CSIT5910 Term Project Report - 21 Fall
Partial Duplicate Video Retrieval based on Deep Metric Learning

Yu Du, Zifan Peng
Department of Computer Science and Engineering, HKUST
{yu.du, zpengao}@connect.ust.hk

Abstract

Under the current development trend of the Internet, video materials and information are becoming more and more abundant. Multiple quotations of videos from different platforms and some authors did not indicate the source will cause a series of copyright issues. Against the backdrop of this, this work addresses the problem of Partial Duplicate Video Retrieval (PDVR). This paper proposes an high-accurate model to query sources of videos based on extracting and embed the features into a high dimension space. Then we employ Network in Network (NiN), which is that we input the embedding features into a DNN neural network to calculate the transition probability of videos as the input weight of the gates in Video Frame Retrieval Memory (VFRM) model which is LSTM-like to derive the source video in the database. We combine the outputs of VFRM and similarity comparison, locate the original video, where the dataset is generated from an independent dataset and is thoroughly tested on the widely used CC_WEB_VIDEO dataset, employing two popular deep CNN architectures (AlexNet, GoogleNet). In the end, we demonstrate it have a high-accuracy, and corresponding videos and corresponding time slot could be extracted from the existing database.

1. Introduction

Currently, the form of information using video as a carrier has become quiet popular. Driven by platforms such as Tik Tok and Bilibili, the concept of "everyone is a video creator" has gradually been recognized by the public. All major platforms have launched efficient and convenient video editing tools to further lower the threshold for video creation. On the video platform, we have noticed that there are a lot of secondary creation phenomena. A single video often contains many non-original material. Therefore, tracing the source of this kind of video and protecting the original is an indispensable component in numerous applications.

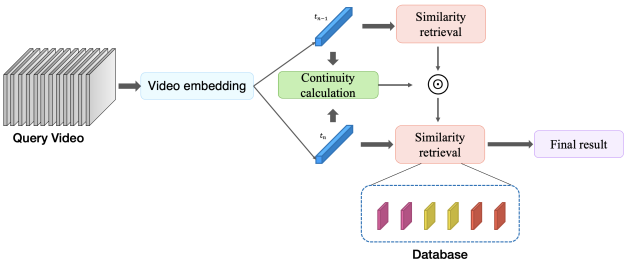

Figure 1. **Pipeline overview –** overview of the whole model pipeline

By simply analyzing the video content, it is easy to think that we can analyze the content composition and originality of the video by comparing the video content. At present, researchers have realized this process by exploring the implementation of Near-Duplicate Video Retrieval [6]. Since the video processing methods are diversified, a primary issue is how to define Near-Duplicate Videos (NDVs). In this project, we have adopted the definition of Wu *et al.* [8]. NDVs are defined as videos that are close to duplicate of each other, but different in terms of photo-metric variations (color, lighting changes), editing operations (caption, logo and border insertion), encoding parameters, file format, different lengths, and other modifications.

At present, there are several typical solutions for the NDVR problem.However, many methods bind data sets and use the same data sets for development and evaluation. This leads to poor performance when the results are applied to different video corpora. There are also methods that point out that the NDVR approach using deep learning has good retrieval results [4]. But the focus of these articles is generally on the analysis of a single short video. They use model training to analyze whether the short video is NDV.

In our project, we consider applying this ability to the analysis of long videos. This engineering application will be closer to real problems. Our long video NVDR is mainly composed of three steps. First, we use CNN features from

