Video Generation Using 3D Convolutional Neural Network

Shohei Yamamoto
Grad. School of Information
Science and Technology
The University of Tokyo
yamamoto@mi.t.u-tokyo.ac.jp

Tatsuya Harada
Grad. School of Information
Science and Technology
The University of Tokyo
harada@mi.t.u-tokyo.ac.jp

ABSTRACT

Recently, content generation using neural network has been widely studied. Motivated by this recent progress, we studied the generation of videos using only a label as input. In our method, we iteratively minimize two objective functions at the same time : an objective function to evaluate how close the video is to the target class and another to evaluate how natural-appearing the video is. Our proposed method uses the cross-entropy error between the target label and the output of 3D convolutional neural network (C3D) as the objective function for evaluating how close the video is to the target class and uses the Euclidean distance between the input video and the video decoded from our temporal convolutional auto-encoder ("tempCAE") as the objective function for evaluating how natural-appearing the video is. We conducted an experiment evaluating the generated videos using a crowdsourcing service and confirmed the utility of our method.

Keywords

Video Generation; 3D Convolutional Neural Network; Temporal Convolutional Auto-encoder

1. INTRODUCTION

Content generation from neural networks (NNs) has been widely studied. It can be considered that we can recognize the characteristics of an NN by generating an image from it. Simonyan et al. [1] studied the generation of images from convolutional neural networks (CNNs) and Gregor et al. [2] proposed the Deep Recurrent Attentive Writer (DRAW) neural network architecture for image generation. As another approach, image reconstruction from NNs has also been studied [3, 4].

However, studies of content generation have been limited primarily to the case of image generation; video generation has been little studied yet. Addressing this lack, this paper proposes a novel method for generating videos using only a label as input. In this method, we iteratively minimize the cross-entropy error between the target label and the output score of a 3D convolutional neural network (C3D) [5], as an objective measure of how close the video is to the target class, and simultaneously minimize the Euclidean distance between the input video and the video decoded from our proposed temporal convolutional auto-encoder "tempCAE", as a measure of how natural-appearing the video is. The idea of tempCAE is an extension of the idea of convolutional auto-encoder (CAE) [6] to video. We created a very simple dataset of moving objects and conducted generating experiments on it; some generated results are shown in Figure 1.

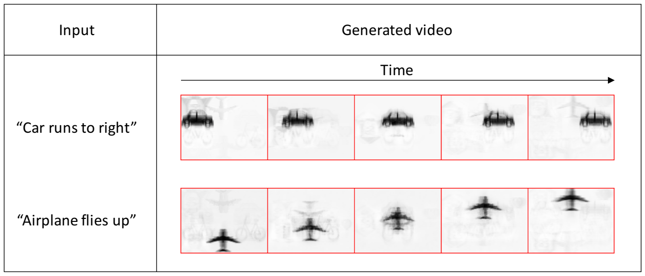

Figure 1: Some examples of video generated by our method. Input is the target label of the video and output is a video of the target label.

Our contributions are the following:

1. We propose a novel method for generating videos from a label alone; the method works by minimizing the cross-entropy error between the target label and the output score of a C3D and minimizing the Euclidean distance between the input video and the video decoded using our proposed temporal convolutional auto-encoder.
2. We propose "temporal convolutional auto-encoder" for extracting temporally local features of videos.
3. We propose a novel method for improving the naturalness of the appearance of generated videos using our temporal convolutional auto-encoder.
4. We demonstrate video generation on a dataset we created and conduct an evaluation experiment using a crowdsourcing service.

2. RELATED WORK

In this section, we introduce the relevant concepts of image reconstruction, image generation, and video reconstruction.

