修　士　論　文

Weakly-Supervised Hierarchical Reinforcement Learning for Video Summarization

(映像要約のための
弱教師つき階層型強化学習)

指導教員　山崎 俊彦　准教授

東京大学大学院
情報理工学系研究科
電子情報学専攻

氏　名　48-186468　陳　亦言

提　出　日　2020 年 8 月 14 日

Abstract

Nowadays, video data are increasing explosively on the Internet. Processing video data is a challenging task. Video summarization can summarize the long original video into a short and concise summary to help us with processing video data. Deep learning based methods regard video as a sequence of frame features. Deep conventional neural networks (CNN) are used to extract the feature vectors for each frame of the video. Then summarization networks are designed to process the temporal information between frame features, like long short-term memory (LSTM). In this thesis, to solve this video summarization task, I apply reinforcement learning methods with a hierarchical structure to video summarization. Besides, I design a sub-reward and a subgoal for the hierarchical structure to enhance the ability of processing long temporal dependence and avoid the sparse reward problem. Task-level labels are used to train my model. Compared with supervised learning methods, my model requires a much smaller number of labels. The amount of required annotations have been reduce to 1/20 with our methods. Experiments on two benchmark datasets by two different metrics show that my proposal has improve the performance effectively.

Contents

1. Introduction **1**

1.1. Video Summarization 1

1.2. Reinforcement Learning 3

1.3. Proposal and Contributions 6

1.4. Organization of This Thesis 7

2. Related Works **8**

2.1. Video Summarization 8

2.1.1. Supervised Methods 8

2.1.2. Unsupervised Methods 11

2.1.3. Weakly Supervised Methods 15

2.2. Reinforcement Learning 17

2.2.1. Deep Q Network 17

2.2.2. Policy Gradient 18

2.2.3. Hierarchical Reinforcement Learning 19

3. Approaches **22**

3.1. Basic Procedures 23

3.2. Hierarchical Structure 24

3.3. Manager Network 25

3.4. Worker Network 27

3.5. Reward Function 28

3.6. Optimization 29

4. Experiments **32**

4.1. Dataset 33

4.2. Evaluation Metrics 34
4.2.1. F score 34
4.2.2. Rank correlation coefficient 36
4.3. Training Setting 37
4.4. Evaluation 38
4.4.1. Quantitative Evaluation 38
4.4.2. Qualitative Evaluation 41

5. Conclusions and Future Works **45**
5.1. Conclusions 45
5.2. Future Works 46

References **50**

Publications **54**

Acknowledgements **56**

List of Figures

1.1. Overview of video summarization 2
1.2. Explanation of Reinforcement Learning. 4
1.3. Illustration of sparse reward problem. 6

2.1. The repeating module in an LSTM contains four interacting layers . 8
2.2. The model vsLSTM [1] is implemented by bidirectional LSTM layers. 9
2.3. The model dppLSTM [1] is an extension of vsLSTM with DPP. . . 10
2.4. The structure of SUM-GAN. 12
2.5. The structure of DR-DSN. 13
2.6. Explanation of web prior. 15
2.7. Illustration of FeUdal networks 20

3.1. Basic procedures of deep video summarization 23
3.2. Hierarchical structure of proposal 24
3.3. Illustration of task-level label 26
3.4. Explanation of training the Manager. 30
3.5. Explanation of training the Worker. 30

4.1. Definition of precision and recall 35
4.2. Experiments are conducted with different sizes of subtask on TV-Sum dataset. When $n = 20$, it shows the best performance. 39
4.3. Examples of summary videos of proposed method comparing with the state-of-the-art. 43
4.4. Examples of summary videos of proposed method comparing with the state-of-the-art. 44

5.1. Illustration of multi-model of video 47

5.2. Comparison between 2D convolution and 3D convolution 49

List of Tables

4.1. Information of datasets 33
4.2. Evaluation by F score on SumMe dataset 39
4.3. Evaluation by F score on TVSum dataset 40
4.4. Kendall's τ and Spearman's ρ correlation coefficients computed between different importance scores and human annotated scores. . . 41

Chapter 1

Introduction

1.1. Video Summarization

Nowadays, there are a lot of videos being uploaded into the Internet. Analyzing video data becomes very important to our daily life, such as action recognition, video surveillance and so on. Image processing has developed a lot in the past. Therefore, a video is usually regarded as a sequence of image frames. However, compared to image data, video data are extremely larger. A 5-minute video can contain over 5 thousands of frames with 30 fps. To solve this problem, researchers pay attention to video summarization. Video summarization is aimed to represents an original long video with a concise short summary. Processing a concise summary can be much more efficient than processing an original video.

There are some tasks similar to video summarization, such as movie trailer [2] and key frame selection [3]. However, these tasks focus on predicting a single score for frames without considering the temporal dependence between different frames. Thus, some important information could be lost in their results and the completeness of the original video could not be promised. Video summarization task considers the temporal dependence between frames and is expected to generate a short summary, which can cover all the important information and the complete story of the original video. As Figure 1.1, video summarization can be divided into three parts, pre-processing, summarization and post-processing. During the pre-processing, a pre-trained deep CNN is used to extract the vector features for each frame. A pre-trained deep CNN is trained on a very large scale of image dataset, such as ImageNet [4]. The features extracted by the deep CNNs [5][6][7] have been

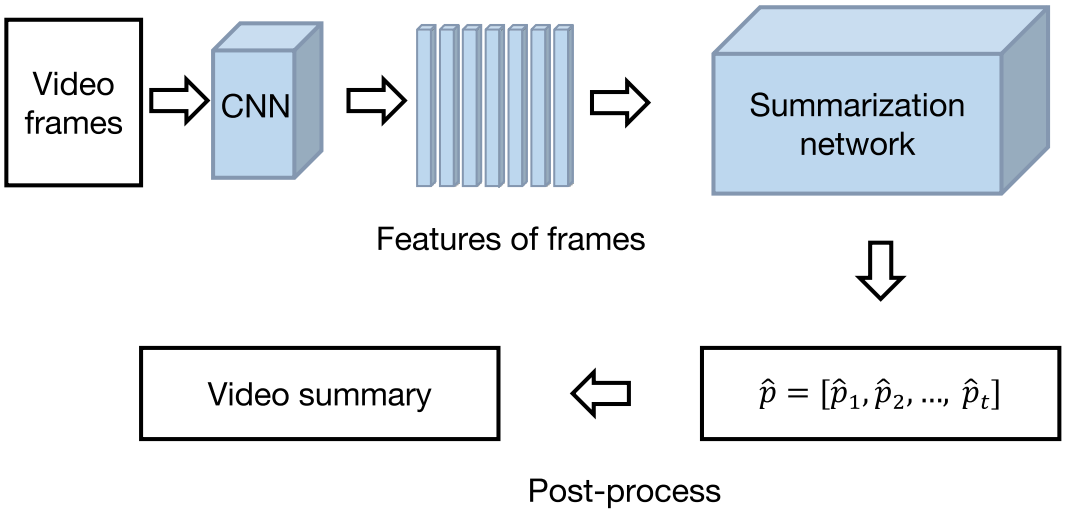

FIGURE 1.1 – Overview of video summarization

applied into several fields of computer vision and achieved a great performance. Then the sequence of frame features are inputted into the summarization network. The summarization network could be composed of different kinds of models, such as recurrent neural network (RNN) and CNN. Then importance scores for each frame are predicted by the summarization network we build. Finally, according to the importance scores and frame features, the video summary is generated after post-processing.

Recent years, researchers have applied deep learning techniques into video summarization. [1] utilizes long short-term memory (LSTM) as summarization network. LSTM can catch up the temporal dependence and predict the importance score for each frame. They also propose a model enhanced by Determinantal Point Process module to improve diversity of result. Temporal segmentation is usually used in the pre-processing as a determinate algorithm. [8] proposes a hierarchical structure, which can predict the temporal segmentation by the neural network. Their model first predicts the temporal segmentation and then the importance score. These kinds of supervised methods require a large number of frame-level annotations. It costs a lot to build a large scale of dataset with annotations for each frame of each video. Besides, annotations tend to be subjectivity when various people are asked to annotate each small temporal interval for the same

video.

Some researchers explore methods requiring no annotations such as in [9][10]. [9] utilizes generative adversarial network with a well designed encoder and decoder. They hold the view that the distance of the distributions of a good summary and the original video should be small. Then they can use a discriminator to distinguish the generated summary and the original video. [10] proposes a reinforcement learning method. They define a diversity-representativeness reward to train the summarization network by policy gradient. However, their reward can only be obtained when the whole summary is generated. This kind of reward is sparse to evaluate a long sequence of actions.

The existing weakly supervised methods require only category labels for each video, such as in [11][12]. However, the existing methods require much more web videos to train the target dataset. Prior knowledge is obtained from the web videos. For example, [12] train a variational autoencoder on the web videos to get a prior distribution. Then their model is trained on the target dataset using the prior distribution as the prior knowledge. This kinds of methods require too many web videos other than the target dataset.

1.2. Reinforcement Learning

Reinforcement learning focuses on how an agent interacts with an environment. As shown in Figure 1.2, the agent takes action a_t according to its current state s_t and receives a reward r_t from the environment. Each action will change its state from s_t to s_{t+1} and lead to positive reward or negative reward. The data $< s_t, a_t, r_t, s_{t+1} >$ required for training are sampled by agent's interacting with environment. Aimed at achieving a good result, the agent adjusts the policy π of action according to the received rewards. In this way, researchers have no need to annotate and collect the data. And it can be widely applied into various complicated tasks. With deep learning technique developing, deep reinforcement learning attracts a lot of attentions from researchers.

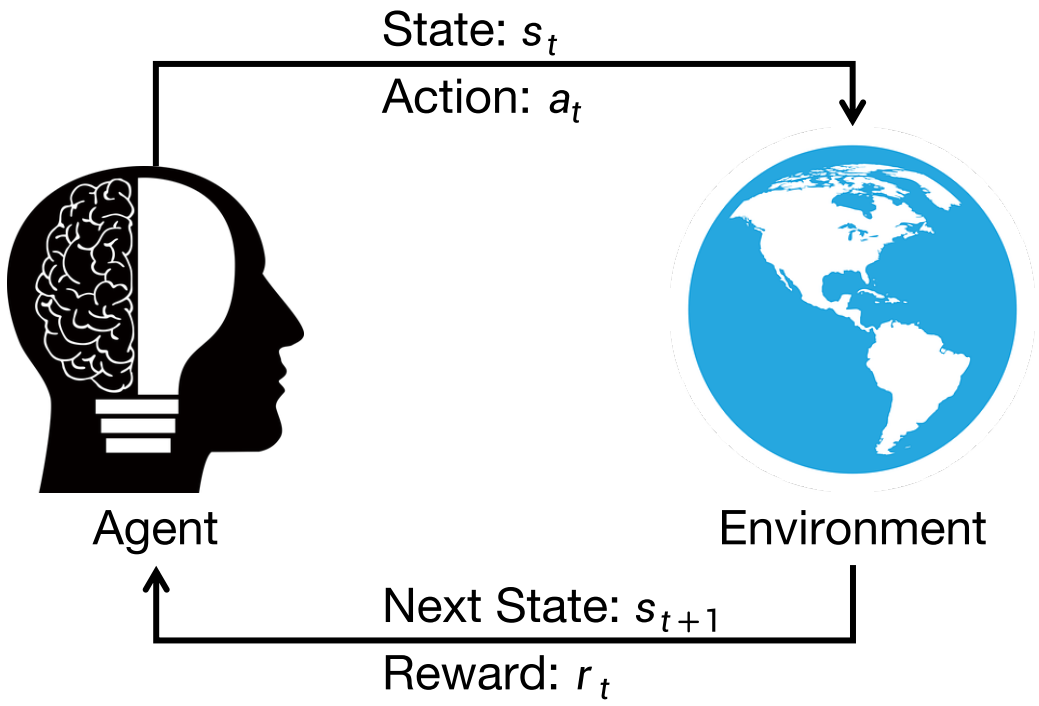

FIGURE 1.2 – Explanation of Reinforcement Learning.

At present, there are two most popular kinds of reinforcement learning methods, Q learning and policy gradients. Q learning method is aimed at approximating the Q function. The Q function is defined as the expected return if we take actions following the current policy.

$$Q^{\pi}(s_t, a_t) = E[R_t]. \tag{1.1}$$

The return is a weighted summing up of successive rewards. It is defined as following:

$$R_t = r_t + \lambda r_{t+1} + \lambda^2 r_{t+2} + ... \tag{1.2}$$

If the Q function can be well evaluated, the optimized policy can be represented as $\pi^* = \text{argmax}_a Q(s, a)$. The Q function is approximated by $< s_t, a_t, r_t, s_{t+1} >$ following the Bellman equation:

$$Q^{\pi^*}(s_t, a_t) = r_t + \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}). \tag{1.3}$$

A deep neural network can be used to model the Q function. Thus, this neural network can be trained by gradient descent with the objective function L. This kinds of Q leaning is Deep Q Network (DQN).

$$L = \frac{1}{2}[r + \lambda \text{max}_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]. \tag{1.4}$$

On the other hand, policy gradients method directly optimizes in the action space. In Q learning, Q function is to evaluate total expected rewards (return) R_t for the current state s_t and action a_t. If we want to know which action to take, we need to compare the returns of all actions available, namely $\text{argmax}_{a_t} Q(s_t, a_t)$. Differently, policy gradient usually utilizes a neural network to predict the probability of action. The best action to take in a specific state should have the highest probability. And when we sample the data $< s_t, a_t, r_t, s_{t+1} >$, all the actions are sampled according to the probability. In policy gradient, the network π_θ (θ is the parameter of network) is also trained to maximize the total expected rewards. According to REINFORCE [13] algorithm, the derivative of the objective function is as follows:

$$\bigtriangledown_\theta J(\theta) = E[\bigtriangledown_\theta log P(a_t) R_t], \tag{1.5}$$

where $P(a_t)$ is the probability of action a_t.

These two kinds of methods have their own strength and weakness. The Q function of Q learning can be too complex to learn in some problems. Policy gradient usually converge faster than DQN but it tends to be local optimal. Q learning usually requires a discrete action space, while policy gradient can be easily applied into continuous action space. As for sampling the data, Q learning show a much better efficiency to sample data and thus it is much more stable than policy gradient.

There are also some methods combining this two kinds of methods together. Actor-critic method use an actor to take action and a critic to evaluate the state-action pair. The actor is trained by policy gradient and the critic is trained to approximate the Q function.

