WHU-NERCMS AT TRECVID 2023: AD-HOC VEDIO SEARCH (AVS) AND DEEP VIDEO UNDERSTANDING (DVU) TASKS

Jiangshan He[*,1] **, Ruizhe Li**[*,2] **, Jiahao Guo**[*,2]**, Hong Zhang**[1] **, Mingxi Li**[2] **, Zhengqian Wu**[1,2] **,**
Zhongyuan Wang[‡] **, Bo Du**[‡]**, Chao Liang**[‡,†]

Hubei Key Laboratory of Multimedia and Network Communication Engineering

National Engineering Research Center for Multimedia Software

School of Computer Science, Wuhan University

cliang@whu.edu.cn

ABSTRACT

The WHU-NERCMS team participated in the Ad-hoc Vedio Search (AVS) and Deep Video Understanding (DVU) tasks at TRECVID 2023. For AVS task, we chose to utilize embedding models and incorporated some of the successful practices from the previous teams. For automatic runs, we applied classic Language-Image pre-training models CLIP and its various variants: SLIP, BLIP, BLIP-2, LaCLIP. Besides, we also applied the Diffusion model to turn the text query into abundant generated pictures to attain the so-called "mean image query".All models used any other training data with any annotation except V3C2. For those using feedback runs, we used Top-K Feedback and a modified algorithm Quantum-Inspired Interactive Ranking Aggregation (QI-IRA)[1] that adjusts models' weight with relevance feedback based on our finding published in AAAI'24. The official evaluations[2] show that the proposed strategies rank 1st in both automatic and interactive tracks. For DVU task, we propose a four-stage method to solve the task. Of these four stages, two of them we consider very important. First, the shot based instance search allows accurate identification and tracking of characters at an appropriate video granularity. Second, using LLM in QA can help us answer the questions by giving background knowledge. We rank first in all four groups in this year's DVU task.

[*]indicates equal contributions.
[‡]indicates instructors.
[†]indicates corresponding author.
[1]indicates members attending Ad-hoc Video Search (AVS) task.
[2]indicates members attending Deep Video Understanding (DVU) task.

1 AVS Task

1.1 Introduction

The Ad-hoc search task necessitates participants to appropriately model a user's text query to search for video shots that align with the textual description. Each query can return a maximum of 1000 shot IDs. This year's dataset remains consistent with the previous year, being V3C2[3], which comprises 9760 videos with a total duration of 1300 hours. The number of queries has reduced from 30 last year to 20 this year. Our approach involves weighting the similarity under different Language-Image models between textual queries and keyframe images from the dataset to obtain automatic results. Additionally, this year, we have incorporated a new interaction method to enhance the quality of automatic results. Our framework is shown in Fig.1.

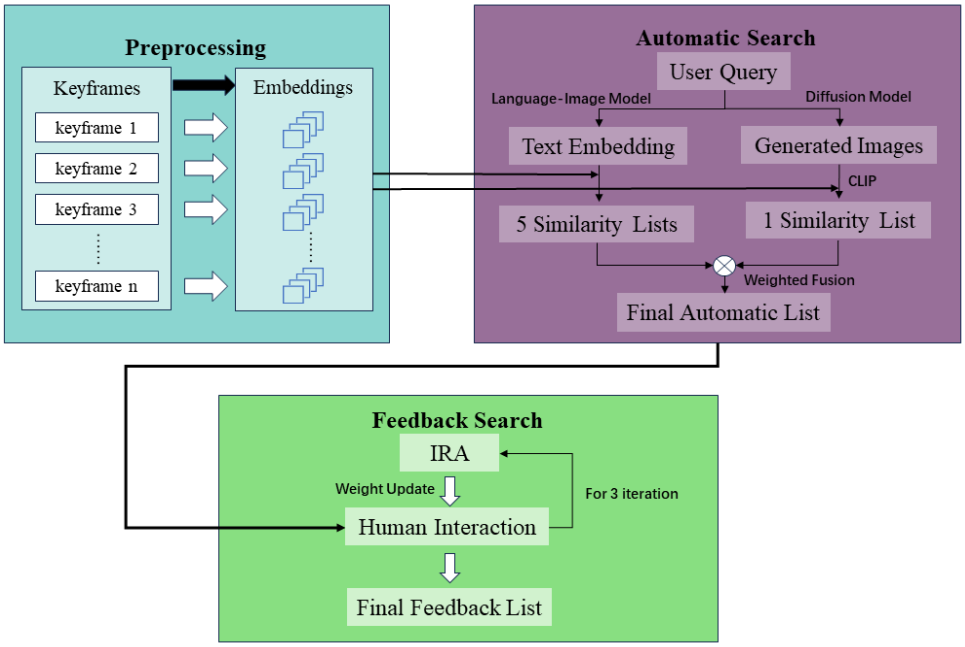

Figure 1: Our framework

1.2 Our Method

We used the following Language-Image pre-trained models to construct our system.

