A Spherical Convolution Approach for Learning Long Term Viewport Prediction in 360 Immersive Video

Chenglei Wu,[1] **Rui-Xiao Zhang,**[1, 2] **Zhi Wang,**[3*†] **Lifeng Sun**[1, 4*]
[1]Beijing Key Lab of Networked Multimedia, Department of Computer Science and Technology, Tsinghua University,
[2]BNRist, Department of Computer Science and Technology, Tsinghua University,
[3]Tsinghua Shenzhen International Graduate School, Tsinghua University,
[4]Key Laboratory of Pervasive Computing (Tsinghua University), Ministry of Education, China
{wucl18@mails., zrx17@mails., wangzhi@sz., sunlf@}tsinghua.edu.cn

Abstract

Viewport prediction for 360 video forecasts a viewer's viewport when he/she watches a 360 video with a head-mounted display, which benefits many VR/AR applications such as 360 video streaming and mobile cloud VR.

Existing studies based on planar convolutional neural network (CNN) suffer from the image distortion and split caused by the sphere-to-plane projection. In this paper, we start by proposing a spherical convolution based feature extraction network to distill spatial-temporal 360 information. We provide a solution for training such a network without a dedicated 360 image or video classification dataset.

We differ with previous methods, which base their predictions on image pixel-level information, and propose a semantic content and preference based viewport prediction scheme. In this paper, we adopt a recurrent neural network (RNN) network to extract a user's personal preference of 360 video content from minutes of embedded viewing histories. We utilize this semantic preference as spatial attention to help network find the "interested" regions on a future video. We further design a tailored mixture density network (MDN) based viewport prediction scheme, including viewport modeling, tailored loss function, etc, to improve efficiency and accuracy.

Our extensive experiments demonstrate the rationality and performance of our method, which outperforms state-of-the-art methods, especially in long-term prediction.

Introduction

Viewport prediction is one of the most challenging tasks in 360 video, which forecasts a viewer's viewport when he/she watches a 360 video with a head-mounted display (HMD). Many virtual reality (VR) applications, e.g., 360 video streaming (Xiao et al. 2017) or mobile cloud VR (Hou et al. 2018) benefit from such technique. Because a user's future viewport is related to the 360 video contents, researchers have proposed to incorporate multi-modal information to predict future viewport, e.g., dedicated saliency detection network (Nguyen, Yan, and Nahrstedt 2018) for 360 video and temporary head movement.

Previous studies often extract visual features from 360 videos with the conventional planar CNN, such as the VGG16 network (Simonyan and Zisserman 2014) used by (Nguyen, Yan, and Nahrstedt 2018) or C3D network (Tran et al. 2015). However due to the image distortion and split caused by the sphere-to-plane projection of 360 video, these planar CNN networks are not capable of extracting visual features effectively, thus sabotaging the accuracy of the later viewport prediction process. This problem imposes an urgent requirement for a dedicated 360 feature extraction network. However, as the planar feature extraction networks are trained on planar image classification datasets (e.g., ImageNet (Deng et al. 2009)) or video classification datasets (e.g., UCF101 (Soomro, Zamir, and Shah 2012)), there also lacks a 360 image or video classification dataset for pre-training 360 feature extraction network.

On the other hand, previous work (Xu et al. 2018; Fan et al. 2017) bases their predictions on the saliency or optical flow information of 360 videos. However, a user's future viewport doesn't always correlate with this visual information. For example, Hu et al 2017 found that in a sightseeing video[1] where the visually salient foreground is the video shooter, human viewers tend to watch the visually un-salient background view. Furthermore, these kinds of information describe more of the pixel information of 360 video frames, rather than the semantic content information, but human viewers tend to have their personal preferences of certain objects or video contents. A content and preference aware viewport prediction network is called for these problems.

In this paper, we start by proposing and training a dedicated spherical CNN (S2CNN) based 360 feature extraction network. We design a convolutional RNN based visual feature extraction network to distill spatial-temporal information from sequential 360 video data. We provide a two-step training process and a method for converting planar video classification dataset to sphere domain, which allows us to pre-train the rotational invariant feature extraction network. We conduct a series of video classification experiments with our 360 video dataset to study the effects of different kernel shape, network depth, etc.

We then propose a preference-aware viewport prediction

*Corresponding author.
†Zhi is also with the Peng Cheng Laboratory, Shenzhen.

[1]https://aliensunmin.github.io/project/360video/

network. In Fig. 1, we show an overview of our network. Our network takes input of multi-modal data, including past viewing history, future video, and head motion, to make accurate and explainable predictions. Specifically, we encode a user's viewing history, i.e., the video content that he/she viewed, with our 360 feature extraction network, as well as the future video content. We adopt an RNN network to process up to minutes of viewing histories and use the output at each timestamp, along with past head motion and future video, to predict the viewport of several seconds later. In this way, the RNN network can capture the semantic information that is important for accurate prediction, in the long run, i.e., a user's personal preference.