Applying the reinforcement learning into different fields is usually let the agent

take a series of actions. In some cases, the reward is hard to define and we can only receive a global reward when all the actions are taken. This kind of reward is too sparse to evaluate all the actions. As shown in Figure 1.3, four actions are taken and one global reward is received. The black numerical value is the actual reward for each action. But we can't receive it from the environment. The global reward is averaged to evaluate each action. In this way, it can be seen that the second negative action is evaluated as a positive action.

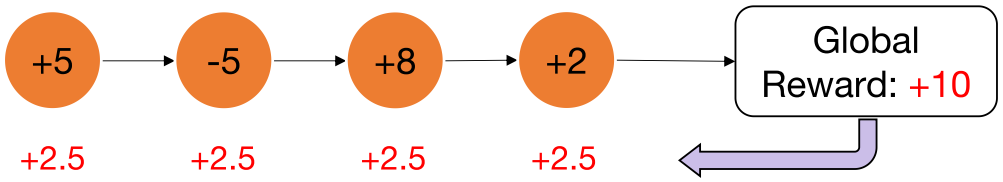

FIGURE 1.3 – Illustration of sparse reward problem.

Hierarchical reinforcement learning is aimed to solve this sparse reward problem. [14] proposes a practical framework to implement the idea of hierarchical reinforcement learning. Their agent consists of a Manager and a Worker. The Manager deals with subtask and learn the information of subtask to help the Worker take action.

1.3. Proposal and Contributions

In this thesis, I aim to improve the reinforcement learning methods for video summarization task. I propose a hierarchical reinforcement learning based video summarization method, which only requires a small number of annotations. The main idea of my proposal is to divide a whole task into several subtasks, where each subtask is assigned a subgoal to achieve. I also define a task-level annotation to enhance the performance. The number of task-level annotations is much smaller and it can also reduce the subjective differences between different annotators who are only required to annotate each subtask. More importantly, my proposal can solve the existing long dependence problem and sparse reward problem. The work [15] published has shown the improvement of this proposal.

There are the main contributions as follows:

- I propose a divide-and-conquer method to divide the whole task into several subtask. In this way, it is possible to solve the long dependence problem.
- A hierarchical structure is proposed based on the divide-and-conquer idea. My agent network consists of two networks, a Manager and a Worker. The Manager is trained evaluate the subtask while the Worker is trained to evaluate each frame within the subtask. The Manager also share the information to the Worker. In this way, the performance could be improved.
- A subgoal and sub-reward are well designed. The subgoal is set by the Manager and is used to guide the Worker to achieve the subgoal for the subtask. The sub-reward gives a reward signal to the agent and it is used with the global reward. With help of them, we can avoid the sparse reward problem.
- A weakly supervised method is proposed. Subtask labels are designed to train the Manager. Compared to the supervised methods requiring frame labels, a much smaller number of annotations is needed. This can reduce the cost of time and labor to build a large scale of video summarization dataset.
- I compare my method with supervised and unsupervised methods. Two kinds of metrics are used to evaluate the performances. Experiments on two benchmark dataset shows the improvement of my proposal.

1.4. Organization of This Thesis

In this thesis, I begin to introduce the related works of video summarization and reinforcement learning in Chapter 2. Then I will introduce my proposal of using hierarchical reinforcement learning for video summarization in Chapter 3. Chapter 4 shows the experiments to evaluate the performance on two benchmark datasets. Finally, I will draw the conclusion and talk about my future work.

Chapter 2

Related Works

2.1. Video Summarization

Video summarization is the task to generate a short summary for a given long original video. Recently, researches on video summarization have been explored extensively and achieved great advances. Summarization networks are designed to process the temporal dependence and predict the importance scores of each frame. Then frames with higher importance scores are included into the generated summary. The existing methods can be roughly divided into three kinds: supervised methods, unsupervised methods and weakly supervised methods.

2.1.1. Supervised Methods

Supervised methods utilize the frame-level annotations to train the summarization network to predict the importance scores [1][8]. The video data is a sequence of frames. Therefore, recurrent neural networks (RNN) are applied to capture the

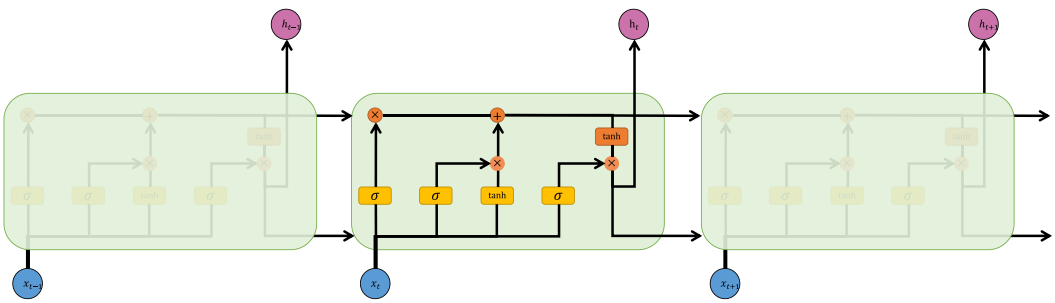

FIGURE 2.1 – The repeating module in an LSTM contains four interacting layers

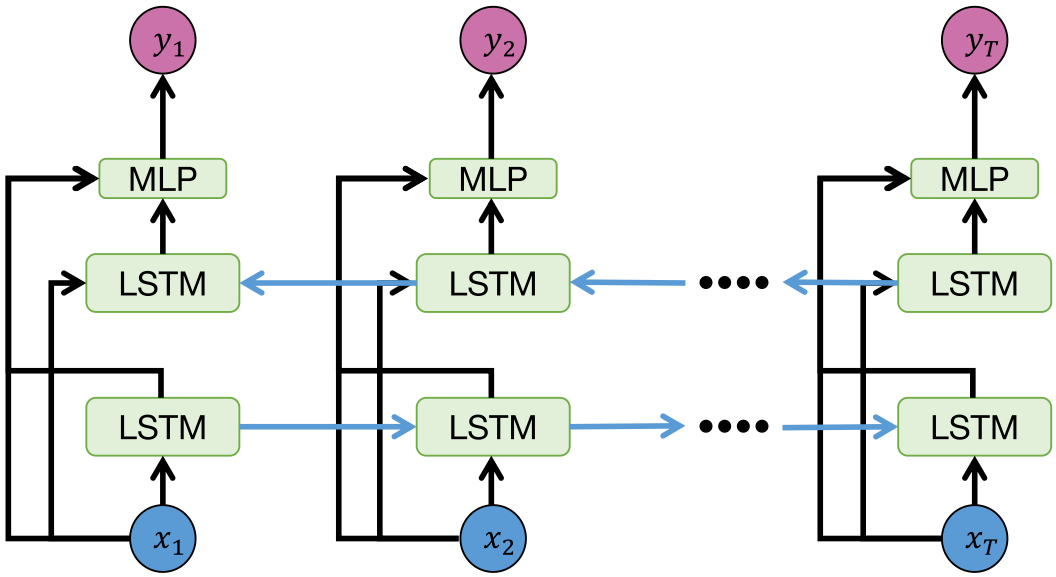

FIGURE 2.2 – The model vsLSTM [1] is implemented by bidirectional LSTM layers.

temporal information of the sequence. There are some variants of RNN and the most popular is long short-term memory (LSTM) [16]. LSTM is designed to deal with vanishing gradient problem and gradient exploding that are the common problems in traditional RNN. Compared to RNN, LSTM can process the long temporal information better by the gate mechanism. As shown in Figure 2.1, a common LSTM unit consists of a cell, an input gate, an output gate and a forget gate. The cell is the values over arbitrary time intervals and the three gates control the information into and out of the cell. In this way, LSTM can enhance the ability of processing temporal sequence data.

[1] began to utilized LSTM as the summarization network to predict the importance scores. They define two kinds of models: vsLSTM and dppLSTM. As show in Figure 2.2, vsLSTM is implemented by bidirectional LSTM layers, which could consider temporal information together in the past and in the future. The hidden outputs $\{h_t^{forward}, h_t^{backward}\}$ of bidirectional LSTM and the frame feature x_t are concatenated to be input into a MLP layer to predict the importance score y_t for each frame.

$$y_t = f_\theta([h_t^{forward}, h_t^{backward}, x_t]), \tag{2.1}$$

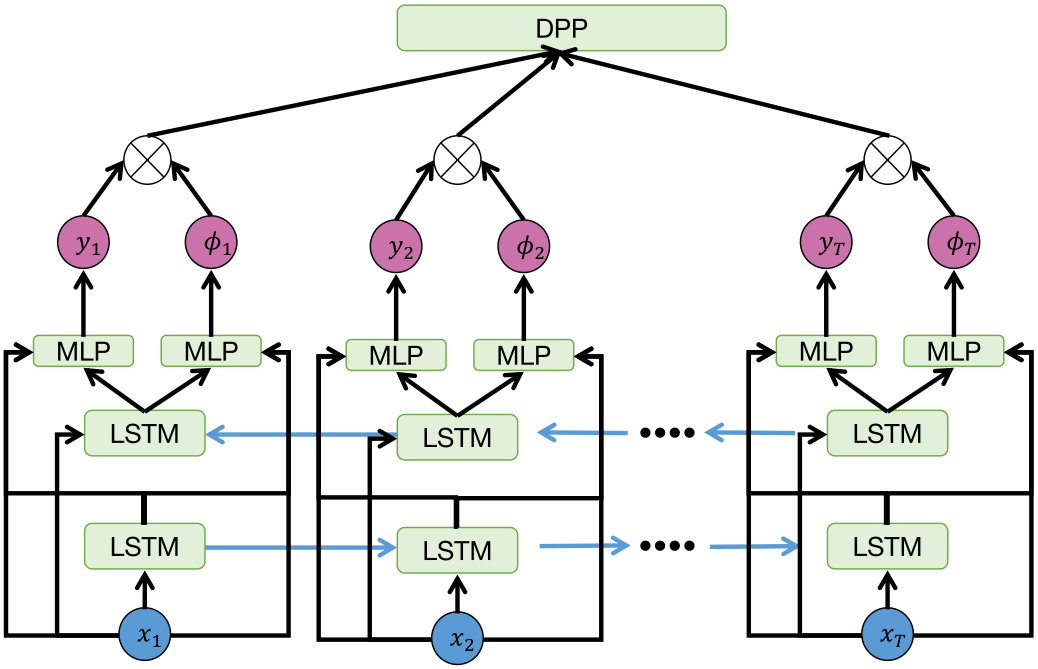

FIGURE 2.3 – The model dppLSTM [1] is an extension of vsLSTM with DPP.

where $f_\theta(\cdot)$ represents the MLP layer and θ is the parameters.

As show in Figure 2.3, dppLSTM is an extension of vsLSTM, which is enhanced by modeling pairwise repulsiveness. Determinantal point process (DPP) is utilized to reduce the redundant frames and increase the diversity of the selected frames. They noted that the diversity of a summary can only be measured when the generation procedure is finished. LSTM has no capability to measure the diversity on the whole summary. It is also at the risk of causing higher recall but lower precision. But with help of DPP, the extensive LSTM can solve these problems.

For a ground set Z of N items (in their case, all frames of a video), an $N \times N$ kernel matrix L stores the pairwise frame-level similarity. DPP encodes the probability to sample any subset from the ground set [17][18]. The probability of a subset z is proportional to the determinant of the corresponding principal minor of the matrix L_z:

$$P(z \in Z; L) = \frac{\det(L_z)}{\det(L + I)}, \tag{2.2}$$

where I is the $N \times N$ identity matrix. If there are two identical items, the determinant of the subset matrix equals zero.

To implement the L matrix, they use the output of LSTM:

$$L_{tt'} = y_t y_{t'} \phi_t^T \phi_{t'}, \tag{2.3}$$

where the similarity between frame x_t and $x_{t'}$ are modeled with the inner product of ϕ_t and $\phi_{t'}$:

$$\phi_t = f_S(h_t^{forward}, h_t^{backward}, x_t), \tag{2.4}$$

$$\phi_{t'} = f_S(h_{t'}^{forward}, h_{t'}^{backward}, x_{t'}). \tag{2.5}$$

These kinds of supervised methods require labels for each frame to train a LSTM-based model. Consequently, it is hard to build a large scale of dataset for video summarization, which cost a lot of time and labour. Besides, the annotations from different users tend to be subjective.

2.1.2. Unsupervised Methods

Collecting annotations for each frame of each video is very tedious. Unsupervised methods with well-designed criteria [19][10] require no annotation. Therefore, a much larger number of videos can be utilized to train the model. Some researches have focused on video summarization by clustering [19][20]. The similar frames are aggregated in the same cluster and the center of each cluster was selected into the final summary. It is also popular to use the generative adversarial network (GAN) [21]. In GAN, there are two networks, one generator and one discriminator. The generator is trained to generate fake data to fool the discriminator. The discriminator is trained to distinguish the ground truth data and the fake data.

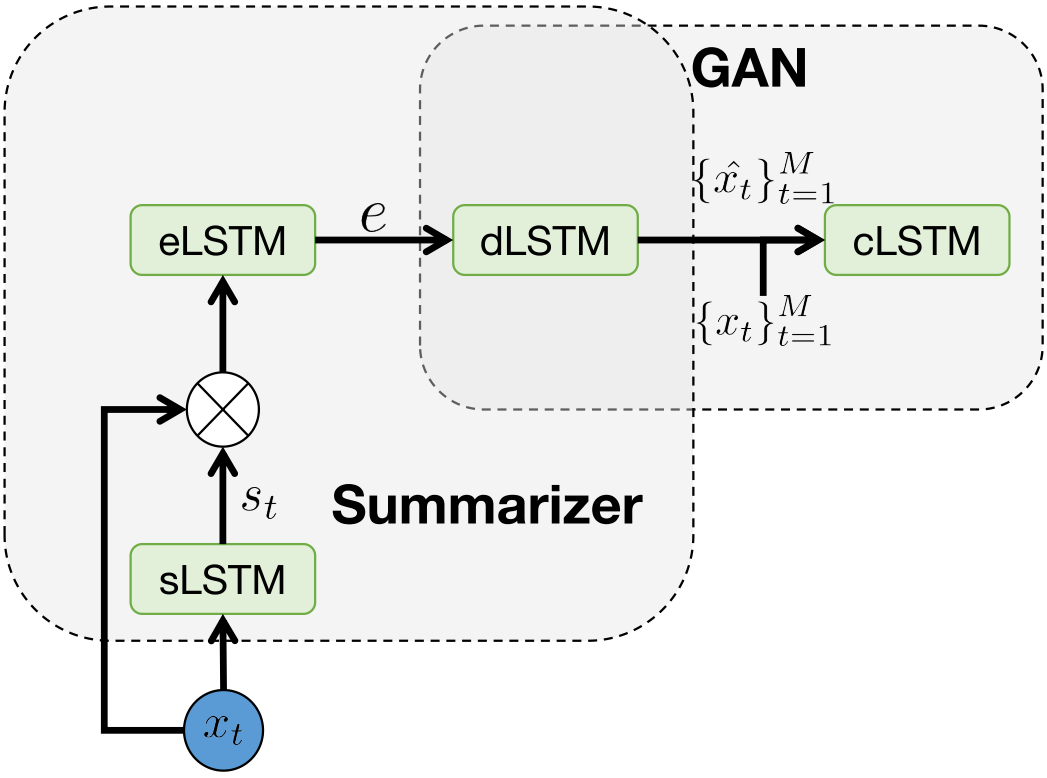

FIGURE 2.4 – The structure of SUM-GAN.