1.2.1 Embedding Models

CLIP*[4]: RN50x4, RN50x16, RN50x64, RN50, RN101, ViT-B/16, ViT-B/32, ViT-L/14.

SLIP†[5]: ViT-Small, ViT-Base, ViT-Large, ViT-Base(CC3M), ViT-Base(CC12M).

*https://github.com/openai/CLIP
†https://github.com/facebookresearch/SLIP

BLIP[‡][6]: ViT-B(COCO), ViT-B(Flickr30k), ViT-L(COCO), ViT-L(Flickr30k).

BLIP-2[§][7]: BLIP2(COCO).

LaCLIP[*][8]: LaCLIP(CC12M).

In the previous year's competition, CLIP received extensive utilization. Consequently, we chose it and its latest variants to enhance the system's ability to match text and images. Similarly, to augment its text understanding capabilities, we introduced LaCLIP, which underwent retraining using the extended corpus from LLM.

For all the mentioned models, we extract image vectors for each keyframe in the dataset and store them in a database for retrieval. When a user inputs a text query, we then extract the corresponding text vectors and calculate similarity with the saved image features of the corresponding model. For different pre-trained models of the same type, we add up the similarity scores for the same keyframes with equal weights and normalize them. This can be represented by the formula:

$$s_i^{id} = norm(\sum_{p \in M} s_p^{id}) \tag{1}$$

Here, "id" represents the shot id corresponding to the keyframe, "i" denotes the i th type of model, and "p" represents different pre-trained models of the same type. After this computation, we obtain five lists of similarity scores.

Furthermore, we referred to the practices of last year's team[9]. Based on textual input, we employed the Diffusion model[¶] to generate 1000 images and utilized CLIP's Vit-B/32 model to extract image vectors. For each keyframe image, its text vector needs to be summed and normalized after calculating similarity with the image vectors of these 1000 images. Here, we can view this operation as generating a "mean image query", effectively transforming cross-modal tasks into comparisons within the image modality. In our testing, this method performed even better in certain queries where CLIP underperformed. With the Diffusion model, we obtained another list of similarity scores.

1.2.2 Ranking Aggregation

With the mentioned cross-modal models, we acquire multiple score lists s_i, and the final score of one keyframe in this query can be defined as:

$$s^{id} = \sum w_i s_i^{id} \tag{2}$$

which means we need to select an appropriate weight w_i, to achieve a relatively optimal result. Typically we set the respective weights heuristically. Based on the practices of previous years' teams and our experimental data in TRECVID AVS 2022, our model parameters in automatic runs are set as shown in Table 4.

1.2.3 Relevance Feedback

In the Relevance-Feedback (R) runs, we fit the weights according to the users' feedback. In the following part, we will introduce a modified algorithm based on Quantum-Inspired Interactive Ranking Aggregation (QI-IRA)[1] via users' feedback on the relevance of the results.

[‡]https://github.com/salesforce/BLIP
[§]https://github.com/salesforce/LAVIS
[*]https://github.com/LijieFan/LaCLIP
[¶]https://huggingface.co/runwayml/stable-diffusion-v1-5

A document can be determined by the relevance and non-relevance of a query in the quantum-inspired approaches to information retrieval[10].

$$|d\rangle = \alpha|r\rangle + \tilde{\alpha}|\neg r\rangle \tag{3}$$

where $|d\rangle$ is the state of the document to be retrieved, $|r\rangle$ is the relevant state of the query, $|\neg r\rangle$ is the complementary state of $|r\rangle$, α and β are the coefficients that subject to $|\alpha|^2 + |\tilde{\alpha}|^2 = 1$.Hence,

$$||\langle r|\, d\rangle||^2 = |\alpha|^2 = s \tag{4}$$

represents the similarity between the document and the query.

Given a specific query and the weights set heuristically, we can get an initial ranking result. After one round of the user's interaction, there are the images set with positive feedback Φ_+ and the set with negative feedback Φ_-.

To ensure that as many positive feedback images as possible appear in the Top 30 results, we want to decrease the weight of models that include negative feedback and increase the weight of models that include positive feedback. Considering both positive and negative feedback, in the perspective of equation (3), a new weight $\tilde{w_i}$ of the i th method can be defined as:

$$\begin{aligned}
\tilde{w_i} &= \frac{1}{|\Phi_+|}\sum_{d_j\in\Phi_+} ||\langle d_j|r\rangle||^2 + \frac{1}{|\Phi_-|}\sum_{d_k\in\Phi_-} ||\langle d_k|\neg r\rangle||^2 \\
&= \frac{1}{|\Phi_+|}\sum_{d_j\in\Phi_+} s_{i,d_j} + \frac{1}{|\Phi_-|}\sum_{d_k\in\Phi_-} \tilde{s}_{i,d_k} \\
&= \frac{1}{|\Phi_+|}\sum_{d_j\in\Phi_+} s_{i,d_j} + \frac{1}{|\Phi_-|}\sum_{d_k\in\Phi_-} 1 - s_{i,d_k} \\
&= \frac{1}{|\Phi_+|}\sum_{d_j\in\Phi_+} s_{i,d_j} - \frac{1}{|\Phi_-|}\sum_{d_k\in\Phi_-} s_{i,d_k} + 1
\end{aligned} \tag{5}$$

where d_j is images marked as positive feedback,s_{i,d_j} is the similarity of the image to the query by the i-th model.d_k and s_{i,d_k} are the same.

