](#) (`cmd/ctrl + shift + p` or `F1`).
4. Continue following the instructions in the [README file](#).



(3): Reopen in Container in VS Code.

If you run into issues with (3), try `docker pull <DOCKER-BASE-IMAGE>` to get a clearer error message. You can find `<DOCKER-BASE-IMAGE>` in the corresponding Dockerfile.

Get Started

Artifact-Free Approach

- ❑ Create index
- ❑ Implement retrieval strategy
- ❑ Running `./retrieve.py <OPTIONS>` will create multiple files.
 - Possible options are specified in **README** file.

```
zcat run.txt.gz:
```

query ID	-	document ID	rank	score	-
74	Q0	950c12d12803 ... b6	0	9.558411251356873	pyterrier
74	Q0	6075d2b6a65d ... 36	1	8.716015411249087	pyterrier
⋮				⋮	⋮

```
cat retrieval-metadata.yml:
```

- ❑ Resources
- ❑ ...

Get Started

PyTerrier Artifact Approach

PyTerrier is a Python framework that enables the construction of declarative retrieval pipelines.

What is [artifact sharing](#)?

- Artifacts are trained models, pre-built indexes
- Proposed at [SIGIR'25](#)
- Functionality in PyTerrier Python package
 - Trained model: `pyterrier_bm25 = pt.Artifact.from_url(f"tira:dataset_id/ows/pyterrier-BM25-on-default")`
 - Pre-built index: `index = pt.Artifact.from_url(f"tira:dataset_id/ows/pyterrier-index-default")`
- Improves reproducibility & saves time
- Some PyTerrier Artifacts are available in TIRA (cf. [overview](#))

Before running the [PyTerrier Artifacts Approach](#) to see how artifacts are used in development, first complete the [PyTerrier Artifacts tutorial](#).

Get Started

Ideas:

- Build a custom index(es) and merge them
 - Use LLMs for query independent stopping/
context-aware term weights [Paper]
 - Index title/ metadata/ whole text
- Change document/ query representations
- Implement reranker(s)
- Perform query rewriting
 - (In-) Dependent of query intent/ type
- Combine n retrieval models
 - Assign weights based on LLM-generated relevance judgments
(i.e., models whose retrieved documents score highest receive greater weight) [Paper]
- Query expansion [Paper] via ...
 - ... RM3 [Paper]
 - ... LLMs [Paper]

Start coding and compare your ideas to **baselines**.

Assignment

- Implement one or more retrieval systems
- Due Date: Monday, 15.12.2025*, 23:59
- Deliverable: TIRA submission(s)

* Date pending confirmation from Jena.

TIRA Account

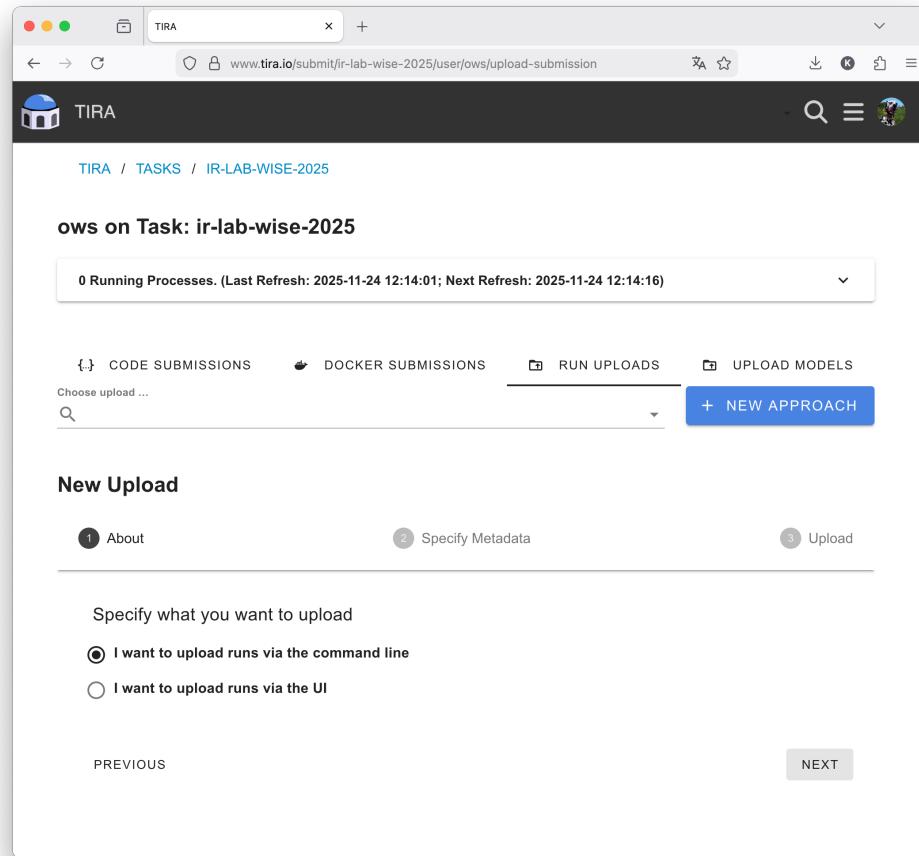
TIRA Integrated Research Architecture

1. Sign up to TIRA.
 - You may use Login in with GitHub.
2. Go to GET STARTED.
3. Go to IR Lab Jena/Kassel/Radboud WiSe 2025.
4. Register your team.

TIRA Token

Upload Runs

(1) Go to IR Lab Jena/Kassel/Radboud WiSe 2025, (2) click on SUBMIT.



Display personal token for uploading runs to TIRA by clicking on NEXT.