DeepGRU: Deep Gesture Recognition Utility

Mehran Maghoumi[1,2(✉)] and Joseph J. LaViola Jr.[2]

[1] NVIDIA, Santa Clara, CA 95051, USA
[2] University of Central Florida, Orlando, FL 32816, USA
{mehran,jjl}@cs.ucf.edu

Abstract. We propose DeepGRU, a novel end-to-end deep network model informed by recent developments in deep learning for gesture and action recognition, that is streamlined and device-agnostic. DeepGRU, which uses only raw skeleton, pose or vector data is quickly understood, implemented, and trained, and yet achieves state-of-the-art results on challenging datasets. At the heart of our method lies a set of stacked gated recurrent units (GRU), two fully-connected layers and a novel global attention model. We evaluate our method on seven publicly available datasets, containing various number of samples and spanning over a broad range of interactions (full-body, multi-actor, hand gestures, *etc*.). In all but one case we outperform the state-of-the-art pose-based methods. For instance, we achieve a recognition accuracy of 84.9% and 92.3% on cross-subject and cross-view tests of the NTU RGB+D dataset respectively, and also 100% recognition accuracy on the UT-Kinect dataset. We show that even in the absence of powerful hardware, or a large amount of training data, and with as little as four samples per class, DeepGRU can be trained in under 10 min while beating traditional methods specifically designed for small training sets, making it an enticing choice for rapid application prototyping and development.

Keywords: Deep learning · Gesture recognition · Action recognition

1 Introduction

With the advent of various input devices, gesture recognition has become increasingly relevant in human-computer interaction. As these input devices get more capable and precise, the complexity of the interactions that they can capture also increases which, in turn, ignites the need for recognition methods that can leverage these capabilities. From a practitioner's point of view, a gesture recognizer would need to possess a set of traits in order to gain adoption: it should capture the fine differences among gestures and distinguish one gesture from another with a high degree of confidence, while being able to work with a vast number of input devices and gesture modalities. Concurrently, a recognition method should enable system designers to integrate the method into their workflow with the least amount of effort. These goals are often at odds: the recognition power of a recognizer usually comes at the cost of increased complexity and decreased flexibility of working across different input devices and modalities.

The original version of this chapter was revised: the name of an author was tagged incorrectly. The correction to this chapter is available at https://doi.org/10.1007/978-3-030-33720-9_54

G. Bebis et al. (Eds.): ISVC 2019, LNCS 11844, pp. 16–31, 2019.
https://doi.org/10.1007/978-3-030-33720-9_2

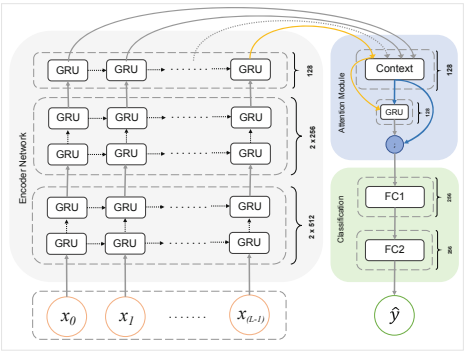

Fig. 1. Our proposed model for gesture recognition which consists of an *encoder network* of stacked gated recurrent units (GRU), the *attention module* and the *classification* layers. The input $\mathbf{x} = (x_0, x_1, ..., x_{(L-1)})$ is a sequence of vector data of arbitrary length and the output is the predicted class label $\hat{y}$. See Sect. 3 for a thorough description.

With these contradicting goals in mind, we introduce *DeepGRU*: an end-to-end deep network-based gesture recognition utility[1] (see Fig. 1). DeepGRU works directly with raw 3D skeleton, pose or other vector features (*e.g.* acceleration, angular velocity, *etc.*) produced by noisy commodity hardware, thus requiring minimal domain-specific knowledge to use. With roughly 4 million trainable parameters, DeepGRU is a rather small network by modern standards and is budget-aware when computational power is constrained. Yet, we achieve state-of-the-art results on various datasets.

Contributions. Our main contributions are devising a novel network model that works with raw vector data and is: **(1)** intuitive to understand and easy to implement, **(2)** easy to use, works out-of-the-box on noisy data, and is easy to train, without requiring powerful hardware **(3)** achieves state-of-the-art results in various use-cases, even with limited amount of training data. We believe **(1)** and **(2)** make DeepGRU enticing for application developers while **(3)** appeals to seasoned practitioners. To our knowledge, no prior work specifically focuses on model simplicity, accessibility for the masses, small training sets or CPU-only training which we think makes DeepGRU unique among its peers.