SUM-GAN [9] uses GAN with a defined summarizer. As shown in Figure 2.4, summarizer consists of a selector (sLSTM), a encoder (eLSTM) and a generator (dLSTM). cLSTM is the discriminator. For a sequence of frame features $\{x_1, x_2, ..., x_t, ..., x_M\}$, the sLSTM predicts importance scores $\{s_1, s_2, ..., s_t, ..., s_M\}$ for each frame. Each frame is weighted by the importance scores and then forwarded into the eLSTM. The eLSTM encodes it to a code e. And the dLSTM generates a sequence of frame features $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_t, ..., \hat{x}_M\}$. They hold the view that the distance between the distribution of original videos $\{x_1, x_2, ..., x_t, ..., x_M\}$ and their summaries $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_t, ..., \hat{x}_M\}$ is supposed to be small. Therefore, the cLSTM is trained to distinguish original videos $\{x_1, x_2, ..., x_t, ..., x_M\}$ and generated summaries $\{\hat{x}_1, \hat{x}_2, ..., \hat{x}_t, ..., \hat{x}_M\}$. Their eLSTM and dLSTM can also be considered as an autoencoder (AE) [22].

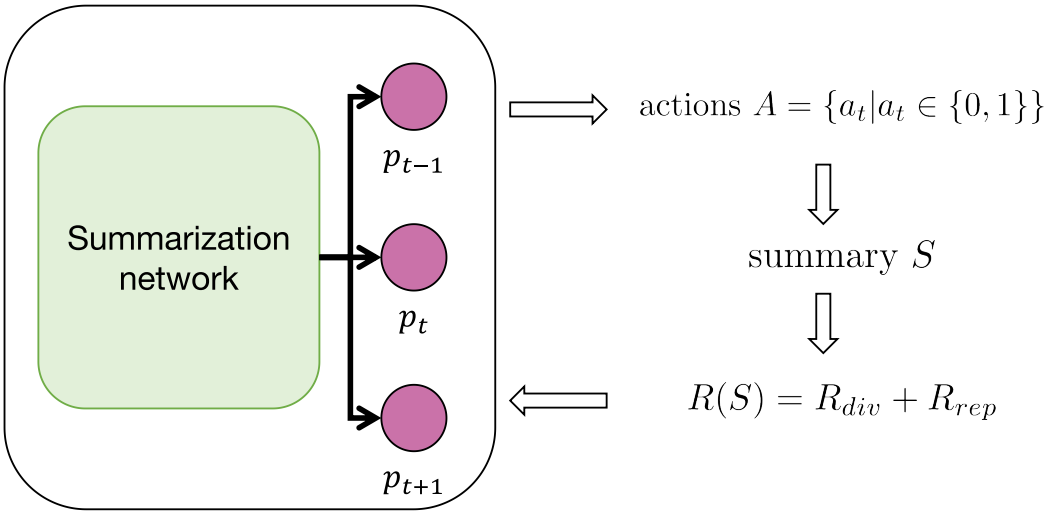

FIGURE 2.5 – The structure of DR-DSN.

DR-DSN [10] uses the reinforcement learning method with well-designed diversity and representativeness rewards. As shown in Figure 2.5, the main difference of DR-DSN from the supervised methods is that the predicted importance scores are regarded as the parameters of action distributions. Their model is also implemented by bidirectional LSTM to predict the importance score for each frame. Each importance score is regarded as a probability p_t of a binary distribution. The action is sampled from the distribution:

$$a_t \sim \text{Bernoulli}(p_t), \tag{2.6}$$

where $a_t \in \{0, 1\}$ represents whether to select the i-th frame into summary or not. The summary is generated by a series of actions. Then the reward is computed using the generated summary. The whole summarization network is trained by policy gradient.

In DR-DSN, they defines a global reward, which consists of the diversity reward and representativeness reward. Diversity reward is defined to measure the degree of diversity of one generated summary. It is computed through measuring the dissimilarity among all the selected frames in the feature space. Given a generated summary $Y = \{y_i | a_{y_i} = 1, i = 1, ..., |Y|\}$, the R_{div} is computed as the mean of the

pairwise dissimilarities among all the selected frames $\{x_1, x_2, ..., x_t\}$:

$$R_{div} = \frac{1}{|Y|(|Y|-1)} \sum_{t \in T} \sum_{t' \in Y, t' \neq t} d(x_t, x_{t'}), \tag{2.7}$$

where $d(\cdot, \cdot)$ is the dissimilarity function:

$$d(x_t, x_{t'}) = 1 - \frac{x_t^t x_{t'}}{||x_t||_2 ||x_{t'}||_2}. \tag{2.8}$$

The R_{div} is aimed to make the generated summary much more diverse. However, this definition doesn't consider the temporal information. And when two frames are far away from each other temporally, they are absolutely diverse from each other. So they set $d(x_t, x_{t'}) = 1$ if $|t - t'| > \lambda$.

Representativeness reward is defined to measure how well the generated summary can represent the original video. To achieve this, they define the representativeness as the k-medoids problem [23]. Particularly, they expect the DS-DSN can choose a set of medoids to minimize the mean of squared errors between video frames and their nearest medoids. Therefore, the R_{rep} is defined as the following:

$$R_{rep} = exp(-\frac{1}{T} \sum_{t=1}^{T} min_{t' \in Y} ||x_t - x_{t'}||_2). \tag{2.9}$$

The R_{rep} encourages the DR-DSN to select frames that are close to the cluster centers in the feature space. These two rewards are combined to be the diversity-representativeness reward:

$$R_{dr} = \frac{1}{2} R_{div} + \frac{1}{2} R_{rep}. \tag{2.10}$$

Note that the R_{dr} is a kind of global reward, which is computed when the whole summary is generated. It is an evaluation for the whole generation. Then the summarization network is trained by policy gradient using this global reward.

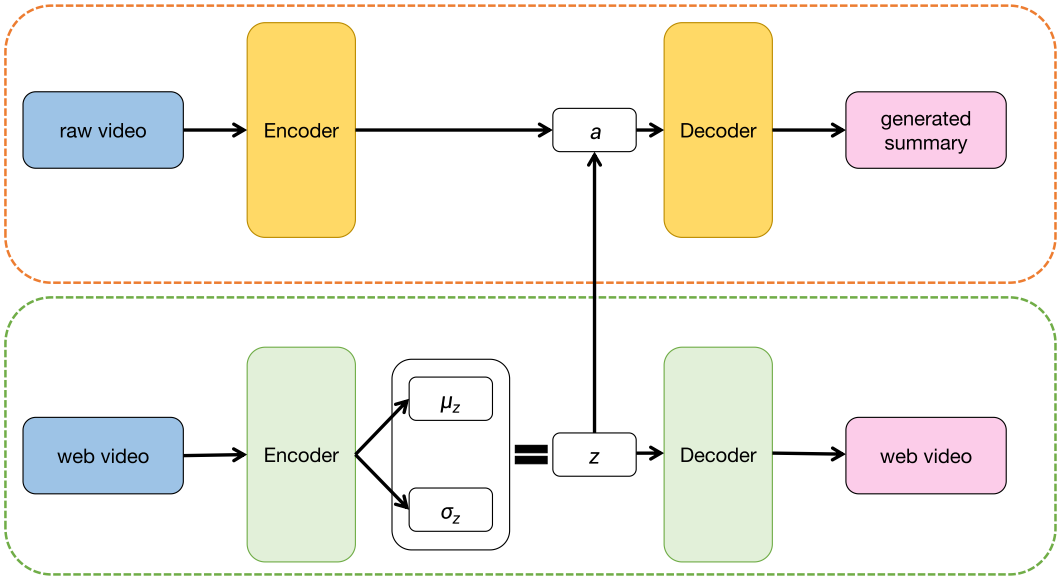

FIGURE 2.6 – Explanation of web prior.

2.1.3. Weakly Supervised Methods

Weakly supervised methods use a smaller number of labels than supervised methods. The existing weakly supervised methods utilize the prior knowledge to train the summarization network. While supervised methods require the ground truth score for each frame of one video, weakly supervised methods require only the category label of one video. Annotating the category of one video costs much less time and labour than annotating the importance for each frame. Nowadays, there are a lot of edited web videos with category labels on web repositories (like YouTube). Therefore, researchers are motivated to utilize the large collection of web videos.

[12] proposes to get a web prior by a large collection of web videos. Their model is variational encoder-decoder. But different from the standard variational autoencoder (VAE) which assumes the latent variable z to be drawn from latent Gaussian (e.g., $p(z) = N(0, I)$), their prior distribution is learned from the web videos with category labels. As show in Figure 2.6, web prior z is gotten by training a VAE with web videos and this web prior is used to help train the other VAE with the targeted video datasets. With data augmentation, the performance of video summarization can be improved.

[11] also proposes to utilize a large collection of web videos. They use videos with category labels to pretrain a classification network. The weights of the pre-trained classification network is freezed. Their summarization network is trained by deep Q-learning. The action is to remove redundant frames from the whole sequence of frames. The generated summary is input into the classification network. They define two rewards to utilize the prior knowledge. The global recognisability reward is to use the classification results of video summaries:

$$r_t^g = \delta(\hat{y} == y) - 5(1 - \delta(\hat{y} == y)) \quad s.t. \quad t = T, \tag{2.11}$$

where $\hat{y} = y$ means that the generated summary can be classified as the expected category. Therefore, this reward will be 1 for the good summary and -5 for the bad summary.

The local relative importance reward is computed when action a_t is taken to remove frame causing the change from s_t to s_{t+1}. The classification network classifies s_t and s_{t+1}, resulting in ξ_t and ξ_{t+1}, which represent the rank of the true category.

$$\begin{aligned} r_t^l &= 0.05(1 - a_t) + h(\xi_t, \xi_{t+1}) \\ s.t. \quad h(\xi_t, \xi_{t+1}) &= tanh(\frac{\xi_t - \xi_{t+1}}{\eta}), \quad t < T, \end{aligned} \tag{2.12}$$

where η is a scaling factor, $h(\xi_t, \xi_{t+1})$ measures the importance of the removed frame. Therefore, if the classification result of the left frame sequence becomes better, a higher reward can be received.

In a word, the existing weakly supervised methods require only category label for each video. However, a large scale of web videos are required to obtain the prior knowledge.

2.2. Reinforcement Learning

Reinforcement learning has been explored a lot in recent years. It focuses on how the agent interacts with an environment. The environment get a reward feedback to the agent. The agent is learned by experience data sampled by interacting with the environment. The way in which reinforcement learning method trains the agent is much more similar with the learning procedure of human beings. It can be applied into various kinds of task. Therefore, reinforcement learning becomes very popular and attracts the attention from researchers.

2.2.1. Deep Q Network

[24] utilized a deep neural network to approximate the Q function and achieved great performance on many Atari games. They use a deep neural network to approximate the Q value function. The input of the Q network is state (image frame in their case). The output of the Q network is the evaluated return value of each possible action. As shown in Section 1.2, DQN is trained by optimizing the objective function L.

$$L(\theta) = E_\theta[(y - Q(s, a; \theta))^2], \tag{2.13}$$

where $y = r + \lambda max_a Q(s, a; \theta)$. Reward r is received from the environment. In other word, experiences of the agent at each time-step can be noted as $< s_t, a_t, r_t, s_{t+1} >$. These experiences are sampled to train the DQN. They propose the experience replay mechanism to store these experiences in a memory. Then they apply the minibatch updates to data samples in the replay memory. Compared with updating the network at each step of experience, this replay mechanism can utilize data more efficiently and avoid the inefficient problems caused by the strong correlation between the consecutive experiences. More importantly, training online could be stuck in a local optima, or even diverge extremely.

[25] proposes to use two same neural networks, double DQN. One is just trained similarly as DQN. The other one is copied from the first model, which is updated at

last episode. Let us note the parameters of the first model as θ and the parameters of the second model as $\theta^{'}$. They have:

$$y = r + \lambda Q(s, argmax(Q(s, a; \theta); \theta^{'}). \tag{2.14}$$

Namely, the first model is used to select the action and the second model is used to evaluate the value. In this way, they can avoid the overoptimistic value estimates caused by using the same network to select and evaluate an action.

[26] proposes dueling DQN, which divides the Q function into state function and advantage function. They notice that in some states, rewards for all actions are similar. They let the state function focus on the state only and the advantage function focus on the action.

$$Q(s, a) = V(s) + A(s, a). \tag{2.15}$$

However, directly using the equation 2.15 has the unidentifiable problem. Although the form is different, it works similarly to the traditional Q function. Therefore, in practical, the following equation is used.

$$Q(s, a) = V(s) + A(s, a) - \frac{1}{||a||} \sum_{a} A(s, a). \tag{2.16}$$

DQN techniques have developed a lot these years. In [27], researchers verify the existing techniques and combine most of improvements together to achieve great performance including DQN, DDQN, Prioritized DDQN, Dueling DDQN, A3C, Distributional DQN and Noisy DQN.

