Video Snippet Retrieval from Image and Text Queries for Gaming (Computer Vision)

Sirisha Akkanapragada
sakkanap@stanford.edu

Abstract

YouTube walkthrough gameplay videos are long in the magnitudes of 6-12 hours. It is hard for gamers to traverse through long videos when they are stuck in a particular segment of the game. We have developed a CNN model and further clustered frames’ vector representations using approximate nearest neighbors. A user can upload a query image and the model would return the top-k videos with all the frames of the particular quest from the videos database available locally. Additionally, the user can also provide a video to segment and the model would return a snippet of the video in which the query image is present.

1 Introduction

Gaming is a fun and interesting domain and it is estimated that 43% of Americans often play video games. It is also found that most of these gamers enjoy strategy or role playing games. But, most of the players get stuck at some point in the gameplay and after several attempts at re-trying the game level, they quit playing the game. This is not ideal from a company perspective like PlayStation because a churn from a game would mean lesser in-game virtual currency purchases, which would further cause a dip in the revenue. The goal of this project is to give gamers small snippets of videos that guide them to proceed with the gameplay and also make their experience enjoyable and seamless.
2 Problem Description

We propose a methodology for suggesting snippets of gameplay video to gamers to help them proceed through a game quest and further improve their gameplay experience on PlayStation platform. Let \( V = \{ V_1, V_2, ..., V_{|V|} \} \) be a database which contains \(|V|\) gameplay videos, where \( V.F = \{ f_1, f_2, ..., f_{|V.F|} \} \) is a video containing \(|V.F| \) successive frames. Let \( I_q \) be a query image or \( T_q \) be a query text then the model outcome would be \( R^{[k]} \), a set of top-k videos where \( R^{i} \) is a video with \(|R.s.f| \) successive selected frames \( \{ f_{s1}, f_{s2}, ..., f_{|R.s.f|} \} \).

3 Relevant work

Visual search refers to searching visual features from the data and querying it from a database. There are several variants to it. The most common being image-to-image retrieval where the query is an image and the database contains a set of images. There are other variants image-to-video, video-to-video and video-to-image. The problem we are investigating is image-to-video retrieval. Early research used image-to-image retrieval methods as mentioned in Sivic et al. In this scenario all the frames in the video were considered as different images and a bag-of-words model was used to retrieve frames without looking into the temporal aspect of videos. Araujo et al. used binarized Fisher Vectors for video retrieval. Further, Zhang et al. performed video retrieval by developing a CNN model followed by clustering the frames and compared the image query with the clustered frames descriptors to return videos that have the query image in it.

But, the problem we are solving is slightly unique. We want to retrieve frames that contain the image, but also frames that might not contain the image in particular, but are part of the quest/level the gamer is playing. So, we need to understand the spatial as well as temporal aspects of the videos. Li et al. suggested a CNN, Channel Attention and convolutional LSTM(CNN-CAtt-ConvLSTM) approach. Further, Pfeuffer et al in their paper suggested the position to place the LSTM for video segmentation.

4 Challenges

An assumption we are making for this problem is that each game’s quest’s landscape would be different from the other and hence, would form different visual feature representations. If the entire game has a monochromatic landscape, it would be hard to differentiate game quests. Another challenge with regard to this problem is that the gameplay would be different from user to user and the sequence of quests could be different making it hard to understand the temporal aspects of the video.

5 Dataset

For this problem, we collected videos for one particular game - Horizon Zero Dawn for which we require snippet recommendations. The videos are from YouTube[9] and there are separate videos for each quest making labelling simpler.

5.1 Same Resolution Video Data

Description: 22 videos with 360p resolution have been downloaded for this task since there are 22 quests for this particular game. All these videos have been uploaded by the same user. Each video can be of different lengths and is between 960 seconds to 4020 seconds and an average video length of 1939 seconds.

Dataset size: After the pre-processing, there are a total of 42,810 images across 22 classes with an average of 1945 frames/images per class.

5.2 Different Resolutions Video Data

Description: Since a user could upload a different resolution image, the model has to be trained on different resolution videos as well as videos from different users. The model is trained on 66 videos
with 360p, 720p and 1080p resolution. There is text of the quest on the top left corner of the video, the model could be learning the length of the text, so we added videos of a different language—Italian.

**Dataset size:** After the pre-processing, there are a total of 106,794 images across 22 classes with an average of 1639 frames/images per class.

### 5.3 Other Games Video Data

The model is trained on a single game Horizon Zero Dawn. But, to test if the model is generating embeddings that are not particular to a game and can be generalized to other games, we have included 3 long videos (test dataset) each of length 11 hours from God of War and Assassins’s Creed.

### 6 Pre-processing

For a video, 24 frames need to be sampled to identify movement in the video. All the videos in the current dataset capture 30 frames in a single second. So, we have sampled the video to extract every 30th frame of the video. The first 60 extracted frames (i.e., 60 seconds of the video) for all the videos were either black images or they displayed the video publisher’s name. Since, this would create noise in our model, we have removed these frames from the model. Also, all the images have been resized to either 256*256*3 or 224*224*3 depending on the model we are implementing by centering, cropping or zooming. The images have also been normalized to have a mean 0 and a standard deviation of 1 for each channel.