Simonyan et al. [1] studied image generation using CNNs trained on the ImageNet dataset [7]; they computed the target score of the output of the CNN and took the image whose score was highest as the generated result. Gregor et al. [2] proposed the Deep Recurrent Attentive Writer (DRAW) neural network architecture for image generation; their method incorporates attention mechanisms into recurrent neural network (RNN) architectures. Mansimov et al. [8]

MM ’16, October 15-19, 2016, Amsterdam, Netherlands
© 2016 ACM. ISBN 978-1-4503-3603-1/16/10. . . $15.00
DOI: http://dx.doi.org/10.1145/2964284.2967287

extended Gregor et al.'s architecture for generating images from captions with attention.

Image reconstruction using CNNs has also been studied. Reconstruction is different from generation in that the network is given an image as input. Mahendran and Vedaldi [3] reconstructed input images from an intermediate layer of the CNN by minimizing the Euclidean distance of a feature at that layer. In addition, Dosovitskiy and Brox [4] reconstructed input images from an intermediate layer of AlexNet [9] by using an up-convolutional neural network.

As for research in video reconstruction, although Srivastava et al. [10] investigated video reconstruction and future prediction using long short-term memory (LSTM), they focused only on video reconstruction with a video given as an input to LSTM and did not focus on generating video from a label alone.

In this way, as the topic of the content generation, image reconstruction, image generation, video reconstruction have been studied. However, the video generation has not been studied yet. Then, this paper proposes a novel method for generating videos from a label alone, and we conduct an experiment evaluating the generated videos using a crowdsourcing service and confirmed the utility of our method.

3. METHOD

In this section,after describing an existing image-generation method, C3Ds, convolutional auto-encoder (CAE), and our temporal convolutional auto-encoder ("tempCAE"), we introduce our video-generation method. A C3D and tempCAE are modules of our system for video generation.

3.1 Image Generation

As an existing method for image generation, we introduce Simonyan et al.'s method [1]. Simonyan et al. studied image generation using a CNN trained on the ImageNet dataset [7]. They regarded an image X in the following expression as a generated image:

$$\underset{X}{\operatorname{argmax}}\, S_\theta(X) - \lambda\|X\|^2, \qquad (1)$$

where θ is the target label, S_θ is the score of the target label in the classification layer of the CNN, and λ is a regularization parameter. In other words, they sought an L2-regularized image such that the score S_θ is high. By this method, we can find an image that the CNN strongly favors as a target image. In our evaluation experiment, we demonstrate the video-generation by an extension of Simonyan et al.'s method to video generation as a comparison method.

3.2 3D Convolutional Neural Network

A 3D convolutional neural network (C3D) is a classifier for video classification using CNN. In the conventional method for image classification, we convolve only for the horizontal and vertical directions (= 2D convolution). We can extract the spatial information of the input image by 2D convolution. In the C3D method, by contrast, we convolve not only for the horizontal and vertical directions but also for the temporal directions. From experiments conducted by Tran et al. (the C3D developers), it is known that 3D convolution can preserve the temporal information of the input signals in the resulting output features.

3.3 Convolutional Auto Encoder

Convolutional auto encoder (CAE) [6] is an application of AE. In the method of CAE, we convolve the input image in the same way as CNN and reconstruct the input image from feature map. So, the formula of CAE is as follows.

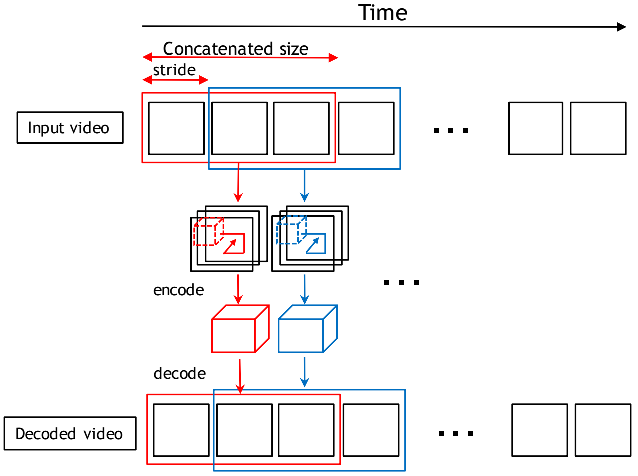

Figure 2: Our proposed temporal convolutional auto-encoder. In this module, we concatenate several frames (concatenated size) of the video to an input 'multi-channels image' and encode and decode input video.