To improve the effectiveness and efficiency of the viewport prediction network, we first apply the viewer's personal preference as the context information as spatial attention to the future video. This attention mechanism calculates the semantic relevance between a region on future video and user's preference, thus indicating which region is important for viewport prediction and improving the prediction accuracy. We model the viewport prediction probability as von Mises-Fisher[2] (vMF) mixture model and propose a MDN based viewport prediction network. Such design not only reduces the decision space but also improves the correlation during the parameterization process.

The main contributions can be summarized as threefold:

▷ We propose a 360 feature extraction network based on S2CNN to distill spatial-temporal information and a training method that doesn't require a dedicated 360 dataset.

▷ We extract user's preference from up to minutes of recent viewing histories and utilize the attention mechanism to achieve preference-aware viewport prediction.

▷ We present a sophisticated MDN based viewport prediction scheme, including viewport modeling, tailored loss function, etc., to improve the efficiency and accuracy of the prediction process.

We compare our method with several baseline methods to validate its effectiveness. Compared with the state-of-the-art method, our network significantly improve the long-term prediction accuracy, e.g., 32.3% for 6 seconds later.

Related Work

Deep Viewport Prediction

Unlike gaze prediction (Xu et al. 2018; Koulieris et al. 2016), viewport prediction predicts a user's head pose. Despite the center of the viewport may slightly differ from a user's actual gaze point, user's head pose determines which part of 360 video or 3D scene is showed or rendered. Viewport prediction has demonstrated its importance in many popular VR/AR applications. For example, in viewport adaptive 360 video streaming (Xiao et al. 2017), researchers adopt viewport prediction to prefetch video tiles in advance, .i.e., part of the video, to improve user experience. In cloud-based VR/AR gaming (Hou et al. 2018), this technique enables pre-rendering of 3D scenes in advance.

We first review the state-of-the-art viewport prediction methods. Fan et al. 2017 proposed to use the visual saliency, optical flow of the 360 video, and user's head motion to predict the future viewport. They adopt an LSTM network to predict the user's viewport at each video frame recurrently. Their method can achieve FoV prediction of at most 1 second later. Xu et al. 2018 proposed to encode the optical flow and saliency with a single encoder. The resulting embedding is concatenated with auser's historical head motion to make displacement prediction of the user's future viewport. Nguyen et al. 2018 trained a dedicated 360 saliency detection network based on human fixation on 360 videos, and predict users' future viewport using image saliency map and head orientation map recurrently. They were able to predict the future viewport of 2.5 seconds later. Xu et al. 2017 adopted the DRL algorithms to learn a viewport prediction network from 360 video frame and user's view traces.

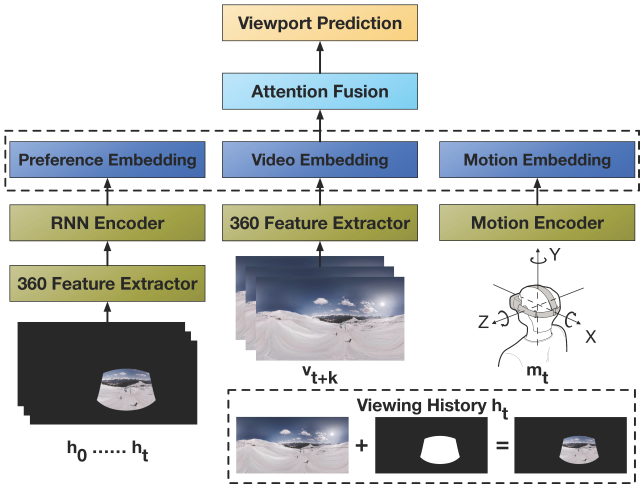


Figure 1: Predict future viewport $\bar{h}_{t+k}$ at time t

These work all processed 360 videos with conventional planar CNN and models, which have been proven to work poorly (Zhao et al. 2018) for 360 images. In our work, instead of using pre-trained saliency or optical flow detection network on planar images, we design and train a spherical CNN based 360 feature extraction network with our own dataset. Also these studies predicted the future viewport in a *recurrent*s style, in which the predicted viewport at timestamp t is used to predict the viewport at timestamp $t+1$.

Our Method To the best of our knowledge, this is the first work that addresses the problem of video content-based long-term viewport prediction (46% accuracy for 6s later, larger than 40% for 10s later). (Nguyen, Yan, and Nahrstedt 2018) predicts viewport of at most 2.5s and mentions the result of 6s later (less than 34% after 6s). Other work only predicts short-term viewport, e.g., next frame prediction in (Xu et al. 2018) and 1s later in (Xu et al. 2017).

Spherical Convolution and Attention

The demand for analyzing spherical data in omnidirectional vision, molecular modeling, etc., has motivated many researchers to investigate the convolution on the 3D sphere. Cohen et al. 2018 and Esteves et al. 2018 propose spherical

[2]https://en.wikipedia.org/wiki/Von_Mises-Fisher_distribution

CNN to directly sampling feature from spherical images and achieves lossless rotation-invariant 360 convolution.