![Figure 2: Sample Data](image)

### 7 Methods

Figure 3 illustrates the overview of the model. The frames of each of the videos is fed into a CNN model that is trained to minimize the cross entropy loss between the quests of the game. The model architecture is a pre-trained ResNet-152 model. The learning rate is found by increasing the learning rate and plotting the loss. Once, the loss has stopped decreasing, we determine that point as the optimal learning rate. We unfreeze the layers and also create a differential learning rate wherein the first few layers have a small learning rate while the later layers have a higher learning rate.

After training the CNN model, the second last layer of the CNN model (feature vector representation) which is a (512, 1) vector in the case of a ResNet 152 architecture is used for clustering. When a query image is uploaded by a user, we generate the feature representation for it. The most accurate way to get the similar frames is to compare the query image feature vector with every other frame in the cluster space using cosine similarity. But, this approach does not scale well and has O(N) complexity. To improve performance, we are using the approximate nearest neighbors algorithm to compute the similar feature vectors instead of cosine similarity.

In the scenario wherein a user uploads a query image as well as a video to segment, we pass the video through the trained CNN model to generate feature vector representations and cluster them. The query image is then compared with the clustered feature representations to get similar frames from the video.
8 Experiments

i) Same Resolution Video Data:

Table 1: Experiments (Same Resolution Videos)

<table>
<thead>
<tr>
<th>Model Configuration</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>0.245</td>
</tr>
<tr>
<td>ResNet-34 with pre-trained weights</td>
<td>0.159</td>
</tr>
<tr>
<td>ResNet-34 with training</td>
<td>0.107</td>
</tr>
<tr>
<td>ResNet-50 with pre-trained weights</td>
<td>0.084</td>
</tr>
<tr>
<td>ResNet-50 with training</td>
<td>0.066</td>
</tr>
</tbody>
</table>

The metric we are using for the evaluation of the model is error rate. Note, for the baseline model the error rate is 24.5% on the validation set. It drops to 6.6% error rate for a trained ResNet-50 model.

ii) Different Resolutions Video Data:

Table 2: Experiments (Different Resolution Videos)

<table>
<thead>
<tr>
<th>Model Configuration</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>0.275</td>
</tr>
<tr>
<td>ResNet-34 with pre-trained weights</td>
<td>0.156</td>
</tr>
<tr>
<td>ResNet-34 with training</td>
<td>0.122</td>
</tr>
<tr>
<td>ResNet-50 with pre-trained weights</td>
<td>0.080</td>
</tr>
<tr>
<td>ResNet-50 with training</td>
<td>0.062</td>
</tr>
<tr>
<td>ResNet-152 with training</td>
<td>0.059</td>
</tr>
</tbody>
</table>

The training and validation dataset have a combination of different resolutions video datasets. Adding more data with different resolutions as well as different language data resulted in similar error rate(6.2%) as same resolution videos approach(6.6%) for the same model type. This shows that the model is generalizing on different types of videos. Further, we trained Resnet-152 model for 10 epochs which resulted in an error rate of 5.9%.

9 Results

Figure 4 shows the results for a query image from Horizon Zero Dawn and the model outputs the game quest as well as the most similar images to the query image. The model also returns the indices of the similar image which can be used to find the position of the image in the video. We take the top 3 similar images and retrieve the position of the images in the database of videos. Every frame from

---

1The models are re-trained by unfreezing the layers and choosing a lower learning rate for the earlier layers and higher learning rate for later layers
that position, is then compared with the query image to find if it is similar or not. Once, the similarity is pretty low, we then stop querying the database and return the snippet of video (sequence of frames).

Figure 4: Results-Horizon Zero Dawn

Figure 5 shows the results for a query image from God of War. In this scenario, the user provides a video link as well. The model has not trained on this game. Looking at the similar images, it looks like the model has generated good feature representations and this model can be generalized to other role playing games. Also, the performance of retrieving similar images is 1.12 ms.

Figure 5: Results-God of War

10 Analysis

We also performed error analysis by looking into the confusion matrix for the most accurate model (ResNet-152), it seemed that the lower resolution images were more likely to get misclassified compared to the high resolution images. To overcome the issue of a user uploading a blurred or a low resolution image, we have also trained a GAN to convert a low resolution image to a high resolution image.

Figure 6: Converting Low Resolution to Higher Resolution

11 Conclusion

We have demonstrated that the architecture works for multiple games though it is trained only on one game. The next steps for this model is further increasing the performance of video retrieval since we were using a p2.xlarge instance and there was less space to include a lot of videos. Also, in our scenario since we are only looking at the spatial features in the CNN model, we need to look at the temporal aspect of the model using sequence models like RNN and LSTM.
References