$$\begin{aligned} encode : \boldsymbol{h}^k &= \sigma(\boldsymbol{x} * W^k + b^k) && (2)\\ decode : \boldsymbol{y} &= \sigma(\sum_{k\in H} \boldsymbol{h}^k * W'^k + b'^k) && (3) \end{aligned}$$

Therein, $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{h}^k$ respectively denote the input image, the output image, and the k-th feature map. Also, H identifies the group of feature map, and $*$ denotes the 2D convolution. So, when we train the CAE, we train the parameters W^k, W'^k, b^k, and b'^k in the above formula.

3.4 Temporal Convolutional Auto Encoder

As an auto-encoder for video, we propose the temporal auto-encoder "tempCAE". The idea of tempCAE is an extension of the idea of CAE [6] to the case of video. We show an overview of the method in Figure 2. In this method, we concatenate some frames (concatenated size) of the video to an input "multi-channels image" and encode the input to the feature map in the same way as 3D convolution. After encoding, we decode in the same way as normal auto encoder (AE). So, the formula of tempCAE is as follows.

$$\begin{aligned} encode : \boldsymbol{h}[i, CS] &= \sigma(\boldsymbol{x}[i, CS] * W + b) && (4)\\ decode : \boldsymbol{y}[i, CS] &= \sigma(\boldsymbol{h}[i, CS] \cdot \tilde{W} + b') && (5) \end{aligned}$$

Therein, CS denotes the size of concatenation (= concatenated size). Also, $\boldsymbol{x}[i, CS]$, $\boldsymbol{h}[i, CS]$, and $\boldsymbol{y}[i, CS]$ respectively denote an input "multi-channels image" to which we concatenated the some frames (from the i-th frame to the $i+CS$-th frame of the video), the feature map, and the output image. Also, $*$ and $\cdot$ respectively denote the 3D convolution and the linear multiplication. In the training step of tempCAE, we train the parameters W, W', b, and b' in the above formula. We can extract temporally local features of videos using tempCAE.

3.5 Video Generation

We establish two objective functions as follows.

$$\begin{aligned} \boldsymbol{L}_{\text{C3D}}(\boldsymbol{X}) &= -\sum_{\theta}^{N} \text{C3D}(\boldsymbol{X}, \theta) \log Y(\theta) && (6)\\ \boldsymbol{L}_{\text{tempCAE}}(\boldsymbol{X}[i, CS]) &= \|\text{tempCAE}(\boldsymbol{X}[i, CS]) - \boldsymbol{X}[i, CS]\|^2 && (7) \end{aligned}$$

