NII_UIT participation in AVS and DVU tracks of TRECVID 2023

Thuyen Tran Doan[1], Khiem Le[1], Thua Nguyen[1], Tien Do[1], TruongAn PhamNguyen[1], Tien-Dung Mai[1], Duy-Dinh Le[1], and Shin'ichi Satoh[2]

[1] University of Information Technology, VNU-HCMC, Vietnam
[2] National Institute of Informatics, Japan

Abstract. This report detail NII_UIT team participation in TRECVID2023 Ad-hoc Video search (AVS) and Deep Video under standing (DVU) task. On AVS task, we introduce a pipeline for effectively integrating multiple visual-embedding models and incorporates additional clues from the original query, such as word counts and object categories, to enhance the ranks of the final results. In the main task, our best performing run achieved a score of **0.166** xinfAP (ranks 7^{th}) in the Fully automatic (**F**) category and a score of **0.1876** xinfAP (ranks 2^{nd}) in the Manually-assisted (**M**) category respectively. As for DVU task, we try to employ latest off the shelf Deep learning approach to built a baseline results for new comer teams.

1 Introduction

NII_UIT teams has been long time participants in TRECVID for many years. Our works primarily including Instance search (INS) and Video summarization (VSUM, MSUM, etc...). Going along with merging and shifting of TRECVID tracks, this year we participated in Ad-hoc Video search (AVS) and Deep Video understanding (DVU) task.

2 Ad-hoc Video search

2.1 Our Approach

Firstly, visual-embedding methods, such as CLIP[RKH+21], SLIP[MKWX21], BLIP[LLXH22], ViFi-CLIP[RkM+23a], and others, facilitate the embedding of both natural language descriptions and images into a common high-dimensional vector space. By doing so, text-to-image retrieval systems become capable of receiving a text-based query and producing a ranked list of results that are closely aligned with the input query. As a result, our approach involves incorporating these methods into our system. However, by observing through experiments, we notice that the individual methods exhibit limited performance.

Secondly, many of the previous works, such as [FS93,LZT+10], have shown that the fusion process is effective in combining outputs across multiple systems. This can be explained due to the multiple effects as outlined in [VC99]. Therefore, we apply the fusion method to show the consensus among the retrieval results.

Thirdly, keywords in queries can help enhance the retrieval results, especially the object noun phrases and words that indicate numbers. In our system, we extract object categories and counts to filter out undesired results from text-to-image models by providing the exact number of object occurrences in that frame.

2.1.1 Overview of the system

Our system consists of three parts:

- The first part is the "Multimedia Database". This part is designed to store two main types of data: the "Frame Collection Database," which consists of N frames, and the storage of extracted frame-level features. To accomplish this, various models such as CLIP [RKH+21], BLIP [LLXH22], and ViFi-CLIP[RkM+23a] are used. The features are then stored and indexed in the "Vector Database" using FAISS [JDJ19]. Additionally, YOLO-NAS is employed to extract object location, counts, and categories, which are stored in the "General Database".
- The second part is the "Retrieval Part". Given a text query, the textual feature is extracted using the models mentioned above. FAISS is used to search these features in their respective "Vector Database efficiently." Finally, the top-k nearest similar frames are retrieved.

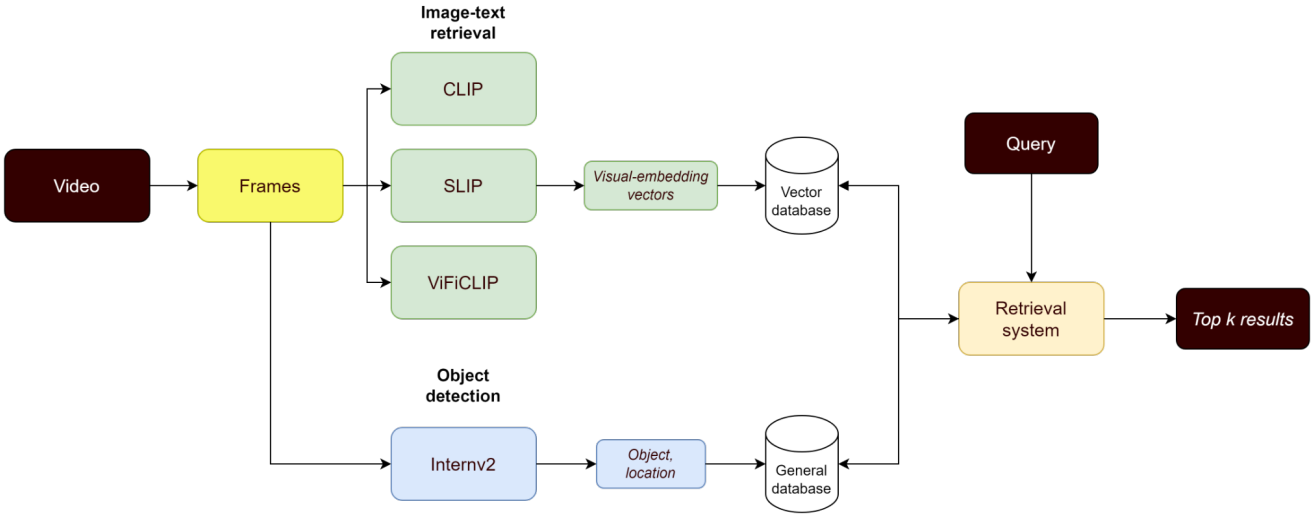

Fig. 1: Our summary framework for TRECVID Ad-hoc Video Search task 2023

2.1.2 Dataset The Trecvid 2023 dataset consists of 9760 Videos (V3C2) with a total duration of 1300 hours. The mean duration of each video is approximately 8 minutes. Additionally, the organizers have provided us with 1,425,454 shots, which were extracted from these 9760 videos using shot boundary detection techniques. However, we have noticed instances of shot overlap, leading to the possibility of repeated frame selection.

