Conditional GAN with Discriminative Filter Generation for Text-to-Video Synthesis

Yogesh Balaji[1] *, **Martin Renqiang Min**[2], **Bing Bai**[2], **Rama Chellappa**[1] and **Hans Peter Graf**[2]
[1]University of Maryland, College Park
[2]NEC Labs America - Princeton
yogesh@cs.umd.edu, {renqiang, bbai}@nec-labs.com, rama@umiacs.umd.edu, hpg@nec-labs.com

Abstract

Developing conditional generative models for text-to-video synthesis is an extremely challenging yet an important topic of research in machine learning. In this work, we address this problem by introducing Text-Filter conditioning Generative Adversarial Network (TFGAN), a conditional GAN model with a novel multi-scale text-conditioning scheme that improves text-video associations. By combining the proposed conditioning scheme with a deep GAN architecture, TFGAN generates high quality videos from text on challenging real-world video datasets. In addition, we construct a synthetic dataset of text-conditioned moving shapes to systematically evaluate our conditioning scheme. Extensive experiments demonstrate that TFGAN significantly outperforms existing approaches, and can also generate videos of novel categories not seen during training.

1 Introduction

Generative models have gained much interest in the research community over the last few years for unsupervised representation learning. Generative Adversarial Networks (GANs) [Goodfellow *et al.*, 2014] have been one of the most successful generative models till date. Following its introduction in 2014, significant progress has been made towards improving the stability, quality and the diversity of the generated images [Salimans *et al.*, 2016][Karras *et al.*, 2017]. While GANs have been successful in the image domain, recent efforts have extended it to other modalities such as text [Wang *et al.*, 2018a], graphs [Wang *et al.*, 2018b], etc.

In this work, we focus on the less studied domain of videos. Generating videos are much harder than images because the additional temporal dimension makes generated data extremely high dimensional, and the generated sequences must be both photo-realistically diverse and temporally consistent. We tackle the problem of text-conditioned video synthesis where the input is a text description and the goal is to synthesize a video corresponding to the input text. This problem has many potential applications, some of which include generating synthetic data for machine learning tasks, domain adaptation, multimedia applications, etc.

Two recent works that address the problem of text-conditioned video generation include [Li *et al.*, 2018] and [Pan *et al.*, 2017]. Both these methods are variants of conditional GAN applied to the video data. In spite of some successes, they have the following limitations: (1) They employ 3D transposed convolution layers in the generator network, which constrains them to only produce fixed-length videos. (2) Their models are trained on low-resolution videos - results are shown only at a 64×64 resolution. (3) Text conditioning is performed using a simple concatenation of video and text features in the discriminator: Such a conditioning scheme may perform well on certain datasets, but maybe inadequate for capturing rich video-text variations.

In this work, we aim to address all the concerns above. First, to model videos of varying lengths, we use a recurrent neural network in the latent space and employ a shared frame generator network similar to [Tulyakov *et al.*, 2018]. Second, we present a model for generating higher-resolution videos by using a ResNet-style architecture in the generator and the discriminator network. Third, we propose a new multi-scale text-conditioning scheme based on discriminative convolutional filter generation to strengthen the associations between the conditioned text and the generated video. We call our model Text-Filter conditioning GAN (TFGAN). Finally, we construct a synthetic moving shapes dataset to extensively evaluate the effectiveness of the proposed conditioning scheme. Constructing this synthetic dataset is extremely useful as it captures rich text-video variations that are lacking in the datasets currently used for text-to-video synthesis [Li *et al.*, 2018; Pan *et al.*, 2017]. We demonstrate the effectiveness of our approach on (1) real-world datasets such as Kinetics Human Action dataset [Kay *et al.*, 2017] that has complex videos with high diversity but has relatively simple text-video variations, and (2) synthetic dataset that has simple video dynamics but models complex text-video associations.

In summary, our contributions in this work are as follows: (i) A new conditional GAN with an effective multi-scale text-conditioning scheme based on discriminative convolutional filter generation is proposed; (ii) A synthetic dataset for studying text conditioning in video generation is presented; (iii) A framework for generating complex video sequences and capturing rich text-video variations is presented.

*Work done during internship at NEC Laboratories America

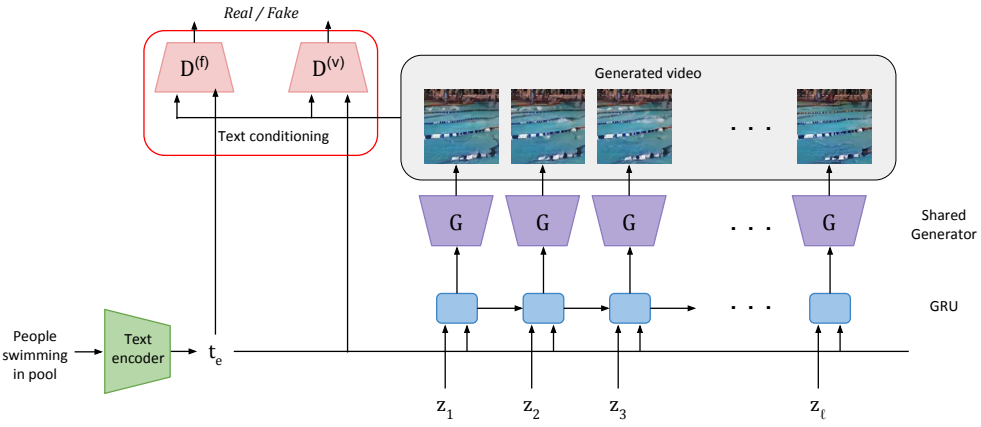

Figure 1: Our TFGAN framework. The box highlighted in red is where the conditioning is performed and is expanded in Fig. 2.

2 Related Work

Two popular approaches to generative modeling include GANs [Goodfellow *et al.*, 2014] and Variational Autoencoders(VAEs) [Kingma and Welling, 2014]. GANs are formulated as a 2-player *minimax* game between a generator and a discriminator network, while VAEs are based on variational inference where a variational lower bound of observed data log-likelihood is optimized. Among the two approaches, GANs have generated significant interest as they have been shown to produce images of high sample fidelity and diversity [Karras *et al.*, 2017] [Brock *et al.*, 2018].

A variant of GAN models is conditional GANs where the generator network is conditioned on input variables of interest. Such a conditioning input can be labels [Odena *et al.*, 2017], attributes [Yan *et al.*, 2016], text [Zhang *et al.*, 2017; Xu *et al.*, 2018] or even images [Zhu *et al.*, 2017]. We focus on text conditioning since it is relevant to this work. One of the first works to perform text-conditioned image synthesis is [Reed *et al.*, 2016]. Their method was only shown to synthesize low-resolution images. To improve the resolution, [Zhang *et al.*, 2017] proposed stacking multiple GAN architectures, each producing images of increasing resolution. While the above two methods perform conditioning using the global text representation, [Xu *et al.*, 2018] adopts an attention mechanism to focus on fine-grained word-level representations to enable improved conditioning.

While image generation is a well studied problem, there has been very little progress in the domain of video generation. [Vondrick *et al.*, 2016] proposed a GAN architecture based on 3D convolutions to generate video sequences, but it can only generate fixed-length videos. [Tulyakov *et al.*, 2018] proposed using a recurrent neural network in the latent space to model videos of varying lengths. While these models are not designed to handle video generation from text, [Li *et al.*, 2018] and [Pan *et al.*, 2017] perform text-conditioned video synthesis by using the sentence embedding as a conditional input. However, both of these conditional generative models are based on 3D convolutions, they can only produce fixed-length low-resolution videos. In this work, we address this issue by developing an architecture capable of producing higher-resolution videos of varying lengths.

A key component in our method is a novel conditioning scheme based on discriminative convolutional filter generation. Generating the future frames guided by filters generated from previous frames have been explored in Dynamic Filter Networks [Jia *et al.*, 2016]. A similar approach is taken in [Li *et al.*, 2018], where filters are generated conditioned on text. Both these approaches apply the generated filters on the intermediate responses of the generator network. This is computationally expensive and technically challenging especially for deeper generatator architectures as we need to generate a large number of semantically meaningful filters. In contrast, applying multi-scale text-conditioned filters in the discriminator architecture as done in our approach, is computationally easier and effective as demonstrated by our experimental results.

3 Method

We first provide a formal description of the problem being addressed. We are given access to N data points $\{(\mathbf{v}^{(n)}, \mathbf{t}^{(n)})\}_{n=1}^{N}$ sampled from an underlying joint distribution $p(\mathbf{v}, \mathbf{t})$ in the video-sentence space. Here, each $\mathbf{v}^{(n)} \in \mathbb{R}^{T \times C \times W \times H}$ is a video clip and $\mathbf{t}^{(n)}$ is a sentence description. We are interested in learning a model capable of sampling from the unknown conditional distribution $p(\mathbf{v}|\mathbf{t})$. Similar to conditional GANs [Mirza and Osindero, 2014], we formulate the problem as learning a transformation function $\mathbf{G}(\mathbf{z}, \mathbf{t})$ from a known prior distribution $P_Z(\mathbf{z})$ and the conditional input variable $\mathbf{t}$ to the unknown conditional distribution $p(\mathbf{v}|\mathbf{t})$. The function $\mathbf{G}$ is optimized using an adversarial training procedure.

3.1 Model Framework

Our conditional GAN framework is shown in Fig. 1, in which the video generator employs a RNN as in MoCoGAN [Tulyakov *et al.*, 2018]. Unlike MoCoGAN, our model does not have separate **Mo**tion and **Co**ntent modeling or any discrete motion/action vector. The novelty of our proposed framework involves a muti-scale text-conditioning scheme,

in which the text description $\mathbf{t}$ is passed to a text encoder $\mathbf{T}$ to get a text representation $\mathbf{t}_e$.

$$\mathbf{t}_e = \mathbf{T}(\mathbf{t}), \tag{1}$$

where $\mathbf{t}$ is a word embedding matrix containing pre-trained distributed representation vectors of the words in a given text description. The text encoder $\mathbf{T}(\cdot)$ is a CNN containing several convolutional layers and a fully connected layer (RNN-based model can also be used). The encoded text representation along with a sequence of noise vectors are passed to a GRU recurrent neural network to produce a trajectory in the latent space as follows,

$$\mathbf{h}_m = GRU([\mathbf{h}_{m-1}, \mathbf{t}_e, \mathbf{z}_m]), \tag{2}$$
$$\mathbf{z}_m \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{3}$$

where $m = 1, \ldots, l$ and l denotes the number of frames in the video sequence. This sequence of latent vectors is then passed to a shared frame generator model G to produce the video sequence,

$$\mathbf{v}_m = \mathbf{G}(\mathbf{h}_m), \quad m = 1, \ldots, l. \tag{4}$$

The generated video is then fed to two discriminator models - $\mathbf{D}^{(f)}$ and $\mathbf{D}^{(v)}$. $\mathbf{D}^{(f)}$ is a frame-level discriminator that classifies if the individual frames in the video are real/fake, whereas the video discriminator $\mathbf{D}^{(v)}$ is trained to classify the entire video as real/fake. The discriminator models $\mathbf{D}^{(f)}$ and $\mathbf{D}^{(v)}$ also, respectively, take the text encoding $\mathbf{t}_e$ as input to enforce text conditioning. Unlike the MoCoGAN model [Tulyakov *et al.*, 2018], we employ much deeper ResNet-style architectures in generator and discriminator networks for high-quality video generation. More details can be found in the supplementary material[1].

3.2 Text-Filter Conditioning

To build strong conditional models, it becomes important to learn good video-text associations in the GAN model. A standard technique is to sample negative $(\mathbf{v}, \mathbf{t})$ pairs (wrong associations) and train it as fake class, while the correct $(\mathbf{v}, \mathbf{t})$ pairs are trained as real class in the discriminator network. Since the generator is updated using the gradients from the discriminator, it becomes important to effectively fuse the video and text representations in the discriminator so as to make the generator condition on the text reasonably well. Previous methods [Li *et al.*, 2018; Pan *et al.*, 2017] use a simple concatenation of text and video features as the feature fusion strategy. We found that this simple strategy produces poor conditional models in datasets where there are rich text-video variations (refer to Section. 4 for more details).

