Generative Adversarial Graph Convolutional Networks for Human Action Synthesis

Bruno Degardin[1,3,4] João Neves[2,3] Vasco Lopes[2,3,4] João Brito[4] Ehsan Yaghoubi[3] Hugo Proença[1,3]

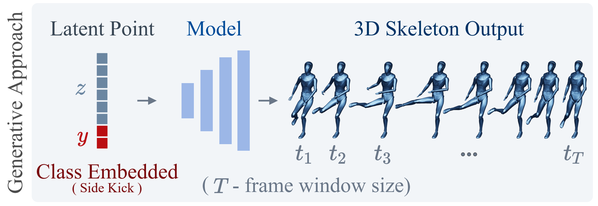
[1]IT - Instituto de Telecomunicações [2]NOVA LINCS [3]Universidade da Beira Interior [4]DeepNeuronic

Paper, Code & Pretrained Models:
https://github.com/DegardinBruno/Kinetic-GAN

2022 WACV WAIKOLOA HAWAII JAN 4-8

Problem Definition and Contribution

Goal: Synthesising human actions with realistic body movements (unconstrained environment).



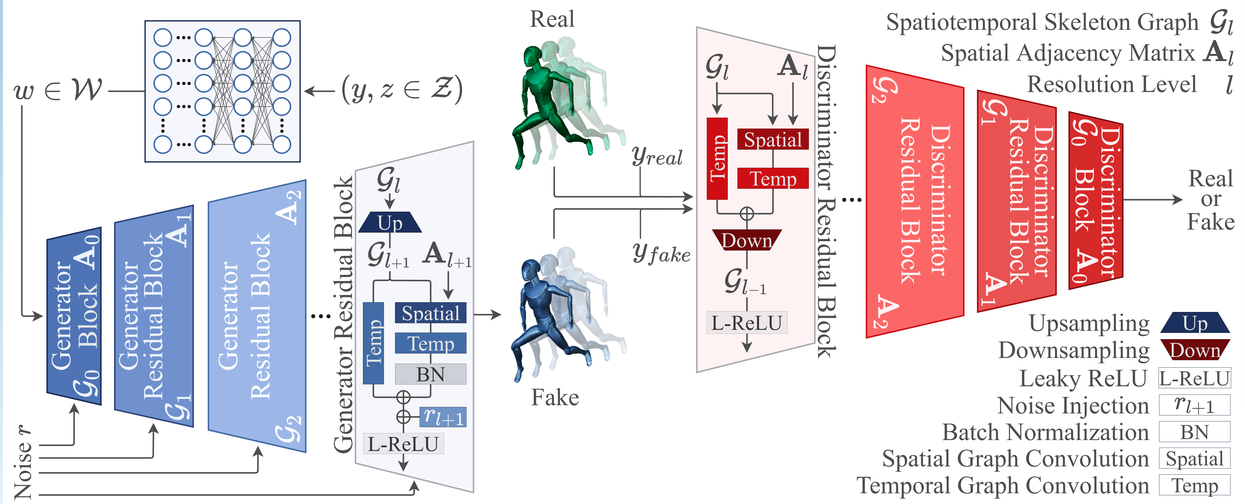
Motivation:

- Extracting 3D skeleton pose becomes difficult due to RGB and depth noises.
- Current approaches are limited in conditioning desirable actions and generate global movement.

Key Contributions:

- A new scalable Generative Adversarial Graph Convolutional Network to synthesise human actions.
- Architecture extension to a conditional model, latent space disentanglement and stochastic variation to generate desired actions (120) with increased diversity.
- Kinetic-GAN achieves state-of-the-art performance on NTU RGB+D, NTU-120 RGB+D and Human3.6M.

Method



- Embedded class representation $\boldsymbol{y}$ is concatenated to $\boldsymbol{z}$ in the generator and $\mathcal{G}$ (channel-wise) in the discriminator.
- Noise injector provides a second input to the generator to produce variations between samples (same latent point).
- Non-linear mapping network to produce an intermediate latent space $\mathcal{W}$ to allow less entangled latent factors.
- Wasserstein distance with gradient penalty loss is used to drive the learning process of Kinetic-GAN.

Problem Formulation

Main idea: Kinetic-GAN employs graph convolutions at each graph's resolution level l to overcome structural information loss from conventional convolutional kernels.

Spatiotemporal Graph Convolution:

- The graph convolution, in one frame, is computed as:

$$\mathcal{S}(\mathbf{X}_l) = \sum_{i=1}^{p} \mathbf{\Lambda}_{l_i}^{-\frac{1}{2}} (\mathbf{A}_{l_i} \odot \mathbf{M}_l) \mathbf{\Lambda}_{l_i}^{-\frac{1}{2}} \mathbf{X}_l \mathbf{W}_{l_i}, \quad (1)$$

  where $\mathbf{W}_{l_p}$ are the stacked weight vectors.
- One-dimensional kernels on the temporal axis:

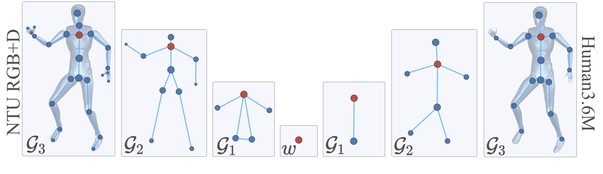
$$\mathcal{T}\big(\mathcal{S}(\mathbf{X}_l)\big) = \mathcal{S}(\mathbf{X}_l) * \mathbf{w}_l \quad (2)$$

  with temporal kernel $\mathbf{w}_l \in \mathbb{R}^{1\times t\times C}$ with t frames.