To address this issue, we have decided to re-implement the VIBRO[HSJB22] video preprocessing technique for frame selection. Through this re-implementation, we were able to obtain **3,349,659** unique frames, ensuring that frame duplication is effectively handled.

2.1.3 Text-to-Image retrievals In the image-text retrieval part of our system, we use these methods:

1. CLIP[RKH+21]: We utilize pretrained models of both transformer-based (L/14 and B/32) and CNN-based (RN50x16 and RN50x4) architectures. These models have dimensions of 768, 512, 512, and 512, respectively. OpenAI provides these models[3]. Additionally, we make use of pretrained models such as CLIP-L/14, trained on the DataComp dataset, and CLIP-H/14, trained on Laion2B[4].
2. BLIP[LLXH22]: We extract a 256-dimensional image-level feature using a ViT-B/16 model that is pretrained on 129M image-text pairs[5].
3. clip-bnl: We extract a 512-dimensional image-level feature using CLIP-bnl. This model is obtained by retraining CLIP (ViT-B/32) with bidirectional negation learning on the re-purposed MSR-VTT dataset[XMYR16]. It is provided by Wang et al[WCHL22][6].
4. SLIP[MKWX21]: We extract a 512-dimensional image-level feature using the ViT-S/16 variant of SLIP. This model is pretrained on a 15M subset of the YFCC100M dataset and then fine-tuned on Conceptual Captions 12M (CC12M)[7].

Moreover, considering that the original data consists of a series of frames, we find it meaningful to incorporate text-to-video methods. In light of this, we utilize two methods, namely XCLIP and ViFi-CLIP, to extract frame-level features.

1. XCLIP[NPC+22]: This method adapts the pretrained language-image models to video recognition. We extract a 768-dimensional video-level feature using with a pretrained model (ViT-L/14 CLIP model) on Kinetics-600[KCS+17] [8].
2. ViFi-CLIP[RkM+23b]: A simple baseline to diminish the gap from text-image retrieval to text-video retrieval. We extract a 512-dimensional video-level feature using a pretrained model (ViT-B/16 CLIP model) on Kinetics-400[9].

[3] https://github.com/openai/CLIP
[4] https://github.com/mlfoundations/datacomp
[5] https://github.com/salesforce/LAVIS
[6] https://github.com/ruc-aimc-lab/nT2VR
[7] https://github.com/facebookresearch/SLIP
[8] https://github.com/microsoft/VideoX/tree/master/X-CLIP
[9] https://github.com/muzairkhattak/ViFi-CLIP

For each text-image retrieval model, the queries are inputted and their textual features are extracted using their respective text encoder. These textual features are then subjected to a similarity search, utilizing the L2 Euclidean distance metric, within their corresponding image-feature indexed database. As a result of this process, a ranked list of the top K shot IDs is generated. In this particular setup, we have chosen K to be **7000**.

CLIP B/32	CLIP L/14	CLIP L/14 DataComp	CLIP H/14 Laion2B	CLIP RN50x16	CLIP RN50x4	CLIP-RN101	SLIP base	SLIP small	BLIP	CLIP-bnl	CLIP-finetuned	XCLIP	ViFi-CLIP
0.0659	0.0607	0.0793	0.0953	0.0688	0.0672	0.0603	0.0362	0.0396	0.0513	0.0864	0.0815	0.0268	0.0135

Table 1: This table show performance of each individually methods on Trecvid 2022.

2.1.4 Data Fusion In Information Retrieval (IR), numerous prior studies have demonstrated the benefits of data fusion in enhancing result quality. Fusion methods can be broadly classified into four categories: score-based methods, rank-based methods, probabilistic methods, and voting-based methods. However, for the purpose of this competition and based on experiments conducted on the Trecvid 2022 ground truth, we have selected two prominent fusion methods for each of their category: **CombMNZ**[FS93] (score-based) (see Tab. 2) and **PosFuse**[VC99] (probabilistic-based) (see Tab. 3).

It is evident from both tables that pretrained CLIP models from DataComp can significantly increase the xinfAP by 0.0079 (from 0.1626 to 0.1705).

Selected text-retrieval model(s)				
CLIP (B/32)	x			
CLIP (L/14)	x	x	x	x
CLIP (L/14) DataComp	x	x	x	
CLIP (RN50x16)	x			x
CLIP (RN50x4)		x	x	x
BLIP (B/16)	x	x	x	x
CLIP-bnl	x	x	x	x
CLIP-finetuned	x	x	x	x
XCLIP		x	x	x
ViFi-CLIP	x	x		x
Fusing result (xinfAP)	**0.1560**	**0.1547**	**0.1519**	**0.1498**

Table 2: Using the CombMNZ fusion method, the xinfAP scores on the Trecvid 2022 groundtruths are generated by combining the outputs of multiple text-retrieval models, with the selected models denoted by x.