Our proposed Text-Filter conditioning GAN (TFGAN) model aims at improving text conditioning for video generation models. In TFGAN, we employ a scheme based on generating discriminative convolutional filters from text features, which are then convolved with image features in the discriminator network. This scheme, which we call Text-Filter conditioning, is shown in Fig. 2.

[1]Supplementary material can be found at `https://tinyurl.com/y3jedfxy`

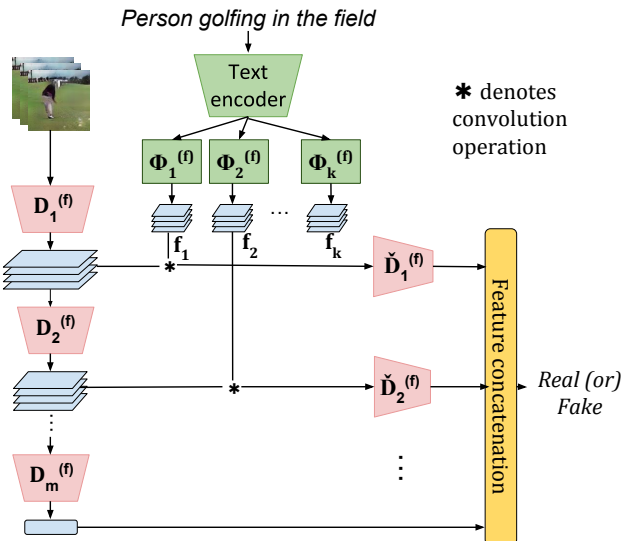

Figure 2: Illustration of our Text-Filter conditioning strategy.

Let the discriminator network $\mathbf{D}^{(f)}$ (equivalently for $\mathbf{D}^{(v)}$) be divided into multiple sub-networks $\{\mathbf{D}_i^{(f)}\}_{i=1}^m$ such that the entire discriminator response can be represented as a cascade of these sub-network responses,

$$\mathbf{D}^{(f)}(\mathbf{x}) = \mathbf{D}_m^{(f)} \circ \mathbf{D}_{m-1}^{(f)} \circ \ldots \mathbf{D}_1^{(f)}(\mathbf{x}), \tag{5}$$

where these sub-networks $\{\mathbf{D}_i^{(f)}\}_{i=1}^m$ can be as small as a single layer, or can be a cascade of multiple layers. Let $\mathbf{d}_i^{(f)}$ denote the output of the i^{th} sub-network of the frame discriminator,

$$\mathbf{d}_i^{(f)} = \mathbf{D}_i^{(f)} \circ \mathbf{D}_{i-1}^{(f)} \ldots \mathbf{D}_1^{(f)}(\mathbf{v}).$$

From the text features, we generate a set of convolution filters $\{\mathbf{f}_i^{(f)}\}_{i=1}^m$ for frame-level discrimination,

$$\mathbf{f}_i^{(f)} = \Phi_i^{(f)}(\mathbf{t}_e), \quad , i = 1, \ldots, m, \tag{6}$$

where $\Phi_i^{(f)}$ (equivalently $\Phi_i^{(v)}$ at video-level) is a discriminative convolutional filter generation network at frame level. Similarly, let $\mathbf{d}_i^{(v)}$ denote the output of the i^{th} sub-network of the video discriminator, and $\{\mathbf{f}_i^{(v)}\}_{i=1}^m$ denote the discriminative convolutional filters at the video level. The frame-level filters $\mathbf{f}_i^{(f)} \in \mathbb{R}^{n_{in} \times n_{out} \times w \times h}$, whereas the video-level filters $\mathbf{f}_i^{(v)} \in \mathbb{R}^{n_{in} \times n_{out} \times c \times w \times h}$. Here, n_{in}, n_{out} are the number of input and output channels of $\mathbf{d}^{(i)}$, and c, w and h are the filter dimensions.

Each filter $\mathbf{f}_i^{(f)}$ is now convolved with the discriminator response $\mathbf{d}_i^{(f)}$, and this convolved output is passed through additional convolution layers $\tilde{\mathbf{D}}_i^{(f)}$ to get a vector $\tilde{\mathbf{d}}_i^{(f)}$ as output. These vectors $\{\tilde{\mathbf{d}}^{(i)}\}_{i=1}^m$ are then concatenated to obtain the final frame-text representation $\mathbf{O}^{(f)}$, which is used to classify the frame-text pair as real or fake.

$$\tilde{\mathbf{d}}_i^{(f)} = \tilde{\mathbf{D}}_i^{(f)}(\mathbf{f}_i^{(f)} * \mathbf{d}_i^{(f)}), \tag{7}$$
$$\mathbf{O}^{(f)} = [\tilde{\mathbf{d}}_1^{(f)}, \tilde{\mathbf{d}}_2^{(f)}, \ldots, \mathbf{d}_m^{(f)}], \tag{8}$$

where [.] denotes vector concatenation. The video-text representation $\mathbf{O}^{(v)}$ is generated using a similar procedure. Since the generated convolutional filters $\{\mathbf{f}_i\}$ are applied to discriminator sub-network outputs $\{\mathbf{d}_i\}$ from different layers, the resulting text conditioning effectively imposes semantic constraints extracted from input text to the generated individual frames and video clips at different feature abstraction levels.

3.3 Training Algorithm

The discriminator models $\mathbf{D}^{(f)}$ and $\mathbf{D}^{(v)}$, and the generator model $\mathbf{G}$ are trained using an adversarial game as done in the standard conditional GANs [Mirza and Osindero, 2014]. However, since we employ deep Resnet-style architectures for generator and discriminator networks, it was important to stabilize the GAN training. We use the regularizer as proposed in [Mescheder *et al.*, 2018] where the norm of the discriminator gradients is penalized. The equations for optimization updates are provided in the Supplementary material.

4 Experiments

This section discusses the experimental validation of our TFGAN model. We first describe a benchmark synthetic dataset we created for the task of text-to-video generation that captures rich video-text associations. The performance of our system is studied exhaustively using this dataset. Then, results are presented on two challenging real-world video dataset - Kinetics human action video dataset [Kay *et al.*, 2017] and Epic-Kitchens dataset [Damen *et al.*, 2018]. Further experiments and ablation studies are shown in the supplementary material. Our code is publicly available at `https://github.com/minrq/CGAN_Text2Video`.

4.1 Moving Shapes Dataset

Dataset Creation

To better understand the task of text-to-video synthesis, we created a dataset of moving shapes where a shape moves along a trajectory as described by the corresponding text description. Two versions of the dataset were created – one with static background which we call *Shapes-v1 dataset* and one with dynamic background which we call *Shapes-v2 dataset*. Some samples from these datasets are shown in Fig. 3. More details about the dataset creation are provided in the Supplementary material.

Quantitative Evaluation

One important benefit of creating the synthetic dataset is that it provides a framework for quantitative evaluation of the text-conditioning. First, five attribute classifiers (shape, size, color, motion and direction classifiers) are trained on the real data using the ground truth attributes (we have access to ground-truth attributes as they were stored while creating the dataset). These trained attribute classifiers are then used to verify if the attributes of a generated video match the attributes corresponding to the input text. This *attribute classification accuracy* is used as a metric to compare GAN models. To evaluate a GAN model, videos are generated for each text desciption in the test set, and the average attribute classification accuracy for these generated videos is reported. Higher the score, better is the model.

We compare TFGAN with the following models:

- Simple concat (SC): a conditional GAN model trained using simple text-video feature concatenation in the discriminator network
- Simple concat + Multiscale D (SC+MD): a conditional GAN model with discriminator network employing a multi-scale architecture. More specifically, outputs at the intermediate layers of the discriminator network are extracted and pooled together to get a multi-scale feature representation. This feature is then used to classify if the video-text pair is real or fake. Similar to the previous model, a simple feature concatenation is used to perform text conditioning.
- Text-filters on generator (TF on G): Instead of applying the text-filters on the discriminator network as done in TFGAN, filters extracted from the text are convolved with the intermediate responses of the generator network. In the discriminator network, simple feature concatenation is used to perform the conditioning. Note that this model is similar to the approach in [Li *et al.*, 2018] where the filters extracted from text are applied to the gist image (which is a part of the generator network).

More details about the architecture and hyper-parameters are given in the supplementary material. Some sample generations of TFGAN model are shown in Fig. 3.

Table 1a reports the quantitative evaluation on the *Shapes* dataset. We observe that TFGAN with text-filter conditioning achieves the best performance compared to other models. Only marginal performance gain is obtained by using filters in the generator network. We argue that this is because generating discriminative filters are much easier than generative filters. Also, the use of multi-scale architecture in discriminator alone does not improve text conditioning. This shows that performance improvement of TFGAN does not come from multi-scale architecture in itself, but in how the text conditioning is applied.

Exploratory Experiments

Sentence Interpolation: In this experiment, we depict conditional interpolation whereby frames in a video transition corresponding to the interpolation between two sentence descriptions. Let S_1 and S_2 denote the two sentences that are interpolated, and $(\mathbf{t}^{(f)}_{S_1}, \mathbf{t}^{(v)}_{S_1})$ and $(\mathbf{t}^{(f)}_{S_2}, \mathbf{t}^{(v)}_{S_2})$ denote their corresponding feature representation obtained from the text encoder $\mathbf{T}$. For each frame to be generated, the corresponding conditioning feature is obtained by a linear interpolation between these two representations:

$$(\mathbf{t}^{(f)}_i, \mathbf{t}^{(v)}_i) = (1-\alpha)(\mathbf{t}^{(f)}_{S_1}, \mathbf{t}^{(v)}_{S_1}) + \alpha(\mathbf{t}^{(f)}_{S_2}, \mathbf{t}^{(v)}_{S_2})$$

Instead of using a fixed text representation $(\mathbf{t}^{(f)}, \mathbf{t}^{(v)})$ to condition all frames in the generator, we use $(\mathbf{t}^{(f)}_i, \mathbf{t}^{(v)}_i)$ as input to the frame i. Some sample interpolations are shown in Fig. 4. We observe smooth interpolations corresponding to the input sentence transitions. When interpolating between blue square and red square, some intermediate frames are generated with pink shade. Interestingly, none of the samples in the dataset contain pink color. In the second figure,

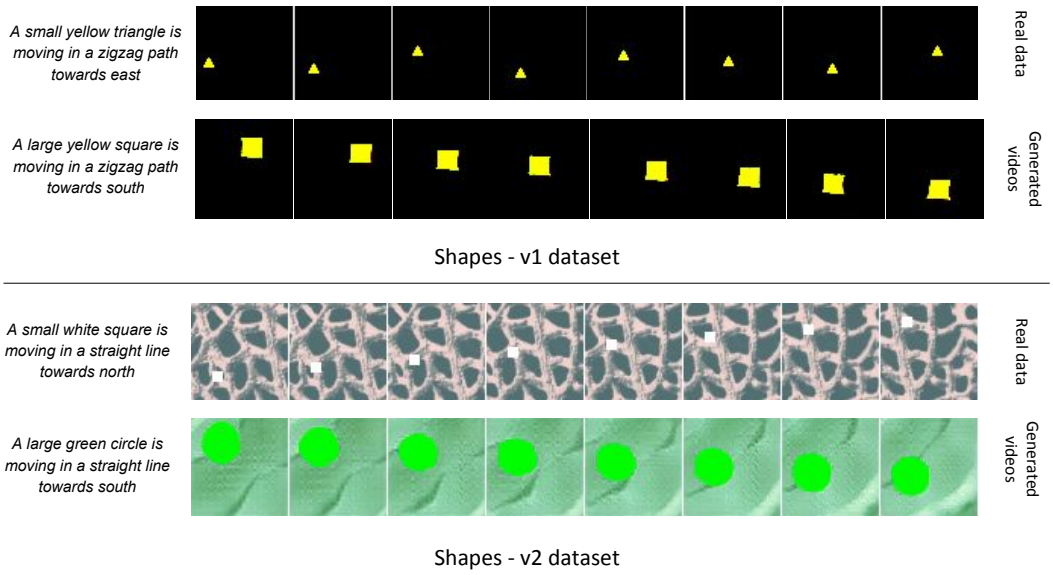

Figure 3: Samples from the Moving Shapes Dataset: In each set of images, the top row corresponds to the original video and the bottom two rows correspond to the video generated by our TFGAN model (Better viewed in color).