2 Related Work

Recognition with Hand-Crafted Features. Despite the success of end-to-end methods, classical methods that use hand-crafted features to perform recognition have been used with great success [18,21,49–51]. As Cheema *et al.* [9] showed, these methods can achieve excellent recognition results. They compared the performance of five algorithms (AdaBoost, SVM, decision trees *etc.*) on Wii controller gestures and concluded that, in some cases, the seemingly simple linear classifier can recognize a set of 25

[1] Reference implementation is available at: https://github.com/Maghoumi/DeepGRU.

gestures with 99% accuracy. Weng *et al*. [51] leveraged the spatio-temporal relations in action sequences with naïve-Bayes nearest-neighbor classifiers [6] to recognize actions. Xia *et al*. [53] used hidden Markov models (HMM) and the histogram of 3D joint locations to recognize gestures. Vemulapalli *et al*. [49] represented skeletal gestures as curves in a Lie group and used a combination of classifiers to recognize the gestures. Our approach differs from all of these methods in that we use the raw data of noisy input devices and do not hand-craft any features. Rather, our encoder network (Sect. 3.2) learns suitable feature representations during end-to-end training.

Recurrent Architectures. The literature contains a large body of work that use recurrent neural networks (RNN) for action and gesture recognition [10,14,16,23,24,29,33,43,48,52]. Shahroudy *et al*. [38] showed the power of recurrent architectures and long-short term memory (LSTM) units [20] for large-scale gesture recognition. Zhang *et al*. [55] proposed a view-adaptive scheme to achieve view-invariant action recognition. Their model consisted of LSTM units that would learn the most suitable transformation of samples to achieve consistent viewpoints. Avola *et al*. [2] used a LSTM architecture in conjunction with hand-crafted angular features of hand joints to recognize hand gestures. Contrary to these methods, we only use gated recurrent units (GRU) [12] as the building block of our model. We show that GRUs are faster to train and produce better results. Also, our method is designed to be general and not specific to a particular device, gesture modality or feature representation. Lastly, we leverage the attention mechanism to capture the most important parts of each input sequence.

Attention Mechanism. When using recurrent architectures, the sub-parts of a temporal sequence may not all be equally important: some subsequences may be more pertinent to the task at hand than others. Thus, it is beneficial to learn a representation that can identify these important sub-parts to aid recognition, which is the key intuition behind the attention model [3,31]. Even though the attention model was originally proposed for sequence to sequence models and neural machine translation, it has been adapted to the task of gesture and action recognition [5,28,41]. Liu *et al*. [28] proposed a global context-aware attention LSTM network for 3D action recognition. Using a global context, their method selectively focuses on the most informative joints when performing recognition. Song *et al*. [41] used the attention mechanism with LSTM units to selectively focus on discriminative skeleton joints at each gesture frame. Baradel *et al*. [5] leveraged the visual attention model to recognize human activities purely using image data. They used GRUs as the building block of their recurrent architecture.

Contrary to some of this work, DeepGRU only requires pose and vector-based data. Our novel attention model differs from prior work in how the context vector is computed and consumed. GCA-LSTM [28] has a multi-pass attention subnetwork which requires multiple initialize/refine iterations to compute attention vectors. Ours is single-pass and not iterative. Our attention model also differs from STA-LSTM [41] which has two separate temporal and spatial components, whereas ours has only one component for both domains. VA-LSTM [55] has a view-adaptation subnetwork that learns transformations to consistent view-points. This imposes the assumption that input data are spatial or view-point dependent, which may prohibit applications on non-spatial data (*e.g*. acous-

tic gestures [36]). Our model does not make any such assumptions. As we show later, our single-pass, non-iterative, spatio-temporal combined attention, and device-agnostic architecture result in less complexity, fewer parameters, and shorter training time, while achieving state-of-the-art results, which we believe sets us apart from prior work.

3 DeepGRU

In this section we provide an in-depth discussion of DeepGRU's architecture. In our architecture, we take inspiration from VGG-16 [39], and the attention [3,31] and sequence to sequence models [42]. Our model, depicted in Fig. 1, is comprised of three main components: an encoder network, the attention module, and two fully-connected (FC) layers fed to softmax producing the probability distribution of the class labels. We provide an ablation study to give insight into our design choices in Sect. 5.

3.1 Input Data

The input to DeepGRU is raw input device samples represented as a temporal sequence of the underlying gesture data (*e.g.* 3D joint positions, accelerometer or velocity measurements, 2D Cartesian coordinates of pen/touch interactions, *etc.*). At time step t, the input data is the column vector $x_t \in \mathbb{R}^N$, where N is the dimensionality of the feature vector. Thus, the input data of the entire temporal sequence of a single gesture sample is the matrix $\mathbf{x} \in \mathbb{R}^{N \times L}$, where L is the length of the sequence in time steps. Each input example sequences could have different number of time steps. We use the entire temporal sequence as-is without subsampling or clipping. When training on mini-batches, we represent the i^{th} mini-batch as the tensor $\mathbf{X}_i \in \mathbb{R}^{B \times N \times \widetilde{L}}$, where B is the mini-batch size and $\widetilde{L}$ is the length of the longest sequence in the i^{th} mini-batch. Sequences that are shorter than $\widetilde{L}$ are zero-padded.