Although spherical CNN shows its strong potential in processing spherical data, it is still at an early age, and the official implementations lack the efficiency for real-world deployment. Therefore, researchers have also investigated how to apply the well-developed planar CNN for sphere signal processing. The most common technique is to alter the shape of the convolution kernels of planar CNN. Su et al. 2017 propose a knowledge distillation based solution for processing spherical data by learning different kernel for each row of the spherical image and transferring a pre-trained conventional CNN model to the spherical network. (Zhao et al. 2018; Jiang et al. 2019) are the other attempts to apply traditional CNN in the spherical data processing. These solutions benefit from the efficiency of traditional CNN and overcome the problem of image distortion and split. However, they still can't achieve rotation-invariant convolution since traditional CNN only has the ability of translational-invariant convolution.

We further discuss some related work about the attention mechanism. Attention mechanism was first proposed in (Bahdanau, Cho, and Bengio 2014) to help deep learning network to selectively focus on part of the input sequence, similar to human attention. This method has then been applied to almost every deep learning tasks, due to its ability to significantly improve network performance. The spatial and channel-wise attention are proposed in (Chen et al. 2017), in which Chen et al. adopt the attention mechanism to effectively integrate spatial, channel-wise, and multi-layer visual attention in the convolution stage for image captioning. This method enables the network to select semantically important features on the demand of the sentence context.

Method

In this section, we first present some of the necessary background of spherical convolution S2CNN, and our study on 360 visual feature extraction. We then present the details of our preference-aware viewport prediction network.

360 Feature Extraction Network

S2CNN Background As a recently proposed group convolution method, S2CNN intends to extract lossless rotation-invariant features from 360 sphere signals. Despite lacking a thorough investigation into the feasibility of S2CNN based 360 video feature extractor, its design is suitable and essential for sequential 360 video data processing. due to the object split on the boundary area and image distortion. For example, in Fig. 2a, we show a distorted and split surfer in the south pole area of a 360 video frame. These problems are inevitable for 360 video projection formats, such as equirectangular and cube-map. Another problem of 360 video is the object rotation caused by 360 video's panorama view of the real-world scene. In Fig. 2b, we show a video frame of a skiing athlete whose body is rotated upside down. which happens very often with unaligned 360 camera orientation or dynamic object movement,

Compared to planar CNN, S2CNN is defined on 3D

(a) Distortion and split (b) Rotation

Figure 2: Distortion and rotation problems in 360 videos

sphere, i.e., S2 sphere[3], and samples features directly from sphere. S2CNN first defines a S2 convolution, denoted as $S2Conv$, to convert a S2 signal to the representation on SO(3) rotation group[4]. Here we take an input spherical images X of width b, height b and 3 channels as example. The input image is of shape $3 \times b \times b$. The width and height correspond to the axis latitude α and longitude β on sphere, in which $0 \leq \alpha < 2\pi, 0 \leq \beta < \pi$. The output of $S2Conv$ with c channels and bandwidth b is a tensor of shape $c \times b \times b \times b$. The last three dimensions correspond to the axis α, β and γ, in which $0 \leq \gamma < 2\pi$ defines a rotation around the axis through the point (α_i, β_j). The convolutions on the SO(3) rotation group can then be continued with $SO3conv$.

Spatial-Temporal Feature Extraction Our 360 feature extraction network is composed of two stages, i) spatial feature extraction, ii) temporal feature aggregation. For spatial feature extraction, our network is composed of 1 $S2Conv$ layer and 4 $SO3Conv$ layers with near-identity kernels. The channels are increased from 3, 48, 72, 240, 312 to 528 in the final layer. We reduce the resolution b from 240 to $64, 48, 36, 20$ and 14 in the final layer.

During the temporal aggregation stage, we intend to preserve the spatial information. The temporal aggregation network is composed of a dimension-reduction layer and a convolutional RNN network (Xingjian et al. 2015). In convolutional RNN network, the original hidden state update based on fully connected layers is replaced with convolution network. However in practice, we found that simply replacing the planar convolution with $SO3Conv$ layer is impossible to train due to extremely large GPU memory consumption.

As we mentioned above, the dimension γ of $SO3Conv$ feature map represents the rotation around axis α, β, which is relatively negligible for our task since we only predict the location of viewport. By pooling an $SO3Conv$ feature map over the γ axis, we can obtain a 3D feature map on the sphere which still the contains the location information latitude α and longitude β. but the embedding dimensionality is significantly reduced and can be processed with $S2Conv$.

Thus we first conduct a global pooling over the γ axis on the spatial embeddings and adopt a convolutional RNN network to aggregate the spatial-temporal information. Specifically, we adopt a convolutional gated recurrent network (GRU) with one hidden layer. After temporal aggregation, we obtain a feature map with size $528 \times 14 \times 14$.

[3]https://en.wikipedia.org/wiki/N-sphere

[4]https://en.wikipedia.org/wiki/3D_rotation_group

Preference-Aware Multi-Model Fusion

Preference Encoder We adopt a GRU to extract user preference from L seconds of viewing histories. As shown in Fig. 1, we first project user's past viewport back to the corresponding video frame. For each second, the sample frequency is also F timestamp per second. The viewing history of each second is thus a sequence of F masked images, showing the contents that user viewed. We encode these images, i.e., $F \times 3 \times 240 \times 240$ input tensor, with our feature extraction network and obtain a history representation h_t of shape $528 \times 14 \times 14$, which is embedded to the same space as the video embedding and allows the network to easily correlate the viewing histories and future video.