Method	Shape	Color	Size	Motion	Direction
Shapes-v1 dataset					
SC	70.18	99.23	83.69	96.83	99.00
SC + MD	65.25	99.91	74.31	99.12	99.11
TF on G	72.91	99.41	82.98	**99.51**	99.81
TFGAN	**97.66**	**99.99**	**98.60**	99.40	**100.00**
Shapes-v2 dataset					
SC	55.88	96.04	73.71	83.30	93.51
SC + MD	56.12	97.71	69.15	87.20	92.00
TF on G	59.76	95.75	82.16	85.00	95.40
TFGAN	**81.10**	**99.99**	**89.82**	**99.80**	**100.00**

(a) Attribute classification accuracy (in %) on *Shapes* dataset

Attribute	Accuracy (in %)
Shape	96.21
Color	99.78
Size	98.77
Motion	96.23
Direction	99.42

(b) Attribute classification accuracy on novel categories on *Shapes-v1* dataset

Table 1: Quantitative analysis: Shapes dataset

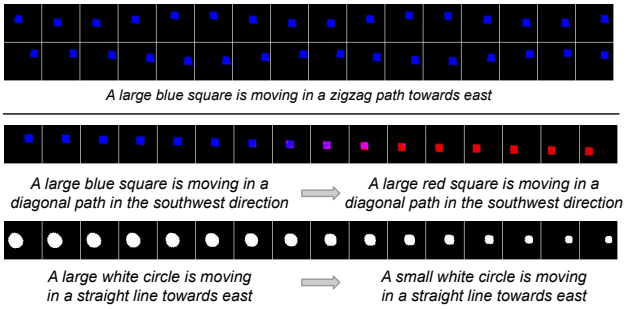

Figure 4: Exploratory experiments on *Shapes-v1* dataset. The image on the top shows the long sequence experiment where we generate 32-length sequence from a model trained for 16 frames. The top row of this video are the first 16 frames and the bottom row corresponds to the next 16. The images on the bottom illustrate the interpolation experiments where we generate a video corresponding to a smooth transition between two input sentences.

a smooth decrease in object size is observed as the object moves in the specified trajectory.

Generating Novel Categories: To determine if the model has learned to generalize and not naively memorize the dataset, this experiment aims to study the ability of TFGAN to produce videos not seen during training. Of the 360 unique parameter configurations in the Shapes dataset, $n = 20$ configurations are held-out from the training set. After the model is trained on this dataset, the attribute classification accuracy in measured on the held-out configurations. The results are reported in Table 1b. Good accuracy on held-out categories are achieved and this illustrates the ability of TFGAN to generalize.

Long Sequence Generation: One of the benefits of using a RNN-based GAN model is that it helps generate video sequences of varying length. To demonstrate this, we perform an experiment where TFGAN model is trained on 16-length video sequences, and made to generate 32-length sequences at test time. Fig. 4 shows the output of one such generated

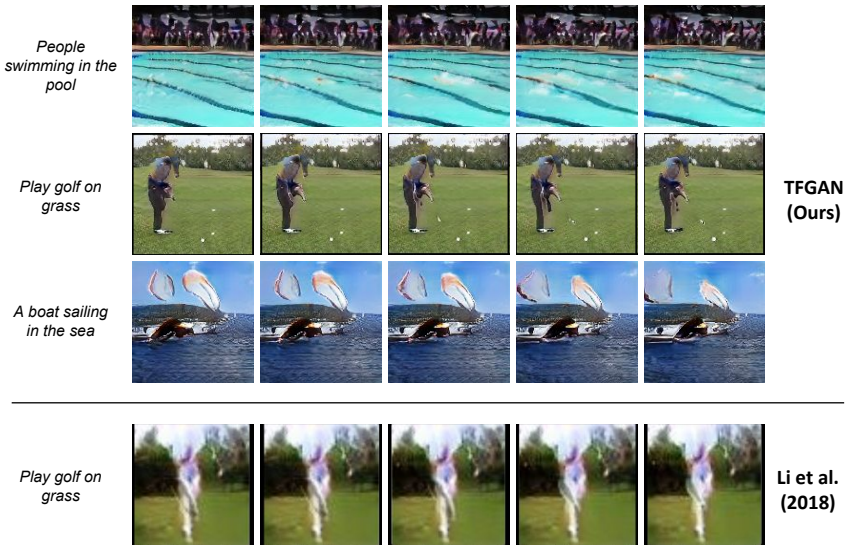

Figure 5: Sample generations by models trained on Kinetics dataset

32-length sequence. We observe that the model is able to clearly perform the zig-zag motion beyond 16 frames, the frame length for which it was trained for.

4.2 Kinetics Dataset

To illustrate the practical relevance of our approach, we perform experiments on real-world video datasets. We use the dataset proposed in [Li *et al.*, 2018] for this purpose. This dataset contains videos of human actions, and was curated from YouTube and Kinetics human action video dataset [Kay *et al.*, 2017]. The following action classes are represented in the dataset - 'biking in snow', 'playing hockey', 'jogging', 'playing soccer ball', 'playing football', 'kite surfing', 'playing golf', 'swimming', 'sailing' and 'water skiing'. This is an extremely challenging dataset for the task of video generation due to the following reasons: (1) videos are extremely diverse with large variations within each video, and (2) many videos have low-resolution and poor-quality video frames.

Method	Acc. (%)	FID-img	FID-vid
In-set	78.1	-	-
T2V([Li *et al.*, 2018])	42.6	82.13	14.65
Ours-SC	74.7	33.51	7.34
Ours-TFGAN	**76.2**	**31.76**	**7.19**

Table 2: Quantitative evaluation on Kinetics dataset

Some qualitative results on the Kinetics dataset are shown in Fig. 5. We observe that TFGAN is able to produce videos of much higher quality than the comparison method [Li *et al.*, 2018]. Fine-grained motions such as golf swing is generated by TFGAN while [Li *et al.*, 2018] produces a blobby region. One reason for improved generation quality is that TFGAN was successfully trained on 128×128 image resolution, while [Li *et al.*, 2018] was trained on 64×64. More qualitative results and generated videos are provided in the supplementary material.

The following metrics are used for quantitative evaluation: (1) Classification accuracy: As done in [Li *et al.*, 2018], a modified version of inception score is computed, in which a video classification model is trained on real data, and the accuracy on generated data is reported. The performance is reported on the following categories as done in [Li *et al.*, 2018]: 'kite surfing', 'playing golf', 'biking in snow', 'sailing', and 'swimming'. (2) Frame-level FID: Frames are extracted from the videos and the *Fréchet Inception Distance* between the real data frames and the generated frames are compute. As originally proposed [Heusel *et al.*, 2017], pre-trained Inception network is used extract the features for computing the score (3) Video-level FID: Features of the penultimate layer are extracted from 3D Resnet-50 model trained on the entire Kinetics dataset [Kay *et al.*, 2017], and the FID score is computed between the real and generated videos. Note that lower the FID scores, better are the models. Quantitative results computed using these metrics are reported in Table. **??**. In-set refers to the performance obtained on the test set of real videos. We observe that TFGAN achieves significantly higher scores than [Li *et al.*, 2018] over all three metrics. In this dataset, the performance gain primarily comes from using stronger generative models as the text descriptions are relatively simple and the degree of text-video variation is limited.

5 Conclusion

In this work, we address the problem of generating videos conditioned on text. A novel conditioning scheme is proposed in which text conditioning is performed using convolution operations acting on discriminator feature maps with filters generated from text. To better understand the text conditioning, we construct a synthetic dataset and show that our conditioning scheme achieves superior performance compared to other techniques. Finally, by using deeper architectures in the discriminator and generator networks, we generate high-quality videos on the challenging Kinetics dataset.

References

[Brock *et al.*, 2018] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale GAN training for high fidelity natural image synthesis. *CoRR*, abs/1809.11096, 2018.

[Damen *et al.*, 2018] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680. 2014.

[Heusel *et al.*, 2017] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems 30*, pages 6626–6637. 2017.

[Jia *et al.*, 2016] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 667–675, 2016.

[Karras *et al.*, 2017] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[Kay *et al.*, 2017] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR 2014*, April 2014.

[Li *et al.*, 2018] Yitong Li, Martin Renqiang Min, Dinghan Shen, David E. Carlson, and Lawrence Carin. Video generation from text. In *AAAI*, 2018.

[Mescheder *et al.*, 2018] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018.

[Mirza and Osindero, 2014] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *ICLR Workshop*, 2014.

[Odena *et al.*, 2017] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier GANs. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2642–2651, 2017.

[Pan *et al.*, 2017] Yingwei Pan, Zhaofan Qiu, Ting Yao, Houqiang Li, and Tao Mei. To create what you tell: Generating videos from captions. 2017.

[Reed *et al.*, 2016] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1060–1069, 2016.

[Salimans *et al.*, 2016] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems 29*, pages 2234–2242. 2016.

[Tulyakov *et al.*, 2018] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Vondrick *et al.*, 2016] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances in Neural Information Processing Systems 29*, pages 613–621. 2016.

[Wang *et al.*, 2018a] Heng Wang, Zengchang Qin, and Tao Wan. Text generation based on generative adversarial nets with latent variables. In *Advances in Knowledge Discovery and Data Mining - 22nd Pacific-Asia Conference, PAKDD 2018, Melbourne, VIC, Australia, June 3-6, 2018, Proceedings, Part II*, 2018.

[Wang *et al.*, 2018b] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. Graphgan: Graph representation learning with generative adversarial nets. In *AAAI*, 2018.

[Xu *et al.*, 2018] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[Yan *et al.*, 2016] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, 2016.

[Zhang *et al.*, 2017] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017.

[Zhu *et al.*, 2017] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

Supplementary material
Conditional GAN with Discriminative Filter Generation for Text-to-Video Synthesis

Yogesh Balaji[1,2], **Martin Renqiang Min**[1], **Bing Bai**[1] and **Rama Chellappa**[2]
Hans Peter Graf[1]
[1]NEC Labs America - Princeton
[2]University of Maryland, College Park

1 Training Algorithm

The equations for the optimization updates of TFGAN model are written below. For brevity, we write the equations only for the frame-level discriminator $\mathbf{D}^{(f)}$. Similar equations have to be repeated for $\mathbf{D}^{(v)}$.

$$\min_{\mathbf{G}} \max_{\mathbf{D}^{(f)}} \quad L_{real} + L_{fake}, \tag{1}$$

$$L_{real} = \mathbb{E}_{(\mathbf{v},\mathbf{t})\sim p_{real}} \Big[\log(\mathbf{D}^{(f)}(\mathbf{v}, \mathbf{T}(\mathbf{t}))) + \frac{\gamma}{2} \|\nabla \mathbf{D}^{(f)}(\mathbf{v}, \mathbf{T}(\mathbf{t}))\|^2 \Big], \tag{2}$$

$$L_{fake} = \frac{1}{2} \Big[\mathbb{E}_{(\mathbf{v},\mathbf{t})\sim p_{fake}} \log(1 - \mathbf{D}^{(f)}(\mathbf{v}, \mathbf{T}(\mathbf{t}))) + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}} \log(1 - \mathbf{D}^{(f)}(\mathbf{G}(\mathbf{z}, \mathbf{T}(\mathbf{t})), \mathbf{T}(\mathbf{t}))) \Big]. \tag{3}$$

The text encoder $\mathbf{T}$ is optimized as follows

$$\max_{\mathbf{T}} L_T = \mathbb{E}_{(\mathbf{v},\mathbf{t})\sim p_{real}} \log(\mathbf{D}^{(f)}(\mathbf{v}, \mathbf{T}(\mathbf{t}))) + \mathbb{E}_{(\mathbf{v},\mathbf{t})\sim p_{fake}} \log(1 - \mathbf{D}^{(f)}(\mathbf{v}, \mathbf{T}(\mathbf{t}))). \tag{4}$$

In the above set of equations, p_{real} denotes the real data distribution with correct video-text correspondences, whereas p_{fake} refers to the distribution with incorrect video-text correspondences. The incorrect correspondences are generated by randomly shuffling the video-text associations. Eq (1) through (3) are optimized by alternating between the minimization and maximization steps as done in standard GAN training.

