Video Dialog

Web Search and Data Mining

Project Guide

João Magalhães
(jmag@fct.unl.pt)

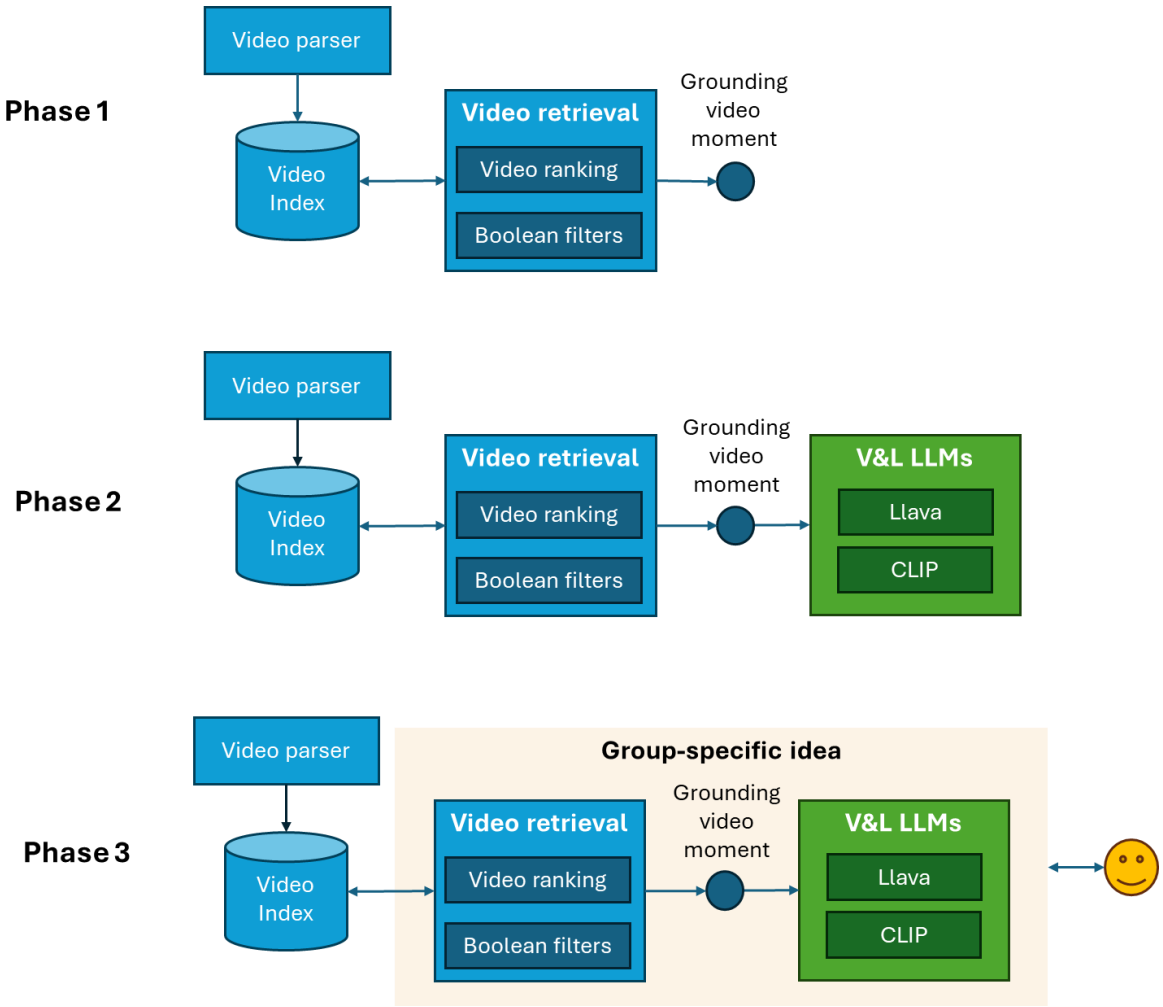
Universidade Nova de Lisboa
2024/2025

1 Introduction

In this project, students will create a pipeline for moment retrieval in videos using transformer-based architectures. The goal is to enable efficient video moment search with accurate results, enabling applications such as personalized finding the correct moment in a video, do question-answering about video content, and more. By leveraging transformer-based models like CLIP, LLava and Llama 3, you will develop a system that can efficiently retrieve video moments based on user input, allowing for complex cross-modal reasoning and inferences.