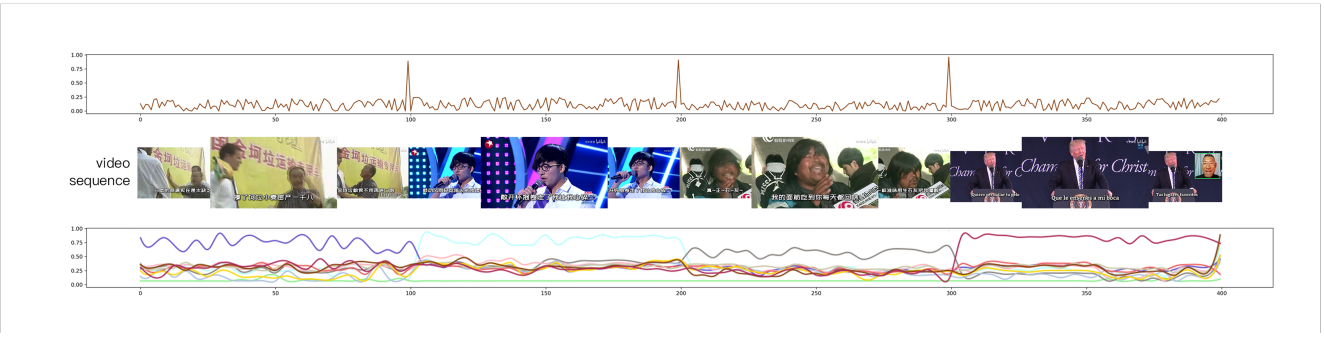

Figure 2. **PDVR – Segmentation and similarity of the query video**

intermediate convolution layers based on Maximum Activation of Convolutions. Second, we leverage a Deep Metric Learning (DML) framework. Through these two steps, we can use a smaller data set to train the retrieval model efficiently and achieve better results. Finally, we apply the above results to long video analysis. We introduced a video segmentation model to optimize the effect of video segmentation and designed a Video Frame Retrieval Memory (VFRM) network model with gates to memorize and predict the target video series. At the same time, we created a small training set to optimize the above two models, the pipeline overview of the whole model is shown in Figure 1.

2. Related works

Based on our project, related work can be divided into two parts. The first part is the implementation of NDVR, and the second part is the segmentation of long videos.

2.1. Implementation of NDVR

Liu *et al.* [6] has done a very good sorting out of the existing methods. According to the granularity of matching between NDVs, the existing NDVR method is divided into video-level, frame-level and hybrid-level matching. Video-level matching aims at solving the NDVR problem at massive scale. Frame-level matching means to compare between individual frames or frame sequences. And the last one hybrid-level matching is trying to combine the advantages of video- and frame-level methods. TRECVID copy detection task is a well-known work. The difference between this work and NDVR lies in the source of the video corpus.

Another important area is Metric learning. This is to learn a distance function to measure similarity-similar objects are close, dissimilar objects are far away. Deep metric learning currently mainly uses the network to extract embedding, and then uses L2-distance to measure the distance in the embedding space [10].

2.2. Segmentation of Long Videos

Some literatures have summarized and sorted out the typical methods of video segmentation [7]. First is about feature extraction. The first category includes features extracted from the entire image, such as color and histogram features. The other type is the area-based feature extraction technology. Finally, there is modeling and classification. There are generally the following methods, Unsupervised Methods, Support Vector Machine, Random Decision Forest, Markov Random Field, Conditional Random Field and Neural Networks.

There are also some typical methods. For example, OSVOS [1] mainly treats each image in the video as an independent picture for processing, without considering the timing information. MaskTrack [3] considers timing information and still treats each frame as a static image. The key method to realize video segmentation in the paper is to combine online and offline strategies.

3. Method

Video data is very huge with a lot of redundancy. Designing an end-to-end model to solve a specific video task will cause the size of the model and the difficulty of training to increase exponentially. To simplify the problem, we first utilize pretrained VGG-16 as backbone network to extract high level features (section 3.1). Then we deploy a deep metric learning (DML) model to map the extracted feature to a embedding space, which denotes the similarity distance between images or videos (section 3.2). After that, we employ NiN and LSTM-like model, where the inputs of LSTM are the outputs of DNN model, which is to obtain weather the two frames of video are from different videos (section 3.3 and 3.4).

3.1. Feature Extraction

To simplify the model structure and the training process, we deploy pretrained VGG-16, which is a CNN backbone network, to extract high-level image and video features.