For the sake of computation simplicity, we can ignore the trailing constants since we only need the change in weight trends. The new weights can be expressed in the following form:

$$\tilde{w_i} = \frac{1}{|\Phi_+|}\sum_{d\in\Phi_+} s_{i,d} - \frac{1}{|\Phi_-|}\sum_{d\in\Phi_-} s_{i,d} \tag{6}$$

To get a smooth update of the weight as well as avoid the updated weights becoming 0, we adopt the weighted average in our experiments:

$$w_i = \alpha\tilde{w_i} + (1-\alpha)w_i, \alpha = 0.9 \tag{7}$$

In interactive submissions, we can conduct a maximum of three rounds, with interactions involving the top 30 items in each round. To maximize interactions, we promote the shots marked as positive feedback to the top of the list and place the shots marked as negative feedback at the end. This way, the same keyframes won't appear in each round, avoiding repetition.

The above feedback interaction can be repeated by iteration.

1.3 Analysis

1.3.1 Model Selection For Fusion

Our performance mainly relies on the fusion of multiple embedding models to achieve superior generalization than a single model. Specific experimental data is shown in Table 1. In model selection, we believe that:

- The more diverse the types of models, the better the results after fusion.
- For models of the same kind, the fewer models that perform poorly, the better the results. A simple example is shown in Table 2.

This is similar to the concept of ensemble learning.

Table 1: Performance of different models and the fused result in TRECVID 2022 AVS

Model Types	Pre-trained type	infAP		
CLIP	RN50x4	0.0873	0.1603	0.2636
	RN50x16	0.0851		
	RN50x64	0.0801		
	RN50	0.0711		
	RN101	0.0833		
	ViT-B/16	0.0949		
	ViT-B/32	0.0864		
	ViT-L/14	0.0797		
SLIP	ViT-Base(CC3M)	0.0369	0.1286	
	ViT-Base(CC12M)	0.0798		
	Vit-Base	0.0584		
	Vit-Small	0.07		
	Vit-Large	0.0569		
BLIP	ViT-Base	0.0745	-	
	ViT-Large	0.0769		
	ViT-B(COCO)	0.1333	0.1857	
	ViT-B(Flickr30k)	0.1293		
	ViT-L(Flickr30k)	0.1447		
	ViT-L(COCO)	0.1623		
LaCLIP	LaCLIP(CC3M)	0.0427	-	
	LaCLIP(CC12M)	0.0931	0.0931	
BLIP2	BLIP2(COCO)	0.1585	0.1585	
Stable-diffusion	v1-5	0.0788	0.0788	

Table 2: Performance comparison of different fusion combinations in TRECVID 2022 AVS

Model Type	Pre-trained type	infAP	infAP after fusion		
BLIP	Base	0.0745	0.149	-	0.1724
	Large	0.0769		-	
	Base-coco	0.1333		**0.1857**	
	Base-flickr	0.1293			
	large-flickr	0.1447	-		
	large-coco	0.1623	-		

1.3.2 The Same Modality

Taking inspiration from last year's team, we found that introducing the diffusion model into retrieval slightly improved performance. Generating a substantial number of images that match the query can be seen as human imagination of what might appear in the dataset. Combined with reliable embedding models, we can get the similarity between the two images.

While, in general, its performance is on par with different types of pre-trained models in our experiments, we observed that the diffusion model has advantages in certain queries, such as tables. For instance, in last year's tasks related to 'a construction site' and 'farm equipment,' its performance significantly outperformed other pre-trained models, even the fused model in Table 3. We attribute this to the fact that the diffusion model's output is closer to real-world scenarios for these specific queries.