The project is divided into three parts:

- **Phase 1 (Understanding Embedding Spaces):** In this phase, you will create an index of video moments using Dual Encoders and OpenSearch. This will allow your system to retrieve relevant moments based on user input.
- **Phase 2 (Large Vision and Language Models):** In the second phase, you will integrate multimodal encoders and decoders to handle cross-modal queries and answer natural language questions.
- **Phase 3:** In the final phase, you will be able to select one of the proposed topics or to suggest your own topic.



2 Understanding Embedding Spaces (20%)

Following a natural conversation, your final prototype will be able to engage in a simple conversation to ground the dialogue on a particular video. Like every human, your prototype will have its own KB of the videos and moments.

In the first phase of the project, you will implement a Question-Answer search framework to answer Natural Questions. See the code examples provided during the laboratory classes.

After that you should play with Llama3 and create a prompt so the model acts like a MNLI where you intend to predict the categories of the moments.

2.1 Text-based Search

- Understand how to use the OpenSearch framework. Read the provided code and the documentation.
- Parse the HuggingFace dataset and create the OpenSearch index mappings for the recipe title and description.
- Index the dataset and test the search functionality.
- You can try to optimize the search results.

2.2 Embeddings Neighborhood

The embeddings space provides you with a mechanism to organize data by their semantic similarity. Dual encoders allow you to create semantic embedding representations of documents and queries. Follow the these steps to explore and understand how the embedding space:

- Revise the index mappings to support k-nn vectors.
- Compute the video text embeddings and store them in a file.
- Add the embeddings to your index and test the search functionality (don't forget that you also need to compute the query embeddings in real time).
- Use different queries to search OpenSearch and explore the neighborhood of those queries.

Discuss how the embeddings space organize data and allow for semantic search. Compare it to the text-based search.

2.3 Constrained Embedding Searches

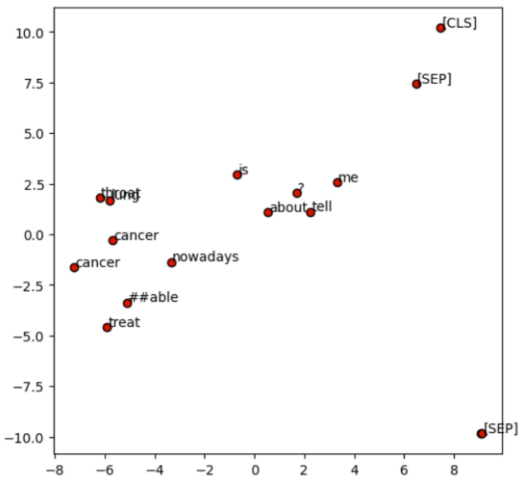
In this phase you should implement different search methods to support searching with natural language questions and search with filters. This is intended for videos with specific characteristics, e.g. actions, entities, length, number of moments, etc. Consider the following possibilities:

- Text based search, e.g., for keyword-based search;
- Embeddings based search; e.g., for semantic search;
- Boolean filters alone, e.g., search by entities;
- Search with boolean filters, e.g. a combination of search methods;

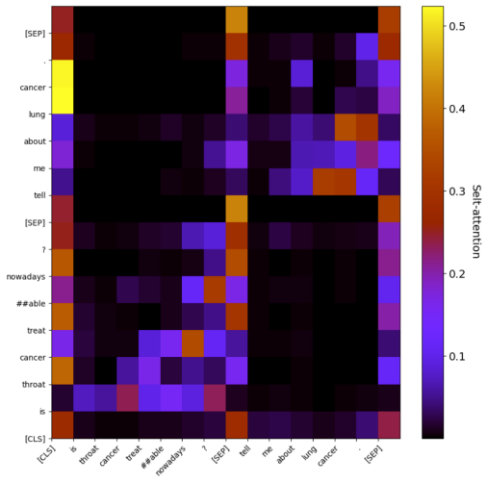
To support the above searches, you should create your own set of mappings to index the video's data. Test different fields and find a strategy to improve results. Using all fields can be sub-optimal and you should propose an optimal set of mappings that support your implementation functionalities.