3.2 Encoder Network

The encoder network in DeepGRU is fed with data from training samples and serves as the feature extractor. Our encoder network consists of a total of five stacked unidirectional GRUs. We prefer GRU units over LSTM units [20] as they have a smaller number of parameters and thus are faster to train and less prone to overfitting. At time step t, given an input vector x_t and the hidden state vector of the previous time step $h_{(t-1)}$, a GRU computes h_t, the hidden output at time step t, as $h_t = \Gamma\big(x_t, h_{(t-1)}\big)$ using the following transition equations:

$$
\begin{aligned}
r_t &= \sigma\ \Big(\big(W_x^r\ x_t + b_x^r\big)\ +\ \big(W_h^r\ h_{(t-1)} + b_h^r\big)\Big) \\
u_t &= \sigma\ \Big(\big(W_x^u\ x_t + b_x^u\big)\ +\ \big(W_h^u\ h_{(t-1)} + b_h^u\big)\Big) \\
c_t &= \tanh\Big(\big(W_x^c\ x_t + b_x^c\big)\ +\ r_t\big(W_h^c\ h_{(t-1)} + b_h^c\big)\Big) \\
h_t &= u_t \circ h_{(t-1)}\ +\ \Big(1 - u_t\Big) \circ c_t
\end{aligned}
\tag{1}
$$

where σ is the sigmoid function, $\circ$ denotes the Hadamard product, r_t, u_t and c_t are reset, update and candidate gates respectively and W_p^q and b_p^q are the trainable weights and biases. In our encoder network, h_0 of all the GRUs are initialized to zero.

Given a gesture example $\mathbf{x} \in \mathbb{R}^{N \times L}$, the encoder network uses Eq. 1 to output $\bar{h} \in \mathbb{R}^{128 \times L}$, where $\bar{h}$ is the result of the concatenation $\bar{h} = \left[h_0;\ h_1;\ \ldots\ ;\ h_{(L-1)}\right]$. This compact encoding of the input matrix $\mathbf{x}$, is then fed to the attention module.

3.3 Attention Module

The output of the encoder network, can provide a reasonable set of features for performing classification. We further refine this set of features by extracting the most informative parts of the sequence using the attention model. We propose a novel adaptation of the global attention model [31] which is suitable for our recognition task.

Given all the hidden states $\bar{h}$ of the encoder network, our attention module computes the attentional context vector $c \in \mathbb{R}^{128}$ using the trainable parameters W_c as:

$$c = \left(\frac{\exp\left(h_{(L-1)}^{\intercal} W_c \bar{h}\right)}{\sum_{t=0}^{L-1} \exp\left(h_{(L-1)}^{\intercal} W_c h_t\right)} \right) \bar{h} \tag{2}$$

As evident in Eq. 2, we solely use the hidden states of the encoder network to compute the attentional context vector. The hidden state of the last time step $h_{(L-1)}$ of the encoder network (the yellow arrow in Fig. 1) is the main component of our context computation and attentional output. This is because $h_{(L-1)}$ can potentially capture a lot of information from the entire gesture sample sequence. However, since the inputs to DeepGRU can be of arbitrary lengths, the amount of information that is captured by $h_{(L-1)}$ could differ among short sequences and long sequences. This could make the model susceptible to variations in sequence lengths. To mitigate this, we jointly learn a set of parameters that given the context and the hidden state of the encoder network would decide whether to use the hidden state directly, or have it undergo further transformation while accounting for the context. This decision logic can be mapped to the transition equations of a GRU (see Eq. 1). Thus, after computing the context c, we additionally compute the auxiliary context c' and produce the attention module's output o_{attn} as follows, where Γ_{attn} is the attentional GRU of the our model:

$$c' = \Gamma_{\text{attn}}\left(c, h_{(L-1)}\right) \qquad o_{\text{attn}} = \left[c\,;\ c'\right] \tag{3}$$

We believe that the novelty of our attention model is threefold. First, it only relies on the hidden state of the last time step $h_{(L-1)}$, which reduces complexity. Second, we compute the auxiliary context vector to mitigate the effects of sequence length variations. Lastly, our attention module is invariant to zero-padded sequences and thus can be trivially vectorized for training on mini-batches of sequences with different lengths. As we show in Sect. 5, our attention model works very well in practice.

3.4 Classification

The final layers of our model are comprised of two FC layers (F_1 and F_2) with ReLU activations that take the attention module's output and produce the probability distribution of the class labels using a softmax classifier:

$$\hat{y} = \text{softmax}\Big(F_2\Big(\text{ReLU}\big(F_1(o_{\text{attn}})\big)\Big)\Big) \tag{4}$$

We use batch normalization [22] followed by dropout [19] on the input of both F_1 and F_2 in Eq. 4. During training, we minimize the cross-entropy loss to reduce the difference between predicted class labels $\hat{y}$ and the ground truth labels y.

4 Evaluation

We evaluate our proposed method on five datasets: UT-Kinect [53], NTU RGB+D [38], SYSU-3D [21], DHG 14/28 [13,15] and SBU Kinect Interactions [54]. We believe these datasets cover a wide range of gesture interactions, number of actors, view-point variations and input devices. We additionally performed experiments on two small-scale datasets (Wii Remote [9] and Acoustic [36]) in order to demonstrate the suitability of DeepGRU for scenarios where only a very limited amount of training data is available. We compute the recognition accuracies on each dataset and report them as a percentage.