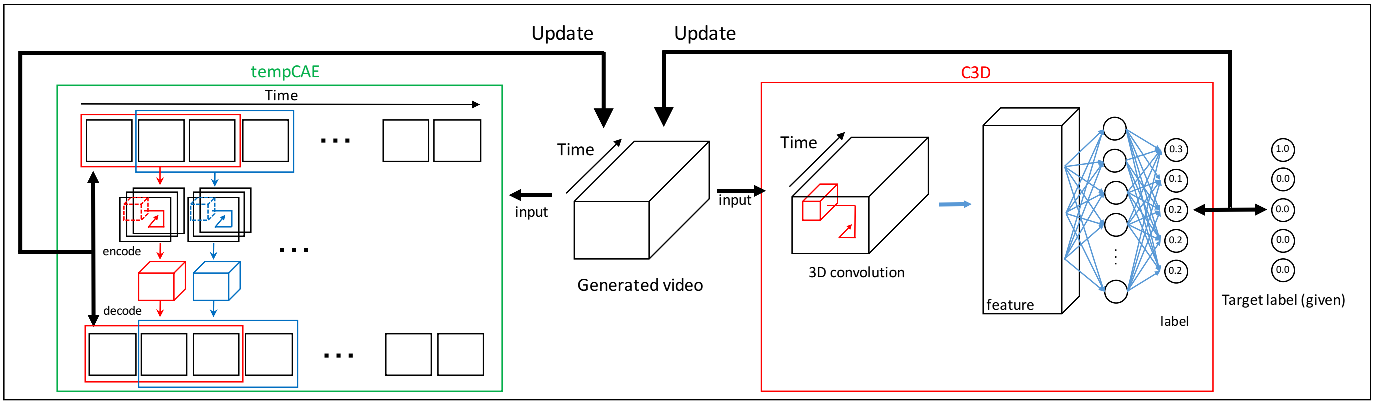

Figure 3: System for generation of videos from a label alone. In this system, we generate video by iteratively minimizing two objective functions (the cross-entropy error between the target label and C3D output, as an objective measure of how close the video is to the target class, and the Euclidean distance between the input video and the video decoded from tempCAE, as a measure of how natural-appearing the video is) at the same time.

Therein, $\mathrm{C3D}(\boldsymbol{X}, \theta)$ denotes the output score of category θ when the video X is input to C3D, and $\mathrm{tempCAE}(\boldsymbol{X}[i, CS])$ denotes the decoded video when the some frames (= from the i-th frame to the $i + CS$-th frame) of video is input to tempCAE, and $Y(\theta)$ denotes the target score of θ (so, $Y(\theta)$ will be 1.0 or 0.0). In short, Function (6) represents the cross-entropy error between the target label and output score of C3D (we call this the "target error"), and Function (7) represents the square of the Euclidean distance between the generated video and video decoded from tempCAE (we call this the "decoded error"). N denotes the number of classes.

The reason we establish the above objective functions is as follows. If the target error is minimized, it means that the video has features similar to the target video. However, just the fact that the target error is minimized doesn't necessarily mean that the generated video appears natural to people who view it. Therefore, we use the decoded error of the video. If the decoded error is minimized, it means that the generated video has features similar to the entire training datasets and that the video looks natural to people. For these reasons, we regard a video X that minimizes above objective functions as a generated result.

After we trained C3D and tempCAE on a video dataset, we use these trained networks to generate videos. For the video generation process, we started with a random video and updated it so that it approached the target class depending on the values of the above objective functions. As the updating method, we used a gradient descent (GD) method as follows.

$$\boldsymbol{X}_{t+1} = \boldsymbol{X}_t - \eta_{C3D} \frac{\partial \boldsymbol{L}_{C3D}}{\partial \boldsymbol{X}_t} - \eta_{tempCAE} \frac{\partial \boldsymbol{L}_{tempCAE}}{\partial \boldsymbol{X}_t} \tag{8}$$

Therein, η_{C3D}, $\eta_{tempCAE}$ respectively denote updating weight of C3D and tempCAE. We show an overview of our system in Figure 3.

4. EXPERIMENTS

We experimented on a simple video dataset that we created for the purpose.

4.1 Datasets

Although large video datasets such as HMDB51 [11] and UCF101 [12] already exist, these datasets are too complicated to use for generating videos, and it is unreasonable to do video-generation experiments on them. As a dataset to use in our video-generation experiments to confirm the utility of our method, it is desirable that the objects of each class are simple and the moving of each class is easy-to-understand. Thus, we created a simple dataset to use in our video-generation experiments. The details of the dataset are as follows.

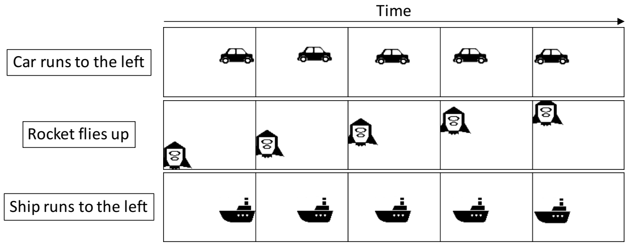

Figure 4: Some examples from the dataset we created.