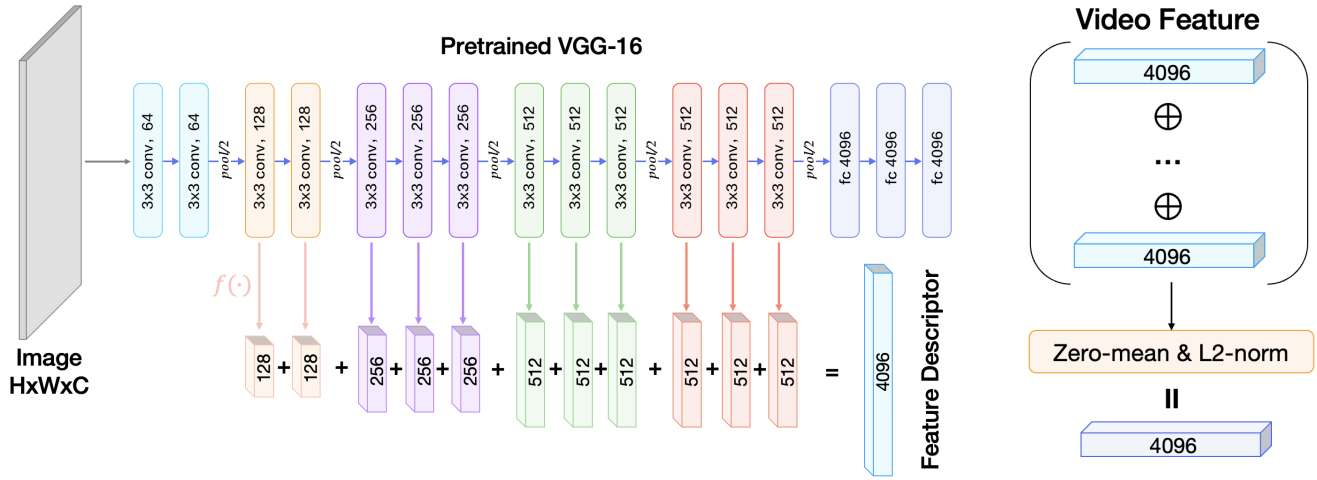

Figure 3. **LHS –** The intermediate feature map we extract from the pretrained VGG network. **RHS –** Process for converting the image feature to video feature.

These features are the outputs of the intermediate convolution layers. We use $\mathcal{M}^l \in \mathbb{R}^{n^l \times n^l \times c^l}$, $l \in \{1, 2, ..., L\}$ to denote the feature map got from the layer output, where n^l is the dimension of the feature map of layer l, c^l is the number of channels, and L is the number of intermediate layers to extract feature map. For every feature map $\mathcal{M}^l$, we do max pooling on each channel, as shown in Eq 1. This operation will get a c^l-dmimension vector v^l, then we will concatenate all $v^l, l \in \{1, 2, ..., L\}$ into one vector to be the extracted feature.

$$v_i^l = max(\mathcal{M}_i^l), \quad i \in \{1, 2, ..., c^l\} \tag{1}$$

To generate video features, uniform sampling is initially applied to select n frames per second for every video and extract the respective features for each of them. The video features are then derived by averaging and normalizing (zero-mean and l_2- normalization) these frame features. So the video feature has the same shape with the image feature, which is 4096-dimension in our experiment. The intermediate feature map we extract and the image feature to video feature process are shown in Figure 3.