Implementation Details. We implemented DeepGRU using the PyTorch [35] framework. The input data to the network are z-score normalized using the training set. We use the Adam solver [25] ($\beta_1 = 0.9, \beta_2 = 0.999$) and the initial learning rate of 10^{-3} to train our model. The mini-batch size for all experiments is 128, except for those on NTU RGB+D, for which the size is 256. Training is done on a machine equipped with two NVIDIA GeForce GTX 1080 GPUs, Intel Core-i7 6850 K processor and 32 GB RAM. Unless stated otherwise, both GPUs were used for training.

Regularization. We use dropout (0.5) and data augmentation to avoid overfitting. All regularization parameters were determined via cross-validation on a subset of the training data. Across all experiments we use three types of data augmentation: **(1)** random scaling with a factor[2] of ± 0.3, **(2)** random translation with a factor of ± 1, **(3)** synthetic sequence generation with gesture path stochastic resampling (GPSR) [45]. For GPSR we randomly select the resample count n and remove count r. We use n with a factor of $(\pm 0.1 \times \widetilde{\text{L}})$ and r with a factor of $(\pm 0.05 \times \widetilde{\text{L}})$. We additionally use a weight decay value of 10^{-4}, as well as random rotation with a factor of $\pm \frac{\pi}{4}$ on NTU RGB+D dataset. This was necessary due to the multiview nature of the dataset.

[2] A factor of ± 0.3 indicates that samples are randomly and non-uniformly (*e.g.*) scaled along all axes to [0.7, 1.3] of their original size.

4.1 UT-Kinect

This dataset [53] is comprised of ten gestures performed by ten participants two times (200 sequences in total). The data of each participant is recorded and labeled in one continuous session. What makes this dataset challenging is that the participants move around the scene and perform the gestures consecutively. Thus, samples have different starting position and/or orientations. We use the leave-one-out-sequence cross validation protocol of [53]. Our approach achieves state-of-the-art results with the perfect classification accuracy of 100% as shown in Table 1.

4.2 NTU RGB+D

To our knowledge, this is the largest dataset of actions collected from Kinect (v2) [38]. It comprises about 56,000 samples of 60 action classes performed by 40 subjects. Each subject's skeleton has 25 joints. The challenging aspect of this dataset stems from the availability of various viewpoints for each action, as well as the multi-person nature of some action classes. We follow the cross-subject (CS) and cross-view (CV) evaluation protocols of [38]. In the CS protocol, 20 subjects are used for training and the remaining 20 subjects are used for testing. In the CV protocol, two viewpoints are used for training and the remaining one viewpoint is used for testing. We create our feature vectors similar to [38]. Also, note that according to the dataset authors, 302 samples in this dataset are corrupted which were omitted in our tests.

Our results are presented in Table 2. Although DeepGRU only uses the raw skeleton positions of the samples, we present the results of other recognition methods that use other types of gesture data. To the best of our knowledge, DeepGRU achieves state-of-the-art performance among all methods that only use raw skeleton pose data.

Table 1. Results on UT-Kinect [53] dataset.

Method	Accuracy	Method	Accuracy
Histogram of 3D joints [53]	90.9	GCA-LSTM *(direct)* [28]	98.5
LARP + mfPCA [1]	94.8	CNN + Feature maps [47]	98.9
ST LSTM + Trust gates [29]	97.0	GCA-LSTM *(stepwise)* [28]	99.0
Lie group [49]	97.1	CNN + LSTM [33]	99.0
ST-NBNN [51]	98.0	KRP FS [11]	99.0
DPRL + GCNN [43]	98.5	**DeepGRU**	**100.0**

Table 2. Results on NTU RGB+D [38] dataset.

Modality	Method	Accuracy		Modality	Method	Accuracy	
		CS	CV			CS	CV
Image	Multitask DL [32]	84.6	–	Pose	STA model [41]	73.2	81.2
	Glimpse clouds [5]	**86.6**	**93.2**		CNN + Kernel feature maps [47]	75.3	–
Pose + Image	DSSCA - SSLM [37]	74.9	–		SkeletonNet [23]	75.9	81.2
	STA model (Hands) [4]	82.5	88.6		GCA-LSTM *(direct)* [28]	74.3	82.8
	Multitask DL [32]	**85.5**	–		GCA-LSTM *(stepwise)* [28]	76.1	84.0
Pose	Lie group [49]	50.1	52.8		DPTC [52]	76.8	84.9
	HBRNN [17]	59.1	64.0		VA-LSTM [55]	79.4	87.6
	Dynamic Skeletons [21]	60.2	65.2		Clips + CNN + MTLN [24]	79.6	84.8
	Deep LSTM [38]	60.7	67.3		View-invariant [30]	80.0	87.2
	Part-aware LSTM [38]	62.9	70.3		DPRL + GCNN [43]	83.5	89.8
	ST LSTM + Trust Gates [29]	69.2	77.7		**DeepGRU**	**84.9**	**92.3**

4.3 SYSU-3D

This Kienct-based dataset [21] contains 12 gestures performed by 40 participants totaling 480 samples. The widely-adopted evaluation protocol [21] of this dataset is to randomly select 20 subjects for training and the use remaining 20 subjects for testing. This process is repeated 30 times and the results are averaged and presented in Table 3.