2.2.2. Policy Gradient

Policy gradient is directly optimized in the action space. As shown in Section 1.2, the derivative of the object function is

$$\bigtriangledown_{\theta} J(\theta) = E[\bigtriangledown_{\theta} logP(a_t) R_t], \tag{2.17}$$

[28] proposes a deterministic policy gradient (DPG) algorithm with continuous action spaces. Rather than predict the probability of action, the agent generates the action $a = \pi_\theta(s)$ deterministically. They also introduce the deterministic actor-critic algorithms, which combine the Q learning and policy gradient. There are an actor and a critic. The critic is also trained to evaluate the action for each state. And the actor is trained by gradient but the return is evaluated by the critic.

$$\bigtriangledown_\theta J(\theta) = E[\bigtriangledown_\theta log P(a_t) Q(s_t, a_t)], \tag{2.18}$$

Actor-critic method combines the strength of DQN and DPG. It also uses the experience replay to improve the data efficiency. Actor-critic method shows a good performance and becomes popular for many fields.

2.2.3. Hierarchical Reinforcement Learning

The sparse reward problem will occur if the reward is only received after a series of actions. Practically, if we apply reinforcement learning in many fields, such as text generation, natural language processing [29][30][31] and object tracking [32][33], our agent usually needs to take a series of actions to finally obtain a reward. However, in such a situation, the sparse reward makes it difficult and inefficient to train a model [34].

Hierarchical reinforcement learning is a promising technology to solve this problem. The basic idea is to define the agent as multi-layer structures dealing with different subtasks to learn several sub policies. The macro policy is learned to choose different sub policies for subtasks. This kind of method can achieve a great performance effectively with the help of domain knowledge to divide the task and define the subtasks well. [14] proposed a flexible and end-to-end framework for hierarchical reinforcement learning. As shown in Figure 2.7, they propose a schematic of FeUdal networks. They divided an agent into the Manager, which learns what the subtask is and the Worker, which learns to finish the subtask as well as possible. For the Manager, it learns a latent state representation s_t and outputs a goal g_t. For the Worker, it predicts the probability of action according

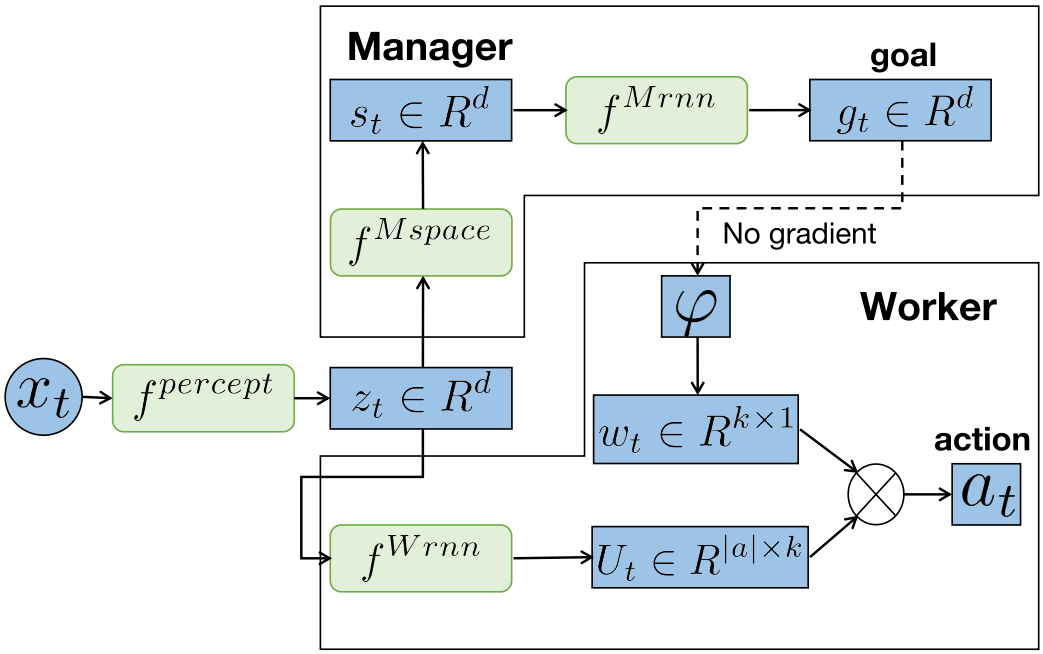

FIGURE 2.7 – Illustration of FeUdal networks

to the state, the environment and the goal from the Manager. The observation x_t from the environment is input into a shared perceptual module layer. The perceptual module layer output a shared intermediate feature z_t, of which dimension is usually smaller than one of x_t. Although the goal vector is input from the Manager to the Worker, there is no gradient between the Manager and the Worker. In other word, the goal vector is detached and input to the Worker.

Recently, many researchers have applied reinforcement learning to their own fields. However, in fields including game, image editing and so on, there are always a series of actions to take. Sparse reward problems occur because in most cases only a sparse global reward can be received after all actions have been taken. [35] utilized hierarchical representations and made the agent learn an internal reward signal to complement the sparse reward. They achieved a great performance in a popular 3D multiplayer first-person video game. [36] applied hierarchical reinforcement learning to text generation. Their method was based on GANs, where the generator consisted of the Manager and the Worker. The Manager received the Leak information from the Discriminator to learn a subgoal for the Worker. [37] focused on the visual storytelling task where a text sequence was generated

according to an ordered image sequence. The Manager took features as input of images and then generated a topic distribution as a subgoal. The Worker generated the text conditioned on the subgoal.

Chapter 3

Approaches

Our proposed method regards the whole video summarization task as the combination of subtasks. In this way, we could define a sub-reward for each subtask to avoid the sparse reward problem. The hierarchical architecture consists of a Manager and a Worker. The Manager sets a subgoal for each subtask. The Worker takes its action following the subgoal. In the following discussion, we define that the whole task is separated into N subtasks. Each subtask processes n frames. The whole task includes $N \times n$ frames. For each step, the input of the Manager is a sequence of frames in one subtask with size n, represented as $[x_{i,1}, x_{i,2}, ..., x_{i,n}]$. The Manager sets a subgoal g_i ($i \in [1, N]$). Conditioned on the subgoal g_i, the Worker predicts the importance score $\hat{p}_{i,t}$ ($t \in [1, n]$) for each frame within i-th subtask. The whole task can be achieved better when the subtasks are achieved well. Our work follows the most common procedures of video summarization task. Therefore, we will introduce the basic procedures first and then focus on the proposal.

3.1. Basic Procedures

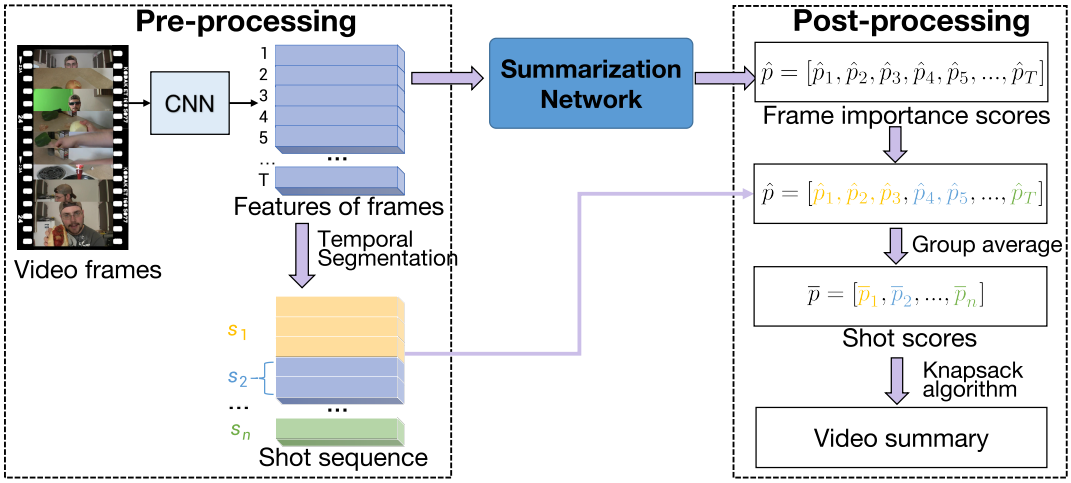

FIGURE 3.1 – Basic procedures of deep video summarization

Video summarization converts an original long video into a short and concise summary. As shown in Figure 3.1, the whole framework consists of several procedures including pre-processing, summarization, and post-processing.

A sequence of video frames is input into a pre-trained deep CNN network. The CNN network is trained for image classification on a large scale of image dataset. It is proved that features extracted by the pre-trained CNN network can be applied to solve many computer vision tasks. Then the features of each frame will be extracted. The former work [1] provides a dataset that uses GoogLeNet to extract the features of each frame. For a fair comparing, I follow them to obtain the sequence of features.

Then the sequence of frame features is subsampled by 2 fps. The summarization network takes the frame sequence as input and predicts the importance scores for each frame. The summary is generated by selecting frames conditioned on the importance score. However, it tends to be unnatural and flashed if we generate summary directly by selecting frame by frame.

In order to promise the generated summary to be natural, temporal segmentation is usually used to divide the sequence of frames into different groups as shots. The most common temporal segmentation method is a kernel temporal

segmentation (KTS) algorithm. KTS algorithm partitions the sequence of frames into shots considering the similarity among frame features [38]. The importance score of each shot is calculated as the average score of the frames included within the shot.

$$\overline{p}_i = \frac{1}{|s_i|} \sum (\hat{p}_j; j \in s_i) \tag{3.1}$$

We select the shot with higher importance score under a given limit of the video duration. Considering that each shot includes a variable number of frames, this kind of selection is of the knapsack problem and I follow the work in [10] to use a near-optimal solution by dynamic programming [23] as the post-processing. Then the summary is generated by combining the selected shots.

3.2. Hierarchical Structure

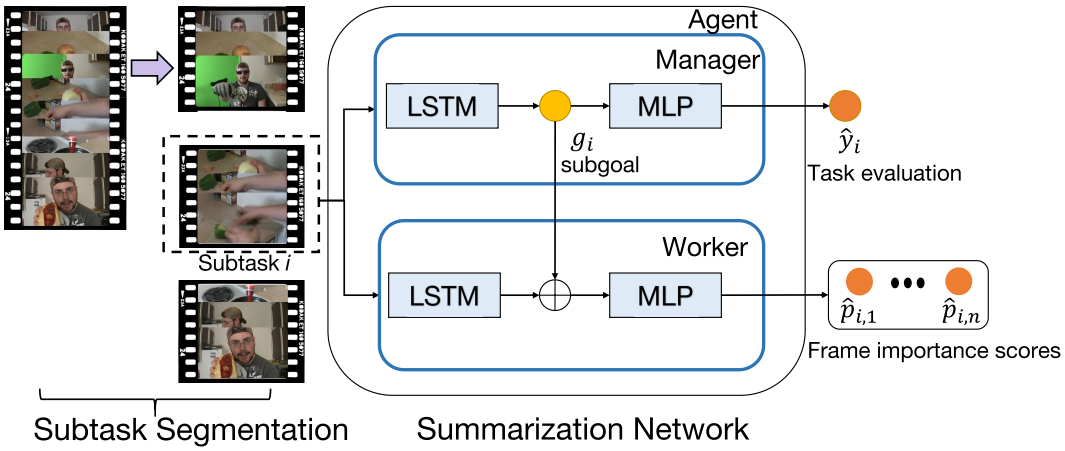

FIGURE 3.2 – Hierarchical structure of proposal

As shown in Figure 3.2, the proposed summarization network uses reinforcement learning with a hierarchical architecture. Note that the hierarchical reinforcement learning is already proposed in [14], but this is the first work to apply hierarchical reinforcement learning to video summarization to the best of my knowledge. Besides, how to divide the whole task into a hierarchical manner requires careful design, which is also my technical contribution. My proposal pays attention not only to the quality of the whole generated summary, but also to the quality of

subtasks. For the quality of the whole summary, I apply a widely used global reward defined as diversity-representativeness reward proposed by [10]. Then, I regard the total summarization task as several subtasks and design an agent with the hierarchical structure using a kind of sub-reward representing the quality of the subtask.

The agent consists of two recurrent neural networks: one is called Manager and the other is called Worker. The Manager predicts a task evaluation for each subtask. And in the meanwhile, the intermediate latent vector of the Manager is detached and input into the Worker. The intermediate latent vector is used as the subgoal for each subtask. In order to achieve the corresponding subgoal, the Worker deals with the frames within the subtask. Both Manager and Worker are implemented by LSTM. I train the Manager with a smaller number of ground-truth annotations compared to supervised methods, and the Worker with the REINFORCE algorithm [13], which belongs to a family of reinforcement learning methods.

In the following section, I are going to introduce the Manager network, the Worker network, and two kinds of rewards used to train the agent in detail.

3.3. Manager Network

Task-level labels can be annotated directly by human beings. Considering that my experiments are conducted on the existing datasets, we can directly derive the task-level labels from the existing frame-level labels of the datasets.

As shown in Figure 3.3, I define the task-level label y_i as 1 if there exists one key frame in a sub sequences and 0 otherwise.

$$y_i = \begin{cases} 1 & \text{if } p_{i,t} = 1, \exists t \in [1, n] \\ 0 & \text{otherwise} \end{cases}, \tag{3.2}$$

where $p_{i,t}$ is the ground-truth label for each frame.

Task-level labels are utilized to train the Manager. It is obvious to see that the number of task-level labels is much smaller than the frame-level labels. And if the size of task becomes bigger, the number of task-level labels becomes smaller.