For a history embedding, the channel contains the semantic information of viewed contents, while the other two dimensions contain the spatial information. For preference extraction, we care about the semantic information rather than location information. Thus we integrate the spatial dimension, i.e., the last two dimensions, to convert the history representation to a vector of shape 528. We then adopt a RNN network, with 1 hidden layer, to process the $L \times 528$ viewing histories. The hidden layer of the RNN network at timestamp t is used to represent the user preference p_t, which is a 784×1 vector.

Motion Encoder We represent user's head orientation with the Euler angles[5], which is 1×3 vector representing the rotations around X, Y, Z axis. For the temporary head movement, we sample 30 timestamps of head orientations from the past 3 seconds to form a input feature of shape 3×30 and embed it with a dense layer to obtain a motion embedding of 128×1.

Preference-Aware Multi-Modal Fusion At each timestamp t, our network first updates user's preference p_t with the new viewing history embedding h_t, in which $1 \leq t \leq L$. The video embeddings of timestamp t is denoted as v_t. And the motion embeddings at t is denoted as m_t. Our network predicts the future viewport of K different time windows w_k later, i.e., the future viewport at timestamp $t + w_k, 1 \leq k \leq K$. For each prediction, the input to the prediction network is the prediction time window w_k, i.e., a scalar, the preference embedding p_t, the motion embedding m_t, and the future video content v_{t+w_k}.

We first apply a preference p_t guided spatial attention to video embedding v_{t+w_k} and help the network locate the interested regions. The video embedding v_{t+w_k} is of shape $528 \times 14 \times 14$ and the preference embedding p_t is of shape 784×1. For each region r on the video embedding v_{t+w_k}, we compute a importance score q. The video embedding is reshaped as $V = \{v_r | r \in 1 \cdots 196\}$, in which $v_r \in R^{528}$.

$$\bar{q} = \quad tanh\big((\boldsymbol{W_v} V + b_v) \oplus \boldsymbol{W_p} p_t\big) \tag{1}$$

$$q = \quad softmax(\boldsymbol{W_{pv}} \bar{q} + b_{pv}) \tag{2}$$

in which $1 \leq r \leq 14 \times 14, \boldsymbol{W_v} \in R^{128 \times 528}, \boldsymbol{W_p} \in R^{128 \times 784}, \boldsymbol{W_{pv}} \in R^{1 \times 128}$. $b_v \in R^{128 \times 1}$ and $b_{pv} \in R^1$ are the model biases. $\oplus$ denotes adding each value of a vector to the corresponding column of a matrix. $v^r_{t+w_k}$ is the feature vector at region r with shape of 528×1. The dimension 128 is the mapping space of p_t and v_{w_k}.

The importance score is calculated using dense layers, which incurs very little overhead. The above equation can be regarded as calculating the relevance of each visual region to user preference. Thus user preference is directly used to guide the attention learning of the video modality. The feature vector $v^r_{t+w_k}$ of each region is multiplied with the attention score q_r to obtain an attended embedding.

For dimensionality reduction of the attended video embedding, We adopt a 1×1 convolution to keep the spatial information. We use a 1×1 convolution with 4 output channels and gain a new video embedding $\bar{v}_{t+w_k}$ with shape $4 \times 14 \times 14$. The representation is flattened to a 1D vector of shape 784×1.

Viewport Prediction Network

MDN based Prediction Network The ground truth viewport at timestamp t is denoted as a 40×20 binary matrix λ^t, in which $\lambda^t_{i,j} = 1$ represents the point (i, j) is within user's viewport. However, in our practice, we find that networks directly predicting the probability of $\lambda^t_{i,j}$ converge very slowly, because the network has a relatively large decision space, i.e., 800 for a 20×40 label. Rather than directly predicting the probability of each point (x, y), we adopt a MDN(Graves 2013; Bazzani, Larochelle, and Torresani 2016) to predict the parameters of a probability density function that describes the overall distribution.

In this paper, we describe the future viewport with vMF mixture model. The vMF mixture can be considered as Gaussian mixture model[6] on 3D sphere. A single vMF distribution is parameterized by a mean direction $\mu = (x_\mu, y_\mu, z_\mu)$ and a concentration scalar $\tau > 0$. For viewport prediction, μ represents the position of a viewport on unit sphere and τ represents its size. We adopt this spherical distribution such that the viewport can be modeled as its actual state on the sphere, rather than its projection on planar image.

For point (x, y), its direction vector is denoted as $n_{i,j}$. The probability of $n_{i,j}$ is calculated as follows:

$$f(n_{i,j}|\mu, \tau) = \frac{\tau e^{\tau \mu^T n_{i,j}}}{2\pi(e^\tau - e^{-\tau})} \tag{3}$$

And the vMF mixture model fits a data distribution with C single vMF distributions. The probability density function of vMF mixture model is defined as:

$$\mathcal{F}(n_{i,j}|\Theta) = \sum_{c=1}^{C} a_c f(n_{i,j}|\mu_c, \tau_c) \tag{4}$$

where $\sum_{c=1}^{C} a_c = 1$ and $\Theta = \{a_c, \mu_c, \tau_c | 1 \leq c \leq C\}$.