4.4 DHG 14/28

This dataset [13] contains 14 hand gestures of 28 participants collected by a near-view Intel RealSense depth camera. Each gesture is performed in two different ways: using the whole hand, or just one finger. Also, each example gesture is repeated between one

Table 3. Results on SYSU-3D [21].

Method	Accuracy	Method	Accuracy
Dynamic skeletons [21]	75.5	VA-LSTM [55]	77.5
ST LSTM + Trust gates [29]	76.5	GCA-LSTM *(stepwise)* [28]	78.6
DPRL + GCNN [43]	76.9	**DeepGRU**	**80.3**

Table 4. Results on DHG 14/28 [13] with two evaluation protocols.

Protocol	Method	Accuracy		Protocol	Method	Accuracy	
		C = 14	C = 28			C = 14	C = 28
Leave–one–out	Chen *et al.* [10]	84.6	80.3	SHREC'17 [15]	HOG^2 [15,34]	78.5	74.0
	De Smedt *et al.* [14]	82.5	68.1		HIF3D [7]	90.4	80.4
	CNN+LSTM [33]	85.6	81.1		De Smedt *et al.* [15,40]	88.2	81.9
	DPTC [52]	85.8	80.2		**DLSTM** [2]	**97.6**	**91.4**
	DeepGRU	**92.0**	**87.8**		**DeepGRU**	94.5	**91.4**

to ten times yielding 2800 sequences. The training and testing data on this dataset are predefined and evaluation can be performed in two ways: classify 14 gestures or classify 28 gestures. The former is insensitive to how an action is performed, while the latter discriminates the examples performed with one finger from the ones performed with the whole hand. The standard evaluation protocol of this dataset is a leave-one-out cross-validation protocol. However, SHREC 2017 [15] challenge introduces a secondary protocol in which training and testing sets are pre-split. Table 4 depicts our results using both protocols and both number of gesture classes.

4.5 SBU Kinect Interactions

This dataset [54] contains 8 two-person interactions of seven participants. We utilize the 5-fold cross-validation protocol of [54] in our experiments. Contrary to other datasets, which express joint coordinates in the world coordinate system, this dataset has opted to normalize the joint values instead. Despite using a Kinect (v1) sensor, the participants in the dataset have only 15 joints.

We treat action frames that contain multiple skeletons similarly to what we described above for the NTU RGB+D dataset, with the exception of transforming the joint coordinates. Also, using the equations provided in the datasets, we covert the joint values them to metric coordinates in the depth camera coordinate frame. This is necessary to make the representation consistent with other datasets that we experiment on. Table 5 summarizes our results.

4.6 Small Training Set Evaluation

The amount of training data for some gesture-based applications may be limited. This is especially the case during application prototyping stages, where developers tend to rapidly iterate through design and evaluation cycles. Throughout the years, various methods have been proposed in the literature aiming to specifically address the need for recognizers that are easy to implement, fast to train and work well with small training sets [26,27,44,46]. Here, we show that our model performs well with small training sets and can be trained only on the CPU. We pit DeepGRU against Protractor3D [27], $3 [26] and Jackknife [46] which to our knowledge produce high recognition accuracies with a small number of training examples [46].

Table 5. Results on SBU Kinect Interactions [54].

Method	Accuracy	Method	Accuracy
HBRNN [17]	80.4	Clips + CNN + MTLN [24]	93.5
Deep LSTM [38]	86.0	GCA-LSTM *(direct)* [28]	94.1
Co-occurance deep LSTM [56]	90.4	CNN + Kernel Feature Maps [47]	94.3
STA Model [41]	91.5	GCA-LSTM *(stepwise)* [28]	94.9
ST LSTM + Trust gates [29]	93.3	**VA-LSTM** [55]	**97.2**
SkeletonNet [23]	93.5	DeepGRU	95.7

Table 6. Rapid prototyping evaluation results with T training samples per gesture class.

Dataset	Method	Accuracy		Dataset	Method	Accuracy	
		$\mathcal{T} = 2$	$\mathcal{T} = 4$			$\mathcal{T} = 2$	$\mathcal{T} = 4$
Acoustic [36]	**Jackknife** [46]	**91.0**	94.0	Wii Remote [9]	Protractor3D [27]	73.0	79.6
	DeepGRU	89.0	**97.4**		$3 [26]	79.0	86.1
					Jackknife [46]	**96.0**	98.0
					DeepGRU	92.4	**98.3**

We examine two datasets. The first dataset contains acoustic over-the-air hand gestures via Doppler shifted soundwaves [36]. This dataset contains 18 hand gestures collected from 22 participants via five speakers and one microphone. The soundwave-based interaction modality is prone to high amounts of noise. The second dataset contains gestures performed via a Wii Remote controller [9] and contains 15625 gestures of 25 gesture classes collected from 25 participants. These datasets are vastly different from other datasets examined thus far in that samples of [36] are frequency binned spectrograms (165D) while samples of [9] are linear acceleration data and angular velocity readings (6D), neither of which resemble typical skeletal nor positional features.