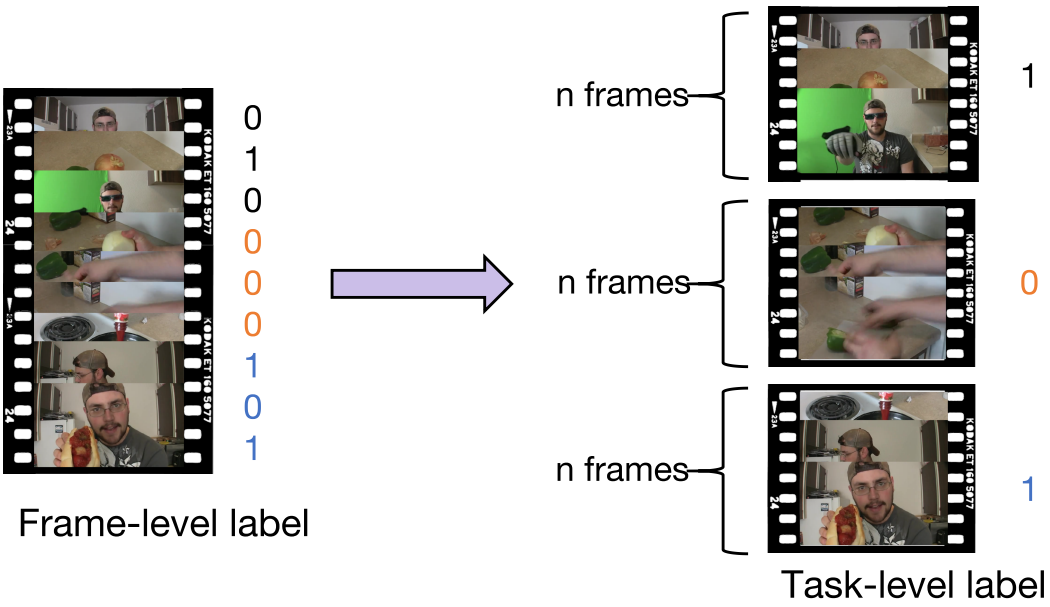

FIGURE 3.3 – Illustration of task-level label

The Manager consists of an LSTM and multilayer perceptron (MLP). As mentioned above, for each step the Manager takes the sub sequence of $[x_{i,1}, x_{i,2}, ..., x_{i,n}]$ (n is set empirically) as input. And then we take the last hidden state as the subgoal g_i of the i-th subtask.

$$h_{i,t} = f_{\beta_m}\left(x_{i,t}, h_{i,t-1}\right), x_{i,t} \in [x_{i,1}, x_{i,2}, ..., x_{i,n}], \tag{3.3}$$

$$g_i = h_{i,n}, \tag{3.4}$$

$$\hat{y}_i = \text{sigmoid}\left(w_m \cdot g_i + b_m\right), \tag{3.5}$$

where f_{β_m} represents the LSTM of the Manager with parameter β_m, w_m and b_m are parameters of MLP, $\hat{y}_i$ is the predicted probability of whether the i-th sub

sequence of frames includes a key frame and $h_{i,t}$ is the hidden state ($h_{i,0} = h_{i-1,n}$ and $h_{0,0}$ is zero-initialized).

3.4. Worker Network

The Worker is also composed of an LSTM and MLP. The whole sequence of frames is input into the Worker subtask by subtask. For each subtask, the Worker outputs the importance score for each frame conditioned on the subgoal given by the Manager. Dealing with each subtask, our Worker takes a sub sequence of frames $[x_{i,1}, x_{i,2}, ..., x_{i,n}]$ (n is set empirically) as input and a subgoal g_i given by the Manager. The Worker predicts the importance scores for each frame.

$$h_{i,t} = f_{\beta_w}\left(x_{i,t}, h_{i,t-1}\right), x_{i,t} \in [x_{i,1}, x_{i,2}, ..., x_{i,n}], \tag{3.6}$$

$$h^{'}_{i,t} = w_1\left[g_i, h_{i,t}\right] + b_1, t \in [1, n], \tag{3.7}$$

$$\hat{p}_{i,t} = \text{sigmoid}\left(w_w \cdot h^{'}_{i,t} + b_w\right), \tag{3.8}$$

where f_{β_w} represents the LSTM with parameter β_w, $h_{i,t}$ is the hidden state ($h_{i,0} = h_{i-1,n}$ and $h_{0,0}$ is zero-initialized), w_1 and b_1 are parameters of a linear layer that combines the subgoal and the hidden state, w_w and b_w are parameters of MLP, $[g_i, h_{i,t}]$ is to concatenate two vectors and $\hat{p}_{i,t}$ is the predicted importance score for frame $x_{i,t}$.

After predicting the importance score of each frame, we regard the importance score as the probability parameter of a Bernoulli distribution. This determines the probability of a frame being selected for the summary.

$$a_{i,t} \sim \text{Bernoulli}\left(\hat{p}_{i,t}\right), a_{i,t} = 0 \text{ or } 1, \tag{3.9}$$

where $a_{i,t} = 1$ is the action and means that $x_{i,t}$ is selected in the summary.

To evaluate the policy of our agent, we sample actions from $N \times n$ Bernoulli distributions. Following these actions, we can get the generated summary. Then we can use the reward defined by us to evaluate the generated summary.

3.5. Reward Function

[10] defines the diversity-representativeness reward R_{dr}. This reward can only be computed using the whole generated summary. Thus, it could be regarded as a kind of global reward. This reward is combined by two parts: diversity reward R_{div} and representativeness reward R_{rep}.

R_{div} is used to measure the diversity of the generated summary and it is computed by summing up the dissimilarity of frame feature within the selected subset. It is defined as following:

$$R_{div} = \frac{1}{|Y|(|Y|-1)} \sum_{t \in Y} \sum_{t' \in Y, t' \neq t} d\left(x_t, x_{t'}\right), \tag{3.10}$$

$$d\left(x_t, x_{t'}\right) = 1 - \frac{{x_t}^T x'_t}{||x_t||_2 ||x_{t'}||_2}, \tag{3.11}$$

where $d\left(\cdot, \cdot\right)$ is the dissimilarity function and $t \in Y$ means that frame x_t is selected into the summary.

R_{rep} is to measure how well the generated summary can represent the original video and it can be obtained by considering the minimal distance between each frame feature with others within the selected subset.

$$R_{rep} = \exp\left(-\frac{1}{n \times N} \sum_{t=1}^{n \times N} \min_{t' \in Y} ||x_t - x_{t'}||_2\right), \tag{3.12}$$

where $t \in Y$ means that frame x_t is selected into the summary.

Then the diversity-representativeness reward R_{dr} is obtained by summing up these two rewards:

$$R_{dr} = \frac{1}{2} R_{div} + \frac{1}{2} R_{rep}. \tag{3.13}$$

This diversity-representativeness reward is given for a generated summary. However, as mentioned before, it is too sparse to evaluate a series of actions. Therefore, we also define a sub-reward R_{sub} to evaluate how well our Worker achieves the subgoal. We compute the average of the outputs of our Worker and compare it with the importance scores of the subtask predicted by our Manager. The sub-reward R_{sub} is defined as follows:

$$\hat{p}_i = \frac{1}{n} \sum_{t=1}^{n} \hat{p}_{i,t}, \tag{3.14}$$

$$R_{sub} = \exp\left(-\frac{1}{N} \sum_{i=1}^{N} ||\hat{p}_i - \hat{y}_i||_2 \right), \tag{3.15}$$

where $\hat{p}_i$ is the average of the probabilities within a subtask, and we use an exponential function to rescale it to get the R_{sub} as our sub-reward.

Finally, the whole reward for our agent is obtained by combining the global reward R_{dr} and the sub-reward R_{sub}:

$$R = \alpha R_{dr} + (1 - \alpha) R_{sub}, \tag{3.16}$$

where α is the hyperparameter.

3.6. Optimization

As shown in Figure 3.4, the Manager is trained by computing the cross entropy loss using task evaluation $\hat{y}_i$ and task-level label y_i, which is defined at Equation 3.2. The loss function of the Manager is as follows:

$$L_m = -\frac{1}{N} \sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i), \tag{3.17}$$

where N is the number of subtasks which a video is divided into, and y_i and $\hat{y}_i$ are defined in Eq. 3.2 and Eq. 3.5, respectively.

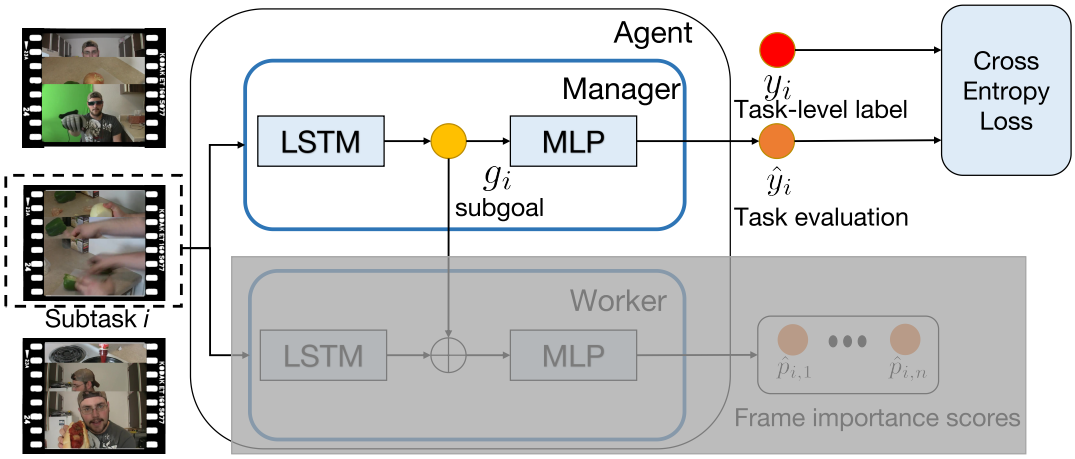

FIGURE 3.4 – Explanation of training the Manager.

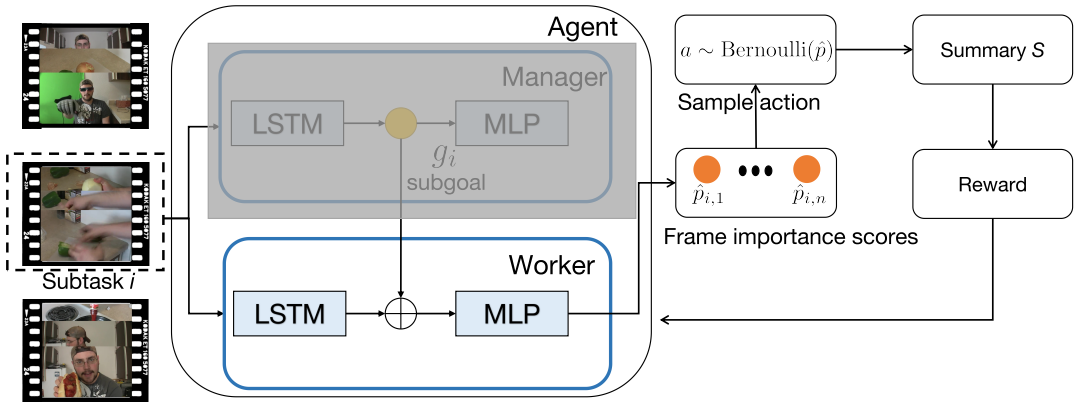

FIGURE 3.5 – Explanation of training the Worker.

As shown in Figure 3.5, the Worker predicts the importance scores first. The importance scores for each frame are regarded as the parameter of the Bernoulli distribution. Actions are sampled from each Bernoulli distribution to select or not the frame into summary. Reward as defined in Equation 3.16 is obtained to evaluate the generated summary. Then the Worker is trained by REINFORCE algorithm [13] with the received reward. It is aimed to learn a policy π_{θ_W} (θ_W is parameter of Worker in our case) through maximizing the expected rewards.

$$J(\theta_W) = E_{p_{\theta_W}(a_{1:n\times N})}[R], \tag{3.18}$$

where $p_{\theta_W}(a_{1:n\times N})$ is the sequence of actions for the whole task. Following the REINFORCE algorithm, we can compute the derivative of the objective function $J(\theta_W)$ in terms of θ_W as follows:

$$\nabla_{\theta_W} J(\theta_W) = E_{p_{\theta_W}(a_{1:n\times N})}\left[R\sum_{t=1}^{n\times N}\nabla_{\theta_W}\log\pi_{\theta_W}(x_t)\right]. \tag{3.19}$$

We approximate R by letting the agent take action c times. To avoid the high variance, we subtract a constant b from the reward.

$$\nabla_{\theta_W} J(\theta_W) \approx \frac{1}{c}\sum_{i=1}^{c}\left[\sum_{t=1}^{n\times N}(R_i - b)\nabla_{\theta_W}\log\pi_{\theta_W}(x_t)\right]. \tag{3.20}$$

Chapter 4

Experiments

In this section, I will introduce the experiments I have done and analyze the results. The content includes dataset, evaluation metrics, training setting and evaluation.

In the dataset section, I am going to introduce two benchmark datasets, SumMe and TVSum. Both of them offer videos with multi-user annotations. I will show the detail information of datasets.

In the evaluation metrics section, I will illustrate two metrics to evaluate the performance, F score and rank correlation coefficient. F score is computed by the intersection between two sets of frames while rank correlation coefficient is computed by two ranking sequences.

In the training setting section, I will introduce the setting for training and comparing the performance. And finally in the evaluation section, I will first present the results evaluated on F score metrics and then results evaluated on rank correlation coefficient. Visualized results are also given. Then I will analyze and discuss the performance for different methods.

TABLE 4.1 – Information of datasets

Dataset	Number of video	Information
SumMe	25	a variety of events
TVSum	50	YouTube videos (10 categories)
OVP	50	Documentary videos
YouTube	39	YouTube videos (Sports, News, etc)

4.1. Dataset

Our proposal is evaluated on two benchmark datasets: SumMe [39] and TVSum [40]. SumMe includes 25 videos ranging from 1 to 6 minutes and for each video there are annotations from 15 to 18 users. TVSum includes 50 videos ranging from 2 to 10 minutes, and for each video there are annotations from 20 users.

Besides, we also consider OVP[1] and YouTube [41], which are annotated with keyframe-based summarization. We follow [1] to process them and get the ground-truth set of keyframes. These two datasets are used as augmentation during the training phrase.

The existing datasets for video summarization offers different ground-truth annotations. There are three kinds of annotations in the four datasets we use: 1) selected keyframes, 2) interval-based keyshots, and 3) frame-level importance scores. For selected keyframes, the annotation is like $gt = [0, 1, 0, ..., 0, 0, 1]$, where $gt_i = 1$ means the i-th frame is annotated into summary. For interval-based keyshots, the annotation is like $gt = [1, 1, 1, 0, 0, 0, ..., 1, 1, 1]$, where there are a lot of successive frames are annotated simultaneously. For frame-level importance score, the annotation is like $gt = [0.5, 0.9, 0.1, ..., 0.2, 0.7]$, where every frame has a importance score. Following [1], we can convert the different annotations into the same format.

From keyframes to keyshots and frame-level scores: Temporally segmentation is first applied to the video. For each segments, which contains at least one keyframe, it is annotated as a keyshot. The importance scores of all frames in a keyshot is 1; otherwise, 0.

[1]Open video project: https://open-video.org/

From keyshots to keyframes and frame-level scores: Keyframes can be selected from each keyshot randomly or by the middle. And the importance scores of frames in keyshots are annotated as 1.

From frame-level scores to keyframes and keyshots: Shots are gotten by temporal segmentation. Then shot-level scores are the average of the frame scores within each shot. Select several shots to achieve the highest sum of scores as keyshots with the total duration under a threshold. The frame with the highest importance score in each keyshot is selected as the keyframe.