Selected text-retrieval model(s)				
CLIP (L/14)	x	x	x	x
CLIP (L/14) DataComp	x			
CLIP (H/14) Laion2B	x			
CLIP (RN50x16)	x	x	x	x
CLIP (RN101)	x	x	x	x
SLIP (S/16)		x		x
BLIP (B/16)	x	x	x	x
CLIP-bnl	x	x	x	x
CLIP-finetuned	x	x	x	x
XCLIP	x	x	x	
ViFi-CLIP	x	x	x	x
Fusing result (xinfAP)	**0.1705**	**0.1626**	**0.1624**	**0.1622**

Table 3: Using the PosFuse fusion method, the xinfAP scores on the Trecvid 2022 groundtruths are generated by combining the outputs of multiple text-retrieval models, with the selected models denoted by x.

2.1.5 Reranking retrieval results using object counts We have observed that incorporating noun roots and count words in the queries can improve the retrieval results. To achieve this, we employ the YOLO-Nas[10] object detector to extract object categories (80 classes from MS-COCO) and their respective counts. Next, for each query, we utilize Spacy[11], a natural language processing library, to extract noun phrases and count words. These extracted phrases are then mapped to their corresponding 80 classes. If a noun phrase does not have a corresponding class, it is discarded. Additionally, in order to broaden the meaning of the original 80 classes, for each of these classes we also generate list of words which have roughly the same meaning. For example, the *person* class has roughly the meaning with [*man,woman,police,girl,adult,child,teacher,...*].

This allows us to apply a filter to the retrieved list obtained from the fusion step, refining the results based on the object categories and counts associated with the query.

2.1.6 Query refining For the manual track, we adopt a similar approach to VIREO[ACB+23] for the manual refinement of the input textual queries. You can find our set of 20 refined queries in Table 4, and the rationale behind these refinements is discussed in the Experiment section.

Query id	Original query	Final
731	A man is seen with a baby	a baby and a man
732	A woman with red hair	A woman with red hair
733	A golf course	A golf course
734	A recording studio	A recording studio
735	A toy vehicle	A toy vehicle
736	A person opens a door and enters a location	a man entering an opened door
737	A woman wearing (dark framed) glasses	A woman wearing (dark framed) glasses
738	A police officer wearing a helmet	A police officer wearing a helmet
739	Two or more persons are seen in front of a chain link fence	Many people in front of a chain link fence
740	A heavy man indoors	A overweight man indoors
741	A red or blue scarf around someone's neck	a person wearing red or blue scarf
742	A child climbs an object outdoors	A child climbs an object outdoors
743	A man is talking in a small window located in the lower corner of the screen	a man is talking nearby a window which is in the bottom of the frame
744	A person taking picture using a cell phone camera	A person taking picture using a smartphone
745	A person wearing gloves while biking	A person wearing gloves while riding a bicycle
746	A man riding a scooter	A man riding a scooter
747	At least two persons are working on their laptops together in the same room indoors.	Many people are working with their laptop together in a room
748	A man carrying a bag on one of his shoulders (excluding backbags)	A man with a bag on one shoulder
749	A person wearing any kind of face or head mask	A person wearing face mask or head mask
750	A man with an earring in his left ear	A man with an earring in his left ear

Table 4: This table show the original Trecvid 2023 queries and their respectively manually refined queries by our team.

2.2 Experiments

2.2.1 Impact of Manually refines queries Based on Tab. 5, it is evident that there is a significant rise in the total number of hits for each of our Manual runs compared to the Automatic runs. Specifically, at a cut-off level of 1000, we achieve an additional 392, 392, 372, and 372 hits, respectively. This increase in hits reflects in the final results as well, as they improve from the initial scores of 0.1660, 0.1605, 0.1640, and 0.1508 (refer to Fig. 2), to 0.1863, 0.1872, 0.1876, and 0.1885, respectively (refer to Fig. 3).

Significantly, query **1747** exhibits a substantial increase in the number of hits. Initially, the query was stated as "At least two persons are working on their laptops together in the same room indoors" but it was subsequently refined to "Many people are working with their laptops together in a room" (see Tab 4). We believe that the retrieval models misunderstood the phrase "at least two" in the original query. However, after refining it to "many" the models were able to capture more contextual information, leading to an increased number of relevant results (increase for 104, 104, 117 and 117 hits in the respective runs).

An additional instance can be observed with query **1740**, where we opted to replace the term "heavy" with "overweight" to better depict a physical attribute of a person. Consequently, this alteration led to an increase of 155, 155, 96, and 96 hits in the respective runs.