For each experiment we use the user-dependent protocol of [9,46]. Given a particular participant, $\mathcal{T}$ random samples from that participant are selected for training and the remaining samples are selected for testing. This procedure is repeated per participant and the results are averaged across all of them. We evaluate the performance of all the recognizers using $\mathcal{T} = 2$ and $\mathcal{T} = 4$ training samples per gesture class to examine a setup with limited training data. Even though deep networks are not commonly used with very small training sets, DeepGRU demonstrates very competitive accuracy in these tests (Table 6). We see that with $\mathcal{T} = 4$ training samples per gesture class, DeepGRU outperforms other recognizers on both datasets.

5 Discussion

Comparison with the State-of-the-Art.[3] Experiment results show that DeepGRU generally tends to outperform the state-of-the-art results, sometimes with a large

[3] Refer to our supplementary material for more details: https://arxiv.org/abs/1810.12514.

Table 7. Ablation study on DHG 14/28 dataset (14 class, SHREC'17 protocol). We examine (respectively) the effects of the usage of the attention model, the recurrent layer choice (LSTM *vs*. GRU), the number of stacked recurrent layers (3 *vs*. 5) and the number of FC layers (1 *vs*. 2). Training times (seconds) are reported for every model. Experiments use the same random seed. DeepGRU's model is boldfaced.

Attn.	Rec. Unit	# Stacked	# FC	Time (sec)	Accuracy	Attn.	Rec. Unit	# Stacked	# FC	Time (sec)	Accuracy
-	LSTM	3	1	162.21	91.78	✓	LSTM	3	1	188.29	92.74
-	LSTM	3	2	164.07	91.07	✓	LSTM	3	2	192.12	92.02
-	LSTM	5	1	246.47	91.90	✓	LSTM	5	1	277.32	92.38
-	LSTM	5	2	251.67	89.52	✓	LSTM	5	2	283.35	92.26
-	GRU	3	1	143.87	93.45	✓	GRU	3	1	170.48	94.12
-	GRU	3	2	148.08	93.33	✓	GRU	3	2	174.00	93.81
-	GRU	5	1	210.83	93.69	✓	GRU	5	1	243.10	93.93
-	GRU	5	2	212.99	93.81	✓	**GRU**	**5**	**2**	**248.66**	**94.52**

Table 8. DeepGRU training times (in minutes) on various datasets.DeepGRU training times (in minutes) on various datasets.

Device	Configuration	Dataset	Time	Device	Configuration	Dataset	Time
CPU	12 threads	Acoustic [36] ($\mathcal{T} = 4$)	1.7	GPU	2 × GTX 1080	SHREC 2017 [15]	5.5
		Wii Remote [9] ($\mathcal{T} = 4$)	6.9			NTU RGB+D [38]	129.6
					1 × GTX 1080	SHREC 2017 [15]	6.2
						SYSU-3D [21]	9.0
						NTU RGB+D [38]	198.5

margin. On the NTU-RGB+D [38], we observe that in some cases DeepGRU outperforms image-based or hybrid methods. Although the same superiority is observed on the SBU dataset [54], our method achieves slightly lower accuracy compared to VA-LSTM [55]. One possible intuition for this observation could be that the SBU dataset [54] provides only a subset of skeleton joints that a Kinect (v1) device can produce (15 compared to the full set of 20 joints). Further, note that VA-LSTM's view-adaptation subnetwork assumes that the gesture data are 3D positions and viewpoint-dependent. In contrast, DeepGRU does not make any such assumptions.

As shown in Table 4, classifying 14 gestures of the DHG 14/28 dataset [13] with DLSTM [2] yields higher recognition accuracy compared to DeepGRU. As previously mentioned, DLSTM [2] uses hand-crafted angular features extracted from hand joints and these features are used as the input to the recurrent network while DeepGRU uses raw input, which relieves the user of the burden of computing domain-specific features. Classifying 28 classes, however, yields similar results with either of the recognizers.

Generality. Our experiments demonstrate the versatility of DeepGRU for various gesture or action modalities and input data: from full-body multi-actor actions to hand gestures, collected from various commodity hardware such as depth sensors or game controllers with various data representations (*e.g*. pose, acceleration and velocity or

frequency spectrograms) as well as other differences such as the number of actors, gesture lengths, number of samples and number of viewpoints. Regardless of these differences, DeepGRU can still produce high accuracy results.

Ease of Use. Our method uses raw device data, thus requiring fairly little domain knowledge. Our model is straightforward to implement and as we discuss shortly, training is fast. We believe these traits make DeepGRU an enticing option for practitioners.

Ablation Study. To provide insight into our network design, we present an ablation study in Table 7. We note depth alone is not sufficient to achieve state-of-the-art results. Further, accuracy increases in all cases when we use GRUs instead of LSTMs. GRUs were on average 12% faster to train and the worst GRU variant achieved higher accuracy than the best LSTM one. In our early experiments we noted LSTM networks overfitted frequently which necessitated a lot more parameter tuning, motivating our preference for GRUs. However, we later observed underfitting when training GRU variants on larger datasets, arising the need to reduce regularization and tune parameters again. To alleviate this, we added the second FC layer which later showed to improve results across all datasets while still faster than LSTMs to train. We observe increased accuracy in all experiments with attention, which suggests the attention model is necessary. Lastly, in our experiments we observed an improvement of roughly 0.5%–1% when the auxiliary context vector is used (Sect. 3.3). In short, we see improved results with the attention model on GRU variants with five stacked layers and two FC layers.