3.2. Deep Metric Learning Model

The Deep Metric Learning model is to learn a embedding function that maps the video feature into an embedding space, where the near duplicate video has a small distance with the original one and those not similar videos will have a larger distance. The distance calculation uses squared Euclidean distance, as shown in Eq 2.

$$D(f_\theta(q), f_\theta(p)) = ||f_\theta(q) - f_\theta(p))||_2^2 \tag{2}$$

The input of the DML model is a triplet (v_i, v_i^+, v_i^-), where v_i is the feature of query video, v_i^+ is the NDV, v_i^- is non-NDV. The objective of DML is to learn an embedding function f_θ that assigns smaller distances to NDV pairs compared to non-NDV ones, which can be describe by Eq 3.

$$D(f_\theta(v), f_\theta(v^+)) < D(f_\theta(v), f_\theta(v^-)) \tag{3}$$

The triplet can be generate from the dataset through query label, the video with same query label is NDV and with different query label is non-NDV. The loss function is defined in Eq 4, where γ is a margin parameter to ensure a sufficiently large difference between the positive-query distance and negative-query distance. If the video distances are calculated correctly within margin γ, then this triplet will not be penalised.

$$\begin{aligned} &L_\theta(v_i, v_i^+, v_i^-) \\ =&max\{0, D(f_\theta(v), f_\theta(v^+)) - D(f_\theta(v), f_\theta(v^-)) + \gamma\} \end{aligned} \tag{4}$$

The pipeline of the DML module is shown in Figure 4. The triplet are fed independently into three siamese DNNs with identical architecture and parameters. The DNNs compute the embeddings of v: $f_\theta(v)$. The architecture of the deployed DNNs is based on three fully-connected layers and a normalization layer at the end leading to vectors that lie on a d-dimensional unit length hypersphere. Here $d = 500$ in our experiment, which is further smaller than the number of feature. Thus the embedding output is a higher-level dense video feature.

3.3. DNN Model

The inputs of the deep neural network is a set of triplets $\mathcal{T}$ created by the DNN training set generator which will be illustrated in Section 4.1. Each triplet contains embedding

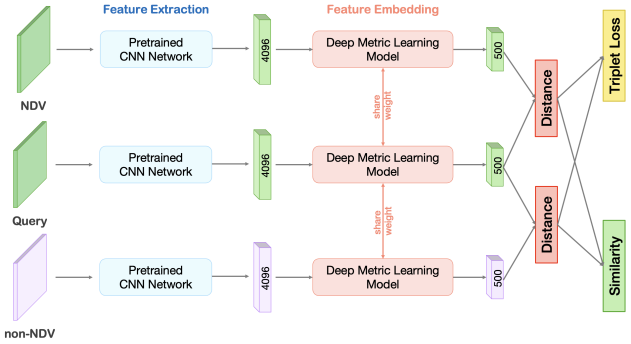

Figure 4. **Pipeline –** The triplet is inputted to the pretrained CNN and get the extracted feature, then to the DML for the feature embedding. For training, we use Triplet Loss function, and for the input of upper module, we calculate the similarity.

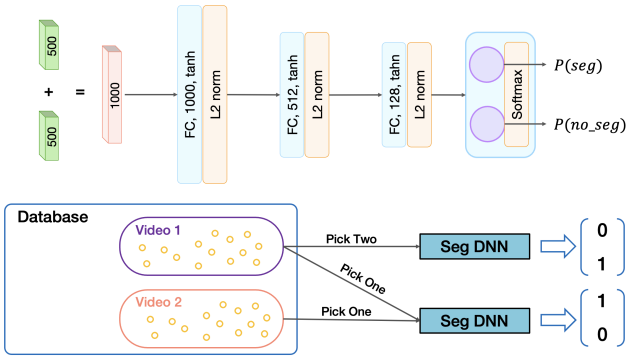

Figure 5. **DNN –** Segmentation DNN Model

features of two frames and a label which denotes it occurs video transition between these two frames or not, more precisely 1 and 0 respectively, which were represented by e_1, e_2 and y, whose shapes are depends on the outputs shape of previous model which extracts the embedding features. We exploited a three fully-connected layers DNN, architecture of which is shown in Figure 5 . The architecture of the deployed DNN is based on three dense *fully-connected layers* and three *normalization layers* after each FC layer before the activate functions. The normalization layers lead to resolve gradient explosion and the scaling factor of BN can effectively identify the neurons that do not contribute much to the network. The activate function is widely-used $Tanh$ function and the last output activate function which outputs the probabilities of segmented and not segmented is $f(v) = Sigmoid(v)$ shown in the figure. The number of neurons in each hidden layer (size of each hidden layer) are fixed to 1000, 512 and 128, which eventually output 2 values.