From the above set of equations, we observe that the objective function is written in terms of the following three loss terms: L_{real} - loss for real samples, L_{fake} - loss for generated samples, and L_T - loss for text encoder. Denoting $L^{(f)}_{real}, L^{(f)}_{fake}, L^{(f)}_T$ as the losses for frame-level discriminator $\mathbf{D}^{(f)}$ and $L^{v}_{real}, L^{v}_{fake}, L^{v}_T$ as the losses for video-level discriminator $\mathbf{D}^{(v)}$, the training algorithm of TFGAN is mentioned in Alg. 1

Algorithm 1 Training algorithm

Require: θ: Parameters of $\mathbf{G}$, $\phi^{(f)}$: Parameters of D_F, $\phi^{(v)}$: Parameters of D_V, $\phi^{(t)}$: Parameters of $\mathbf{T}$
Require: N_{iter}: number of training iterations
Require: α: Learning rate, N_b: batch size
1: **for** t in $1 : N_{iter}$ **do**
2: Sample N_b real samples with correct video-text correspondence $\{(\mathbf{v}^{real}_i, \mathbf{t}^{real}_i)\}_{i=1}^{N_b}$
3: Sample N_b real samples with incorrect video-text correspondence $\{(\mathbf{v}^{fake}_i, \mathbf{t}^{fake}_i)\}_{i=1}^{N_b}$
4: Update $\mathbf{D}^{(f)}$: $\phi^{(f)} = \phi^{(f)} + \alpha\nabla\big[L^{(f)}_{real} + L^{(f)}_{fake}\big]$
5: Update $\mathbf{D}^{(v)}$: $\phi^{(v)} = \phi^{(v)} + \alpha\nabla\big[L^{(v)}_{real} + L^{(v)}_{fake}\big]$
6: Update $\mathbf{G}$: $\theta = \theta - \alpha\nabla\big[L^{(f)}_{real} + L^{(f)}_{fake} + L^{(v)}_{real} + L^{(v)}_{fake}\big]$
7: Update $\mathbf{T}$: $\phi^{(t)} = \phi^{(t)} + \alpha\nabla\big[L^{(f)}_{T} + L^{(f)}_{T} + L^{(v)}_{T} + L^{(v)}_{T}\big]$
8: **end for**

2 Dataset generation - Shapes Dataset

In this section, we describe in detail how *Shapes* dataset was created. Each sample in the *Shapes* dataset has five control parameters: shape type, size, color, motion type and motion direction. Table 1 lists the possible values each parameter can take. The (shape type, color, size) tuple describes the structure of the shape, while (motion type, direction) tuple dictates how the shape moves in the video. For straight line and zig-zag motion, the shape could move in north, south, west and east direction, while for diagonal motion, the possible directions include north-west, north-east, south-west and south-east. The zig-zag motion was generated using a sinusoidal function. To generate a video, we randomly sample the control parameters, generate a random trajectory for the shape according to the control parameters, and overlay the shape according to the generated trajectory. The text description corresponding to the parameter configuration is then generated. We use a fixed template for generating the text.

In *Shapes-v1* dataset, a moving shape described by the parameter configuration is generated and is overlaid in a black background. While this assumption of static background helps study the problem of text-to-video generation, it is hardly true in practice as many videos have dynamic backgrounds. So, we create a second dataset called *Shapes-v2 dataset* which is a version of Moving Shapes dataset with dynamic backgrounds. To generate the background, we choose

texture images from Colored Broadatz [Abdelmounaime and Dong-Chen, 2013] dataset and sample a sequence of patches corresponding to a random trajectory. Each patch in this sequence forms the background of an individual frame in the video. These background textures are blended with the moving shape resulting in videos as shown in Fig. 2. This dataset is much more challenging than the *Shapes-v1 dataset* as the generative models should learn to ground the text description to the moving object but not to the randomly moving background. We would like to point out that using synthetic datasets have been explored in the previous works [Pan *et al.*, 2017], however, we create a much more comprehensive dataset with fine-grained text-video variation for the task of text-to-video generation.

From Table 1, we observe that there are 360 different parameter combinations for *Shapes-v1* and *Shapes-v2* dataset. Samples from the dataset are shown in Fig. 2. All *Shapes* datasets were created with videos containing 16 frames at 64×64 frame resolution.

3 Multiple shapes dataset

The task of text-to-video synthesis becomes much more challenging when there are multiple shapes in the scene. So, we created a third dataset called *Shapes-v3* dataset with two moving shapes. The text annotation in this dataset contains the description of the two moving shapes, and so, the model should learn correct associations between the individual shapes and their respective description. This makes this dataset extremely challenging. From Table 1, we observe that there are $360 \times 360 = 129600$ parameter configurations in this dataset. Some samples from this dataset are visualized in Fig 2

It is hard to perform quantitative evaluation in this dataset due to the presence of two shapes in each video. *Attribute classification accuracy* fails as it does not associate the individual shapes with their respective attributes. For an input text containing blue square and red circle in its description, checking for the existence of (square, circle) and (blue, red) attributes in the generated video is insufficient as the video might contain blue circle and red square. So, we show qualitative results on this dataset.

Samples generated from TFGAN and the simple concatenation scheme on this dataset are shown in Fig 5. We observe that the simple concatenation baseline model fails to generate two shapes. Since the text description is complex, just performing simple concatenation of video and text features fails to condition well in this dataset. This leads to extremely poor quality generations. TFGAN, on the other hand, produces two shapes, each generated according to the given text description. This dataset clearly illustrates how our proposed scheme can effectively model text-video associations in datasets where there are rich and complex text-video variations.

4 Ablation studies

4.1 Effect of filtering at different levels of abstraction

As shown in Figure (2) of main paper, filters extracted from text are convolved at different levels of abstraction in the discriminator network in TFGAN model. In all our experiments, text-filters are applied at two stages – the output of Resnet-1 block (which we call as Stage 1) and the output of Resnet-3 block (which we call as Stage 2). Please refer to Section 5 for more details. The effect of removing one stage of filtering is shown in this experiment. The following settings are considered – applying no text-filters (this corresponds to Simple Concat (SC) scheme discussed in the main paper), applying text-filters only at Stage 1, applying text-filters only at stage 2, and applying text-filters at both stages (this corresponds to TFGAN model). Experiments are performed on *Shapes-v2* dataset, and the results are shown in Table. 2. We observe that applying filters only at single stage gives sub-optimal performance than applying at both the stages. Among S_1 and S_2, applying filters at S_2 is more effective than S_1. However, using the combination of $S_1 + S_2$ (which is the TFGAN model) gives the best performance.

4.2 Effectiveness of text-filter conditioning in different architectures

In the previous experiments, effectiveness of text-filter conditioning on Resnet-based architectures are demonstrated. However, the proposed conditioning scheme is general and can be applied to any generator-discriminator architecture. A natural question to ask is if the text-filter conditioning is effective for other architectures. To investigate this, we use DCGAN architectures for the generator and discriminator models as used in MoCoGAN [Tulyakov *et al.*, 2018]. The text-filters are applied at the outputs of $conv_1$ and $conv_3$ layers in the discriminator network. The results of this experiment on *Shapes-v1* dataset are reported in Table 3. We observe that simple concatenation scheme results in poor conditioning performance. Text-filter conditioning, on the other hand, significantly improves the conditioning. Due to the limitation of the architecture, the performance is not as high as that using Resnet architectures reported in main paper. Nevertheless, even with such poor models, using text-filter conditioning helps improve the performance.

4.3 On different conditioning choices

Some recenly explored choices for class-conditioning in generative models include class-conditional batch normalization [Dumoulin *et al.*, 2017][de Vries *et al.*, 2017] and Projection discriminator [Miyato and Koyama, 2018]. While these schemes are originally used for class-conditioning[Miyato *et al.*, 2018], they can easily be extended to perform text-conditioning by replacing the one-hot encoding of class labels with the output of the text encoder. For experiments using conditional Batchnorm, all batch normalization layers in the generator and the discriminator models are replaced with text-conditional Batchnorm. Comparison of these conditioning schemes with text-filter conditioning on *Shapes-v1* dataset using DCGAN architecture is reported in Table 4. We

Table 1: Simulation parameters

Atribute	Set of values
Shape	{ Square, Circle, Triangle }
Color	{ Red, Blue, Green, White, Yellow }
Size	{ Small, Large }
Motion type	{ Straight Line, Diagonal, Zig-zag }
Direction	{ North, South, East, West } for st.line and zig-zag motion
	{ North-east, North-west, South-east, South-west } for diagonal motion

Table 2: Filtering at different levels of abstraction – Attribute classification accuracy (in %) on *Shapes-v2* dataset

Method	Shape	Color	Size	Motion	Direction	Average
No text-filters (SC)	55.88	96.04	73.71	83.30	93.51	80.49
Text-filters on stage 1	61.21	99.97	75.41	99.90	100.00	87.29
Text-filters on stage 2	67.25	99.98	78.55	100.00	100.00	89.15
Text-filters on stage 1 + stage 2 (TFGAN)	81.10	99.99	89.82	99.80	100.00	94.14

observe that both projection discriminator and conditional batch-normalization performs poorly compared to text-filter conditioning. This demonstrates the effectiveness of the proposed conditioning scheme for improving text conditioning in videos.

4.4 On different design choices

To train ResNet-style architectures, we use R_1 regularization that penalizes the norm of discriminator gradients ($\|\nabla \mathbf{D}(\mathbf{v}, \mathbf{T}(\mathbf{t}))\|^2$ term in Eq 2) as proposed in [Mescheder *et al.*, 2018]. Spectral normalization [Miyato *et al.*, 2018] is another class of stabilizing technique commonly used in GAN models. With respect to the loss function, [Miyato *et al.*, 2018] used Hinge loss in their objective, whereas a standard log-likelihood based GAN loss is used in our formulation. We investigate a combination of these design choices in this experiment. Table 5 reports the performance of these design choices using Resnet achitecture on *Shapes-v2* dataset. We noticed that the spectral normalization by itself did not stabilize our training. After 70000 iterations, generator loss diverged and models collapsed to produce plain black images. However, using R_1 regularization stabilized the models better. Using spectral normalization in addition to R_1 regularization did not improve over using R_1 regularization alone. So, we used R_1 regularization in all our experiments. Also, using hinge loss gave similar performance as the standard GAN loss.

4.5 Importance of regularization

In this experiment, we study the importance of using regularization to stabilize the training. Fig 1 shows the plots of generator and discriminator losses for (1) TFGAN model with Resnet architecture trained on *Shapes-v1* dataset without using any regularization, (2) TFGAN model with Resnet architecture trained on *Shapes-v1* dataset using R_1 regularization. We observe that without regularization, the generator loss keeps increasing over iterations, and generated images collapse beyond a certain point. However, models trained with R_1 regulazation exhibit much more stable behaviour in losses. In our experiments, we obtained higher values of attribute classification accuracy when the models were trained longer. Models on *Shapes-v1* dataset were trained for $150k$ iterations, and *Shapes-v2* and *v3* dataset were trained for $300k$ iterations. So, it was crucial to use R_1 regularization to stabilize these models. Using R_1 regularizer was much more crucial in *Kinetics* and *Cooking* datasets as models did not produce any meaningful generations without the regulaizer.

5 Architecture and Hyper-parameters

The basic resnet block on which generator and discriminator architectures are built upon are shown in Figure. 6.