Timings. We measured the amount of time it takes to train DeepGRU to convergence with different configurations in Table 8. The reported times include dataset loading, preprocessing and data augmentation time. Training our model to convergence tends to be fast: GPU training of medium-sized datasets or CPU-only training of small datasets can be done in under 10 min. We also measured DeepGRU's average inference time per sample both on GPU and on CPU in *micro*seconds. On a single GPU, our methods takes 349.1 µs to classify one gesture example while it takes 3136.3 µs on the CPU.

Limitations. Our method has some limitations which we plan to address in the future. The input needs to be segmented, nonetheless adding support for unsegmented data is straightforward and requires some changes in the training protocol as demonstrated in [8]. In our experiments we observed that DeepGRU performs better with high-dimensional data, thus application on low-dimensional data may require further effort from developers. Although we used a similar set of hyperparameters for all experiments, other datasets may require some tuning.

6 Conclusion

We discussed DeepGRU, a deep network-based gesture and action recognizer which directly works with raw pose and vector data. We demonstrated that our architecture,

which uses stacked GRU units and a global attention mechanism along with two fully-connected layers, was able to achieve state-of-the-art recognition results on various datasets, regardless of the dataset size and interaction modality. We further examined our approach for application in scenarios where training data is limited and computational power is constrained. Our results indicate that with as little as four training samples per gesture class, DeepGRU can still achieve competitive accuracy. We also showed that training times are short and CPU-only training is possible.

References

1. Anirudh, R., Turaga, P., Su, J., Srivastava, A.: Elastic functional coding of human actions: from vector-fields to latent variables. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3147–3155 (2015)
2. Avola, D., Bernardi, M., Cinque, L., Foresti, G.L., Massaroni, C.: Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures. IEEE Trans. Multimed. **21**, 234–245 (2018)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: Proceedings of ICLR (2015)
4. Baradel, F., Wolf, C., Mille, J.: Human action recognition: pose-based attention draws focus to hands. In: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 604–613 (2017)
5. Baradel, F., Wolf, C., Mille, J., Taylor, G.W.: Glimpse clouds: human activity recognition from unstructured feature points. In: The IEEE Conference on Computer Vision and Pattern Recognition (2018)
6. Boiman, O., Shechtman, E., Irani, M.: In defense of nearest-neighbor based image classification. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
7. Boulahia, S.Y., Anquetil, E., Multon, F., Kulpa, R.: Dynamic hand gesture recognition based on 3D pattern assembled trajectories. In: 2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA), pp. 1–6 (2017)
8. Caputo, F.M., et al.: Online gesture recognition. In: Eurographics Workshop on 3D Object Retrieval (2019)
9. Cheema, S., Hoffman, M., LaViola, J.J.: 3D gesture classification with linear acceleration and angular velocity sensing devices for video games. Entertain. Comput. **4**(1), 11–24 (2013)
10. Chen, X., Guo, H., Wang, G., Zhang, L.: Motion feature augmented recurrent neural network for skeleton-based dynamic hand gesture recognition. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 2881–2885 (2017)
11. Cherian, A., Sra, S., Gould, S., Hartley, R.: Non-linear temporal subspace representations for activity recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2197–2206 (2018)
12. Cho, K., et al.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734 (2014)
13. De Smedt, Q., Wannous, H., Vandeborre, J.P.: Skeleton-based dynamic hand gesture recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1–9 (2016)
14. De Smedt, Q., Wannous, H., Vandeborre, J.-P.: 3D hand gesture recognition by analysing set-of-joints trajectories. In: Wannous, H., Pala, P., Daoudi, M., Flórez-Revuelta, F. (eds.) UHA3DS 2016. LNCS, vol. 10188, pp. 86–97. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91863-1_7