2.2.2 Submission results As the results of all techniques we present above (text-image retrievals, data fusion, objects counts for reranking and manually refining input queries), we summarize the results of our runs as follows:

[10] https://github.com/Deci-AI/super-gradients/blob/master/YOLONAS.md
[11] https://github.com/explosion/spaCy

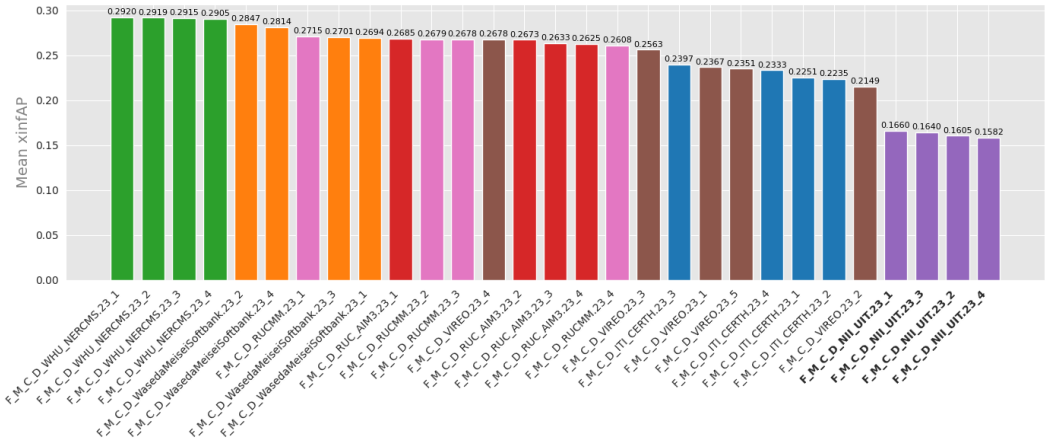

Fig. 2: The overall xinfAP score of the automatic runs in Trecvid Ad-hoc Video Search 2023. Our team's submissions are highlighted in bold text.

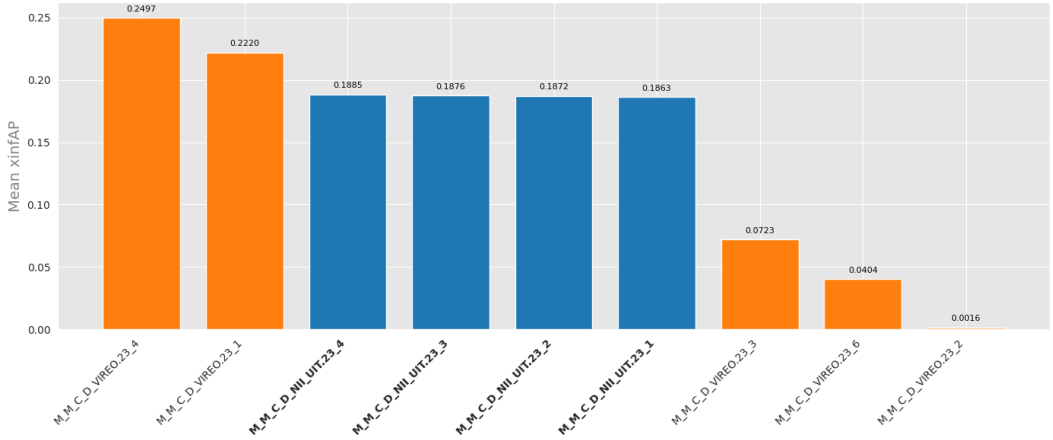

Fig. 3: The overall xinfAP score of the manual runs in Trecvid Ad-hoc Video Search 2023. Our team's submissions are highlighted in bold text.

– *F_M_C_D_NII_UIT.23_1*: This automatic run achieve xinfAP = **0.166** on the main task. It is the result of fusing results which are obtained from CLIP[RKH+21], BLIP[LLXH22] and ViFi-CLIP[RkM+23a] by their respective variants. We use CombMNZ [FS93] as the fusion method in this run.
– *F_M_C_D_NII_UIT.23_2*: Similar to run **1**, we also combine CLIP, BLIP, ViFi-CLIP and XCLIP by their variants. However, in this run, we use PosFuse[LZT+10] - a probabilistic method that learns to combine the ranks of multiple systems. We achieve the mean xinfAP = **0.1605** on the main task.

	Query id	**1731**	**1732**	**1733**	**1734**	**1735**	**1736**	**1737**	**1738**	**1739**	**1740**	**1741**	**1742**	**1743**	**1744**	**1745**	**1746**	**1747**	**1748**	**1749**	**1750**	**Sum of differences**
Run Manually.1 - Run Automatic.1	hit@10	3	0	2	-1	0	0	-2	-1	0	6	-2	-1	0	-2	0	1	0	-1	-3	1	0
	hit@30	2	-4	-4	2	-2	3	0	0	-3	12	-2	-4	0	-4	2	3	6	0	-1	1	7
	hit@100	1	0	-1	-1	-7	5	0	10	-12	32	-1	-2	0	-2	2	9	10	8	14	-2	63
	hit@1000	2	-7	0	13	-13	23	-17	17	-22	155	35	-5	0	21	-35	26	104	-2	100	-4	391
Run Manually.2 - Run Automatic.2	hit@10	3	0	2	-1	0	0	-2	-1	1	7	-2	-1	0	-2	3	1	0	-1	-2	1	6
	hit@30	0	-4	-4	2	-2	1	0	0	0	12	-2	-2	0	-3	5	3	12	0	-2	1	17
	hit@100	-1	0	-1	-1	-7	8	2	10	2	31	-1	-3	0	2	4	10	29	9	14	-3	104
	hit@1000	2	-7	0	13	-13	23	-17	17	-22	155	35	-5	0	21	-35	26	104	-2	100	-4	391
Run Manually.3 - Run Automatic.3	hit@10	-1	0	-1	0	2	-1	1	-1	2	5	-1	3	0	1	0	0	3	2	-1	0	13
	hit@30	-3	4	1	-1	2	-2	-2	-3	-5	15	-1	6	0	5	1	0	10	4	-1	-1	29
	hit@100	2	6	-4	-5	-5	3	-8	-7	-16	27	4	4	0	11	5	0	0	-4	11	2	26
	hit@1000	4	-6	-7	-24	-13	13	42	-15	-26	96	39	3	-1	17	-18	-5	117	14	124	18	372
Run Manually.4 - Run Automatic.4	hit@10	-1	0	-1	0	2	-1	1	-1	3	5	-1	2	0	1	2	0	7	2	-1	0	19
	hit@30	-1	4	1	-1	2	0	0	-3	2	15	-1	6	0	4	3	0	15	4	-1	-1	48
	hit@100	4	6	-4	-5	-5	7	-4	-7	1	28	5	2	0	13	9	1	27	-2	14	2	92
	hit@1000	4	-6	-7	-24	-13	13	42	-15	-26	96	39	3	-1	17	-18	-5	117	14	124	18	372