2.4 Contextual Embeddings and Self-Attention

In this phase of the project you will show your understanding of the self-attention mechanism for vision-language web related tasks. You will also demonstrate your understanding of the Transformer architecture for different tasks.



Word embeddings visualization.



Single-head self-attention visualization.

You should discuss the following 5 points:

1. **Contextual embeddings.** Visualize the contextual word embeddings from layer 0 to layer 11. Observe how they change from layer to layer.
2. **Positional Embeddings.** Consider a text encoder. Insert a sequence of text with the same word repeated 20 times. Visualize the embeddings and the distance across all tokens. Discuss what you observe.
3. **Self-Attention.** Examine the self-attention mechanism of a transformer cross-encoder. Repeat with a dual encoder. Do critical analysis of your observations.
4. **Interpretability.** Compute the attention that each token receives on each layer and visualize. How is this related to the final output of the model?

To address the above questions, you should build on the provided tutorials to visualize word embeddings and self-attention matrices. See the two examples above.

2.5 Tips for Persistent Storage

- Some tasks take a long time to process, e.g. computing the embeddings, hence, you may wish to put the embeddings in some persistent storage.
- Python pickles allow you to (de)serialize objects: https://docs.python.org/3/library/pickle.html
- HDF5 or Pandas DataFrames provide you a more structured solution you can use.
- If you wish to store JSON objects, you can do so in the index, however, remember that whenever you delete the index you will lose this data.

3 Ideas for Group-specific Implementations (20%)

For the third phase you need to implement a part that is unique to your project as a group or individual. The proposed idea should be discussed with the lab instructor or selected from the following list of ideas:

Search/browsing:

1. Improved video search.
2. Video browsing by similarity.
3. Billion scale distributed search.

Conversational agent:

4. Dialog manager with controlled dialog state-tracking.
5. Feedback UI for semi-automatic video annotation.

V&L Explainability:

1. CLIP model explainability.
2. LVLM model explainability.

Literature reviews (min 6 pages, 20 references):

3. Energy and pollution concerns of large vision and language models.
4. Ethical and moral risks of large vision and language models.

Bring your own idea!!! Propose it until May 15.

4 Project Grading, Submissions and Report

The progress accomplished on each phase should be submitted and detailed in a report.

- Phase 1: April 11 15%
- Phase 2: May 12 20%
- Phase 3: June 13 25%

The report should be 12 pages length written during the three phases – annexes are not limited. A suggestion is to write 4 pages on each phase. When you submit the report of each phase, you are allowed to update the text of the previous phase. In exceptional circumstances you may exceed the 12 pages, with high-quality content.

The suggested template is available on Overleaf. The suggested structure is as follows:

1. Introduction (Phase 1,2,3)
2. Algorithms and Implementation (Phase 1,2,3)
   - 2.1. Search and Indexing
   - 2.2. Large Vision and Language Models
   - 2.3. Dialog Manager
3. Evaluation
   - 3.1. Dataset description (Phase 1, 2)
   - 3.2. Baselines (Phase 3)
   - 3.3. Results analysis (Phase 3)
4. Critical discussion (Phase 3)
5. References

5 Readings

The course site will be updated with readings and tips.

- SentenceBERT: https://arxiv.org/abs/1908.10084
- Facebook MNLI: https://huggingface.co/facebook/bart-large-mnli
- Llama 3: https://arxiv.org/abs/2407.21783
- Llava: https://llava-vl.github.io/