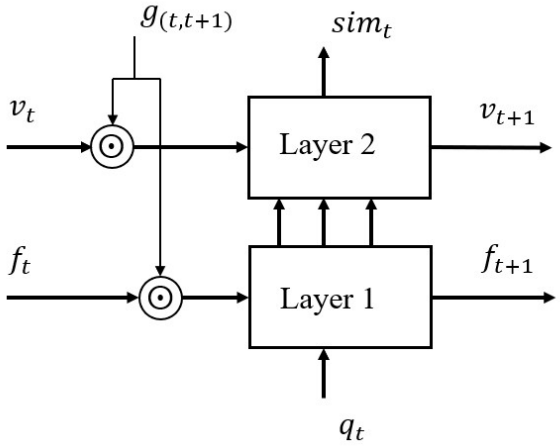

Figure 6. **LSTM –**

3.4. Video Frame Retrieval Memory Model

The query video is a typical time series data and the relationship between two frames can also be used to predict the correct source of a frame. It's easy to notice that frames in the video nearby is supported to have more similarity with each other. We design a Video Frame Retrieval Memory (VFRM) network model with gates to memorize and predict the target video series. The query video is combined by slices of source video, author may cut slices of video from the same or different videos together. So the relationship between two nearby frames in the query video may have three relations: (1) consecutive frames in the same video; (2) discontinuous frame but in the same videos; (3) frames from different videos. We refer to the design of LSTM [2], append two hidden time-series data and build a model to memorize or forget the video and frame information from previous frames.

Firstly, we need to train a model which tells us whether the frame comes from the same video or not from two frames in t and t+1. A deep network structure [5]. will calculate the possibility of the existence of cutting point. Gate result $g_{t,t+1}$ will decides whether video and frame data: v_t and f_t will pass through and affect the weight of next cell or not. Then, two layers designed for denotes similarity of frames will calculate the final similarity with a weight-bias model:

Layer 1:

$$\text{sim}_1 = e_t q_t + b_f f_{t+1} \tag{5}$$

If two nearby frames q_t and q_{t+1} are from the same videos, and q_t is similar to f_t it's very possible that q_{t+1} is similar to f_{t+1} which meets the situation 1 above.

Layer 2:

$$\text{sim}_2 = \text{sim}_1 + b_v v_t \tag{6}$$

And the final result of two layer will be:

$$\text{sim}_f = e_t q_t + b_f f_{t+1} + b_v v_t \tag{7}$$

Finally, the result of this cell v_{t+1} and f_{t+1} will pass to next cell.

3.5. Similarity Computation

In the previous employed models, we calculate the similarity of embedding features by adopting the Euclidean distance between vectors for further calculation, we define distance in the embedding space between two videos as Equation 2, where $\{p_i\}_{i=1}^{M} \in P, \{q_i\}_{i=1}^{N} \in Q$, and P is the set of whole database frames, Q is the set of all query video frames.

Finally, the similarity between two videos we can derive from the distance in the embedding space, the equation is defined as follows, $S(\cdot,\cdot)$ is the similarity between two videos and $max(\cdot)$ is the maximum function.

$$S(q,p) = 1 - \frac{D(f_\theta(q), f_\theta(p))}{\underset{p_i \in P}{max}(D(f_\theta(q), f_\theta(p)))} \tag{8}$$

4. Evaluation

4.1. Datasets

Training Dataset. We leverage CC_WEB_VIDEO [9] dataset, which is deployed for training the deep metric learning model. The collection consists of a set of videos retrieved by submitting 24 frequent text queries to popular video sharing websites, i.e. YouTube, Google Video, and Yahoo! Video. The dataset contains a total of 13,129 videos with 397,965 keyframes. The dataset is originally stored on the server at CityU. Since the video dataset is very huge (411,094 independant files and 90GB in size), using regular downloader would be very slow and may encounter thread blocking problem. To solve this problem, we develop a multi-thread download tool to handle this dataset. The GitHub repository of this tool is here [1].