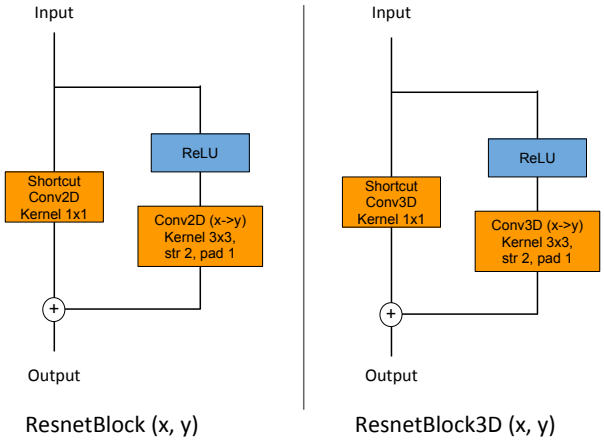

Figure 6: Basic resnet blocks. Shortcut is a shortcut connection with 1×1 convolution used to adjust the number of filters when there is a mismatch between the number of input filters x and output filters y. When $x = y$, Shortcut is an identity map

5.1 Text-Filter conditioning

Discriminator network architectures for *Shapes* and *Kinetics* experiments are given in Tables 7, 8, 10, 11. In all our ex-

Table 3: Attribute classification accuracy (in %) on *Shapes-v1* dataset using DCGAN models

Method	Shape	Color	Size	Motion	Direction	Average
SC (MoCoGAN)	70.45	92.84	82.06	41.70	26.40	62.69
Text-filter conditioning	90.60	96.93	96.55	78.10	71.30	86.70

Table 4: Different conditioning schemes – Attribute classification accuracy (in %) on *Shapes-v1* dataset

Method	Shape	Color	Size	Motion	Direction	Average
Simple concat (SC)	70.45	92.84	82.06	41.70	26.40	62.69
Projection Discriminator	82.53	93.50	83.45	70.10	44.51	74.82
Class-conditional Batchnorm + SC	80.65	80.55	84.76	60.90	39.40	69.25
Text-filter conditioning	90.60	96.93	96.55	78.10	71.30	86.70

periments, the discriminator networks (both $\mathbf{D}^{(v)}$ and $\mathbf{D}^{(f)}$) were divided into three sub-networks ($m = 3$). In *Shapes* experiments, the sub-network $\mathbf{D}_1^{(f)}$ is the section of the network $\mathbf{D}^{(f)}$ till the end of Resnet-1 block, $\mathbf{D}_2^{(f)}$ is the sub-network from Resnet-2 block till the end of Resnet-3 block and $\mathbf{D}_3^{(f)}$ forms the rest of the network. For the Kinetics experiment, the $\mathbf{D}_1^{(f)}$ contains the section of the network $\mathbf{D}^{(f)}$ till Renset-11 block, $D_2^{(f)}$ spans from Renset-20 to Resnet-31 block, and $\mathbf{D}_3^{(f)}$ forms the rest of the network. For DCGAN based experiments, $\mathbf{D}_1^{(f)}$ block is the first *conv* layer, $\mathbf{D}_2^{(f)}$ block spans the next two *conv* layers, and $\mathbf{D}_3^{(f)}$ block contains the last *conv* layer. The same partitioning scheme is used for the video discriminator network $\mathbf{D}^{(v)}$ as well.

Based on the partioning scheme discussed above, the discriminator network is divided into three sub-networks. We apply text-filter conditioning only on the responses of the first two sub-networks. Let us call the response of the first sub-network $\mathbf{D}_1^{(f)}$ and $\mathbf{D}_1^{(v)}$ as stage-1, and the response of the second sub-network $\mathbf{D}_2^{(f)}$ and $\mathbf{D}_2^{(v)}$ as stage-2. An ablation study for the effect of removing conditioning on each of these stages is discussed in Section 4.1. Since the responses of stage-1 and stage-2 contain a large number of channels, the text-filters $\mathbf{f}_i$ to be convolved on them will have to contain the same number of input channels which is computationally expensive. So, the responses of stage-1 and stage-2 are first passed to a bottleneck connection (with 1×1 conv layers) to bring down the number of channels to 32. So, if the output of $\mathbf{D}_1^{(f)}$ was a $b \times n \times k \times w$ map, this transformation will bring it down to $b \times 32 \times k \times w$.

The response of the text encoder $\mathbf{T}$ (architecture is given in Section 5.2) are two vectors $\mathbf{t}^{(f)}$ and $\mathbf{t}^{(v)}$, which are 256 dimensional each. To form the filters $\mathbf{f}^{(f)}$ for frame-discriminator $\mathbf{D}^{(f)}$, we use the text encoding $\mathbf{t}^{(f)}$, and to form the filters $\mathbf{f}^{(v)}$ for frame-discriminator $\mathbf{D}^{(v)}$, we use the text encoding $\mathbf{t}^{(v)}$. The filter generator networks $\Phi_i^{(f)}$ is a fully connected layer $FC(256 \rightarrow 32.32.5.5)$, whereas $\Phi_i^{(v)}$ is a fully connected layer $FC(256 \rightarrow 32.32.3.5.5)$. Hence, the dimensions of $\mathbf{f}_i^{(f)}$ are $32 \times 32 \times 5 \times 5$, and $\mathbf{f}_i^{(v)}$ are $32 \times 32 \times 3 \times 5 \times 5$

The outputs of stage-1 and stage-2 can be convolved with $\mathbf{f}_i^{(f)}$ and $\mathbf{f}_i^{(v)}$ as the number of channels match with the filter dimension. A stride of 1 and padding 2 is used in this convolution so that the output size match the input size. The response of this convolution are then passed onto $\tilde{\mathbf{D}}_i^{(f)}$ and $\tilde{\mathbf{D}}_i^{(v)}$ as mentioned in Figure 2 of the main paper. The following architectures are used for $\tilde{\mathbf{D}}$ networks

- $\tilde{\mathbf{D}}_1^{(f)}$: AvgPool($4 \times 4$) $\rightarrow$ Conv2D($32 \rightarrow 32$, kernel 3×3)+ReLU $\rightarrow$ AvgPool(4×4) $\rightarrow$ Conv2D($32 \rightarrow 128$, kernel 2×2)
- $\tilde{\mathbf{D}}_2^{(f)}$: AvgPool($2 \times 2$) $\rightarrow$ Conv2D($32 \rightarrow 32$, kernel 3×3)+ReLU $\rightarrow$ AvgPool(2×2) $\rightarrow$ Conv2D($32 \rightarrow 128$, kernel 2×2)
- $\tilde{\mathbf{D}}_1^{(v)}$: AvgPool($2 \times 4 \times 4$) $\rightarrow$ Conv3D($32 \rightarrow 32$, kernel $3 \times 3 \times 3$)+ReLU $\rightarrow$ AvgPool($4 \times 4 \times 4$) $\rightarrow$ Conv3D($32 \rightarrow 128$, kernel $1 \times 2 \times 2$)
- $\tilde{\mathbf{D}}_2^{(v)}$: AvgPool($2 \times 2 \times 2$) $\rightarrow$ Conv3D($32 \rightarrow 32$, kernel $3 \times 3 \times 3$)+ReLU $\rightarrow$ AvgPool($2 \times 2 \times 4$) $\rightarrow$ Conv3D($32 \rightarrow 128$, kernel $1 \times 2 \times 2$)

The outputs of $\tilde{\mathbf{D}}_1^{(f)}$ and $\tilde{\mathbf{D}}_2^{(f)}$ are 128 dimensional vectors. Similarly, the output of the third stage of the discriminator netowork $\mathbf{D}_3^{(f)}$ (on which text-conditioning was not applied) is also n dimensional vector. This vector is passed to a FC layer to make it 128 dimensional and is concatenated with the responses of $\tilde{\mathbf{D}}_1^{(f)}$ and $\tilde{\mathbf{D}}_2^{(f)}$. This concatenated representation then classifies if the input frames are real/fake. Similar computation is performed for video-level representation as well.

5.2 Text encoder

For the text encoder, we first obtained the GloVE [Pennington *et al.*, 2014] embeddings of individual words (which are 300-dimensional vectors each), then applied a 1D-CNN based network with the following network architecture:

Conv1D($300 \rightarrow 512$, kernel=3) $\rightarrow$ ReLU $\rightarrow$ MaxPool(2) $\rightarrow$ Conv1D($512 \rightarrow 512$, kernel=3) $\rightarrow$ ReLU $\rightarrow$ MaxPool(2) $\rightarrow$ Conv1D($512 \rightarrow 256$)

There are two-copies of the above architecture - one giving frame-level representation $\mathbf{t}^{(f)}$, and other giving video level representation $\mathbf{t}^{(v)}$. Both these vectors are 256 dimensional. 1D-CNN was sufficient in all our experiments, and

Table 5: Different design choices – Attribute classification accuracy (in %) on *Shapes-v2* dataset

Method	Shape	Color	Size	Motion	Direction	Average
Spectral normalization + hinge loss	Did not converge					
Spectral normalization + standard loss	Did not converge					
R_1 regularization + hinge loss	75.50	99.90	86.31	100.00	100.00	92.34
R_1 regularization + standard loss (TFGAN)	81.10	99.99	89.92	99.80	100.00	94.16
Spectral normalization + hinge loss + R_1 regularization	78.32	99.61	88.90	99.61	100.00	93.29
Spectral normalization + standard loss + R_1 regularization	82.10	99.81	89.98	99.60	99.71	94.24

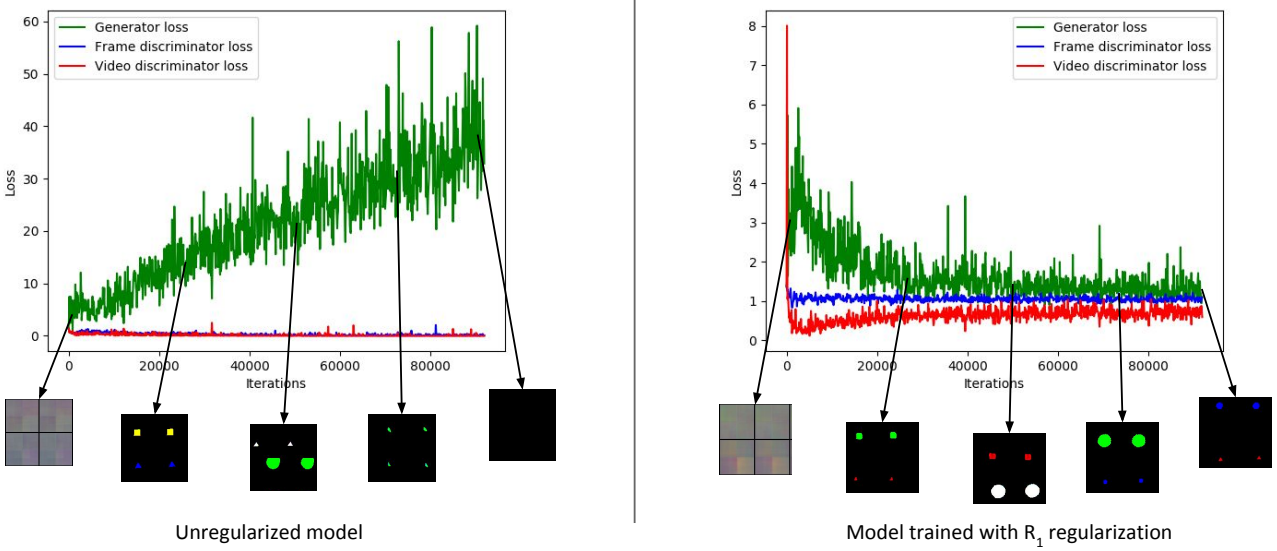

Figure 1: Plots of generator and discriminator losses for unreglarized and regularized models

using RNN-based models did not give any improvement over 1D-CNN network.

5.3 GAN architectures

The architectures used for the discriminator and generator networks for *Shapes* experiments are mentioned in in Tables. 6, 7 and 8. The architectures used for Kinetics and Cooking experiments are given in Tables 9, 11 and 10. Hyperparameter details used in all our experiments are given in Table. 12.

6 Additional results

6.1 Kinetics dataset

Some additional results on the *Kinetics* dataset are shown in Fig 7. We observe that we are able to generate videos of high quality. To illustrate the variations that occur within a class, we generate multiple videos of the same text description. Figure 8 shows one such example for swimming class. We find that out model is capable of generating diverse predictions. To better visualize the temporal motion, we also provide ".gif" files accompanying the generated videos. However, the quality of videos are affected due to gif artifacts, and we suggest the reader to use the images provided in this paper to assess the quality of the generations.