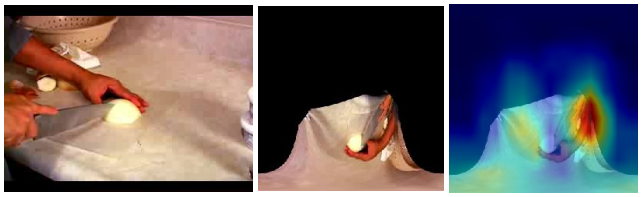
The multi-modal representation (w_k, p_t, m_t, v_{w_k}) is used to predict the parameters Θ of vMF mixture model, which are then used to calculate the probability $\mathcal{F}(n_{i,j}|\Theta)$. The

[5] https://en.wikipedia.org/wiki/Euler_angles

[6] https://en.wikipedia.org/wiki/Mixture_model #Gaussian_mixture_model

(a) Original frame (b) PanoUCF101 (c) Activation

Figure 3: *v_CuttingInKitchen_g02_c02*



(a) Original frame (b) PanoUCF101 (c) Activation

Figure 4: *v_CuttingInKitchen_g05_c02*

predicted $\bar{\lambda}^t_{i,j}$ is then calculated by normalize $\mathcal{F}(n_{i,j}|\Theta)$ to between $[0, 1]$. Thus our MDN network only has to predict $C \times 5$ parameters, which significantly reduces the decision space and makes it converge much faster.

Spherical MSE We use the mean squared error (MSE) to measure the accuracy based on ground truth λ. Similar to previous work(Nguyen, Yan, and Nahrstedt 2018; Fan et al. 2017), we define a threshold t, such that if $\mathcal{F}(n_{i,j}|\Theta) > t$, the point (i, j) is predicted to be within user's viewport. As previous work, we use $t = 0.5$. However, the calculation is based on planar labels, of which pixels on the polar areas have redundant densities, thus contributing more to the final loss. To counter this problem, we use the quadrature weight (Kostelec and Rockmore 2008) to measure the inverse density of point (i,j):

$$\omega_j = \frac{4}{b} sin(\frac{\pi(2j-1)}{2b}) \sum_{s=0}^{b//2-1} \frac{1}{2s+1} sin(\frac{(2j-1)(2s+1)\pi}{2b}) \tag{5}$$

in which $1 \leq i, j \leq b$. We define a spherical MSE loss as follows:

$$Loss = \boldsymbol{E}\big(\omega_j(\bar{\lambda}_{i,j} - \lambda_{i,j})^2\big) \tag{6}$$

Experiments

Our code and more results can be found at Github[7].

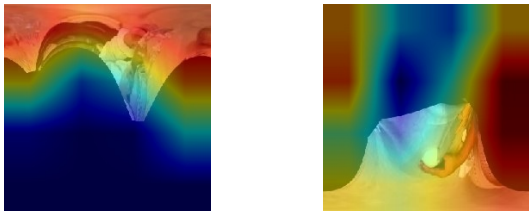
360 Feature Extraction Network

PanoUCF101 Dataset As we mentioned above, training S2CNN is extremely memory consuming. We therefore train the spatial feature extraction and the temporal aggregation network separately. As there lacks a dedicated 360 image or video classification dataset, we provide a method for pre-training the feautre extraction network based on the off-the-shelf planar video classification dataset UCF101.

[7]https://github.com/wuchlei/AAAI20-Viewport-Prediction

Table 1: Top-1 Classification Accuracy

Network		Accuracy
PanoUCF101	**S2CNN**	**65.7**
	S2CNN, equatorial kernel	57.9
	S2CNN, 6 layer	76.5
	S2CNN+ConvGRU, F=4	**70.3**
	S2CNN+ConvGRU, F=8	72.7
	S2CNN+ConvGRU, F=16	73.2



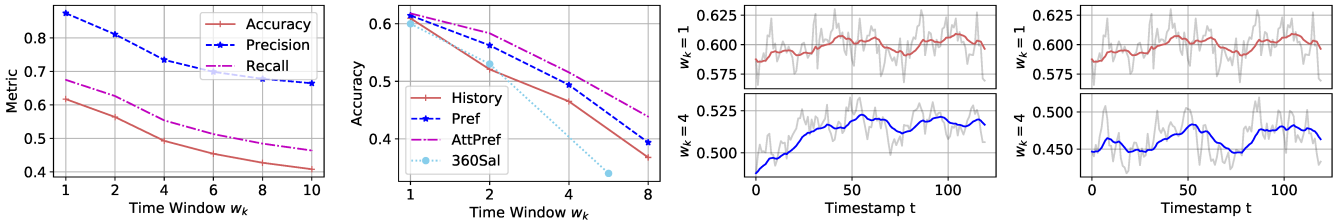
(a) v_CuttingInKitchen_g02_c02 (b) v_CuttingInKitchen_g05_c02

Figure 5: S2CNN with 6 layers and $b = 2$

We convert the planar UCF101 dataset(Soomro, Zamir, and Shah 2012), which contains video clips of 101 human actions classes, to the sphere domain. As shown in Fig. 3 and Fig. 4, we project each video clip to a 360 viewport of random orientation, including its position and rotation. The width and height of the viewport is $160°$ and $120°$, same as the original width-to-height ratio. We then apply a sphere-to-plane projection to store these videos in regular format, with width and height both set to 240, i.e. the input resolution of spatial feature extraction network. We refer to this dataset as the PanoUCF101 dataset. We use 70% of videos as training set, 10% as validation set and 20% for testing. As UCF101 is a relatively small dataset, we use a 0.5 dropout rate through the experiments to avoid over-fitting.