Evaluation dataset. The test dataset was generated semi-manually for specific requirements. We retrieved some original videos from Bilibili (B site), a popular video streaming site with a relatively huge platform traffic, where the videos cover TV series, variety shows, animation, self-publishing and other popular videos. We randomly select video clips of more than a specific duration (5s, by default) from the original videos and stitch them together in a random order by using package in Python. The test dataset contains a total of 1,193 clips with 57,695 frames, whose lengths vary from 15.3s to 614.7s. We extract three frames per second from these clips and perform feature extraction and embedding on these frames, after which we obtained a series of embedding features to input the following model.

Segmentation DNN training dataset. We employ the Network in Network(NiN) [5] for improving the accuracy of the query videos. Thus we generated this dataset automatically for feeding the DNN whose outputs are our gates threshold of following LSTM model, and this dataset [2] contains 18,300 positive samples and 18,300 negative samples. The DNN model is a binary classification model, which classify if the video occurs transition between previous and current frame, and output the probability of these two classes. We label it as positive (more precisely, 1) data if previous and current frames are from different videos and mark it as negative (more precisely, 0) otherwise.

4.2. Evaluation Metrics

To measure detection accuracy, we adopted four widely-accepted metrics, which contains the interpolated precision-recall (PR) curve and accuracy tables. We need to calculate precision and recall by following formulas, where **TP** is **T**rue **P**osivive and **FN** is **F**alse **N**egative, which means the number of retrieved positive and unretrieved positive samples respectively, and **TN** is **T**rue **N**egative, **FP** is **F**alse **P**osivive, which means the number of retrieved negative and unretrieved negative samples respectively.

- **Accuracy**: : Accuracy represents the proportion of correct predictions (both true positives and true negatives) among the total number of cases examined. The Accuracy metric is defined as follows:

$$Accuracy = \frac{TP + TN}{FP + FN + TP + TN} \tag{9}$$

- **Precision:** Precision denotes the proportion of two frames of the video that are correctly classified into transitions among all the transitions. The Precision metric is defined as follows:

$$Precision = \frac{TP}{TP + FP} \tag{10}$$

- **Recall:** Recall represents the proportion of all video transitions that are correctly classified as transitions. The Recall metric is defined as follows:

$$Recall = \frac{TP}{TP + FN} \tag{11}$$

[1] CC_WEB_VIDEO Downloader: https://github.com/Mi-Dora/CC_WEB_VIDEO_Downloader

[2] Seg_train_data: https://drive.google.com/drive/folders/1XYat0tl2vFmquWZsOYW3w19reJMnkYOJ?usp=sharing

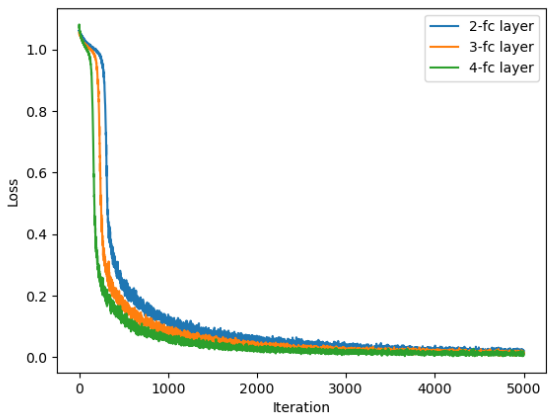

Figure 7. **Loss –** Training Loss of the Deep Metric Learning Model with different number of layers.