6.2 Epic-Kitchens Dataset

In this experiment, we consider Epic-Kitchens dataset [Kay *et al.*, 2017] which is a ego-centric dataset containing videos of people cooking in a kitchen. The videos were recorded using a high-definition head mounted camera. The dataset is annotated with text descriptions of step-by-step instructions of a cooking recipe with the corresponding timestamp as people perform the action. This dataset is extremely challenging for the following reasons: (1) Since this is a head-mounted dataset, there are large variations in the background, and rapid motion can occur between frames. (2) There is high object clutter as the dataset is collected in indoor kitchen environment. From this dataset, we extracted clips of 16-lenght video sequences containing the following action classes: 'take', 'cut', 'dicing', 'pour', 'stir', 'wash', 'grate', 'rinse', 'put'. This resulted in 4793 (video, text) pairs. The qualitative results of training TFGAN on this dataset are shown in Fig. 9. Despite the inherent challenges of this dataset and small size of the training set, TFGAN generates good videos.

6.3 Text to Image Generation

Text-to-image generation is a relatively easier problem than text-to-video generation due to the absence of temporal constraints. Although the focus of this paper is on text-to-video

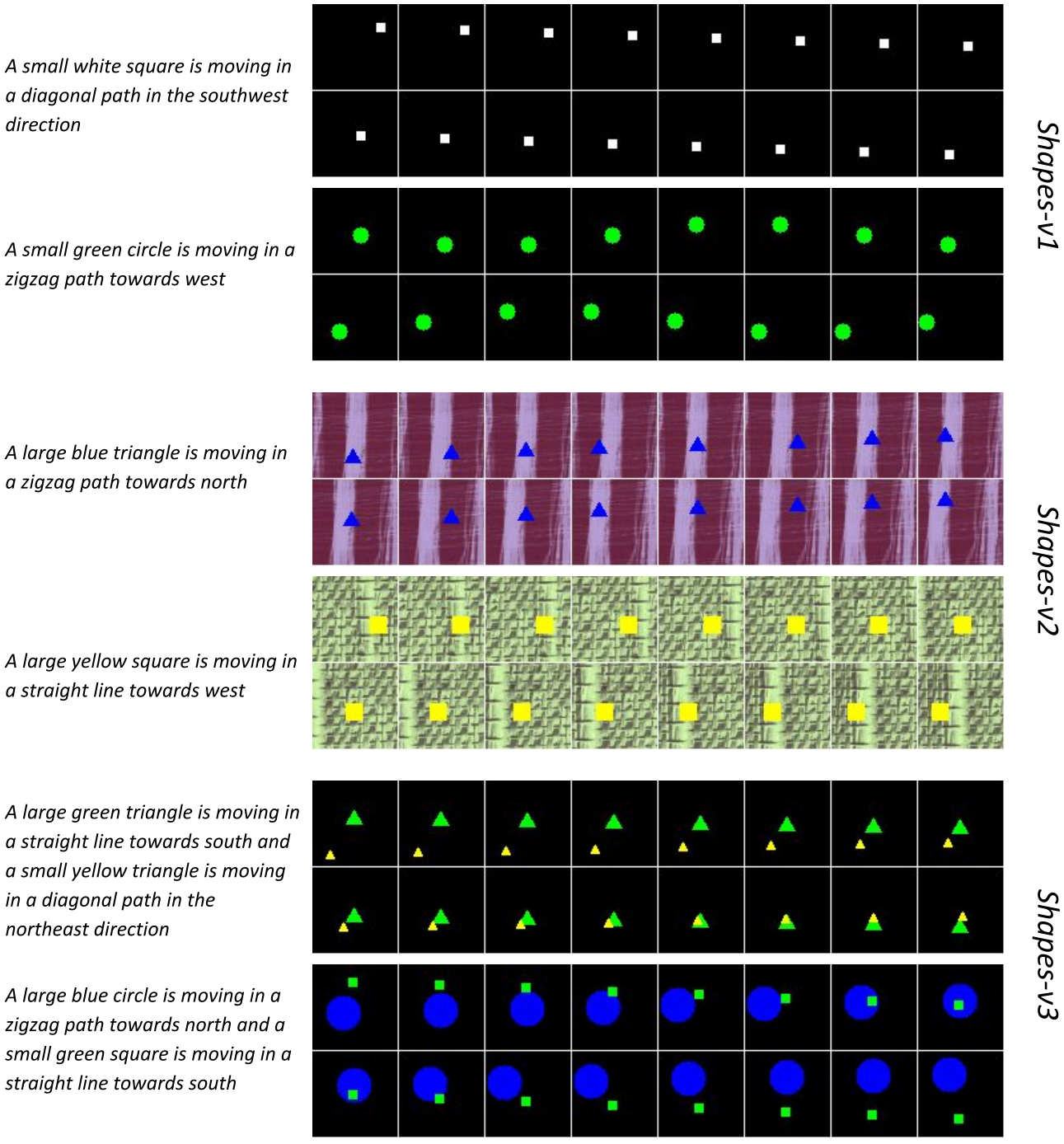

Figure 2: Real samples from Shapes datasets

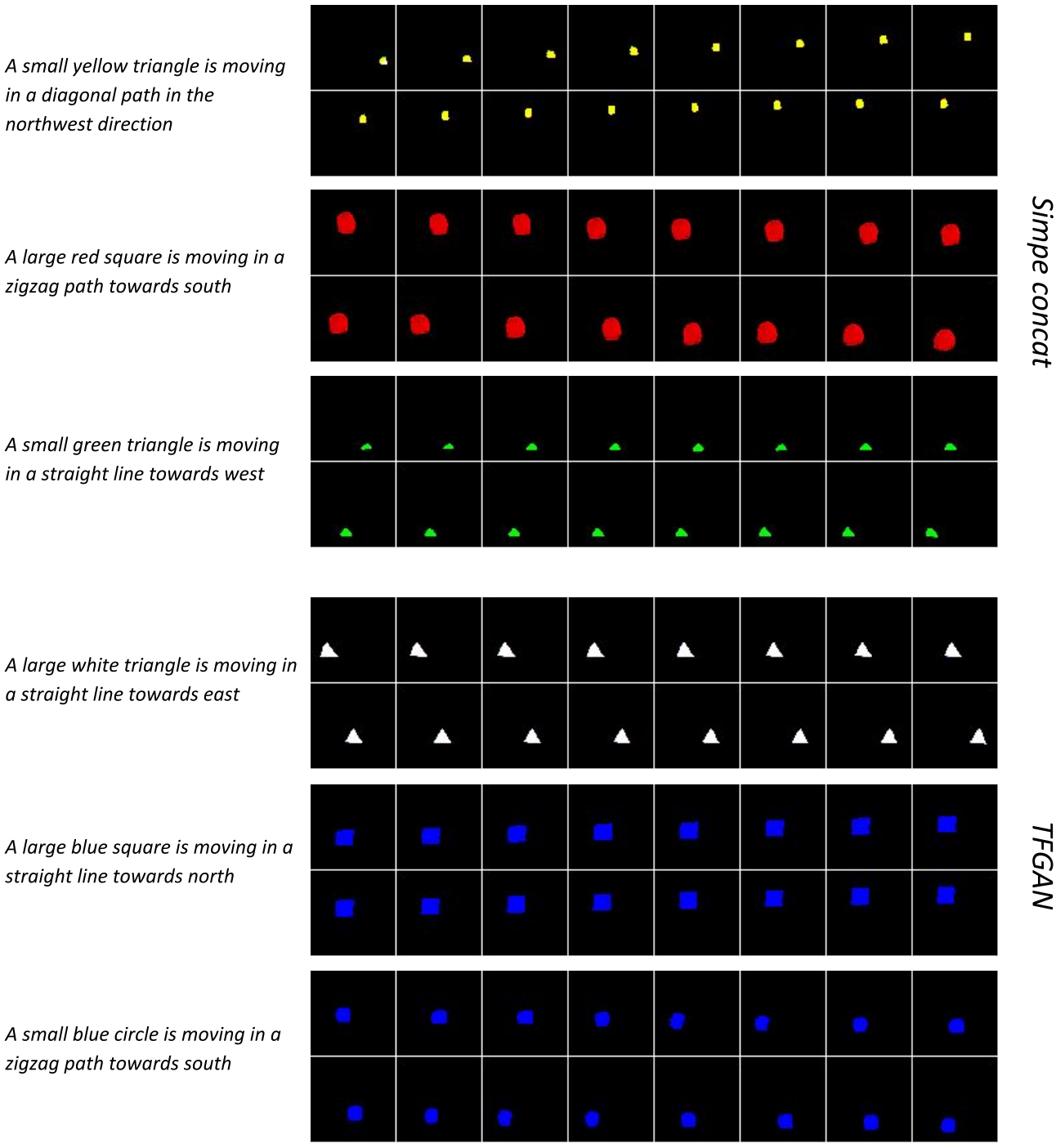

Figure 3: Samples generated by TFGAN and Simple concatenation scheme on *Shapes-v1* dataset

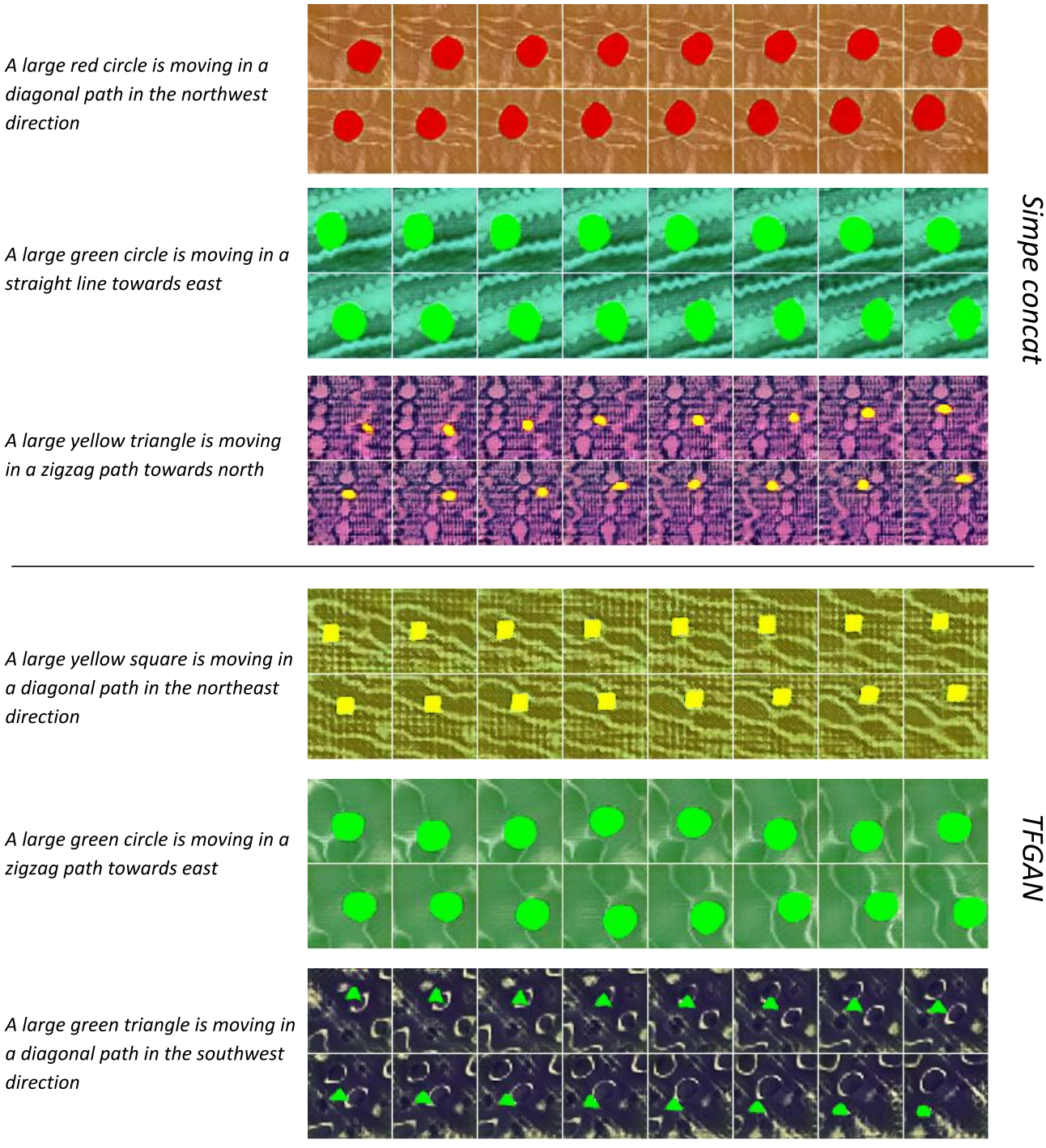

Figure 4: Samples generated by TFGAN and Simple concatenation scheme on *Shapes-v2* dataset