Table 3: Some queries Diffusion model performs better

Model types	infAP	query
Fusion-SLIP	0.5247	703 A construction site
Fusion-CLIP	0.5307	
Diffusion	**0.5902**	
Fusion-SLIP	0.0104	708 A female person bending downwards
Fusion-CLIP	0.2031	
Diffusion	**0.2223**	
Fusion-SLIP	0.0925	719 A piece of heavy farm equipment or machine seen outdoors
Fusion-CLIP	0.1613	
Diffusion	**0.2934**	
Fusion-SLIP	0.0109	728 Two adults are seated in a flying paraglider in the air
Fusion-CLIP	0.0373	
Diffusion	**0.083**	

Table 4: Result of each run

Type	**Run ID**	**Relation**	**infAP**	**Weight(C:S:B:B2:L:D)**
Automatic	F_1	–	0.292	10:3:16:4:3:3
	F_2	–	**0.292**	10:3:16:4:3:6
	F_3	–	0.291	10:3:20:4:3:3
	F_4	–	0.290	13:3:16:4:3:3
Interactive	R_1	F_1 + QI-IRA	0.299	–
	R_2	F_2 + QI-IRA	0.298	–
	R_3	F_3 + QI-IRA	**0.299**	–
	R_4	F_4 + QI-IRA	0.296	–

Table 5: Model of the abbreviation in our method Introduction

Abbreviation	**Description**	**Abbreviation**	**Description**
C	CLIP	S	SLIP
B	BLIP	B2	BLIP-2
L	LaCLIP	D	Diffusion

1.3.3 Interactive Algorithm

Our interactive algorithm has been effective in this year's tasks, but the gains have been relatively modest in Table 4. We believe the main reasons are the following:

- When interacting with the top 30 results in our list, for queries that are either too easy or too challenging, we might get into a condition where all responses are correct or all are incorrect. In such cases, it is limited to distinguish the model's ability.

- our interactions can only gather simple information and cannot access complex semantic information. To illustrate with a simple example, for a query like 'A man wearing black clothes', when an image of 'A woman wearing black clothes' appears, according to the algorithm's principles, we would label it as negative feedback and update the weights based on the image's similarity across different models. However, in reality, we should guide the model to understand that finding 'black clothes' is correct, but finding 'a woman' is incorrect. For models proficient at identifying 'black clothes', their weights will decrease rapidly. If we could provide them with corrected query information, it would undoubtedly be more advantageous. This could potentially be integrated with the concept bank method.

1.4 Conclusion

In this paper, we employed a method based on Ranking Aggregation to fuse similarity lists from various types of embedding models. Besides, we introduced a modified algorithm based on Quantum-Inspired Interactive Ranking Aggregation(QI-IRA) that incorporates user feedback, enhancing performance further by adjusting fusion weights. The experimental results for the TRECVID 2023 AVS task demonstrate that our model exhibits strong performance.

2 DVU Task

2.1 Introduction

Deep Video Understanding (DVU) is a difficult task which requires researchers to develop a deep analysis and understanding of the relationships between different entities in the video, to use known information to reason about other, and to populate a knowledge graph (KG) with all acquired information. The aim of the task is to push the limits of multimedia analysis techniques to address long duration videos holistically and extract useful knowledge to utilize it in solving different kinds of queries.

The DVU task tries to interpret a film from two levels: (1)scene-level: interactions between entities; (2)movie-level: relationships between entities. Instance identification and tracking is a very fundamental and important part of the solution to the different approaches to DVU. This is because many subsequent sessions are dependent on accurate instance search. For example, during the Video Question & Answer (VQA) phase, a large number of questions focus on entities, like **"What is the relationship between entity A to entity B?"**. So you need to identify A and B precisely before you can proceed to the next process. Based on the above analysis, we believe that accurate identification of entities is the basis for problem solving. And in film, the core of the entity is person, because the narrative of the film is usually a human-centered story. So it is very crucial to do a good job of character identification and tracking.

Existing works can be generally categorized into two groups according to the smallest unit for identification and tracking, one for long duration and one for short duration.

For the long duration group, [11, 12] perform character recognition and tracking under the scene. But doing so ignores an important feature of the movie, which is the shot. Unlike traditional face recognition and tracking under surveillance cameras, in which the scene is roughly the same, this aspect in movie encounters many problems. Because the film directors often use different camera language (express emotion or narrative through camera switching, advancement, and movement) to develop the story. But the switching camera makes it difficult to track the correct characters. For example, in the film often appear in the scene of two people talking face to face, the common camera switch is from a person's face close-up switch to another, and in the switch before and after the two people in basically the same position, in this case, if you do not consider the camera switch and directly track the frame of the series, it will be easy to make two different tracks in one trajectory.

For the short duration group, [13] samples every ten frames in the scene, and face detection and recognition in each frame, and by complementing the tracking track with the face track to obtain the complete entity track information. But this also does not take into account the misalignment of the trajectory brought about by the camera switch. And in shorter clips for identification and tracking, the character is likely to turn his back to the camera throughout the video, and we cannot discriminate the identity information of the tracking track through the face.

After considering the above two problems, we propose a character recognition tracking method for prestage of deep video understanding;, which we call shot based instance search. In this method, we slice the scene into shots and perform face detection, recognition and entity tracking under the shots frame-by-frame. Then we determine the identity of the tracking by calculating the overlap between the face box and the tracking box, finally we vote in the same tracking track to get the identity of the whole track. In this way can we avoid the identity mismatch caused by camera switching, and also ensure through the voting mechanism that we can determine the identity information of the whole tracking sequence as long as the character has appeared facial information in one frame, even if the character turns around after.