As [1] does, a single ground-truth set of keyframes is created for training from multiple user-annotated ones of each video. And our task-level labels are gotten from the single ground-truth keyframes. During the test, multiple user annotations are used to compute the metrics. It is noted that SumMe dataset only offers the importance scores for keyframes. Namely, the scores for most frames are indiscriminately remarked as 0 in SumMe dataset. Therefore, we don't compute the rank correlation coefficient for SumMe dataset.

4.2. Evaluation Metrics

4.2.1. F score

In many deep learning tasks, precision and recall are widely utilized to evaluate the performance. As shown in Figure 4.1, precision measures how many selected items are relevant and recall measures how many relevant items are selected. Take binary classification for example. Predictions can be divided into four kinds: 1) true positives, 2) false positives, 3) false negatives and 4) true negatives. When the ground truth of one sample is 1, this sample is true positives if it is predicted as 1; otherwise, this sample is false negatives. When the ground truth of one sample is 0, this sample is false positives if it is predicted as 1; otherwise, this sample is true negatives. Precision is the ratio of true positives to the sum of true positives and false positives. Recall is the ratio of true positives to the sum of true positives and false negatives. Precision can be regarded as a measure of quality, while recall can be regarded as a measure of quantity.

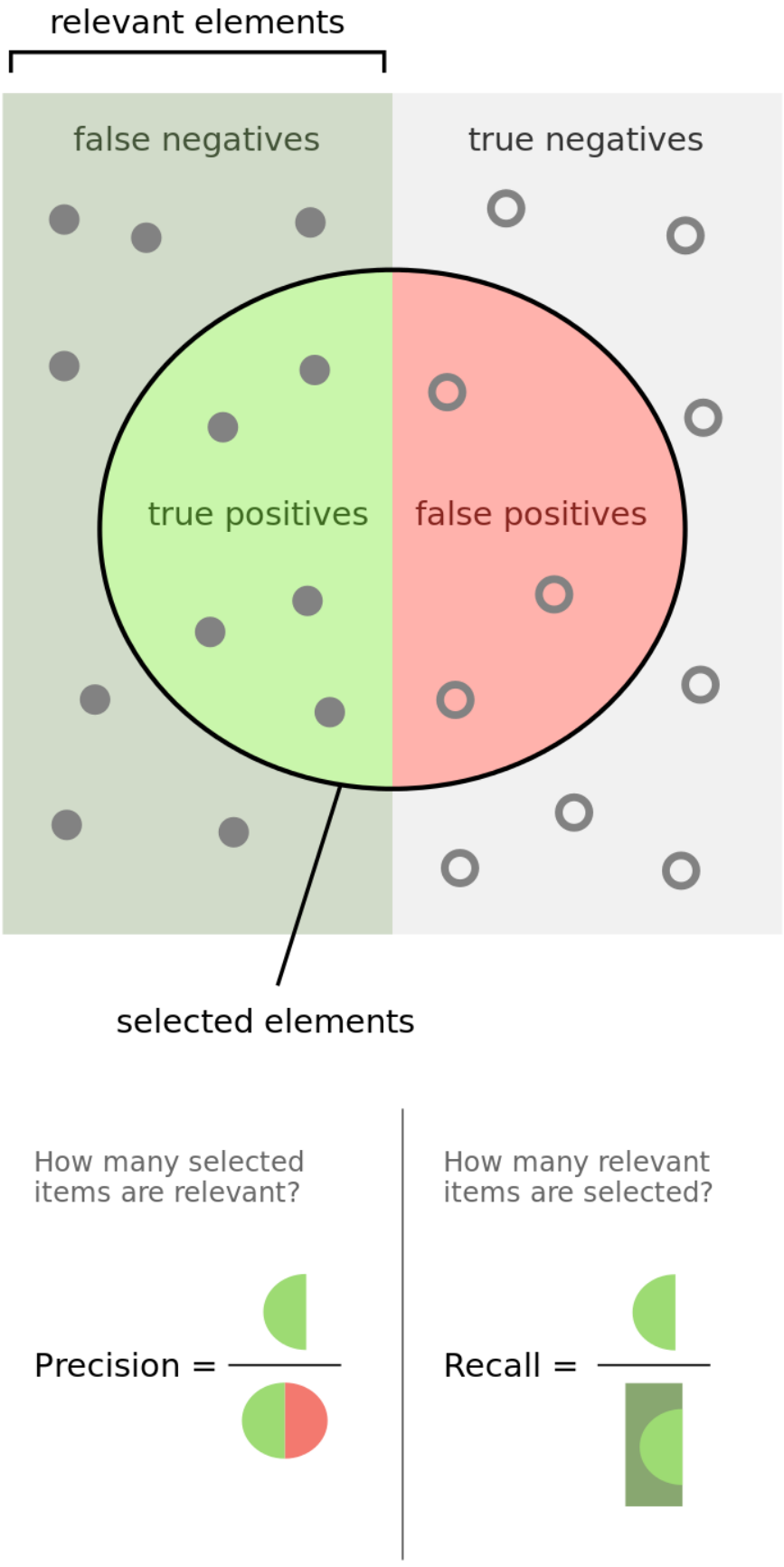

FIGURE 4.1 – Definition of precision and recall

Different tasks have different requirements for precision and recall. To consider a balance between precision and recall, F score is used to evaluate the performance. F score is a harmonic mean of precision and recall. It is defined as follows:

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{4.1}$$

F score is widely used to evaluate the performance of video summarization. However, the generated summary and the ground-truth are a set of frame sequence. Therefore, we need to define the precision and recall for video summarization. Considering the intersection of two videos, the precision and recall could be defined as follows:

$$Precision = \frac{A \bigcap B}{A}, \tag{4.2}$$

$$Recall = \frac{A \bigcap B}{B}, \tag{4.3}$$

where A is the generated summary and B is the ground-truth summary. Then the F score is computed following Equation 4.1

4.2.2. Rank correlation coefficient

In statistics, rank correlation coefficient is a metric to measure the similarity between two rankings. Rank correlation coefficient is between -1 and 1. The higher coefficient means that the agreement between two rankings is higher. There are some popular rank correlation coefficients to measure the agreement including Kendall's τ, Spearman's ρ, Goodman and Kruskal's γ, Somers'D and so on.

[42] proposed to use rank order statistics to evaluate video summarization. As introduced in Chapter 3, summarization network predicts importance scores first. And then the summary is generated according to the importance scores. F score metric is computed using the generated summary. For summarization methods based on importance scores, [42] claimed that the random methods can achieve comparable performance on F score metric because of well-designed pre-processing and post-processing. Therefore, they reported that considering the importance

score rank order between the prediction and the human annotation is a much better metric without the effect of post-processing. They use Kendall's τ [43] and Spearman's ρ [44] correlation coefficients.

Kendall's τ and Spearman's ρ are both non-parametric rank correlation coefficients. Kendall's τ are computed on concordant and discordant paris. Compared to other coefficients, the distribution of Kendall's τ has a better statistics properties. Spearman's ρ are computed on deviations. The numerical value of it is usually much more bigger than Kendall's τ.

SumMe dataset only offers the importance score for keyframe. Namely, importance scores for most of frames are annotated as 0 indiscriminately. Therefore, we only compute the rank correlation coefficients on TVSum dataset.

4.3. Training Setting

Following [1], the experiments are conducted on three settings.

- Canonical: we train our model by 5-fold cross validation (5FCV).
- Augmented: for a target dataset, we randomly divide the target dataset into 5 folds. One fold is used for testing. Others are augmented with OVP and Youtube datasets for training. It is assumed that augmentation can improve the performance by using more data for training.
- Transfer: for our target dataset (SumMe or TVSum), the other three datasets are used as training data. It is interesting to see whether the training model can be used to new datasets.

We train the Manager and the Worker iteratively. We approximate the expected rewards with $c = 10$ and set $\alpha = 0.5$. The size of a subtask is set as $n = 20$. The pseudocode is shown in Algorithm 1.

Algorithm 1 Iteratively Training for Manager and Worker

agent: G_{θ_M,θ_W}, initialize the θ_M and θ_W
for epoch < max epoch **do**
  for each video in training data **do**
    train the manager G_{θ_M}
    get the subgoal g_i from manager
    $\hat{p_t} = G_{\theta_W}(x_t, g_i)$
    importance score $\hat{p} = [\hat{p_1}, \hat{p_2}, ..., \hat{p_n}]$
    Worker takes action $a \sim \text{Bernoulli}(\hat{p})$
    for c **do**
      get Reward $R+ = \alpha R_{dr}(a) + (1-\alpha)R_{sub}$
    end for
    $R = R/c$
    update θ_W by REINFORCE as Eqt. (17)
  end for
end for

4.4. Evaluation

4.4.1. Quantitative Evaluation

To show that our subtasks help to improve the performance, we compare our model with the different sizes of subtask. Our model shows the best performance when the size of subtask is set as $n = 20$. It can be seen in Figure 4.2 that when the size of subtask is 1, the performance is better than that of size 5. It is reasonable because we define the sub-reward using the average of the outputs of the Worker. Therefore, if the size of subtask is not large enough, the bias of the sub-reward could be large.

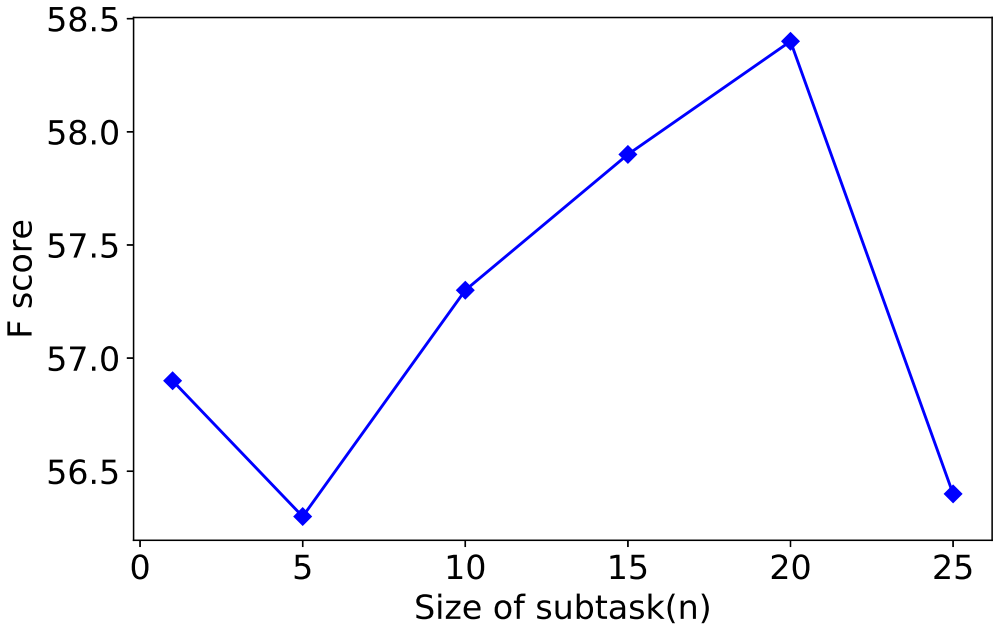

FIGURE 4.2 – Experiments are conducted with different sizes of subtask on TVSum dataset. When $n = 20$, it shows the best performance.

TABLE 4.2 – Evaluation by F score on SumMe dataset

	Methods	canonical	augmented	transfer	labels
supervised	vsLSTM [1]	37.6	41.6	40.7	293
	dppLSTM [1]	38.6	42.9	41.8	293
	SUM-GAN$_{sup}$ [9]	41.7	43.6	-	293
	DR-DSN$_{sup}$ [10]	42.1	43.9	**42.6**	293
unsupervised	SUM-GAN [9]	39.1	43.4	-	-
	DR-DSN [10]	41.4	42.8	42.4	-
weakly supervised	**Our Proposal**	**43.6**	**44.5**	42.4	15

TABLE 4.3 – Evaluation by F score on TVSum dataset

	Methods	canonical	augmented	transfer	labels
supervised	vsLSTM [1]	54.2	57.9	56.9	470
	dppLSTM [1]	54.7	59.6	58.7	470
	SUM-GAN$_{sup}$ [9]	56.3	**61.2**	-	470
	DR-DSN$_{sup}$ [10]	58.1	59.8	**58.9**	470
unsupervised	SUM-GAN [9]	51.7	59.5	-	-
	DR-DSN [10]	57.6	58.4	57.8	-
weakly supervised	**Our Proposal**	**58.4**	58.5	58.3	24

We compare our proposal with previous supervised and unsupervised methods. vsLSTM and dppLSTM [1] are supervised methods. SUM-GAN [9] and DR-DSN [10] are unsupervised methods. SUM-GAN is based on GANs. DR-DSN is based on reinforcement learning and the most related to our work. The existing weakly supervised methods only require category labels for one video, such as in [12]. However, they use a large number of web videos with category labels to learn more accurate and informative video representations. For a fair comparison, we don't include it here. Table 4.2 and Table 4.3 shows the F score performance and the average number of labels for a video, which is required by each method. Our hierarchical reinforcement learning based model achieves improvement on two benchmark datasets and requires much smaller number of task-level annotations than supervised methods.

On the canonical setting, our proposal outperformed the baseline dppLSTM by 5% and 3.7% points on two benchmark datasets. And compared with the state-of-the-art DR-DSN, our proposal improves by 2.2% and 0.8% points. Our hierarchical structure requires a much smaller number of annotations (about 1/20 in both cases) to train the Manager. The whole task is divided into several sub-tasks. Compared to frame-level annotations, our proposal only requires task-level annotations to train the Manager. Therefore it is weakly supervised.

Then, we compared the performance with the supervised version of SUM-GAN and DR-DSN. On the canonical setting, our proposal outperformed SUM-GAN$_{sup}$

by 1.9% and 2.1% points on the two benchmark datasets. Compared with DR-DSN$_{\text{sup}}$, the performance of our proposed method was improved by 1.5% and 0.3% points. Our result was improved by 0.9% and 0.1% points with augmented data respectively on SumMe and TVSum. In particular, our proposal outperformed all the other methods on SumMe with augmented data. However, our proposal was not capable of the transfer task because subgoals between different domains may vary a lot.