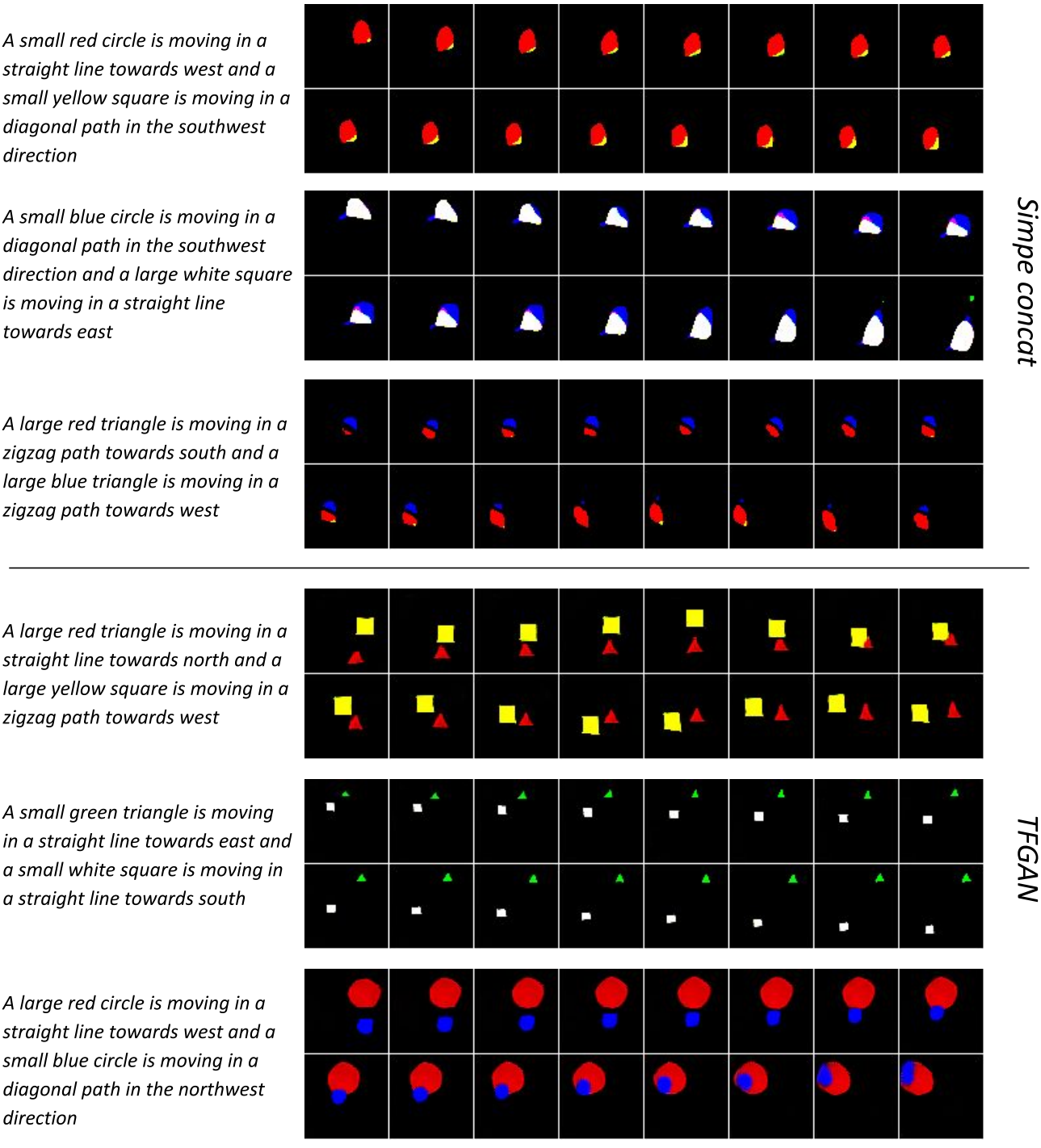

Figure 5: Samples generated by TFGAN and Simple concatenation scheme on *Shapes-v3* dataset

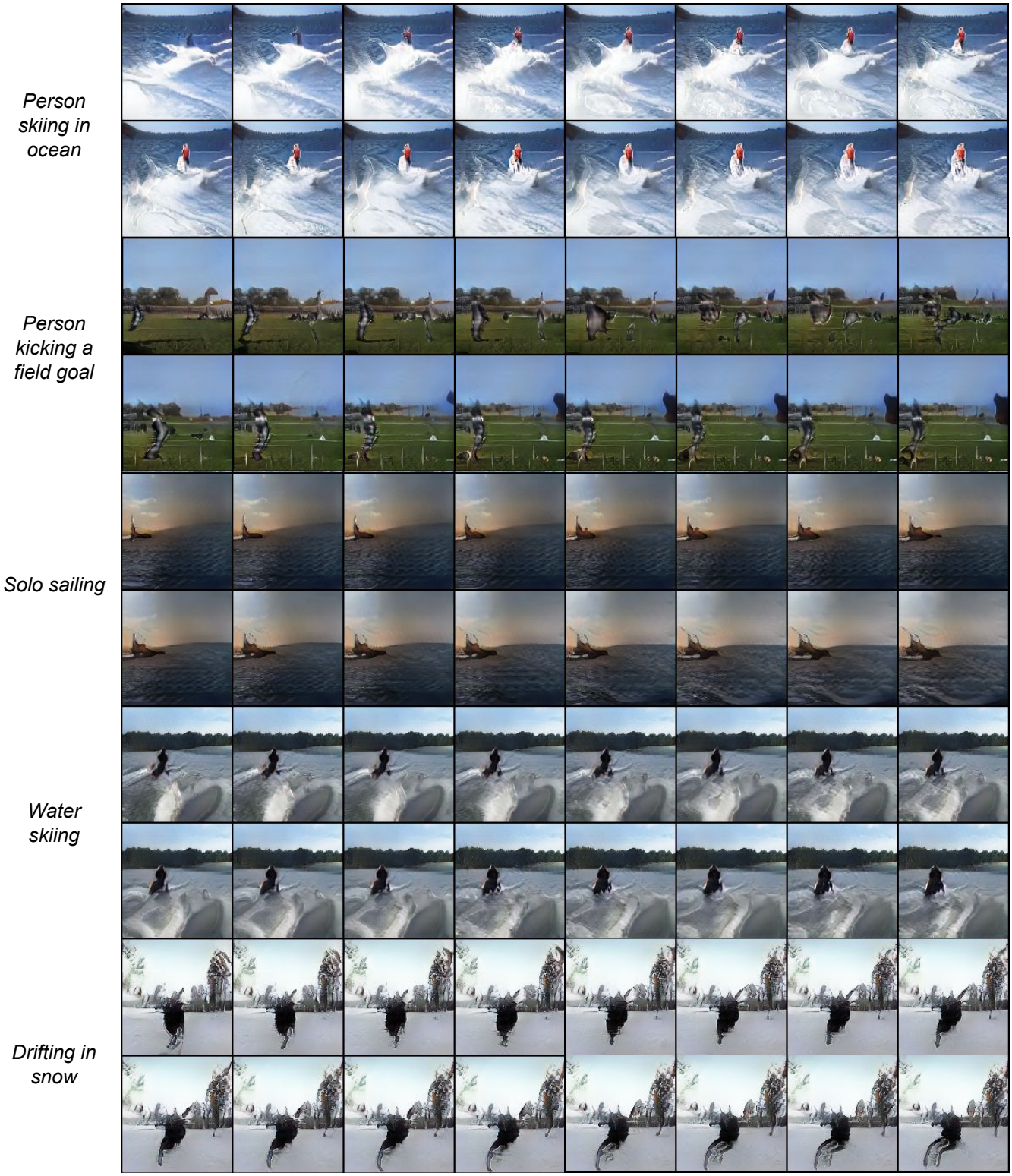

Figure 7: Samples generated by TFGAN model on *Kinetics* dataset

Figure 8: Variations wihin a category - Swimming class produced by TFGAN model

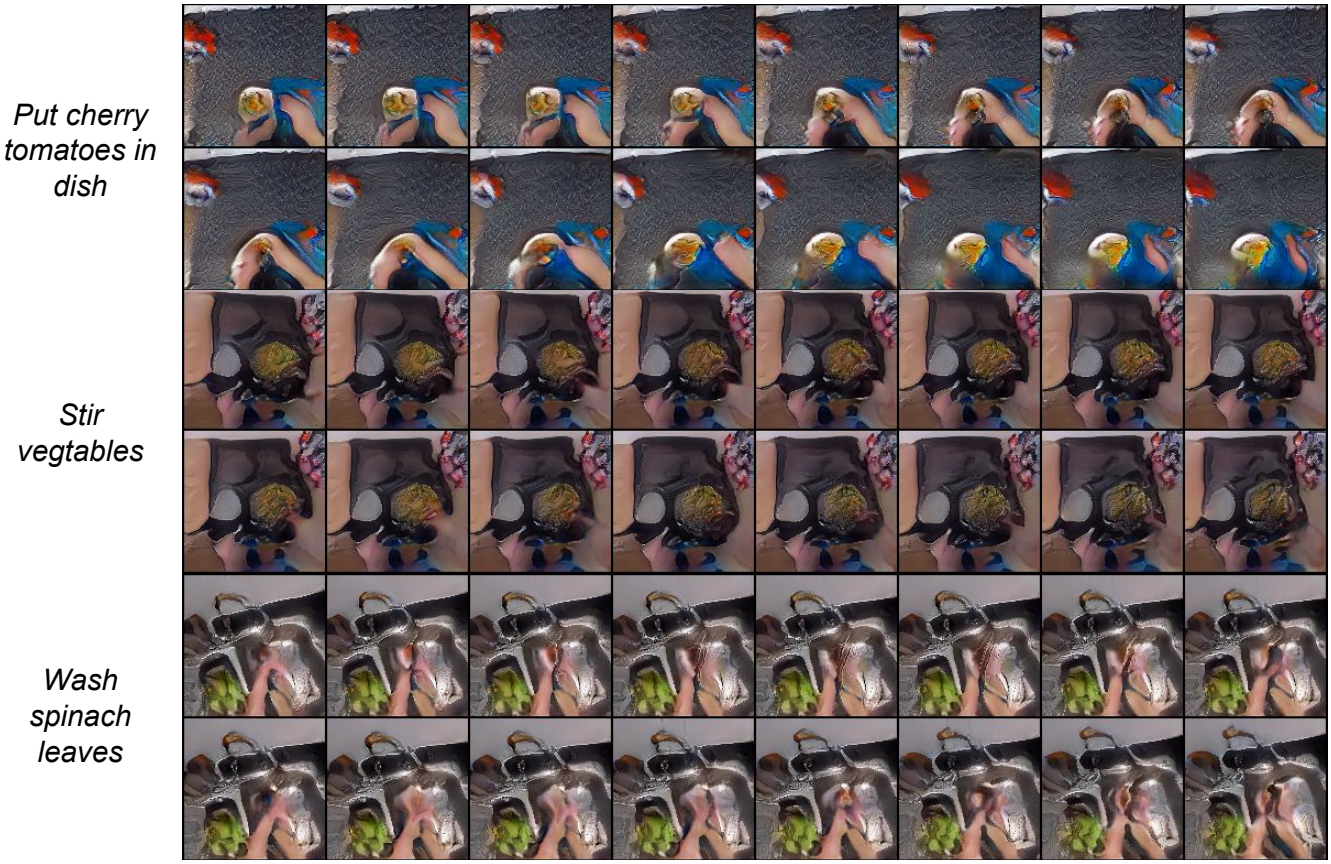

Figure 9: Samples generated by TFGAN model on *Epic-Kitchens* dataset

Table 6: Generator architecture - Shapes dataset

Layer	Output size	Filter
Input	768	-
Fully connected	1024	$768 \rightarrow 1024$
Reshape	$256 \times 2 \times 2$	-
ResnetBlock-0	$256 \times 2 \times 2$	$256 \rightarrow 256$
Upsample	$256 \times 4 \times 4$	-
ResnetBlock-1	$128 \times 4 \times 4$	$256 \rightarrow 128$
Upsample	$128 \times 8 \times 8$	-
ResnetBlock-2	$128 \times 8 \times 8$	$128 \rightarrow 128$
Upsample	$128 \times 16 \times 16$	-
ResnetBlock-3	$128 \times 16 \times 16$	$128 \rightarrow 128$
Upsample	$128 \times 32 \times 32$	-
ResnetBlock-4	$64 \times 32 \times 32$	$128 \rightarrow 64$
Upsample	$64 \times 64 \times 64$	-
ResnetBlock-5	$32 \times 64 \times 64$	$64 \rightarrow 32$
Conv2D	$3 \times 64 \times 64$	$32 \rightarrow 3$
Tanh	$3 \times 64 \times 64$	-