In addition, due to the powerful understanding of text recently demonstrated by LLM, we believe it is also very helpful for answering DVU questions, so we introduced LLM into the QA phase of our model. In this phase, LLM assists us in three ways: firstly, we find that the class of entities has a great impact on the selection of relationship questions, so we use LLM to classify all relationships, and secondly, to take full advantage of LLM's textual interpretation capabilities, we propose a method called shot-scene summary to obtain a scene description and let LLM use this text as the basis for answering part of the scene-level questions and rank average it with the answers we obtained through the knowledge graph. Lastly, By inputting the coarse-grained summaries and subtitles acquired from ASR into LLM, we obtain fine-grained summaries. These fine-grained summaries can serve as background prompt for LLM, enhancing its overall performance.

In general, our contribution can be summarized as follows:

- propose a character recognition and tracking method for movies, which can help us avoid tracking errors due to camera switching and achieve accurate instance search;

- introduce LLM in QA phase. We use LLM to help us classify different entity types and make multiple choice based on Shot-Scene Summary.

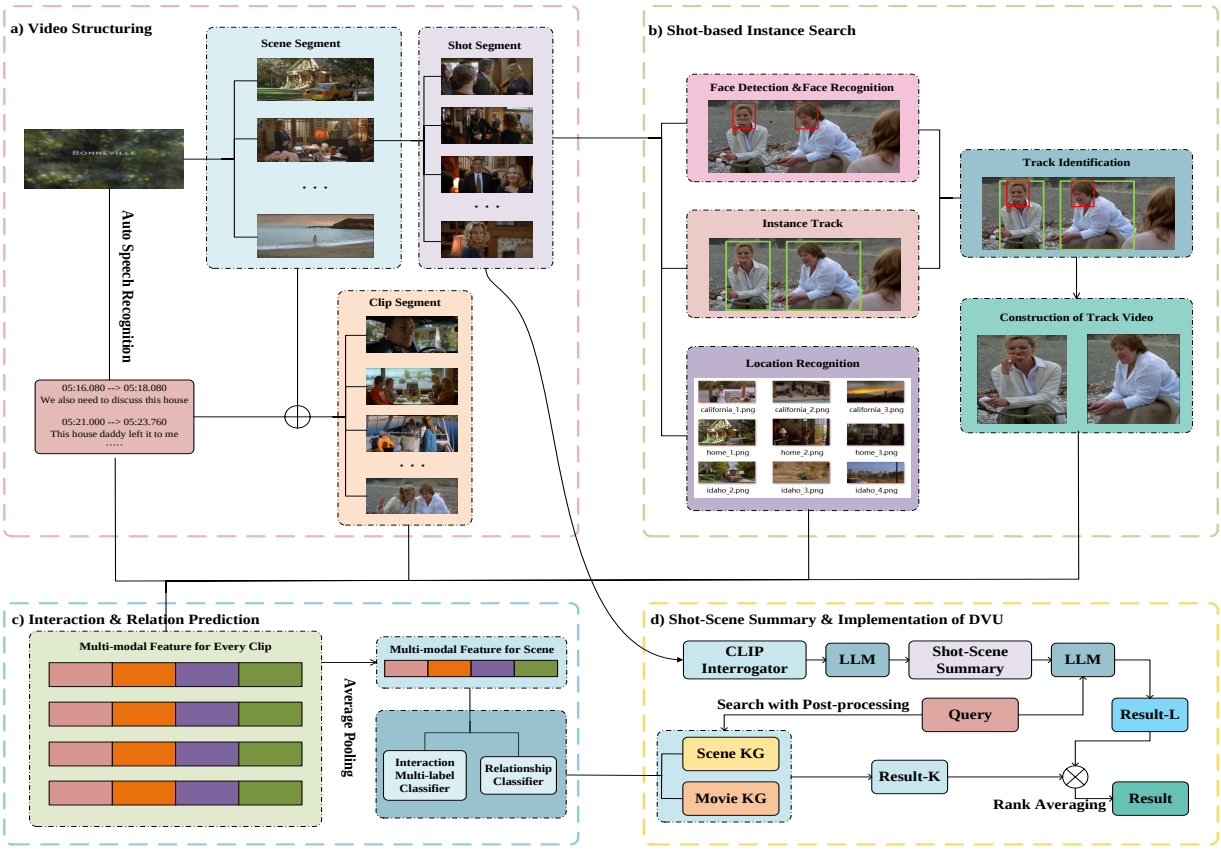

Figure 2: Our four-stage framework

To test the effectiveness of our method, we use dataset and query from NIST TRECVID 2023 DVU task and get good performance.