- the number of classes : 11 classes (a car runs to the right/left, a rocket flies up/down, an airplane flies up/down, a ship runs to the right/left, a bicycle runs to the right/left, a balloon flies up)
- size of dataset : 200 data for each class
- size of videos : 128 * 96 px, 16 frames

We show some examples from our dataset in Figure 4 and other examples in supplemental material.

4.2 Comparison Method

As the comparison method in our video-generation experiments, we set the following two objective functions:

$$\boldsymbol{L}_{\mathrm{C3D}}(\boldsymbol{X}) = -\sum_{\theta}^{N} \mathrm{C3D}(\boldsymbol{X}, \theta) \log Y(\theta) \tag{9}$$

$$\boldsymbol{L}_{\mathrm{norm}}(\boldsymbol{X}) = \|\boldsymbol{X}\|^2 \tag{10}$$

The function (9) represents the cross-entropy error between the target label and the output score of C3D, and the function (10) represents the L2 norm of the generated video. So, in this method, we would like to find an L2-regularised video such that the cross-entropy error between the target label and the output score of C3D is minimized. In short, it can be considered that this method is extension of Simonyan et al.'s image-generation method (mentioned in Section 1) to video-generation. As the updating method, we used a gradient descent (GD) method as follows.

$$\boldsymbol{X}_{t+1} \quad = \quad \boldsymbol{X}_t - \eta_{C3D}\frac{\partial \boldsymbol{L}_{C3D}}{\partial \boldsymbol{X}_t} - \eta_{norm}\frac{\partial \boldsymbol{L}_{norm}}{\partial \boldsymbol{X}_t} \qquad (11)$$

Therein, η_{norm} is a regularization parameter.

4.3 Results

We experimented on our simple video dataset. It took 30 minutes for generating a video using suitable GPU. Although the dataset may look like too simple, in terms of computational cost, it can be considered that such a simple dataset is appropriate for experiments on new method.

We show some of the generated videos in Figure 5 and others in supplemental material. As shown in the figure, the videos generated by our method with tempCAE are more natural-appearing than those generated by the comparison method, and we can easily recognize the video categories. Furthermore, we can see difference in the backgrounds of the generated videos : whereas the backgrounds in the videos generated by the comparison method are very noisy, videos generated by our method have backgrounds that are close to white. However, we can also see faint images of objects of other classes (*e.g*., an airplane appears in the background of the video of a car). It is reasonable to assume that the features of the other classes appear because the dataset features were generated by using tempCAE trained on the entire dataset.

4.4 Evaluation

To evaluate our method quantitatively, it is desirable to have a human check the generated results. Thus, we used CrowdFlower [13] which is a crowdsourcing service where we can ask a member (called a "contributor") to do a variety of tasks using CrowdFlower. We showed contributors some videos generated by the two methods and a list of categories and asked them to choose the appropriate category for each video. The list of categories consisted of 11classes of the dataset, "Other", and "Unclear" (13 classes in total). The number of contributors who participated in our experiment was 300. We show the results and parameters on Table 1. We can clearly see the notable difference in the accuracy: the accuracy of our method was more than twice that of the comparison method.

Table 1: Parameters and accuracy of CrowdFlower.

Method	Iteration	η_{C3D}	η_{tempCAE}	η_{norm}	Accuracy
Ours	50000	10.0	10.0	none	88.2 %
Comparison	50000	10.0	none	5.0	42.7 %

5. APPLICATION

As an application of our method, we generated new videos as shown in Figure 6. In this application, we generated videos from multiple labels at the same time (*e.g*., "Car runs to the right" and "Rocket flies up"). In this way, our method can be applied to a system for the generation of new videos.

