Task 1 (Search Engine)

Information Retrieval Pre

Yan Jiafeng

Department of Information
CUFE

January 10, 2025

Overview

Framework

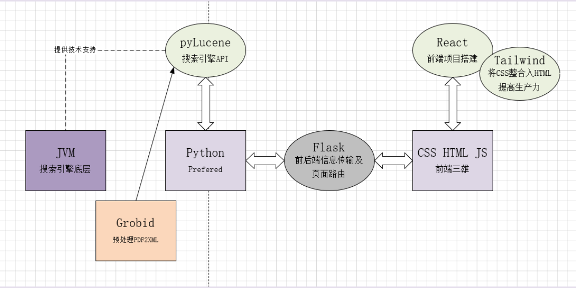


Figure: Main tech stack

- Not yet familiar with Java development. When seen PyLucene packaging, use it decisively
- there are a lot of problems during use
- Through consulting the official documentation basic solution finally.

Lucene

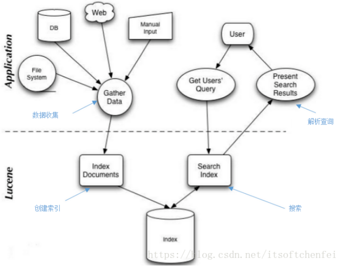


Figure: main frame of Lucene

Due to frequent errors when Pylucene was installed on Windows, I chose to use the SSH connection to the Linux virtual machine to "save the curve".

Points

I implemented a simple page, but relatively more features of the search engine. The general functions are as follows:

- User-friendly front-end design
- Normal query / Boolean query / multi-field query / wildcard query
- paging (Max 5 pages with 15 items each page)
- dynamic digest generation / highlight & customized highlight settings
- spelling correction & customized checking dictionary
- stop words & customized stop words dictionary
- Allow to upload your own pdf files

static display behind

Front-end Design

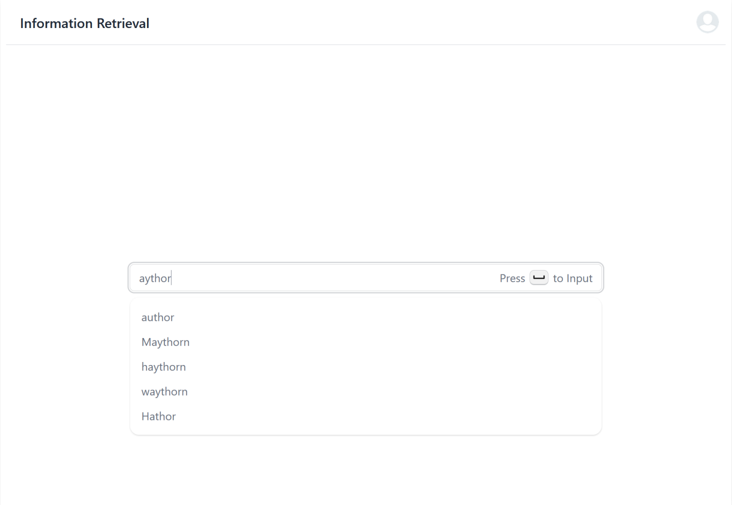


Figure: Home Page & correction

Front-end Design

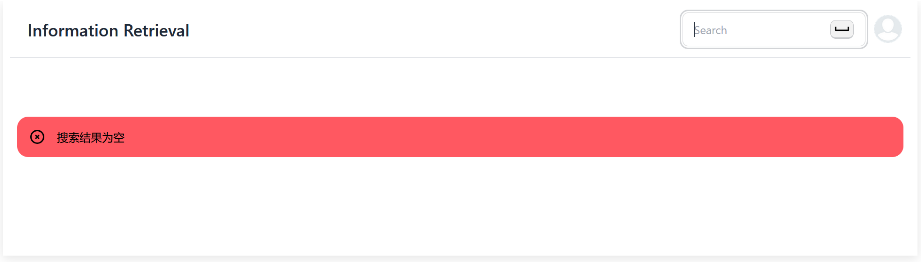
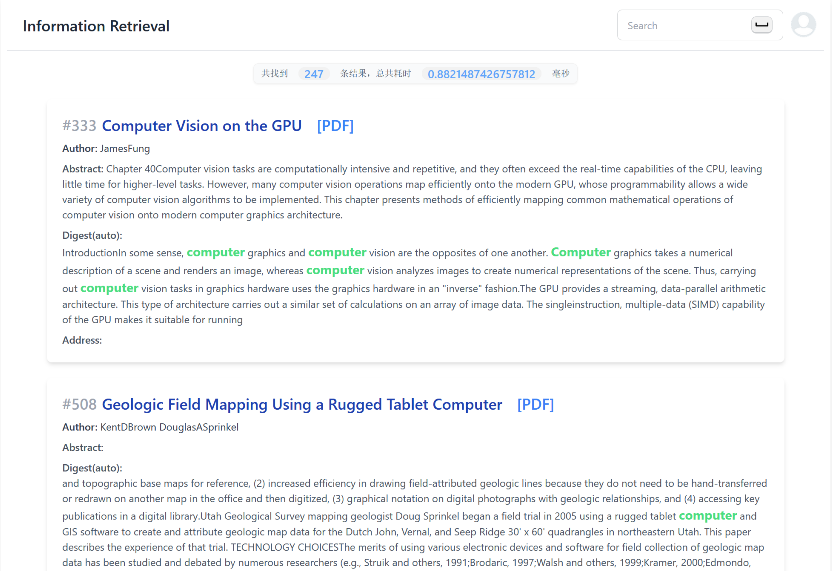