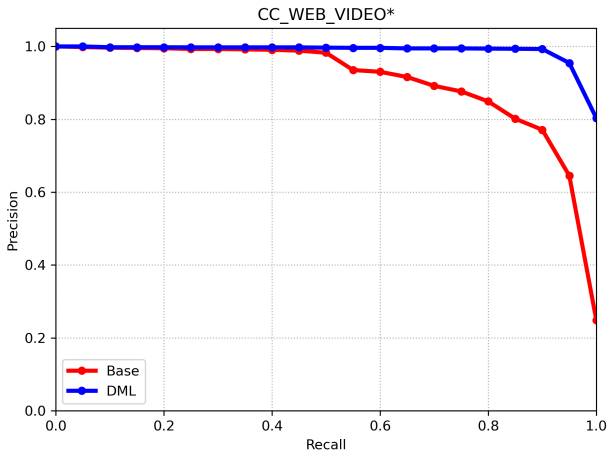

Figure 8. **PR –** Precision-Recall curve of Deep Metric Learning on CC_WEB_VIDEO, comparing with the baseline.

Baseline	DML 2-layers	DML 3-layers	DML 4-layers
95.70%	99.96%	99.99%	99.97%

Table 1. **mAP** – comparing among baseline and DML with different number of layers.

- **mAP:** : Average Precision, which combines both precision and recall, uses the area under the PR curve as a measure. Mean Average Precision (mAP), which is the mean of AP value, evaluates the average AP value for multiple individuals in the validation set, where Q_R is the set of validation, which is defined:

$$mAP = \frac{1}{|Q_R|} \sum_{q \in Q_R} AP(q) \tag{12}$$

5. Results

Figure 7 shows the training loss curve, we can find that the loss decreases very fast and smoothly, which is very reasonable because the input of the DML model is already high-level features extracted by VGG-16 and the DML model is a small model.

In order to evaluate the effectiveness of the DML model, we also use the baseline, which uses the extracted feature to compute the similarity directly, to make a comparison. Figure 8 is the Precision-Recall curve of the DML model and the baseline. We can find that the feature embedding process significantly improve the classification precision on hard sample in the dataset. And the mAP of DML is 99.99%, which also outperforms the 95.70% on the baseline.

At the same time, Precision-Recall graph and accuracy table of Video Segmentation are shown in Figure 9 and

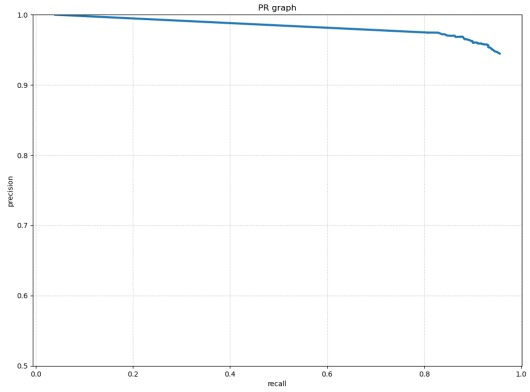

Figure 9. **PR –** Video Segmentation Precision and Recall graph.

Ratio of Positive Sample	0.0	0.2	0.4	0.6	0.8	1.0
Accuracy	0.9732	0.9682	0.9511	0.9336	0.9219	0.9172

Table 2. **Accuracy –** of the segment DNN in various ratios of positive samples.

Table 2, which is highly precise and accurate because of the fact the extracted embedding features are in a high-dimension space.

We also test some videos with noises which are shown in Figure 10, the test query videos contains video split screen, add lots of text and irrelevant emoticons, etc. The model also has a good performance, and the query similarities are shown in Figure 10, but some video are recognized but with lower similarity, which will occur some accuracy issues.