6. CONCLUSION

We have proposed a novel method for generating videos by iteratively minimizing two objective functions at the same time: the objective function for evaluating how close the video is to the target class and another function for evaluating how natural-appearing the video is. As the objective function for evaluating how close the video is to the target class, our proposal uses the cross-entropy error between the target label and the output of C3D, and for evaluating how natural-appearing the video is, the Euclidean distance between the input video and the video decoded from tempCAE is used. We conducted a video-generation experiment on our video dataset and quantitatively evaluated the results using a crowdsourcing service, and thereby confirmed the utility of our method. In our future research, we plan to generate a larger video dataset and develop an architecture for generating videos not only from words but also from short captions.

7. ACKNOWLEDGEMENT

This work was funded by ImPACT Program of Council for Science, Technology and Innovation (Cabinet Office, Government of Japan).

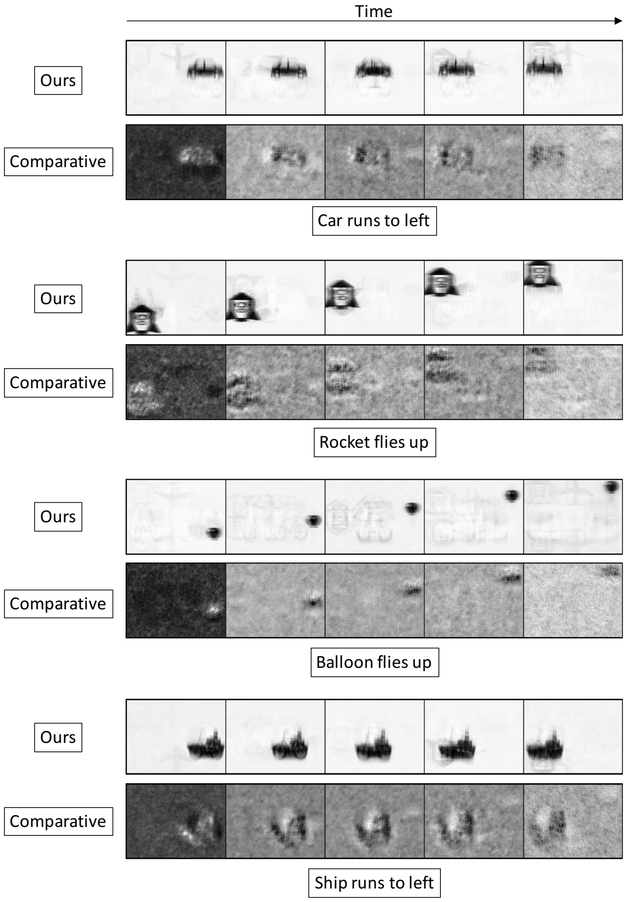

Figure 5: Some videos generated by our method and comparative method.

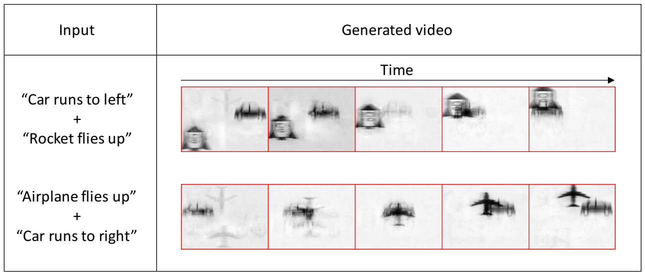

Figure 6: Some examples of videos generated from multiple labels.

8. REFERENCES

[1] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[2] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *ICML*, 2015.

[3] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *CVPR*, 2015.

[4] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *CVPR*, 2016.

[5] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

[6] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In *Artificial Neural Networks and Machine Learning–ICANN 2011*, pp. 52–59. Springer, 2011.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[8] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating images from captions with attention. In *ICLR*, 2016.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[10] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015.

[11] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *ICCV*, 2011.

[12] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[13] CrowdFlower. http://www.crowdflower.com/.