Figure: user-friendly alert

Front-end Design



Query

Figure: Can employ flexible query

Paging

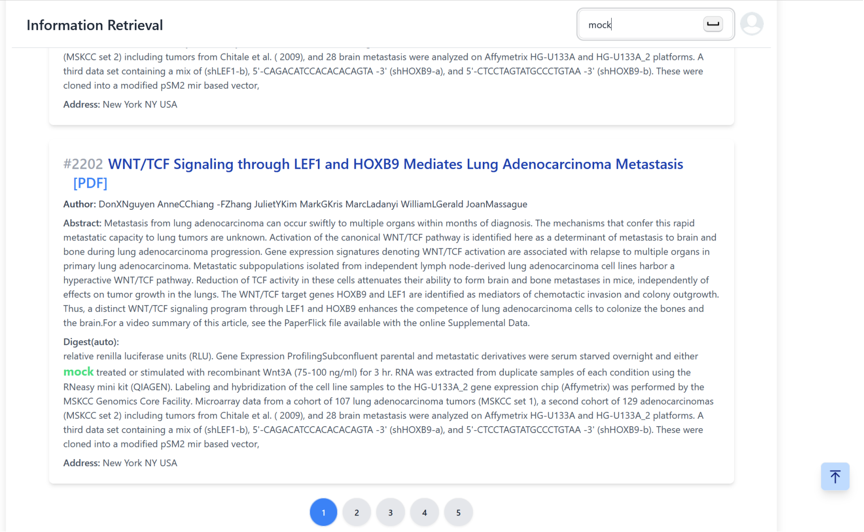


Figure: Paging & TopKey

Customization

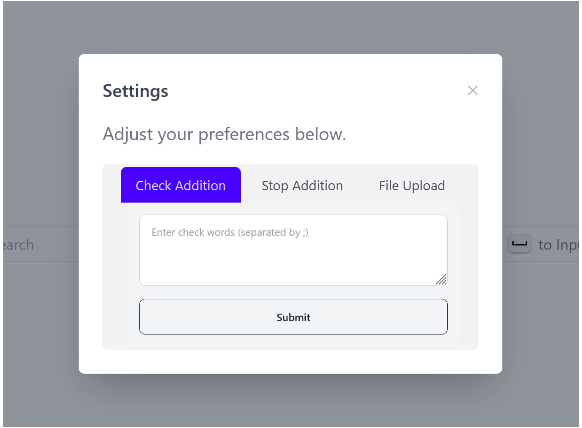


Figure: check addition

Customization

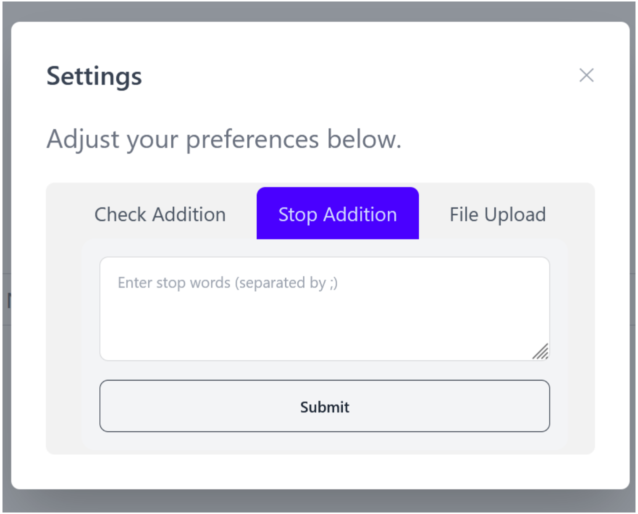


Figure: stop words addition

Customization

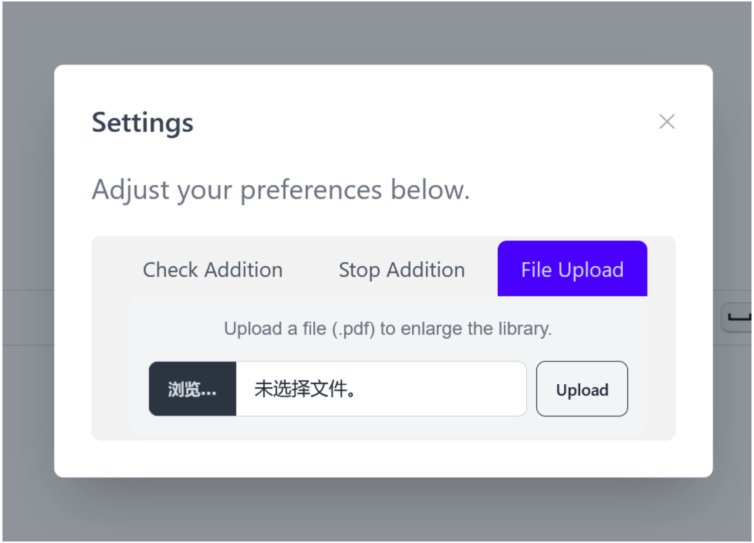


Figure: Upload yourself files

Highlight Settings

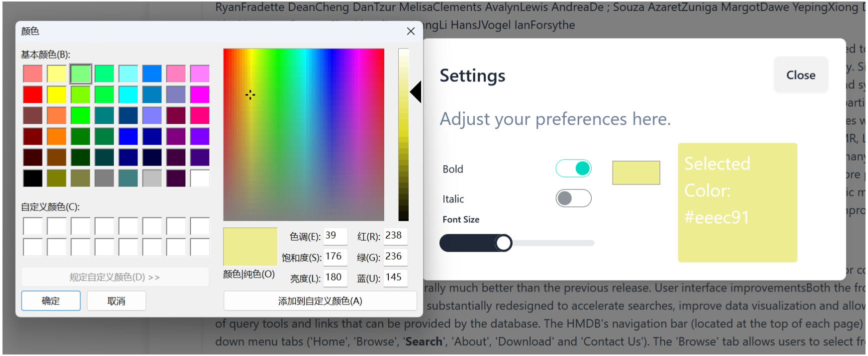


Figure: Highlight Settings

Shorts

-

```python
paths = {
    'title': "{0}teiHeader/{0}fileDesc/{0}titleStmt/{0}title".format(ns),
    'funder': "{0}teiHeader/{0}fileDesc/{0}titleStmt/{0}funder".format(ns),
    'author': "{0}teiHeader/{0}fileDesc/{0}sourceDesc/{0}biblStruct/{0}analytic/{0}author/{0}persName".format(ns),
    'email': "{0}teiHeader/{0}fileDesc/{0}sourceDesc/{0}biblStruct/{0}analytic/{0}author/{0}email".format(ns),
    'affiliation': "{0}teiHeader/{0}fileDesc/{0}sourceDesc/{0}biblStruct/{0}analytic/{0}author/{0}affiliation/{0}orgName".format(ns),
    'address': "{0}teiHeader/{0}fileDesc/{0}sourceDesc/{0}biblStruct/{0}analytic/{0}author/{0}affiliation/{0}address".format(ns),
    'date': "{0}teiHeader/{0}fileDesc/{0}sourceDesc/{0}biblStruct/{0}monogr/{0}imprint/{0}date".format(ns),
    'keyword': "{0}teiHeader/{0}profileDesc/{0}textClass/{0}keywords".format(ns),
    'abstract': "{0}teiHeader/{0}profileDesc/{0}abstract".format(ns),
    'fulltext': '{0}text/{0}body'.format(ns),
}
for field, path in paths.items():
```

Figure: Multi-field

I wanted to implement a more acceptable front-end and checkbox, but due to time reasons could only use the default way in lucene to perform complex queries

Document

Supported query syntax

Standard query parser borrows most of its syntax from the classic query parser but adds more features and expressions on top of that syntax.

A *query* consists of clauses, field specifications, grouping and Boolean operators and interval functions. We will discuss them in order.