Table 5: The provided table illustrates the variation in the number of hits at different cutoff levels (10, 30, 100, 1000). The column headings in the table represent abbreviations for each submission run, where "Manually" corresponds to *M_M_C_D*, and "Automatic" corresponds to *F_M_C_D*.

- *F_M_C_D_NII_UIT.23_3*: This automatic run achieve the mean xinfAP = **0.164** on the main task. By extracting object counts and categories in the input query, we can re-ranking the results from run **1**.
- *F_M_C_D_NII_UIT.23_4*: Similar to run **3**, we apply reranking on the results of run **4**. This automatic run achieves the mean xinfAP = **0.1582** on the main task.
- *M_M_C_D_NII_UIT.23_1*: In this manual run, we firstly manually refine the queries, then the refined queries are input into the same models and fusion method from *F_M_C_D_NII_UIT.23_1*. The performance has significantly risen from **0.166** into **0.186**.
- *M_M_C_D_NII_UIT.23_2*: Similarly, this manual run has, the queries are also manually refined, models and fusion method settings from the setting of *F_M_C_D_NII_UIT.23_2*. The performance has also improve from **0.1605** into **0.1872**.
- *M_M_C_D_NII_UIT.23_3*: In this manual run, we manually refine the queries, and use the same models and fusion method in *F_M_C_D_NII_UIT.23_3*, and re-rank the retrieved results using object counts and categories. The performance has also improved from **0.1640** into **0.1876**.
- *M_M_C_D_NII_UIT.23_4*: In this manual run, similar setting to *M_M_C_D_NII_UIT.23_3*, manually refine the queries, same methods and fusion in *F_M_C_D_NII_UIT.23_4*, and re-rank the retrieved results using object counts and categories. The performance has also improved from **0.1582** into **0.1885**.

3 Deep Video Understanding

The goal of this track is to develop deep multimedia analyzing method for full-length cinematic movie (roughly 1.5 hours long) that can answer a variety of multiple choice questions about said move on both movie level and scene level. There were 5 movies in the test set and an ontology that would be used in both questions and answers. Additionally, the organizer provided with each movie the following data

- A list of scenes that was segmented manually based on movie semantic. This list is also used for the scene level questions.
- A list of entities of interest in that movie. The questions will be about those entities and each entity came with a name and 5-7 images of that entity in the movie. So far there were only 2 kinds of entites: person and location
- The computer generated transcript of the entire movie, produce by OpenAI's whisper tool.

There were a barrage of questions for each movie (well above a hundred questions each), divided into either movie level or scene level. For movie level questions there 2 types of queries:

- *Fill in the graph query* give a list of edges in a knowledge graph. These edges represent relations or interactions between an **Unknown** entity with other entities. The system will have to return a ranked list of best candidates for the unknown entity. To make the task even more challenging Some of the edges will have theirs target left as BLANK ¿
- *Question answering query* is the good old multiple choice in which there are 6 choices per question. Unlike other query types, both the question and the choice in this type of query can be free-form natural English statement with vocabulary span beyond the provided ontology for the dataset.

For scene level question, there were 4 types of queries, however, our team was only able to complete the last 2 types of queries that is:

- *Match scene to text query* give one sentence natural language description of an unknown scene along with the list of 10 scene numbers. The system have to choose which of those 10 scenes best fit the given description.
- *Scene sentiment analysis query* give one scene number and a list of 6 sentiment words. The system have to select the correct sentiment label for that scene. There were nearly a hundred of sentiment words used among all questions of this query types and some of the "words" could be more akin to a short phrase than a simple word, which added to the dificulty of these queries.

Due to the sheer complexity and diversity of these query types, we decided early on that no system could be robust enough to answer all query type, perhaps, each query type deserves its own answering method. Still, our hope for a perfect question answering system soon shattered when we notice that the test query for 2023 is very different from 2022: questions are more challenging, choices are more numerous and natural language are used prominently. With the limited time and resource at hand, we pivoted our strategy to took advantage of the provided transcript along with off-the-shelf pre-trained Large language model to build a strong baseline across the aforementioned query types. Though doing so could mean forgoing the 'deep' part in Deep video understanding, it's still useful in sowing the seed for our future participating in this task.

3.1 Our Approach

For each movie we do some facial search and background search to try to get a list of entities in each scene. These information will be used alongside Universal sentence encoder [CYK+18] narrowing down the choices for each questions and procure some answers.