Table 7: Frame Discriminator architecture - Shapes dataset

Layer	Output size	Filter
Input	$3 \times 64 \times 64$	-
Conv2D	$32 \times 64 \times 64$	$3 \rightarrow 32$
ResnetBlock-0	$64 \times 64 \times 64$	$32 \rightarrow 64$
AvgPool	$64 \times 32 \times 32$	-
ResnetBlock-1	$128 \times 32 \times 32$	$64 \rightarrow 128$
AvgPool	$128 \times 16 \times 16$	-
ResnetBlock-2	$128 \times 16 \times 16$	$128 \rightarrow 128$
AvgPool	$128 \times 8 \times 8$	-
ResnetBlock-3	$128 \times 8 \times 8$	$128 \rightarrow 128$
AvgPool	$128 \times 4 \times 4$	-
ResnetBlock-4	$256 \times 4 \times 4$	$128 \rightarrow 256$
AvgPool	$256 \times 2 \times 2$	-
ResnetBlock-5	$256 \times 2 \times 2$	$256 \rightarrow 256$
Reshape	256.2.2	-
Text-Img feat concat	n_{fused}	-
Fully connected	$n_{fused} \rightarrow 1$	$n_{fused} \rightarrow 1$

Table 8: Video Discriminator architecture - Shapes dataset

Layer	Output size	Filter
Input	$3 \times 16 \times 64 \times 64$	-
Conv3D	$32 \times 16 \times 64 \times 64$	$3 \rightarrow 32$
ResnetBlock3D-0	$64 \times 16 \times 64 \times 64$	$32 \rightarrow 64$
AvgPool3D	$64 \times 8 \times 32 \times 32$	-
ResnetBlock3D-1	$128 \times 8 \times 32 \times 32$	$64 \rightarrow 128$
AvgPool3D-spatial	$128 \times 8 \times 16 \times 16$	-
ResnetBlock3D-2	$128 \times 8 \times 16 \times 16$	$128 \rightarrow 128$
AvgPool3D	$128 \times 4 \times 8 \times 8$	-
ResnetBlock3D-3	$128 \times 4 \times 8 \times 8$	$128 \rightarrow 128$
AvgPool3D	$128 \times 2 \times 4 \times 4$	-
ResnetBlock3D-4	$256 \times 2 \times 4 \times 4$	$128 \rightarrow 256$
AvgPool3D	$256 \times 1 \times 2 \times 2$	-
ResnetBlock3D-5	$256 \times 1 \times 2 \times 2$	$256 \rightarrow 256$
Reshape	256.2.2	-
Text-Img feat concat	n_{fused}	-
Fully connected	$n_{fused} \rightarrow 1$	$n_{fused} \rightarrow 1$

synthesis, our framework is flexible and can be trivially extended to the problem of text-to-image synthesis. This can be accomplished by removing the video-level discriminator $\mathbf{D}^{(v)}$ and the RNN network in the latent space. We train our GAN model with Text-Filter conditioning on the CUB-Birds dataset [Welinder *et al.*, 2010], a benchmark dataset for text-to-image generation. Some of the samples from the generated images are shown in Figure 10 and 11. We observe that our model is able to produce photo-realistic images. We also report Inception score as a quantitative metric. As can be seen from Table. 13, our method achieves higher inception scores than the comparison methods.

References

[Abdelmounaime and Dong-Chen, 2013] Safia Abdelmounaime and He Dong-Chen. New brodatz-based image databases for grayscale color and multiband texture analysis. In *ISRN Machine Vision*, 2013. 2

[de Vries *et al.*, 2017] Harm de Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C. Courville. Modulating early visual processing by language. *CoRR*, abs/1707.00683, 2017. 2

[Dumoulin *et al.*, 2017] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2017. 2

[Kay *et al.*, 2017] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017. 5

[Mescheder *et al.*, 2018] Lars Mescheder, Sebastian Nowozin, and Andreas Geiger. Which training methods for gans do actually converge? In *International Conference on Machine Learning (ICML)*, 2018. 3

[Miyato and Koyama, 2018] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. *CoRR*, abs/1802.05637, 2018. 2

[Miyato *et al.*, 2018] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. *CoRR*, abs/1802.05957, 2018. 2, 3

[Pan *et al.*, 2017] Yingwei Pan, Zhaofan Qiu, Ting Yao, Houqiang Li, and Tao Mei. To create what you tell: Generating videos from captions. 2017. 2

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. 4

Figure 10: Sample generations from TFGAN model trained on CUB-Birds dataset

Figure 11: Sample generations of Text2img synthesis from TFGAN model trained on CUB-birds dataset

Table 9: Generator architecture - Kinetics dataset

Layer	Output size	Filter
Input	768	-
Fully connected	8192	$768 \rightarrow 8192$
Reshape	$512 \times 4 \times 4$	-
ResnetBlock-00	$512 \times 4 \times 4$	$512 \rightarrow 512$
ResnetBlock-01	$512 \times 4 \times 4$	$512 \rightarrow 512$
Upsample	$512 \times 8 \times 8$	-
ResnetBlock-10	$512 \times 8 \times 8$	$512 \rightarrow 512$
ResnetBlock-11	$512 \times 8 \times 8$	$512 \rightarrow 512$
Upsample	$512 \times 16 \times 16$	-
ResnetBlock-20	$256 \times 16 \times 16$	$512 \rightarrow 256$
ResnetBlock-21	$256 \times 16 \times 16$	$256 \rightarrow 256$
Upsample	$256 \times 32 \times 32$	-
ResnetBlock-30	$128 \times 32 \times 32$	$256 \rightarrow 128$
ResnetBlock-31	$128 \times 32 \times 32$	$128 \rightarrow 128$
Upsample	$128 \times 64 \times 64$	-
ResnetBlock-40	$64 \times 64 \times 64$	$128 \rightarrow 64$
ResnetBlock-41	$64 \times 64 \times 64$	$64 \rightarrow 64$
Upsample	$64 \times 128 \times 128$	-
ResnetBlock-50	$32 \times 128 \times 128$	$64 \rightarrow 32$
ResnetBlock-51	$32 \times 128 \times 128$	$32 \rightarrow 32$
Conv2D	$3 \times 128 \times 128$	$32 \rightarrow 3$
Tanh	$3 \times 128 \times 128$	-

[Tulyakov *et al.*, 2018] Sergey Tulyakov, Ming-Yu Liu, Xiaodong Yang, and Jan Kautz. Mocogan: Decomposing motion and content for video generation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[Welinder *et al.*, 2010] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010. 13

[Zhang *et al.*, 2017a] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017. 16

[Zhang *et al.*, 2017b] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *CoRR*, abs/1710.10916, 2017. 16

Table 10: Frame Discriminator architecture - Kinetics dataset

Layer	Output size	Filter
Input	$3 \times 128 \times 128$	-
Conv2D	$32 \times 128 \times 128$	$3 \rightarrow 32$
ResnetBlock-00	$32 \times 128 \times 128$	$32 \rightarrow 32$
ResnetBlock-01	$64 \times 128 \times 128$	$32 \rightarrow 64$
AvgPool	$64 \times 64 \times 64$	-
ResnetBlock-10	$64 \times 64 \times 64$	$64 \rightarrow 64$
ResnetBlock-11	$128 \times 64 \times 64$	$64 \rightarrow 128$
AvgPool	$128 \times 32 \times 32$	-
ResnetBlock-20	$128 \times 32 \times 32$	$128 \rightarrow 128$
ResnetBlock-21	$256 \times 32 \times 32$	$128 \rightarrow 256$
AvgPool	$256 \times 16 \times 16$	-
ResnetBlock-30	$256 \times 16 \times 16$	$256 \rightarrow 256$
ResnetBlock-31	$512 \times 16 \times 16$	$256 \rightarrow 512$
AvgPool	$512 \times 8 \times 8$	-
ResnetBlock-40	$512 \times 8 \times 8$	$512 \rightarrow 512$
ResnetBlock-41	$512 \times 8 \times 8$	$512 \rightarrow 512$
AvgPool	$512 \times 4 \times 4$	-
ResnetBlock-50	$512 \times 4 \times 4$	$512 \rightarrow 512$
ResnetBlock-51	$512 \times 4 \times 4$	$512 \rightarrow 512$
Reshape	512.4.4	-
Text-Img feat concat	n_{fused}	-
Fully connected	$n_{fused} \rightarrow 1$	$n_{fused} \rightarrow 1$

Table 11: Video Discriminator architecture - Kinetics dataset

Layer	Output size	Filter
Input	$3 \times 16 \times 128 \times 128$	-
Conv3D	$32 \times 16 \times 128 \times 128$	$3 \rightarrow 32$
ResnetBlock3D-00	$32 \times 16 \times 128 \times 128$	$32 \rightarrow 32$
ResnetBlock3D-01	$64 \times 16 \times 128 \times 128$	$32 \rightarrow 64$
AvgPool3D	$64 \times 8 \times 64 \times 64$	-
ResnetBlock3D-10	$64 \times 8 \times 64 \times 64$	$64 \rightarrow 64$
ResnetBlock3D-11	$128 \times 8 \times 64 \times 64$	$64 \rightarrow 128$
AvgPool3D	$128 \times 4 \times 32 \times 32$	-
ResnetBlock3D-20	$128 \times 4 \times 32 \times 32$	$128 \rightarrow 128$
ResnetBlock3D-21	$256 \times 4 \times 32 \times 32$	$128 \rightarrow 256$
AvgPool3D	$256 \times 2 \times 16 \times 16$	-
ResnetBlock3D-30	$256 \times 2 \times 16 \times 16$	$256 \rightarrow 256$
ResnetBlock3D-31	$512 \times 2 \times 16 \times 16$	$256 \rightarrow 512$
AvgPool3D-spatial	$512 \times 2 \times 8 \times 8$	-
ResnetBlock3D-40	$512 \times 2 \times 8 \times 8$	$512 \rightarrow 512$
ResnetBlock3D-41	$512 \times 2 \times 8 \times 8$	$512 \rightarrow 512$
AvgPool3D	$512 \times 1 \times 4 \times 4$	-
ResnetBlock3D-50	$512 \times 1 \times 4 \times 4$	$512 \rightarrow 512$
ResnetBlock3D-51	$512 \times 1 \times 4 \times 4$	$512 \rightarrow 512$
Reshape	512.4.4	-
Text-Img feat concat	n_{fused}	-
Fully connected	$n_{fused} \rightarrow 1$	$n_{fused} \rightarrow 1$

Table 12: Hyper-parameter details for Shapes and Kinetics experiments

Parameter	Config
Batch Size	8 videos
Optimizer G	Adam
Learning rate G	0.0001
Adam params G: (β_1, β_2)	$(0.0, 0.99)$
Optimizer D_F	Adam
Learning rate D_F	0.0001
Adam params D_F: (β_1, β_2)	$(0.0, 0.99)$
Regularization γ for D_F	10.0
Optimizer D_V	Adam
Learning rate D_V	0.0001
Adam params D_V: (β_1, β_2)	$(0.0, 0.99)$
Regularization γ for D_V	10.0

Table 13: Inception Score on CUB-Birds dataset

Method	Inception Score
StackGAN [Zhang *et al.*, 2017a]	3.7 ± 0.04
StackGAN v-2 [Zhang *et al.*, 2017b]	3.82 ± 0.06
Ours	4.12 ± 0.18