Basic clauses

A query must contain one or more clauses. A clause can be a literal term, a phrase, a wildcard expression or other expression that

The following are some examples of simple one-clause queries:

- `test`

  selects documents containing the word *test* (term clause).

- `"test equipment"`

  phrase search; selects documents containing the phrase *test equipment* (phrase clause).

- `"test failure"~4`

  proximity search; selects documents containing the words *test* and *failure* within 4 words (positions) from each other. The provided "proximity" is technically translated into "edit distance" (maximum number of atomic word-moving operations required to transform the document's phrase into the query phrase).

- `tes*`

  prefix wildcard matching; selects documents containing words starting with *tes*, such as: *test*, *testing* or *testable*.

- `/.est(s|ing)/`

  documents containing words matching the provided regular expression, such as *resting* or *nests*.

- `nest~2`

  fuzzy term matching; documents containing words within 2-edits distance (2 additions, removals or replacements of a letter) from *nest*, such as *test*, *net* or *rests*.

Field specifications

Most clauses can be prefixed by a field name and a colon: the clause will then apply to that field only. If the field specification is omitted, the query parser will expand the clause over all fields specified by a call to `setMultiFields(CharSequence[])` or will use the default field provided in the call to `parse(String, String)`.

The following are some examples of field-prefixed clauses:

- `title:test`

Figure: Document

w/ LLM

#6 LTM: Lightweight Textured Mesh Extraction and Refinement of Large Unbounded Scenes for Efficient Storage and Real-time Rendering [PDF[4]] [Copy] [Kimi[12]] [REL]

Authors: Jaehoon Choi, Rajvi Shah, Qinbo Li, Yipeng Wang, Ayush Saraf, Changil Kim, Jia-Bin Huang, Dinesh Manocha, Suhib Alsisan, Johannes Kopf

Advancements in neural signed distance fields (SDFs) have enabled modeling 3D surface geometry from a set of 2D images of real-world scenes. Baking neural SDFs can extract explicit mesh with appearance baked into texture maps as neural features. The baked meshes still have a large memory footprint and require a powerful GPU for real-time rendering. Neural optimization of such large meshes with differentiable rendering pose significant challenges. We propose a method to produce optimized meshes for large unbounded scenes with low triangle budget and high fidelity of geometry and appearance. We achieve this by combining advancements in baking neural SDFs with classical mesh simplification techniques and proposing a joint appearance-geometry refinement step. The visual quality is comparable to or better than state-of-the-art neural meshing and baking methods with high geometric accuracy despite significant reduction in triangle count, making the produced meshes efficient for storage, transmission, and rendering on mobile hardware. We validate the effectiveness of the proposed method on large unbounded scenes from mip-NeRF 360, Tanks & Temples, and Deep Blending datasets, achieving at-par rendering quality with 73× reduced triangles and 11× reduction in memory footprint.

#7 MoPE-CLIP: Structured Pruning for Efficient Vision-Language Models with Module-wise Pruning Error Metric [PDF[10]] [Copy] [Kimi[15]] [REL]

Authors: Haokun Lin, Haoli Bai, Zhili Liu, Lu Hou, Muyi Sun, Linqi Song, Ying Wei, Zhenan Sun

Vision-language pre-trained models have achieved impressive performance on various downstream tasks.However, their large model sizes hinder their utilization on platforms with limited computational resources.We find that directly using smaller pre-trained models and applying magnitude-based pruning on CLIP models leads to inflexibility and inferior performance.Recent efforts for VLP compression either adopt uni-modal compression metrics resulting in limited performance or involve costly mask-search processes with learnable masks.In this paper, we first propose the Module-wise Pruning Error (MoPE) metric, accurately assessing CLIP module importance by performance decline on cross-modal tasks.Using the MoPE metric, we introduce a unified pruning framework applicable to both pre-training and task-specific fine-tuning compression stages. For pre-training, MoPE-CLIP effectively leverages knowledge from the teacher model, significantly reducing pre-training costs while maintaining strong zero-shot capabilities.For fine-tuning, consecutive pruning from width to depth yields highly competitive task-specific models.Extensive experiments in two stages demonstrate the effectiveness of the MoPE metric, and MoPE-CLIP outperforms previous state-of-the-art VLP compression methods.