3.1.1 Get a list of people entities appear in each scene To get a list of people entities appear in each scene, we do facial search for each frame of that scene against the images of each entity. The pipline to determine if one entity appear in a frame is as follow:

- We use MTCNN[ZZLQ16] to detect and crop out the facial images of each entity (about 5-7 faces for each entity).
- Then we use ArcFace[DGXZ19] to extract a deep feature vector for each facial images. Let's call this set of feature vector the "entity faces" set
- For each frame in a scene, we also use MTCNN and ArcFace to detect and extract the facial feature vector.
- Using cosine similarity we find 3 closest faces from "entity faces" set to each face in a frame. If all 3 closest faces belong to the same entity, we say that entity appears in the scene.

3.1.2 Get a list of location entities appear in each scene To do background search of each scene, we use the popular CNN model ResNet 50 [KK21] to extract a feature for each given images of each entities, we would got the set of "entity vectors". The for each frame in a scene we also extract its feature vector with ResNet 50 and find 3 closest vectors from "entity vectors" with the current frame vector. If all 3 of the closest vectors belong the the same entity, we say that entity appear in this scene.

3.1.3 Answering 'Fill in the graph' movie level query This query given a knowledge related to an Unknown entity and we have to returned a ranked list of candidate for that unknown. We assume that it is very unusual for an entity to have relations or interaction with itself, the Unknown entity is not among those entities appeared in the given graph, thus we got our list of candidates.

To rank this list of candidates, we simply count the number of scenes in which each candidates appeared alongside those entities mentioned in the given graph. We reason that candidates whom appeared together many times with each entities in the graph may have better chances of relating to or interacting with the given entities and should be rank higher.

3.1.4 Answering 'QA' movie level query We performed 2 runs in this query type. In run 1: we simply use the movie transcript as the context for the given natural language question. We use universal sentence encoder to match each given choices against the context and the question and return the choice with highest score.

In run 2: We find a list of entity mentioned in the question and answers. Then we find a list of scenes where at least one of those entities appear. The use the transcript of those scenes as context then match each choice against the question as in run 1.

3.1.5 Answering 'match scene to text' query Using Universal sentence encoder, we match the transcript of each scene in the choices against the textual description in the question and return the scene number of highest match.

3.1.6 Answering 'scene sentiment analysis' query Again we match the transcript of the given scene number in the question against each choices of 'sentiment word' and return the choice with highest match

3.2 Result and Discussion

For movie level, 'Fill in the graph' query, the result of our only run and a another team's last run is show in figure 4. It can be seen that our naive approach hold up a decent score. We tend to do well in movie with not many entity and these entity does not share many scene like 'Heart machine' and then fall flat on movie 'Little Rock' where there's a lots of entity and many of them share many scenes.

The movie level 'question answering' query is divided into 2 category: computer generated question from information in the ground truth knowledge graph and human generated question using free form natural language. As we

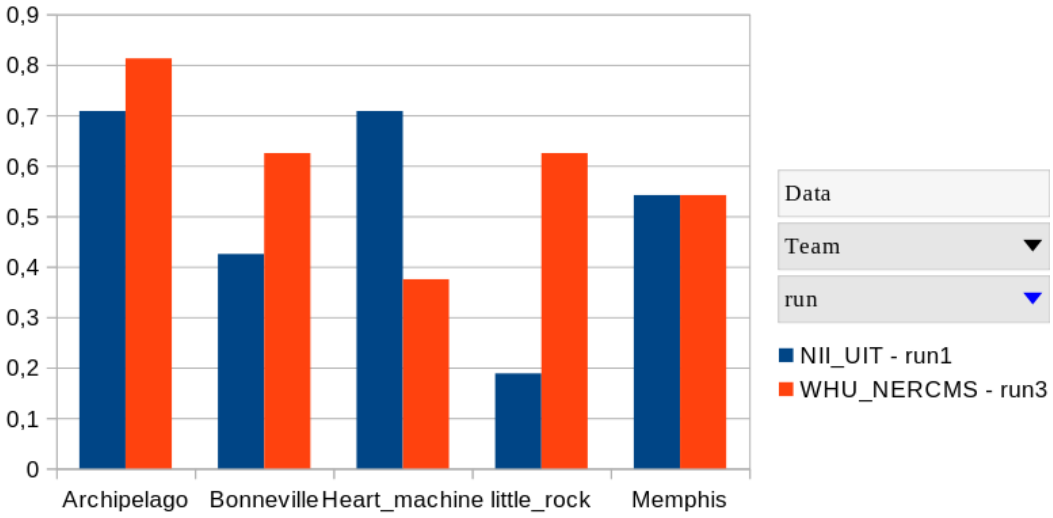

Fig. 4: Movie level query of type: 'Fill in the graph'

leverage Universal sentence encoder for question answering, we got some not-so-terrible score on natural language, human generated question as show in figure 5. Expectedly, when facing computer generated questions, our system perform attrociously, even got a score of absolute zero on the movie Archipelago as show in figure 6 Although we have employ a rudimentary facial recognition and location recognition techniques to find and filter out only scene we thought relevance to the query. The result shown only marginal different between filtering for scene and using all of the movie's transcript for context in Universal sentence encoder.