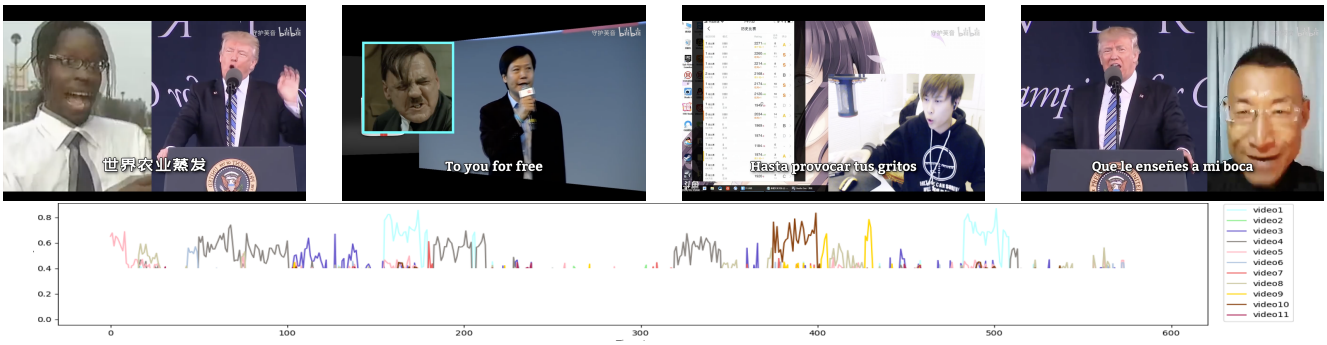

Figure 10. **Video with Noises –** similarities when video have noises

6. Conclusions and Future Work

We presented a new model for Partial Duplicate Video Retrieval (PDVR), which leverages the effectiveness of existing Deep Metric Learning, intermediate convolution and traditional fully-connected layers. We proposed the architecture based on LSTM-like model which expoit Network in Network (NiN) method. The proposed approach was tested on semi-manually generated video clips. Finally, the proposed method has a competitive performance when the requirements is based on a specific video query scenario, and we derived the Precision-Recall and F-1 score figures and tables in experiments.

In future work, we plan to look into further improvement on speed and accuracy of PDVR. We propose to explore more precise and accurate video feature embedding. The current DML method compress the whole video frames into one single embedding, which cannot detect the internal scenario change. We suggest to deploy a slide window on the frame sequence to do the fragment embedding. In additional, we will increase the similarities comparison in some video frames with noises and find some methods to calculate the similarity more accurately and improve the model's resistance and robustness, and we proposed to assess the applicability of generating and applying more diverse and challenging datasets.

Acknowledgement

This work is supported by the course of CSIT5910: Machine Learning and Prof. Nevin L. Zhang, Department of Computer Science & Engineering, The Hong Kong University of Science and Technology.

References

[1] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation, 2017. 2

[2] Klaus Greff, Rupesh K. Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. Lstm: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10):2222–2232, 2017. 4

[3] Anna Khoreva, Federico Perazzi, Rodrigo Benenson, Bernt Schiele, and Alexander Sorkine-Hornung. Learning video object segmentation from static images, 2016. 2

[4] Giorgos Kordopatis-Zilos, Symeon Papadopoulos, Ioannis Patras, and Yiannis Kompatsiaris. Near-duplicate video retrieval with deep metric learning. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017. 1

[5] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network, 2014. 4, 5

[6] Jiajun Liu, Zi Huang, Hongyun Cai, Heng Tao Shen, Chong Wah Ngo, and Wei Wang. Near-duplicate video retrieval: Current research and future trends. *ACM Computing Surveys (CSUR)*, 45(4):1–23, 2013. 1, 2

[7] Mohammad Hajizadeh Saffar, Mohsen Fayyaz, Mohammad Sabokrou, and Mahmood Fathy. Semantic video segmentation: A review on recent approaches, 2018. 2

[8] Xiao Wu, Alexander G. Hauptmann, and Chong-Wah Ngo. Practical elimination of near-duplicates from web video search. In *Proceedings of the 15th ACM International Conference on Multimedia*, MM '07, page 218–227, New York, NY, USA, 2007. Association for Computing Machinery. 1

[9] Xiao Wu, Alexander G Hauptmann, and Chong-Wah Ngo. Practical elimination of near-duplicates from web video search. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 218–227, 2007. 5

[10] Liu Yang and Rong Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2(2):4, 2006. 2