#8 HEAL-SWIN: A Vision Transformer On The Sphere [PDF[11]] [Copy] [Kimi[18]] [REL]

Authors: Oscar Carlsson, Jan E. Gerken, Hampus Linander, Heiner Spiess, Fredrik Ohlsson, Christoffer Petersson, Daniel Persson

High-resolution wide-angle fisheye images are becoming more and more important for robotics applications such as autonomous driving. However, using ordinary convolutional neural networks or vision transformers on this data is problematic due to projection and distortion losses introduced when projecting to a rectangular grid on the plane. We introduce the HEAL-SWIN transformer, which combines the highly uniform Hierarchical Equal Area iso-Latitude Pixelation (HEALPix) grid used in astrophysics and cosmology with the Hierarchical Shifted-Window (SWIN) transformer to yield an efficient and flexible model capable of training on high-resolution, distortion-free spherical data. In HEAL-SWIN, the nested structure of the HEALPix grid is used to perform the patching and windowing operations of the SWIN transformer, enabling the network to process spherical representations with minimal computational overhead. We demonstrate the superior performance of our model on both synthetic and real automotive datasets, as well as a selection of other image datasets, for semantic segmentation, depth regression and classification tasks. Our code will be made available.

Figure: import LLM generation

kimi exmaple

#3 Sparse Semi-DETR: Sparse Learnable Queries for Semi-Supervised Object Detection [PDF[17]] [Copy] [Kimi[23]] [REL]

Authors: Tahira Shehzadi, Khurram Azeem Hashmi, Didier Stricker, Muhammad Zeshan Afzal

In this paper, we address the limitations of the DETR-based semi-supervised object detection (SSOD) framework, particularly focusing on the challenges posed by the quality of object queries. In DETR-based SSOD, the one-to-one assignment strategy provides inaccurate pseudo-labels, while the one-to-many assignments strategy leads to overlapping predictions. These issues compromise training efficiency and degrade model performance, especially in detecting small or occluded objects. We introduce Sparse Semi-DETR, a novel transformer-based, end-to-end semi-supervised object detection solution to overcome these challenges. Sparse Semi-DETR incorporates a Query Refinement Module to enhance the quality of object queries, significantly improving detection capabilities for small and partially obscured objects. Additionally, we integrate a Reliable Pseudo-Label Filtering Module that selectively filters high-quality pseudo-labels, thereby enhancing detection accuracy and consistency. On the MS-COCO and Pascal VOC object detection benchmarks, Sparse Semi-DETR achieves a significant improvement over current state-of-the-art methods that highlight Sparse Semi-DETR's effectiveness in semi-supervised object detection, particularly in challenging scenarios involving small or partially obscured objects.

Q: 这篇论文试图解决什么问题?

A: 这篇论文试图解决基于DETR（Detection Transformer）的半监督目标检测（SSOD）框架中存在的一些限制，特别是与目标查询质量相关的挑战。具体来说，论文指出了以下几个问题:

1. **不准确的伪标签生成**: DETR-based SSOD中的一对一分配策略会产生不准确的伪标签，这会影响模型的学习效率。
2. **重叠预测**: 一对多分配策略会导致预测结果重叠，这会降低模型性能，尤其是在检测小的或部分遮挡的对象时。
3. **小目标和遮挡对象的检测性能差**: 现有的DETR-based SSOD方法在检测小目标或遮挡对象方面表现不佳，因为它们没有使用多尺度特征，如特征金字塔网络（FPN），这在基于CNN的SSOD方法中对识别小目标至关重要。

为了解决这些问题，论文提出了一种名为Sparse Semi-DETR的新型半监督目标检测解决方案。Sparse Semi-DETR通过引入两个主要模块来改进模型:

- **查询细化模块（Query Refinement Module）**：增强目标查询的质量，特别是对于小和部分遮挡的对象，显著提高了检测能力。
- **可靠伪标签过滤模块（Reliable Pseudo-Label Filtering Module）**：选择性地过滤高质量的伪标签，从而提高检测的准确性和一致性。

通过这些改进，Sparse Semi-DETR在MS-COCO和Pascal VOC目标检测基准测试中取得了显著的性能提升，特别是在涉及小或部分遮挡对象的复杂场景中。

Q: 有哪些相关研究?

A: 论文中提到了多个与半监督目标检测（SSOD）相关的研究工作，这些研究可以大致分为以下几个类别:

1. **基于伪标签的方法（Pseudo-Labeling）**：
   - STAC [39]: 引入了一个简单的多阶段SSOD训练方法，结合了伪标签和一致性训练。
   - Teacher-Student框架 [27]: 简化了伪标签的生成过程。
2. **基于一致性正则化的方法（Consistency-Based Regularization）**：
   - DSL [3]: 引入了自适应过滤、聚合教师和不确定性一致性正则化技术以提高泛化能力。

Figure: calling Kimi API