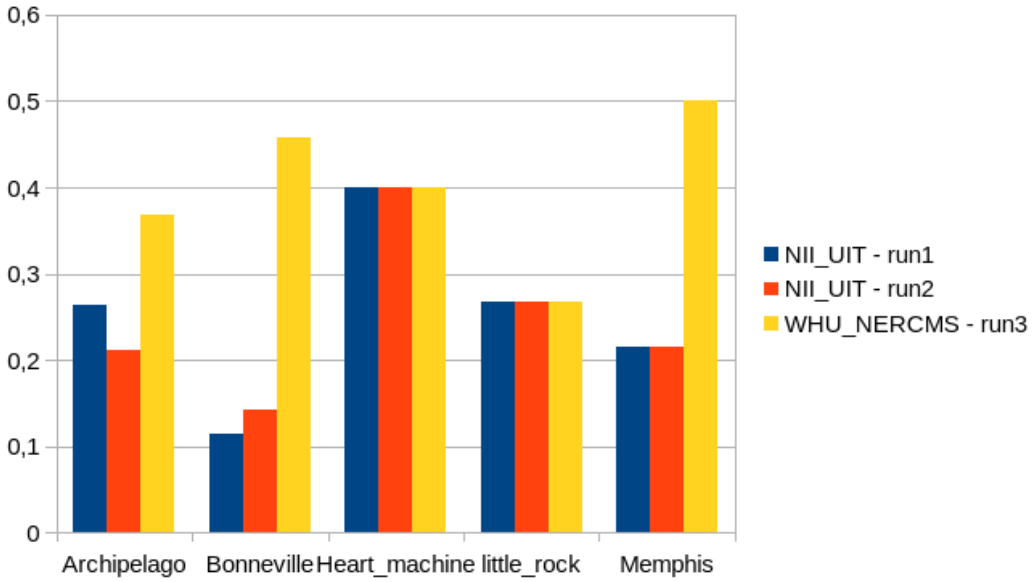

Fig. 5: Movie level query with human generated questions

The scene level query that we try out was a showcase of universal sentence encoder. The result shown in figure 7 and 8 attest to a very strong baseline with comparable performance on some movie.

4 Conclusion

In TRECVID 2023 AVS task, our methodology involved the utilization of various techniques: leveraging visual-embedding of text and image to optimize image retrieval, employed data fusion to effectively merge results from multiple text-image methods, and incorporated re-ranking techniques that incorporated object categories and counts extracted from additional information associated with the original query. Additionally, during the manual task, we made refinements to the input query based on careful observations and deductions drawn from both the query itself and the displayed search results. These combined efforts significantly contributed to the overall improvement of our ad-hoc video search system's effectiveness.