2.2 Our Method

In this section we introduce our four-stage method stage by stage, and our framework overview is shown in Fig. 2. First, we will slice the source video at multiple levels, including, scene, shot and clip, and then we perform character recognition, tracking and location recognition on the shot. We then perform multimodal feature extraction, including the overall video, the tracking box video, and the line text aligned with the video. Finally we train the classifiers to build the knowledge graph and introduce the large language model to help us answer the relevant questions.

2.2.1 Video Structuring

The first step is video structuring. For each movie, we use Youtube ASR API[‖] to generate the subtitles. The Video Segmentation contains three parts, scene segmentation, shot segmentation and clip segmentation. For scene segmentation, we segment the movie using officially provided time stamps. For shot segmentation, we use PySceneDetect[**] to segment the scene videos to facilitate the shot based instance search and shot-scene summary. For clip segmentation,

[‖]https://github.com/jdepoix/youtube-transcript-api
[**]https://github.com/Breakthrough/PySceneDetect.git

based on the subtitle file, the start and end time of each clip are obtained, and the corresponding scene to which clip belongs is located by combining the clip segmentation with the scene segmentation. For the clips which are between two segmented scenes, we add them to the scene which has a longer coincident video. Through this way can we generate the mapping table from scene to clip and the unit for the next process.

2.2.2 Shot-based Instance Search

The second step is the Shot-based Instance Search, which consists of Person Recognition and Track, Construction of Track Video and Location Recognition. By these, we can obtain haracter information and scene information for subsequent processing.

Person Recognition and Track. For characters in shot, we run face detection using SCRFD [14] frame by frame to get bounding box and we use ArcFace [15] to extract the face features of snapshot and detected face. We average the different image features of the same person for identifying. Then we calculate the similarity between face library and datected face and save the similarity matrix to confirm the identity of the detected face. After face recognition, we run person track using faster RCNN [16] and Deepsort [17] on the shot-level to avoid tracking errors due to camera switching. Then we calculate the overlap between the detected face box and the tracking box. If the overlap exceeds the threshold, we assign face identity to the tracking frame. The identity information of the sequence is finally determined by voting. At last, we concatenate different sequences with same identity in all shots of one scene into one sequence.

Construction of Track Video. Based on the character track sequence on the scene and the start and end time of the clip, the track sequence on each clip is generated and saved as video for later usage.

Location Recognition. The location recognition is similar to person recognition, we extract features by ResNet and average the features of different images for the same location. Then we extract the visual feautures of location by ResNet shot by shot, then we calculate the similarity with library to obtain the location identity in the shot.

2.2.3 Interaction Relation Prediction

The third step is the prediction of interaction and relation. We adopt a similar structure with Kukleva[18] for interaction recognition and relation recognition.

Multimodal Features. We use multi-modal features for the prediction module, including visual features and text features.

Visual features. We use TSM [19] to extract vectors of 2048 dimensions as visual features. For each clip, we extract features of every clip in a scene and for each combination of two track sequences. For example, if two persons and one location are recognized in the clip, there should be 6 different combinations of track features as there is active and passive relationships or interaction.

Text features. For each clip, we use pre-trained BERT-base [20] to convert sentences to vectors of 768 dimensions as text features, then we gather all the clip text feature belongs to the same scene. After extracting all the features in the scene, we concatenate the visual feature and text feature and get the feature of one scene.

Interaction and Relation Recognition. Due to the annotations given officially are in scene-level rather than clip-level so we average the clip feature, which means every average scene feature may get multiple labels. Based on the multi-labeled feature, we apply the multi-label classification method to predict interaction between two persons. For the recognition of relation, we use the same method to get the average features as input to predict relationship.

2.2.4 Shot-Scene Summary & Implementation of DVU

Knowledge Graph. For each movie, we can get a list of predicted relations and interactions, then we use the Neo4j to generate the knowledge graph. The interactions and relations are saved in the different graphs. There are two kinds of nodes, the person node and location node. In interaction part, the connecting line is recognized interaction. In relation part, the connecting line is relation.

Shot-Scene Summary. In order to use LLM, we need to provide it with background knowledge in textual form to help us answer the questions, so we propose a shot-scene summary approach. We use CLIP Interrogator to generate picture descriptions for the middle frame of each shot in a scene, because the middle part is often the richest part of a movie information, and after obtaining descriptions of all the shots, these descriptions are stitched together and fed to LLM, allowing it to remove duplicate semantics and regenerate a more refined description for after usage.

Implementation of DVU. In NIST TRECVID 2023 DVU task, there are 5 types of question, in scene-level:

- **Find the Unique Scene (q1)**. Given a full, inclusive list of interactions, unique to a specific scene in the movie, teams should find which scene this is.
- **Find Next or Previous Interaction (q2&q3)**. Given a specific scene and a specific interaction between person X and person Y, participants will be asked to return either the previous interaction or the next interaction, in either direction, between person X and person Y. This can be specifically the next or previous interaction within the same scene, or over the entire movie. This query type will take a multiple choice questions format.
- **Match Scene to Text Description (q4)**. Given a list of natural language descriptions of scenes and a list of scene numbers, match the descriptions with the scene number.
- **Scene Sentiment Classification (q5)**. Given a specific movie scene and a set of possible sentiments, classify the correct sentiment label for each given scene.