Graph Upsampling and Downsampling Paths:

- Graph upsampling: Introducing new vertices and one-dimensional interpolation over the temporal axis.
- Temporal skip connections bring training stability.

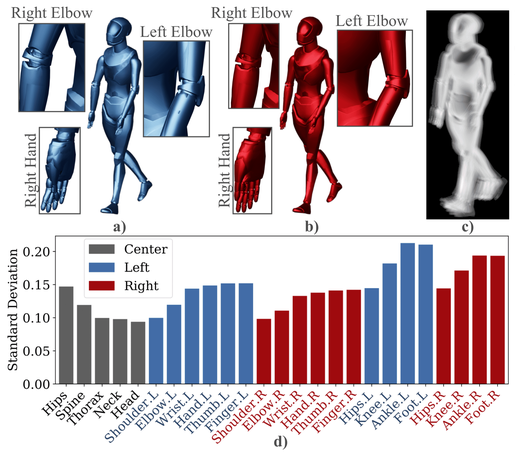
$$\begin{aligned} \mathbf{X}_{l+1} &= \mathcal{T}\big(\mathcal{S}\big(Up(\mathbf{X}_l)\big)\big) + \mathcal{T}\big(Up(\mathbf{X}_l)\big) \\ \mathbf{X}_{l-1} &= Down\big(\mathcal{T}\big(\mathcal{S}(\mathbf{X}_l)\big) + \mathcal{T}(\mathbf{X}_l)\big) \end{aligned} \quad (3)$$



Improving Quality and Diversity

Stochastic Variation: Current approaches are attached to autoregressive techniques and Gaussian process reducing the ability of variation.

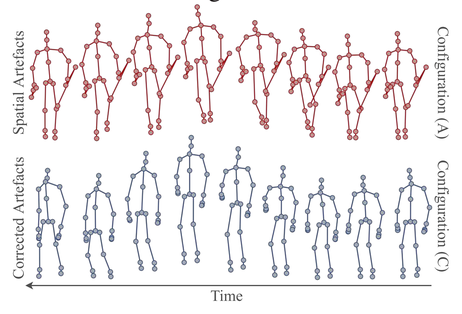
- Random noise is added joint-wise after each generator's graph convolution.
- Variation impact can be learned by assigning weights to every noise's channel.

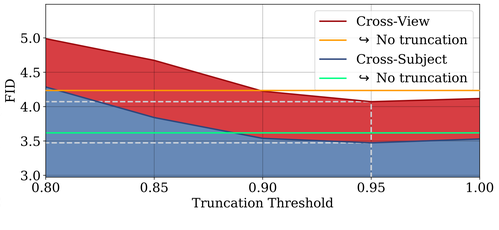


Truncation Trick on $\mathcal{W}$: Latent space $\boldsymbol{z}$ is sampled from $\mathcal{N}(0,1)$ and mapped to $\mathcal{W}$. Ranges of low density (training data) are not well represented, becoming difficult to learn for the generator. Despite some variation losses, during inference time, we scale the deviation of $\boldsymbol{w}$ as:

$$\boldsymbol{w}' = \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}}}\left[f(\boldsymbol{z})\right] + \psi(\boldsymbol{w} - \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}}}\left[f(\boldsymbol{z})\right]), \quad (4)$$

Reducing Spatial Artefacts: Batch normalization omits information concerning individual feature's magnitude (magnified by the generator). Hence, we regularize the use of batch normalization on the generator.





Quantitative and Qualitative Results

	Method	FID	MMD_a	MMD_s	FID	MMD_a	MMD_s
NTU RGB+D		*Cross-Subject*			*Cross-View*		
	c-GAN	27.480	0.919	0.975	31.875	0.993	1.088
	CSGN	6.030	0.873	0.954	7.114	0.910	0.991
	A Proposed	5.621	0.836	0.927	6.528	0.883	0.953
	B + No Residual	19.723	0.892	0.961	21.331	0.971	1.030
	C + Regular BN	4.751	0.815	0.917	5.328	0.867	0.940
	D + Noise Inject	4.698	0.811	0.895	5.102	0.851	0.933
	E + Mapping Net	**3.618**	**0.772**	**0.871**	**4.235**	**0.824**	**0.913**

		Cross-Subject		*Cross-Setup*		
NTU-120	c-GAN	54.403	1.037	1.104	58.531	1.082	1.141
	Kinetic-GAN	**5.967**	**0.819**	**0.906**	**6.751**	**0.847**	**0.934**

		MMD_a	MMD_s
Human3.6M	E2E	0.991	0.805
	EPVA	0.996	0.806
	adv-EPVA	0.977	0.792
	SkeletonVAE	0.452	0.467
	SkeletonGAN	0.419	0.419
	c-SkeletonGAN	0.195	0.218
	c-GAN	0.161	0.187
	SA-GCN	0.146	0.134
	Kinetic-GAN	**0.071**	**0.082**

		MMD_a	MMD_s	MMD_a	MMD_s
NTU-2D RGB+D		*Cross-Subject*		*Cross-View*	
	SkeletonVAE	0.992	1.136	1.079	1.205
	SkeletonGAN	0.698	0.788	0.999	1.311
	c-SkeletonGAN	0.338	0.402	0.371	0.398
	c-GAN	0.334	0.354	0.365	0.373
	SA-GCN	0.285	0.299	0.316	0.335
	Kinetic-GAN	**0.256**	**0.273**	**0.295**	**0.310**