Then, we evaluate the performance using the rank order statistics proposed in [42] introduced in Section 4.2, which is claimed to be a better evaluation metric without the effect of post-processing. Note that SumMe dataset only offers the importance score for keyframe. Namely, importance scores for most of frames are annotated as 0 indiscriminately. Therefore, we only compute the rank correlation coefficients on TVSum dataset. Compared to supervised and unsupervised methods in Table 4.4, our proposal achieved higher correlation coefficients on both Kendall's τ and Spearman's ρ. Compared to supervised methods [1], unsupervised methods [10] improves the diversity by reinforcement learning but achieves lower correlation coefficients. With help of task-level labels, our proposal can promise the diversity and achieve higher correlation coefficients in the meantime.

TABLE 4.4 – Kendall's τ and Spearman's ρ correlation coefficients computed between different importance scores and human annotated scores.

	TVSum	
Methods	τ	ρ
dppLSTM [1]	0.042	0.055
DR-DSN [10]	0.020	0.026
Proposal	**0.078**	**0.116**
Human	0.177	0.204

4.4.2. Qualitative Evaluation

Figure 4.3 shows examples of our generated summaries. The gray bar of histogram is the ground-truth importance score of each frame. The red bar means the

corresponding frame is selected in the summary. In (a), the original video sequence is about keeping a dog's ears clean. The summary results of our proposal included the detail of cleaning the dog's ears (with higher importance scores) and skipped the beginning where there are some unrelated captions (with lower importance scores). However, DR-DSN included more repeated parts and the unrelated beginning parts. In (b) and (c), our proposal also outperforms the DR-DSN method. However, the performance of (c) is not good as (a) or (b). This video is a piece of news about cars. It can be considered that the summary of this kind of video depends on the audio information more than visual information. Therefore, our proposal only utilizing the visual information achieved limited performance in this case. More examples can be found in Figure 4.4.

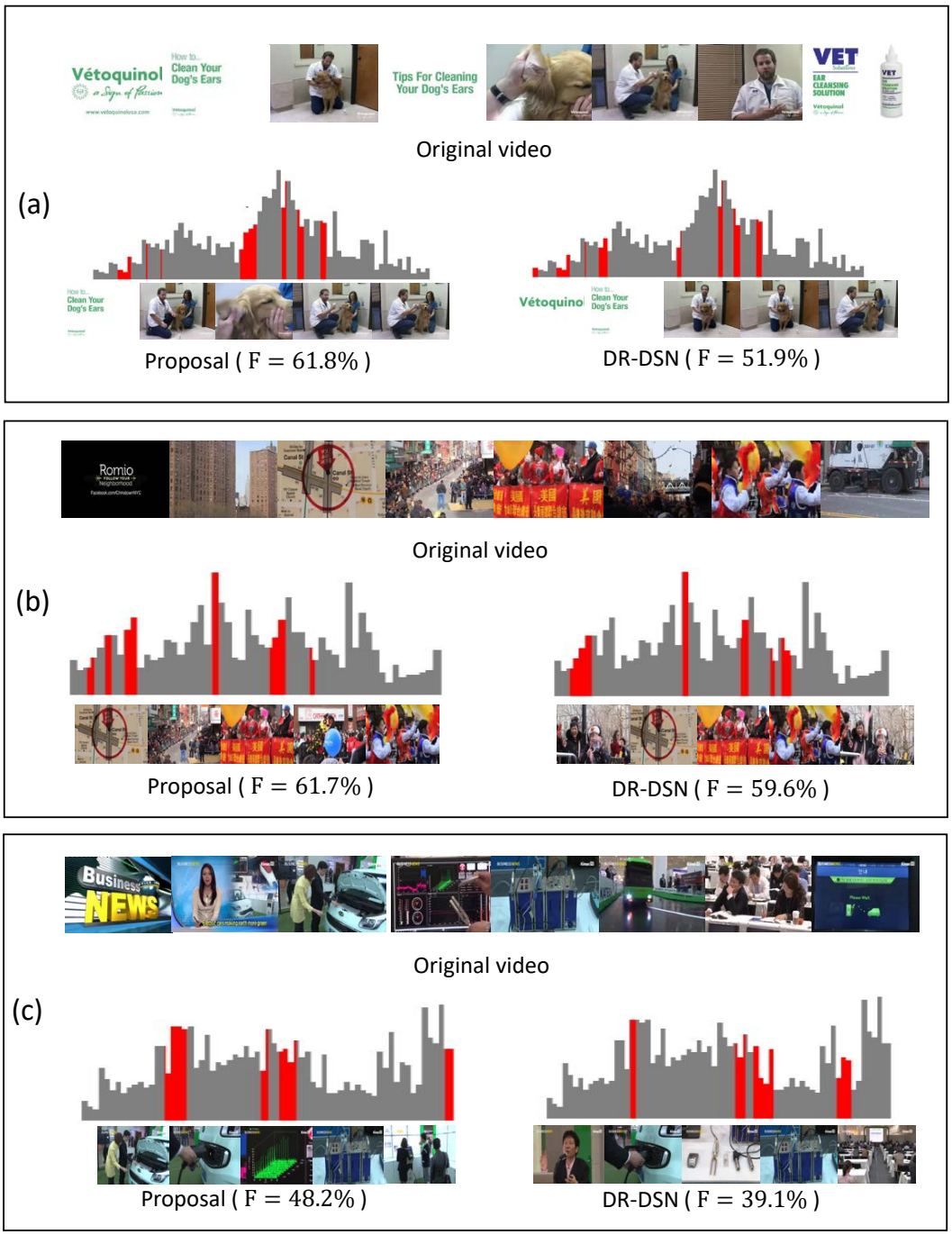

FIGURE 4.3 – Examples of summary videos of proposed method comparing with the state-of-the-art.

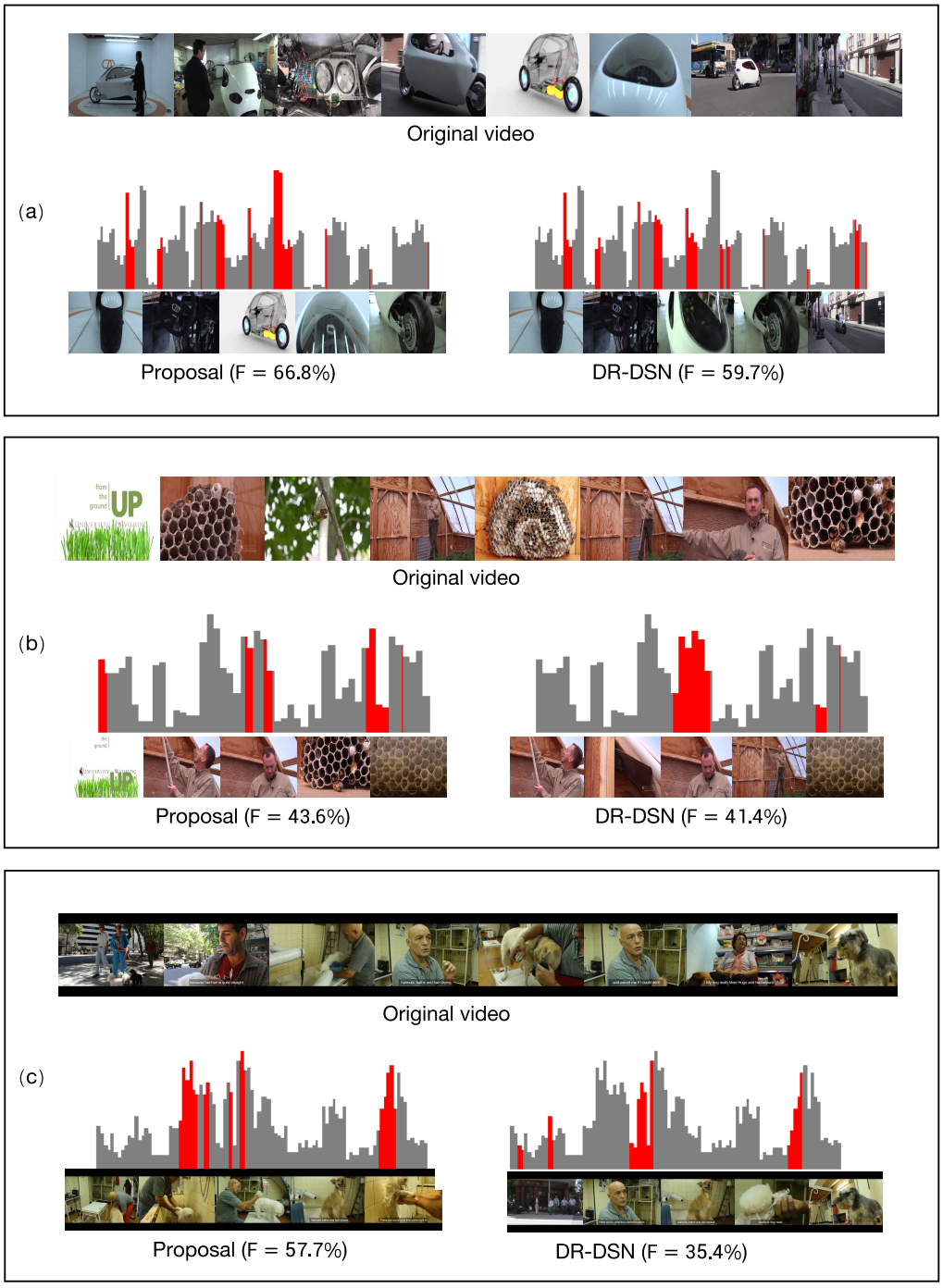

FIGURE 4.4 – Examples of summary videos of proposed method comparing with the state-of-the-art.

Chapter 5

Conclusions and Future Works

5.1. Conclusions

In my thesis, I share my study about video summarization technology. Video summarization is to process a long original video and generate a short but concise summary for it. This task can be regarded as a subset selection with considering the temporal dependence and the completeness of a story.

At the beginning, I first introduce the existing video summarization methods and reinforcement learning methods. Depending on annotations, these methods can be divided into supervised, unsupervised and weakly supervised methods. I introduce the detail of the related works. And I also explain the related reinforcement learning technique, which is used in my approach. Before I start to introduce my approach, I compare these different methods for video summarization. Supervised methods require a large number of frame-level labels but it is very hard to collect a large scale of dataset. Reinforcement learning based unsupervised methods require no annotations but it has sparse reward problem. The existing weakly supervised methods require a large number of web video to train for the target dataset. And all these methods need to deal with the long temporal dependence problem.

Then most importantly, I introduce my approach, which is based on hierarchical reinforcement learning. I design a hierarchical structure for the agent. And a sub-reward is well defined to avoid the sparse reward problem. My approach is based on the idea that the whole task can be divided into several subtasks. In this way, it is expected to process the long temporal dependence subtask by subtask.

Besides, I also define a task-level label to train the Manager of the agent. Compared to labels required by supervised methods, the number of label required by my approach becomes much smaller. Experiments are conducted on two benchmark datasets, SumMe and TVSum. I compare my approach with the existing methods including vsLSTM, dppLSTM, SUM-GAN and DR-DSN. The agent predicts the importance scores for each frame first. Then frames are selected to be combined into the summary according to these importance scores under a given limit. It is implemented by knapsack algorithm.

Therefore, the performance could be evaluated on the generated summary and on the importance scores. I use the F score and rank correlation coefficient to evaluate the performance. The former metric is based on the selected frame set. Precision and recall are defined to be computed on the intersection between two sets for video summarization task. On the other hand, the latter metric is based on the importance score. Kendall's τ and Spearman's ρ coefficients are used to measure the rank correlation. The results on these two metrics show the superiority of my approach.

5.2. Future Works

Some recent works apply other techniques into video summarization like attention mechanism and graph CNN [45][46][47]. These recent works explore a lot of new fields. They are also good directions for us to explore.

Besides, there are two good directions I am going to introduce, 3D convolutional neural network (C3D) and multi-modal. Recent years, C3D has been explored a lot [48][49][50][51]. Video processing task attracts tremendous attention from researchers [52][53][54]. There are a large number of videos being uploaded into the website (YouTube) everyday. 2D convolutional neural networks have achieved a great performance on image processing. Applying 2D CNN into video task can not process the temporal dependence in a convolutional way, in which the temporal information is regarded as an additional dimension. The differences between 2D CNN and 3D CNN are shown in Figure 5.2. Applying 2D convolution

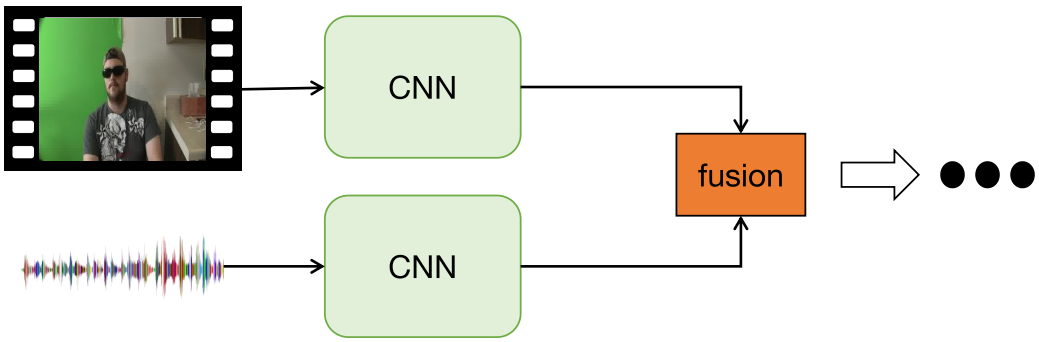

FIGURE 5.1 – Illustration of multi-model of video

into frame sequence results in a 2D output without temporal information. But through applying 3D convolution, we can get a 3D output.

In my future work, I would like to apply the C3D into video summarization. Rather than depend on the RNN to process the temporal information, I am going to use C3D to process the temporal and spatial information together in a convolutional way. This can first allow us to design a compact model without requiring a CNN to extract the frame feature first in a way of the current method. In the current method, we first get vectors as the frame features and regard them as a sequence of frames. Therefore, some spatial information is lost. For example, one video is about a dog running from the left to the right. A pre-trained 2D CNN extracts the vector for a dog but without the information of the movement because the CNN is usually pre-trained on the image classification dataset. Through 3D CNN, there may be no need for a RNN to process the temporal information and it can be more faster to process the data with the total convolutional operations. Embedding multi-model is also my future work. In my approach, I only process the image information of video. However, the image information is not enough for some kinds of video like news. It is expected to enhance the performance by extracting the feature of audio.

As shown in Figure 5.1, audio and visual data are processed by two separated CNNs. And then these two features are fused together.