In movie-level, there are also 2 types of question:

- **Fill in the Graph Space (q1)**. Fill in spaces in the Knowledge Graph (KG). Given the listed relationships, events or actions for certain nodes, where some nodes are replaced by variables X, Y, etc., solve for X, Y etc. Example of The Simpsons: X Married To Marge. X Friend Of Lenny. Y Volunteers at Church. Y Neighbor Of X. Solution for X and Y in that case would be: X = Homer, Y = Ned Flanders.
- **Question Answering (QA) (q2)**. This query type represents questions on the resulting knowledge base of the movies in the testing dataset. For example, we may ask 'How many children does Person A have?', in which case participating researchers should count the 'Parent Of' relationships Person A has in the Knowledge Graph. These queries also contain human-generated questions. These are open domain questions which are not limited to the ontology. This query type will take a multiple choice questions format.

For q1 in scene-level, we transfer the query to the Cypher language, and search in the KG. Finally we get a set of scenes, and we take the top four as our answers. For q2&q3, we have two ways to solve it. One is similar to q1, first to the Cypher language and then search in KG to get answer. And the other is through LLM, we input the query and shot-scene summary generated by clip-interrogator API as background, have LLM rank all options. Then we rank average the answers we gave with the answers given by LLM to get the final answer. In the experiment, we assigned the score of the six answers as 6,5,4,3,2,1 by order, and we think that the accuracy of shot-scene summary and the interpretation ability of LLM are very strong, so we assign 1 to the weight of the answers generated by LLM, the answers we obtained through the KG were assigned a weight of 0.6. For q4&q5, we treat them as video retrieval task, and leverage a VLM ALPRO[21] to handle these.

For q1 in movie-level, we solve them by finding the relation line between the character nodes mentioned in the queries. Specially, we find that there are different relationship libraries between different categories of entities. We use LLM to devide the relation terms into two classes *e.g.*, PP (person-to-person) and PL (person-to-location) to reduce the answer space. For q2, We first obtain a coarse-grained summary of the video through Video captioning model PDVC[22], and then we integrate the results of Automatic Speech Recognition (ASR) and the corresponding role tracking. We feed this combined information into LLM to get a fine-grained summary , which we ultimately provide along with the question to LLM, prompting it to choose the appropriate option. Lastly, It is worth pointing out that the LLM we use is the API for chatGPT provided by OpenAI.

2.3 Analysis

2.3.1 Dataset & Query

We use the dateset from NIST [23], which include 19 training movies (total 25 hr) and corresponding annotations, snapshots, vocabulary and, 5 testing movies (total 7.5 hr) with snapshots and corresponding query from NIST TRECVID 2023 DVU task, and use **MRR** and **ACC** as metrics. **MRR** stands for Mean Reciprocal Rank which is the average of the reciprocal ranks of all instances. **ACC** stands for Accuracy which is calculated by dividing the number of correctly classified instances by the total number of instances. In addition, we also used the relevant data and query of NIST TRECVID 2022 DVU task as test in early stage.

2.3.2 Evaluation and Results

The results in NIST TRECVID 2023 DVU task are listed in Table 6. We assess two levels of problems: Scene-level and Movie-level. For scene-level problems, we categorize them into two groups based on the official criteria. The first group consists of q1, q2, and q3, while the second group includes q4 and q5. We calculate the weighted average of the metrics within each group as the metrics. The metrics for the first group is 40.9%, and for the second group it is 51.2%. As for movie-level problems, we divide q1 and q2 into two groups. The metrics for the first group is 59.6%, and 43.0% for the second group.

Through the analysis of different question and answers, we found that our model has a better performance in the questions that Fill in the graph space. As we mentioned above, this type of questions has strong requirements for the recognition of characters, and our method is accurate in the recognition of characters. But we also found some queries

Table 6: Result in NIST TRECVID 2023 DVU task (%)

Movie	Scene-level				Movie-level	
	q1 (MRR)	q2&q3 (ACC)	q4 (ACC)	q5 (ACC)	q1 (MRR)	q2 (ACC)
Archipelago	12.5	62.5	40	33.3	81.25	33.3
Bonneville	37.5	25	50	66.7	62.5	39.1
Heart_Machine	25	50	60	33.3	37.5	61.5
Little_Rock	18.8	0	60	33.3	62.5	30
Memphis	8.32	12.5	60	66.7	54.2	59.1
Total	26.8		51.2		59.6	43.7

that we didn't do well. The first problem is the data distribution of the training set is not even (long-tail), our method will tend to choose the answer with more training samples. The second problem is indistinguishable action by visual aspect alone, such as "talk to" and "asks". Our method has difficulty distinguishing similar pairs of these answers. We also tried to add audio features to assist this problem, but it did not work. We believe that the dimension of some commonly used audio features (MFCC, etc.) is small (less than 100). Compared with the text-visual features (6912 dimensions), audio features can provide little information.