Training of Spatial Feature Extraction Network We extract one frame per second from the videos of our PanoUCF101 and embed it with our 360 feature extraction network. The representation is integrated to contain only feature information and used to predict its class with a simple two-layer predictor. We use a cross-entropy loss with softmax layer to train this network. The network is trained with a Adam optimizer with learning rate $3e-4$ for 100 epochs. We adopt an early stop if both the loss and prediction accuracy on the validation set is not improved for more than 1 epoch.

In our model design, we use a relatively unusual channel settings. This is due to fact that training S2CNN network is extremely memory consuming. In our experiments, we are only able to put a batch of at most 4 to 8 images in a GTX 1080Ti GPU with 11GB memory. Therefore we adopt a group normalization method(Wu and He 2018) to avoid gradient vanishing, rather than batch normalization. We assign 24 channels to each group and the mean and variance are computed within each group for normalization. Each epoch takes about 2 hours to train on a computer with 2 GTX 1080Ti GPU and the total training time is around 200 hours.



(a) Accuracy of AttPref (b) Baseline comparison (c) Our method (d) *History* method

Figure 6: Experiment results

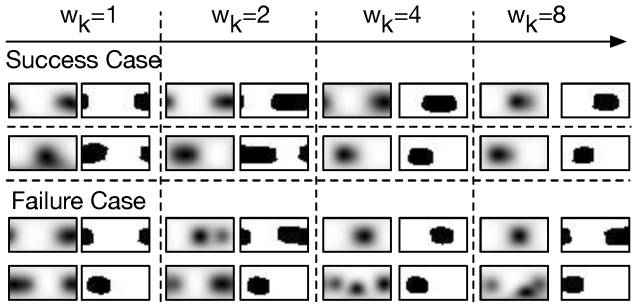


Figure 7: Visualization of viewport prediction

Training of Temporal Aggregation Network We then train our temporal aggregation network with different frame rate, i.e., $F = \{4, 8, 16\}$. The hidden state of the final video frame is integrated and used to predict the video class, also with a simple two-layer predictor. The embedding of each video frame is pre-processed with the spatial feature extraction network in advance. Thus we only have to to train the temporal aggregation RNN network. We further improve the training efficiency by initialize the predictor's weight with the model from previous spatial network training and freeze its weights, until the whole model is performing acceptably.

Evaluation Results We investigate the prediction accuracy of our spatial feature extraction with different setups, including kernel shape[8] and network depth (one more SO3Conv layer with 1200 output channel and $b = 4$). In Table 1, we present the top-1 accuracies, in which our model with near-identity kernel achieves 65.7% accuracy and outperforms the the model with equatorial kernel. The model with 6 layers performs best with an accuracy of 76.5%.

In Fig. 3 and Fig.4, we also present some visualization results. In Fig. 3c and Fig. 4c, we adopt the Grad-CAM method (Selvaraju et al. 2017) to highlight the activation region of input images, i.e., the red regions. We can observe that our network can correctly locate the semantically important regions, which are barely recognizable even for a human viewer.

However the 4×4 feature embedding doesn't preserve enough spatial information, as shown in Fig. 5. And despite that S2CNN with 6 layers improves the prediction accuracy, it also increases the model size by $2\times$ larger. Since our goal is not to design a dedicated video classification network, but a general feature extraction network, we choose the network setup with 5 layers.

We then investigate the impact of frame sampe rate on temporal aggregation network. We can found in Table 1 that the more frequent temporal sampling improves performance of the RNN network. However the marginal improvement is not very significant, thus for the sake of efficiency, we use a temporal sample frequency of $F = 4$.

Note that there's no comparison between our results and the results of state-of-the-art planar methods on the original dataset. Because by projecting the standard videos to sphere, they only cover less than half of the images, which means the resolution of our input videos are much smaller. Also the planar feature extraction networks used by these methods, such as VGG16, is much deeper than our network.

360 Viewport Prediction Network

Baseline Methods We compare our method with 3 baseline methods. Our method is denoted as *AttPref*.

(I) We use only the viewing history h_t of current time t and w_k, m_t, v_{w_k} as input features, denoted as *History*. The dimensionality of viewing history h_t is reduced with a 1×1 convolution as the attended video embedding.

(II) We concatenate the preference p_t with w_k, m_t, v_{w_k} as the input features, denoted as *Pref*. These two baseline networks are trained with the same setup as our method for abalation study.

(III) We also compare with the state-of-the-art method (Nguyen, Yan, and Nahrstedt 2018) which is based on 360 saliency information, denoted as *360Sal*.