[55] collects Audio-Visual Event datasets for audio-visual event localization. There are five modules in their network: feature extraction, audio-guided visual

attention, temporal modeling, multi-modal fusion and temporal labeling. In addition to fusing two modal features from separated branches, they use an audio-guided visual attention mechanism. A visual attention map is generated conditioned on audio and visual features. [56] applies multi-modal technique to video captioning. They define a multi-model CNN and modality-aware aggregation. In a word, multi-modal technique is a good direction to explore for video task. It is promising to apply the multi-model technique into video summarization. And it is also challenging to consider an effective way of fusion.

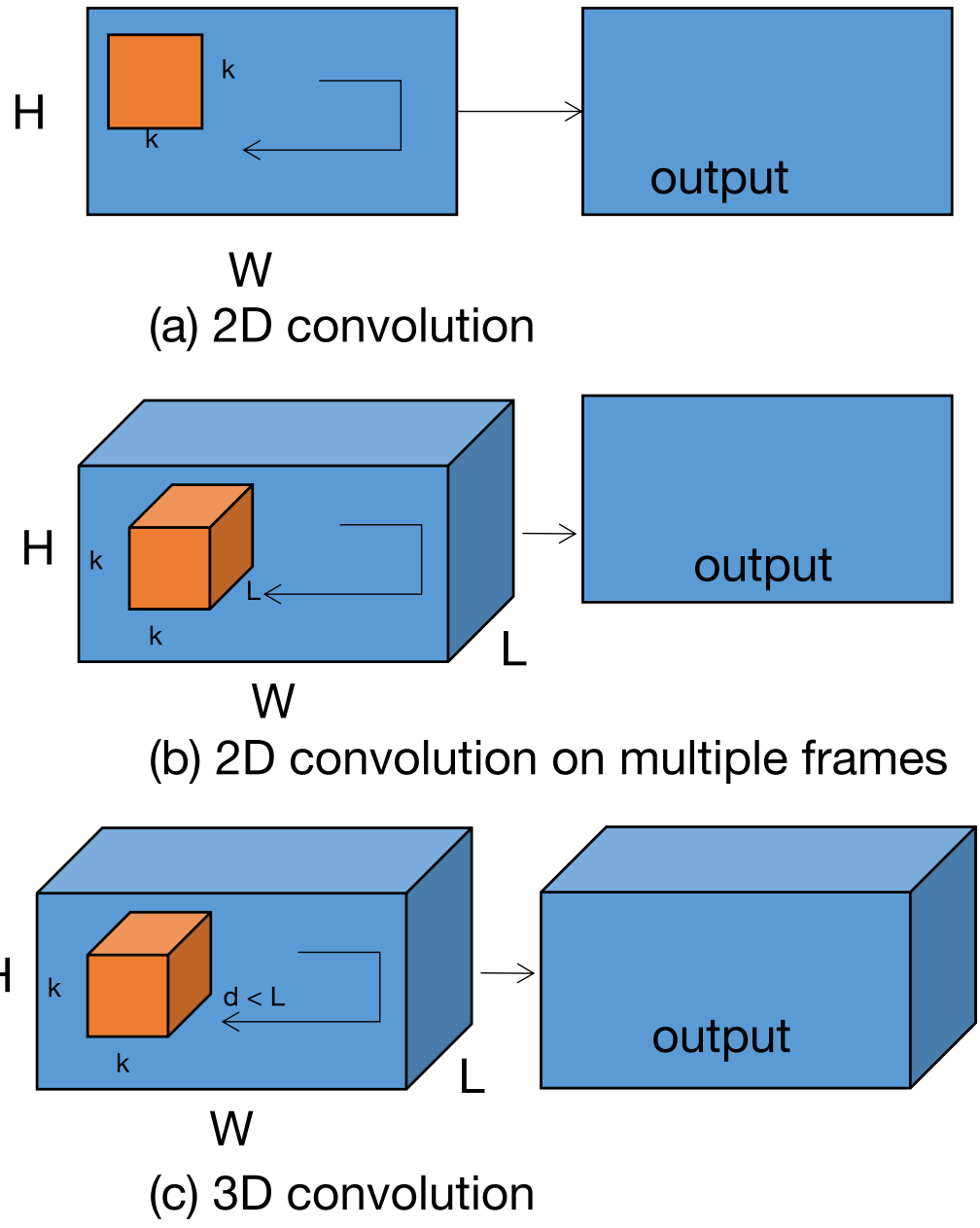

FIGURE 5.2 – Comparison between 2D convolution and 3D convolution

References

[1] K. Zhang, W.-L. Chao, F. Sha, and K. Grauman, "Video summarization with long short-term memory," in *European conference on computer vision*. Springer, 2016, pp. 766–782.

[2] J. R. Smith, D. Joshi, B. Huet, W. Hsu, and J. Cota, "Harnessing ai for augmenting creativity: Application to movie trailer creation," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1799–1808.

[3] F. Dirfaux, "Key frame selection to represent a video," in *Proceedings 2000 International Conference on Image Processing (Cat. No. 00CH37101)*, vol. 2. IEEE, 2000, pp. 275–278.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[5] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[8] B. Zhao, X. Li, and X. Lu, "Hsa-rnn: Hierarchical structure-adaptive rnn for video summarization," in *CVPR*, 2018.

[9] B. Mahasseni, M. Lam, and S. Todorovic, "Unsupervised video summarization with adversarial lstm networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 202–211.

[10] K. Zhou, Y. Qiao, and T. Xiang, "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[11] K. Zhou, T. Xiang, and A. Cavallaro, "Video summarisation by classification with deep reinforcement learning," *arXiv preprint arXiv:1807.03089*, 2018.

[12] S. Cai, W. Zuo, L. S. Davis, and L. Zhang, "Weakly-supervised video summarization using variational encoder-decoder and web prior," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 184–200.

[13] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[14] V. Alexander, Sasha, O. Simon, S. Tom, H. Nicolas, J. Max, S. David, and K. Koray, "Feudal networks for hierarchical reinforcement learning," in *ICML*, 2017.

[15] Y. Chen, L. Tao, X. Wang, and T. Yamasaki, "Weakly supervised video summarization by hierarchical reinforcement learning," in *Proceedings of the ACM Multimedia Asia*, 2019, pp. 1–6.

[16] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[17] B. Gong, W.-L. Chao, K. Grauman, and F. Sha, "Diverse sequential subset selection for supervised video summarization," in *NeurIPS*, 2014.

[18] A. Kulesza, B. Taskar *et al.*, "Determinantal point processes for machine learning," *Foundations and Trends® in Machine Learning*, vol. 5, no. 2–3, pp. 123–286, 2012.

[19] P. Mundur, Y. Rao, and Y. Yesha, "Keyframe-based video summarization using delaunay clustering," *International Journal on Digital Libraries*, vol. 6, no. 2, pp. 219–232, 2006.

[20] A. Aner and J. R. Kender, "Video summaries through mosaic-based shot and scene clustering," in *European Conference on Computer Vision*. Springer, 2002, pp. 388–402.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[22] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[23] M. Gygli, H. Grabner, and L. Van Gool, "Video summarization by learning submodular mixtures of objectives," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3090–3098.

[24] M. Volodymyr, K. Koray, S. David, G. Alex, A. Loannis, W. Daan, and R. Martin, "Playing atari with deep reinforcement learning," in *NIPS Deep Learning Workshop*, 2013.

[25] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Thirtieth AAAI conference on artificial intelligence*, 2016.

[26] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1995–2003.

[27] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[28] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," 2014.

[29] H. Guo, "Generating text with deep reinforcement learning," in *NIPS Workshop*, 2015.

[30] H. Ji, C. Jianshu, H. Xiaodong, G. Jianfeng, L. Lihong, D. Li, and O. Mari, "Deep reinforcement learning with a natural language action space," in *ACL*, 2016.

[31] N. Karthik, Y. Adam, and B. Regina, "Improving information extraction by acquiring external evidence with reinforcement learning," in *EMNLP*, 2016.

[32] D. Zhang, H. Maei, X. Wang, and Y.-F. Wang, "Deep reinforcement learning for visual object tracking in videos," *CoRR*, vol. abs/1701.08936, 2017.

[33] L. Ren, X. Yuan, J. Lu, M. Yang, and J. Zhou, "Deep reinforcement learning with iterative shift for visual tracking," in *ECCV*, September 2018.

[34] K. Tejas, D., N. Karthik, R., S. Ardavan, and T. Joshua, B., "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *NIPS*, 2016.

[35] M. Jaderberg, W. Czarnecki, I. Dunning, L. Marris, G. Lever, A. G. Castañeda, C. Beattie, N. C. Rabinowitz, A. S. Morcos, A. Ruderman, N. Sonnerat, T. Green, L. Deason, J. Z. Leibo, D. Silver, D. Hassabis, K. Kavukcuoglu, and T. Graepel, "Human-level performance in first-person multiplayer games with population-based deep reinforcement learning," *CoRR*, vol. abs/1807.01281, 2018.

[36] L. Jiaxian, Guo andSidi, C. Han, Z. Weinan, Y. Yong, and W. Jun, "Long text generation via adversarial training with leaked information," in *AAAI*, 2018.

[37] H. Qiuyuan, G. Zhe, C. Asli, W. Dapeng, W. Jianfeng, and H. Xiaodong, "Hierarchically structured reinforcement learning for topocally coherent visual story generation," in *AAAI*, 2019.

[38] D. Potapov, M. Douze, Z. Harchaoui, and C. Schmid, "Category-specific video summarization," in *ECCV*, 2014, pp. 540–555.

[39] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *ECCV*, 2014, pp. 505–520.

[40] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes, "Tvsum: Summarizing web videos using titles," in *CVPR*, 2015, pp. 5179–5187.

[41] S. E. F. De Avila, A. P. B. Lopes, A. da Luz Jr, and A. de Albuquerque Araújo, "Vsumm: A mechanism designed to produce static video summaries and a novel evaluation method," 2011, pp. 56–68.

[42] M. Otani, Y. Nakashima, E. Rahtu, and J. Heikkila, "Rethinking the evaluation of video summaries," in *CVPR*, 2019, pp. 7596–7604.

[43] M. G. Kendall, "The treatment of ties in ranking problems," *Biometrika*, vol. 33, no. 3, pp. 239–251, 1945. [Online]. Available: http://www.jstor.org/stable/2332303

[44] D. Zwillinger and S. Kokoska, *CRC standard probability and statistics tables and formulae*. CRC, 1999.

[45] X. He, Y. Hua, T. Song, Z. Zhang, Z. Xue, R. Ma, N. Robertson, and H. Guan, "Unsupervised video summarization with attentive conditional generative adversarial networks," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2296–2304.

[46] S. Xiao, Z. Zhao, Z. Zhang, X. Yan, and M. Yang, "Convolutional hierarchical attention network for query-focused video summarization." in *AAAI*, 2020, pp. 12 426–12 433.

[47] J. Wu, S.-H. Zhong, and Y. Liu, "Mvsgcn: A novel graph convolutional network for multi-video summarization," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 827–835.

[48] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[49] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7083–7093.

[50] A. Diba, V. Sharma, L. V. Gool, and R. Stiefelhagen, "Dynamonet: Dynamic action and motion network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 6192–6201.

[51] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3d residual networks for action recognition," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 3154–3160.

[52] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[53] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[54] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2556–2563.

[55] Y. Tian, J. Shi, B. Li, Z. Duan, and C. Xu, "Audio-visual event localization in unconstrained videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 247–263.

[56] Y. Tian, C. Guan, J. Goodman, M. Moore, and C. Xu, "Audio-visual interpretable and controllable video captioning," in *CVPR Workshops*, 2019.

Publications

International Conferences and Workshops

[1] Yiyan Chen, Li Tao, Xueting Wang, and Toshihiko Yamasaki. Weakly Supervised Video Summarization by Hierarchical Reinforcement Learning. *In ACM Multimedia Asia (MMAsia ' 19), December 15 - 18, 2019, Beijing, China.*

Domestic Conferences and Symposia

[2] Yiyan Chen, Li Tao, Xueting Wang, Toshihiko Yamasaki. Reinforcement Learning on Video Summarization with Hierarchical Structure. Image Engineering Technical Group (IE), Feb. 27-28, 2020, Hokkaido.

Acknowledgements

It is very honor for me to have experience to spend two years in doing research at The University of Tokyo. Being a foreign student, I was excited but also nervous about my life in Tokyo before I came here. But I became to feel at home in a short time because I am very lucky to meet with some kind and friendly people. At the beginning, I would like to express my sincere respect and gratitude to my supervisor Prof. Toshihiko Yamasaki, who gives me the chance to start my life at The University of Tokyo. Prof. Yamasaki is always very kind, helpful, patient and professional. During my time here, he not only guides me in the research but also gives me a lot of chances to open my eyes. He has supported me several times to participate the international conferences to communicate with researchers from all the world. It is quite lucky to be supervised by Prof. Yamasaki, who is patient and professional to guide me how to do research, how to write a research paper and how to give a nice speech to communicate with others.

Then I am very thankful to Prof. Kiyoharu Aizawa. I have received a lot of advice from him during the laboratory meetings. I have learnt a lot from his advice. I am also very grateful to Dr. Wang, who is like a bigger sister to provide help and advice to me in my research and life in Japan.

I also want to show my gratitude to all lab members. They give me a feeling of living in Tokyo with a big family. Many thanks to the secretaries of our lab, Ms. Matsubayashi and Ms. Egawa, who are always very kind and helpful to me. I remember that the first day I came to the lab they are the first ones to come to say hello to me. Thanks to Ms. Yiwei, being my tutor to help me get familiar with life at UTokyo. Thanks to all phD students, Mr. Furuta, Mr. Inoue, Mr. Sourav, Mr Ikuta, Mr. Tao, Mr.Zhong and Ms. Yiwei sharing their advice and their research. I was very shy at the beginning. But I become to talk with others to share my idea because I feel at home here.

I start my master course in September. But Yu, Kato, Kaneko, Kosugi, Tanaka, Tsubota, Yi, Lin, Kawarada and Luwei regard me as the students of the same year. I am very grateful that I can spend a lot of time with them. Thanks to them for

sharing the life at UTokyo, doing research together and holding the Dokkikai every month. I enjoy a lot participating the conferences and hanging out with them.

Finally, I want to express the gratitude to my parents, who support me to go abroad to pursuing higher education. Thanks to my friend, Mr. Guo, listening a lot to me and giving me advice even though in Sydney far away from me.

May you all good health.

Yiyan Chen

August 14, 2020