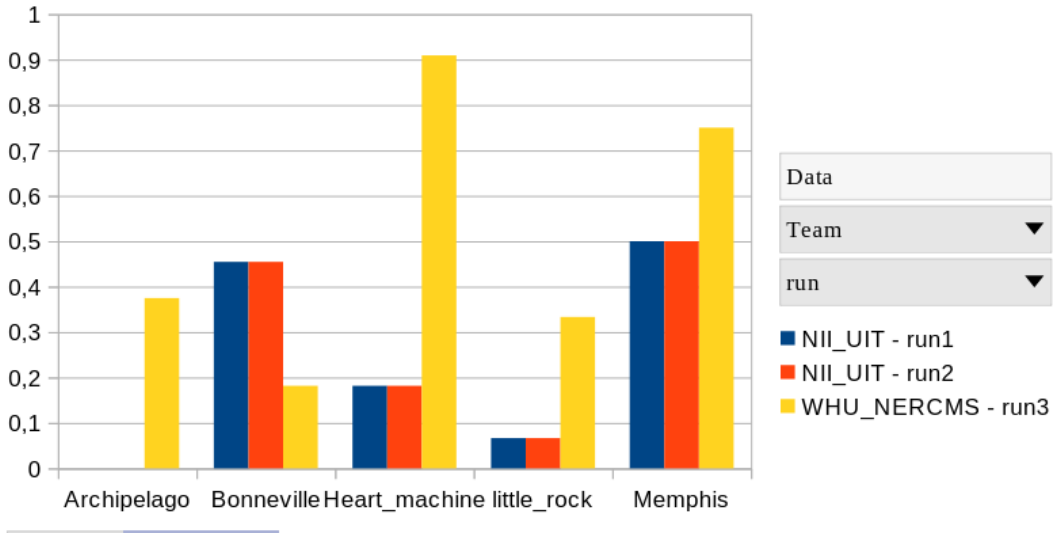

Fig. 6: Movie level query with computer generated questions

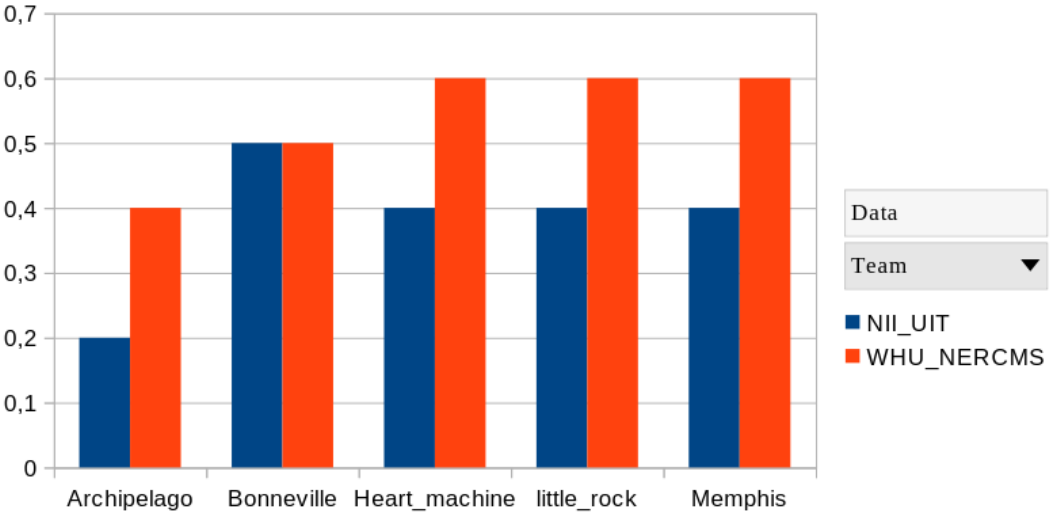

Fig. 7: Scene level, matching scene with description query

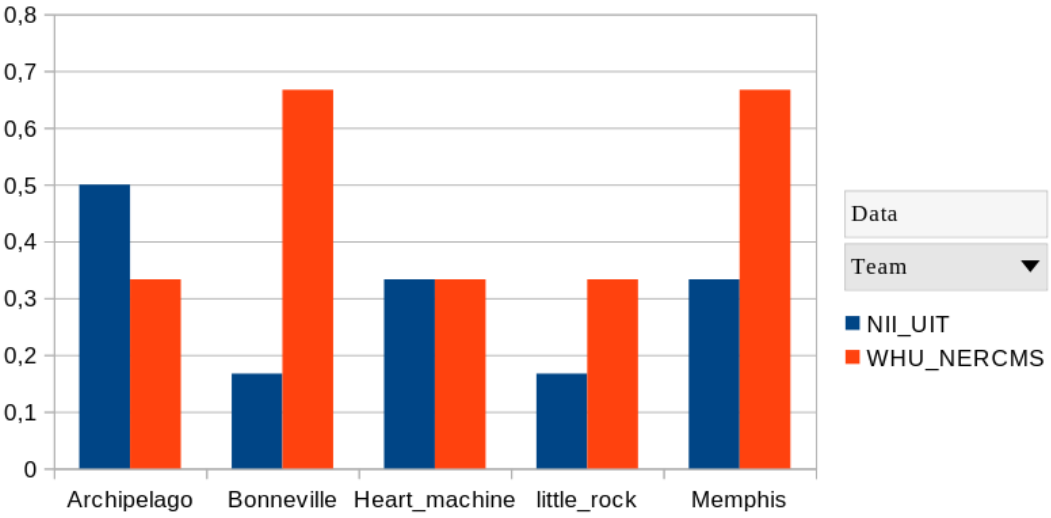

Fig. 8: Scene level, matching scene with description query

In DVU task, our performance was decisively beaten by better team. Yet, it can still serves as a strong baseline for new comer into this task. Moreover it's a testament of the robustness found in various deep learning model in multimedia especially with the help of Automatic Speech Recognition and large language model.

References

ACB+23. George Awad, Keith Curtis, Asad Butt, Jonathan Fiscus, Afzal Godil, Yooyoung Lee, Andrew Delgado, Eliot Godard, Lukas Diduch, Jeffrey Liu, et al. An overview on the evaluated video retrieval tasks at trecvid 2022. *arXiv preprint arXiv:2306.13118*, 2023.

CYK+18. Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

DGXZ19. Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019.

FS93. Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *TREC*, volume 500-215 of *NIST Special Publication*, pages 243–252. National Institute of Standards and Technology (NIST), 1993.

HSJB22. Nico Hezel, Konstantin Schall, Klaus Jung, and Kai Uwe Barthel. Efficient search and browsing of large-scale video collections with vibro. In *International Conference on Multimedia Modeling*, pages 487–492. Springer, 2022.

JDJ19. Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

KCS+17. Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

KK21. Brett Koonce and Brett Koonce. Resnet 50. *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pages 63–72, 2021.

LLXH22. Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

LZT+10. David Lillis, Lusheng Zhang, Fergus Toolan, Rem W Collier, David Leonard, and John Dunnion. Estimating probabilities for effective data fusion. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 347–354, 2010.

MKWX21. Norman Mu, Alexander Kirillov, David Wagner, and Saining Xie. Slip: Self-supervision meets language-image pre-training. *arXiv preprint arXiv:2112.12750*, 2021.

NPC+22. Bolin Ni, Houwen Peng, Minghao Chen, Songyang Zhang, Gaofeng Meng, Jianlong Fu, Shiming Xiang, and Haibin Ling. Expanding language-image pretrained models for general video recognition. 2022.

RKH+21. Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

RkM+23a. Hanoona Rasheed, Muhammad Uzair khattak, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Finetuned clip models are efficient video learners. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

RkM+23b. Hanoona Rasheed, Muhammad Uzair khattak, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Finetuned clip models are efficient video learners. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

VC99. Christopher C Vogt and Garrison W Cottrell. Fusion via a linear combination of scores. *Information retrieval*, 1(3):151–173, 1999.

WCHL22. Ziyue Wang, Aozhu Chen, Fan Hu, and Xirong Li. Learn to understand negation in video retrieval. In *ACMMM*, 2022.

XMYR16. Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016.

ZZLQ16. Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE signal processing letters*, 23(10):1499–1503, 2016.