2.4 Conclusion

In this paper, we propose a hierarchical DVU method with shot-based instance search and large language model. We introduce shot-based instance search to cope with track error, and we also use LLM to answer questions for us by providing background knowledge for LLM. We have evaluated our method in the NIST TRECVID 2023 DVU task, and obtain competitive results.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (U1903214, 62372339, 62371350, 61876135). We also would like to express our gratitude to the person who have helped us in both missions. We would like to express our sincere gratitude to Yingfei Sun for her previous contribution to DVU and Weixi Song & Xinghan Li for their previous contribution to AVS. And the numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

References

[1] Chunyu Hu, Hong Zhang, Chao Liang, and Hao Huang. Qi-ira: Quantum-inspired interactive ranking aggregation for person re-identification. In *The 38th Annual AAAI Conference on Artificial Intelligence (AAAI)*. Springer, 2024.

[2] George Awad, Keith Curtis, Asad A. Butt, Jonathan Fiscus, Afzal Godil, Yooyoung Lee, Andrew Delgado, Jesse Zhang, Eliot Godard, Baptiste Chocot, Lukas Diduch, Jeffrey Liu, Yvette Graham, , and Georges Quénot. An overview on the evaluated video retrieval tasks at trecvid 2022. In *Proceedings of TRECVID 2022*. NIST, USA, 2022.

[3] Luca Rossetto, Heiko Schuldt, George Awad, and Asad A Butt. V3c–a research video collection. In *International Conference on Multimedia Modeling*, pages 349–360. Springer, 2019.

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

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

[6] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[7] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023.

[8] Lijie Fan, Dilip Krishnan, Phillip Isola, Dina Katabi, and Yonglong Tian. Improving clip training with language rewrites. In *NeurIPS*, 2023.

[9] Kazuya Ueki, Yuma Suzuki, Hiroki Takushima, Hideaki Okamoto, Hayato Tanoue, and Takayuki Hori. Waseda meisei softbank at trecvid 2022. In *Proceedings of the TRECVid 2022 Workshop*, pages 1–5, 2022.

[10] Sagar Uprety, Dimitris Gkoumas, and Dawei Song. A survey of quantum theory inspired approaches to information retrieval. *ACM Computing Surveys (CSUR)*, 53(5):1–39, 2020.

[11] Penggang Qin, Jiarui Yu, Yan Gao, Derong Xu, Yunkai Chen, Shiwei Wu, Tong Xu, Enhong Chen, and Yanbin Hao. Unified qa-aware knowledge graph generation based on multi-modal modeling. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 7185–7189, New York, NY, USA, 2022. Association for Computing Machinery.

[12] Raksha Ramesh, Vishal Anand, Zifan Chen, Yifei Dong, Yun Chen, and Ching-Yung Lin. Leveraging text representation and face-head tracking for long-form multimodal semantic relation understanding. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 7215–7219, New York, NY, USA, 2022. Association for Computing Machinery.

[13] Siyang Sun, Xiong Xiong, and Yun Zheng. Two stage multi-modal modeling for video interaction analysis in deep video understanding challenge. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22, page 7040–7044, New York, NY, USA, 2022. Association for Computing Machinery.

[14] Ming Chen, Peng Du, and Jieyi Zhao. Scrfd: Spatial coherence based rib fracture detection. In *Proceedings of the 2018 5th International Conference on Biomedical and Bioinformatics Engineering*, pages 105–109, 2018.

[15] 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.

[16] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[17] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple Online and Realtime Tracking with a Deep Association Metric. *arXiv e-prints*, page arXiv:1703.07402, March 2017.

[18] Anna Kukleva, Makarand Tapaswi, and Ivan Laptev. Learning interactions and relationships between movie characters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9849–9858, 2020.

[19] Ji Lin, Chuang Gan, Kuan Wang, and Song Han. Tsm: Temporal shift module for efficient and scalable video understanding on edge devices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[21] Dongxu Li, Junnan Li, Hongdong Li, Juan Carlos Niebles, and Steven C.H. Hoi. Align and prompt: Video-and-language pre-training with entity prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4953–4963, June 2022.

[22] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6847–6857, 2021.

[23] Keith Curtis, George Awad, Shahzad Rajput, and Ian Soboroff. Hlvu: A new challenge to test deep understanding of movies the way humans do. In *Proceedings of the 2020 International Conference on Multimedia Retrieval*, pages 355–361, 2020.