Dataset and Training Setup We train our preference-aware viewport prediction network with the public dataset(Wu et al. 2017), which contains the records of 48 users viewing 18 videos. We use the 9 videos from its first experiment, as previous work (Nguyen, Yan, and Nahrstedt 2018), in which users watch a 360 video freely without tasks. We use 35 users' records as the training set, 5 for validation and 8 for test. We use the model performs best on the validation set to carry out the following experiments on the test set. Throughout the experiments, we use a 0.5 dropout rate to avoid over-fitting.

Videos and viewing histories are processed with our 360 feature extraction network and stored in local disk. We use a viewing history length $L = 120$, i.e., two minutes of h_t.

[8]https://github.com/jonas-koehler/s2cnn

The prediction time window w_k is set to $\{1, 2, 4, 8\}$, i.e., we predict the future viewport of 1, 2, 4, 8 seconds later. For each record of a user watching a video, we sample the 120s viewing history every 45 seconds. Thus for each sequence of viewing history, we make 120 predictions. The viewport prediction network is trained with our spherical MSE loss and Adam optimizer for 200 epochs with a learning rate of $1e-4$ on a dual GTX 1080Ti GPU computer. We use a training batch size of 6, validation and test batch size of 10.

Visualization of Viewport Prediction In Fig. 7, we present some interesting success and failure cases of viewport prediction. In the first row of success prediction, we observe that even with rapid head movements at $w_k = 1$ and $w_k = 2$, our network is still able to perform properly. While it fails to predict the viewport at $w_k = 4$, it predicts the correct viewport at $w_k = 8$. In the second row, we observe that despite rapid head movement causes our network to not predict correctly at $w_k = 1$ and $w_k = 2$, it's able to predict the correct position at $w_k = 4$ and $w_k = 8$. These findings suggest that the prediction of our network is not driven by the head motion, but by the video content and its correlation to user preference.

The bottom two rows show two cases of failure predictions with some wrong predictions Although we observe that that our network tries to model the probability distribution of future viewport with more than one vMF distributions. A further study could be conducted to evaluate the effects of the number of mixtures.

Impact of Prediction Time Window We first evaluate the effects of different time window w_k. We use three metrics to evaluate the prediction performance, i.e., accuracy, precision and recall. Specifically, accuracy is defined as previous work(Nguyen, Yan, and Nahrstedt 2018; Fan et al. 2017) to be the intersection over union (IOU) index of the set of point predicted to be within future viewport, i.e., $\bar{\lambda}_{i,j} > 0.5$, and the point within viewport on label, i.e., $\lambda_{i,j} = 1$.

In Fig. 6a, we show the prediction accuracy of our method with time window $w_k \in \{1, 2, 4, 6, 8, 10\}$. We observe that all three metrics decrease as the time window increases. For the viewport prediction of 10s later, our network is still able to achieve a more than 40% accuracy. A interesting finding is that compared to accuracy and recall, our network has a much higher precision This can be interpreted as the network tends to make cautious guesses of future viewports, as can be observed in Fig. 7. Also note that we only train our network to predict time window $w_k \in \{1, 2, 4, 8\}$, while our network still performs ideally when predicting the viewport of 6s or 10s later. This confirms that our model can adapt to new data including i.e., time window and the corresponding video.

In Fig. 6b, we show the accuracy of our method compared with the three baselines. We observe that for short-term prediction, the four methods performs similarly. This is mostly because that head motion and recently viewed content play more important roles in short-term prediction and the four methods all take input of these features. However, in terms of long-term viewport prediction, our method outperforms *360Sal* method by a large margin, e.g., 32.3% improvement for $w_k = 6$. This confirms our motivation that pixel information is important but not enough for viewport prediction. Semantic content information and long-term preference can greatly improve the prediction accuracy.

The prediction accuracies of *History* and *Pref* demonstrate that the removal of network modules, such as attention and preference, degrades the performance of long-term prediction. Compared with *History*, the accuracy is reduced by 16.2% for $w_k = 8$. Comparing *History* and *360Sal*, we also observe that even without preference-aware attention, the dedicated 360 video feature extraction network still improves the prediction accuracy by a noticeable margin.

Impact of User Preference To further validate the importance of user preference, we plot the time-varing change of prediction accuracy for different future time window t in Fig. 6c and Fig. 6d. At each timestamp t, the hidden state or the preference embedding p_t is obtained by recurrently processing all past viewing histories $[h_1, h_2, \cdots, h_t]$. Thus a larger t indicates that the input preference embedding p_t contains more past viewing histories. We evaluate if more preference information can improve the perdition accuracy.

In Fig. 6c, we first show the time-varying prediction accuracy with our method, in which the transparent line are the real accuracies and the solid line are the values smoothed with moving average. We observe that bottom subplot of long-term prediction $w_k = 4$ shows a remarkable tendency of accuracy improvement, which demonstrate the effectiveness of extracting user preference.

In Fig. 6d, we show the same figure of *History*. We observe that the top subfigure of Fig. 6c and Fig. 6d are almost identical, indicating that long-term preference doesn't improve the short-term prediction accuracy. As the *History* only takes the the viewing history of past 1 second as input, in the bottom subfigure, its long-term prediction accuracy shows no correlation with the timestamp t, confirming that preference is the reason that improves accuracy in Fig. 6c.

Conclusion

In this paper, we address the major and unique challenges in 360 viewport prediction, such as 360 feature extraction, multi-modal learning, etc. Through meticulous scheme design, we propose a preference-aware viewport prediction network, tailored for 360 immersive scenarios. We start by improving the 360 feature extraction with the S2CNN and design a attention based viewport prediction framework, which embeds user's long-term preference of 360 content and uses it as spatial attention to achieve effective multi-modal data learning. We carefully design a MDN network based viewport predictor and a matching spherical loss to accelerate the training process. We prove the performance of our network through extensive experiments.

Acknowledgements

This work is supported in part by NSFC under Grant No. 61521002, No. 61936011, No. 61872215 and SZSTI under Grant No. JCYJ20180306174057899.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Bazzani, L.; Larochelle, H.; and Torresani, L. 2016. Recurrent mixture density network for spatiotemporal visual attention. *arXiv preprint arXiv:1603.08199*.

Chen, L.; Zhang, H.; Xiao, J.; Nie, L.; Shao, J.; Liu, W.; and Chua, T.-S. 2017. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5659–5667.

Cohen, T. S.; Geiger, M.; Köhler, J.; and Welling, M. 2018. Spherical cnns. *arXiv preprint arXiv:1801.10130*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Esteves, C.; Allen-Blanchette, C.; Makadia, A.; and Daniilidis, K. 2018. Learning so (3) equivariant representations with spherical cnns. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 52–68.

Fan, C.-L.; Lee, J.; Lo, W.-C.; Huang, C.-Y.; Chen, K.-T.; and Hsu, C.-H. 2017. Fixation prediction for 360 video streaming in head-mounted virtual reality. In *Proceedings of the 27th Workshop on Network and Operating Systems Support for Digital Audio and Video*, 67–72. ACM.

Graves, A. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Hou, X.; Dey, S.; Zhang, J.; and Budagavi, M. 2018. Predictive view generation to enable mobile 360-degree and vr experiences. In *Proceedings of the 2018 Morning Workshop on Virtual Reality and Augmented Reality Network*, VR/AR Network '18, 20–26. New York, NY, USA: ACM.

Hu, H.-N.; Lin, Y.-C.; Liu, M.-Y.; Cheng, H.-T.; Chang, Y.-J.; and Sun, M. 2017. Deep 360 pilot: Learning a deep agent for piloting through 360 {\deg} sports video. *arXiv preprint arXiv:1705.01759*.

Jiang, C.; Huang, J.; Kashinath, K.; Marcus, P.; Niessner, M.; et al. 2019. Spherical cnns on unstructured grids. *arXiv preprint arXiv:1901.02039*.

Kostelec, P. J., and Rockmore, D. N. 2008. Ffts on the rotation group. *Journal of Fourier Analysis and Applications* 14(2):145–179.

Koulieris, G. A.; Drettakis, G.; Cunningham, D.; and Mania, K. 2016. Gaze prediction using machine learning for dynamic stereo manipulation in games. In *2016 IEEE Virtual Reality (VR)*, 113–120.

Nguyen, A.; Yan, Z.; and Nahrstedt, K. 2018. Your attention is unique: Detecting 360-degree video saliency in head-mounted display for head movement prediction. In *2018 ACM Multimedia Conference on Multimedia Conference*, 1190–1198. ACM.

Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; and Batra, D. 2017. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, 618–626.

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

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

Su, Y.-C., and Grauman, K. 2017. Learning spherical convolution for fast features from 360 imagery. In *Advances in Neural Information Processing Systems*, 529–539.

Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; and Paluri, M. 2015. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 4489–4497.

Wu, Y., and He, K. 2018. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19.

Wu, C.; Tan, Z.; Wang, Z.; and Yang, S. 2017. A dataset for exploring user behaviors in vr spherical video streaming. In *Proceedings of the 8th ACM on Multimedia Systems Conference*, MMSys'17, 193–198. New York, NY, USA: ACM.

Xiao, M.; Zhou, C.; Liu, Y.; and Chen, S. 2017. Optile: Toward optimal tiling in 360-degree video streaming. In *Proceedings of the 2017 ACM on Multimedia Conference*, MM '17, 708–716. New York, NY, USA: ACM.

Xingjian, S.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; and Woo, W.-c. 2015. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, 802–810.

Xu, M.; Song, Y.; Wang, J.; Qiao, M.; Huo, L.; and Wang, Z. 2017. Modeling attention in panoramic video: A deep reinforcement learning approach. *arXiv preprint arXiv:1710.10755*.

Xu, Y.; Dong, Y.; Wu, J.; Sun, Z.; Shi, Z.; Yu, J.; and Gao, S. 2018. Gaze prediction in dynamic 360 immersive videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5333–5342.

Zhao, Q.; Zhu, C.; Dai, F.; Ma, Y.; Jin, G.; and Zhang, Y. 2018. Distortion-aware cnns for spherical images. In *IJCAI*, 1198–1204.