15. De Smedt, Q., Wannous, H., Vandeborre, J.P., Guerry, J., Le Saux, B., Filliat, D.: Shrec'17 track: 3D hand gesture recognition using a depth and skeletal dataset. In: 10th Eurographics Workshop on 3D Object Retrieval (2017)
16. Devineau, G., Moutarde, F., Xi, W., Yang, J.: Deep learning for hand gesture recognition on skeletal data. In: 2018 13th IEEE International Conference on Automatic Face Gesture Recognition (FG 2018), pp. 106–113 (2018)
17. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1110–1118 (2015)
18. Fernández-Ramírez, J., Álvarez-Meza, A., Orozco-Gutiérrez, Á.: Video-based human action recognition using kernel relevance analysis. In: Bebis, G., et al. (eds.) ISVC 2018. LNCS, vol. 11241, pp. 116–125. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03801-4_11
19. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.R.: Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580 (2012)
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
21. Hu, J., Zheng, W., Lai, J., Zhang, J.: Jointly learning heterogeneous features for rgb-d activity recognition. IEEE Trans. Pattern Anal. Mach. Intell. **39**(11), 2186–2200 (2017)
22. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift, pp. 448–456 (2015)
23. Ke, Q., An, S., Bennamoun, M., Sohel, F., Boussaid, F.: Skeletonnet: mining deep part features for 3-D action recognition. IEEE Signal Process. Lett. **24**(6), 731–735 (2017)
24. Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F.: A new representation of skeleton sequences for 3D action recognition. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 4570–4579. IEEE (2017)
25. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
26. Kratz, S., Rohs, M.: The $3 recognizer: Simple 3D gesture recognition on mobile devices. In: Proceedings of the 15th International Conference on Intelligent User Interfaces (2010)
27. Kratz, S., Rohs, M.: Protractor3D: a closed-form solution to rotation-invariant 3D gestures. In: Proceedings of the 16th International Conference on Intelligent User Interfaces (2011)
28. Liu, J., Wang, G., Duan, L., Abdiyeva, K., Kot, A.C.: Skeleton-based human action recognition with global context-aware attention lstm networks. IEEE Trans. Image Process. **27**(4), 1586–1599 (2018)
29. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal LSTM with trust gates for 3D human action recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 816–833. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_50
30. Liu, M., Liu, H., Chen, C.: Enhanced skeleton visualization for view invariant human action recognition. Pattern Recogn. **68**(C), 346–362 (2017)
31. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (2015)
32. Luvizon, D.C., Picard, D., Tabia, H.: 2D/3D pose estimation and action recognition using multitask deep learning. In: The IEEE Conference on Computer Vision and Pattern Recognition, vol. 2 (2018)
33. Núñez, J.C., Cabido, R., Pantrigo, J.J., Montemayor, A.S., Vélez, J.F.: Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recogn. **76**(C), 80–94 (2018)

34. Ohn-Bar, E., Trivedi, M.M.: Joint angles similarities and HOG2 for action recognition. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops (2013)
35. Paszke, A., et al.: Automatic differentiation in pytorch. In: NIPS-W (2017)
36. Pittman, C.R., LaViola Jr., J.J.: Multiwave: complex hand gesture recognition using the doppler effect. In: Proceedings of the 43rd Graphics Interface Conference. pp. 97–106 (2017)
37. Shahroudy, A., Ng, T., Gong, Y., Wang, G.: Deep multimodal feature analysis for action recognition in RGB+D videos. IEEE Trans. Pattern Anal. Mach. Intell. **40**(5), 1045–1058 (2018)
38. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: NTURGB+D: a large scale dataset for 3D human activity analysis. In: IEEE Conference on Computer Vision and Pattern Recognition (2016)
39. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. CoRR abs/1409.1556 (2014)
40. Smedt, Q.D., Wannous, H., Vandeborre, J.: Skeleton-based dynamic hand gesture recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1206–1214 (2016)
41. Song, S., Lan, C., Xing, J., Zeng, W., Liu, J.: An end-to-end spatio-temporal attention model for human action recognition from skeleton data. AAAI. **1**, 4263–4270 (2017)
42. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
43. Tang, Y., Tian, Y., Lu, J., Li, P., Zhou, J.: Deep progressive reinforcement learning for skeleton-based action recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (2018)
44. Taranta, II, E.M., LaViola Jr., J.J.: Penny pincher: a blazing fast, highly accurate $-family recognizer. In: Proceedings of the 41st Graphics Interface Conference, pp. 195–202 (2015)
45. Taranta II, E.M., Maghoumi, M., Pittman, C.R., LaViola Jr., J.J.: A rapid prototyping approach to synthetic data generation for improved 2D gesture recognition. In: Proceedings of the 29th Symposium on User Interface Software and Technology, pp. 873–885. ACM (2016)
46. Taranta II, E.M., Samiei, A., Maghoumi, M., Khaloo, P., Pittman, C.R., LaViola Jr., J.J.: Jackknife: a reliable recognizer with few samples and many modalities. In: Proceedings of the 2017 Conference on Human Factors in Computing Systems, pp. 5850–5861 (2017)
47. Tas, Y., Koniusz, P.: CNN-based action recognition and supervised domain adaptation on 3D body skeletons via kernel feature maps. In: BMVC (2018)
48. Tewari, A., Taetz, B., Grandidier, F., Stricker, D.: Two phase classification for early hand gesture recognition in 3D top view data. In: Bebis, G., et al. (eds.) ISVC 2016. LNCS, vol. 10072, pp. 353–363. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-50835-1_33
49. Vemulapalli, R., Arrate, F., Chellappa, R.: Human action recognition by representing 3D skeletons as points in a lie group. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 588–595 (2014)
50. Vrigkas, M., Mastora, E., Nikou, C., Kakadiaris, I.A.: Robust incremental hidden conditional random fields for human action recognition. In: Bebis, G., et al. (eds.) ISVC 2018. LNCS, vol. 11241, pp. 126–136. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-03801-4_12
51. Weng, J., Weng, C., Yuan, J.: Spatio-temporal naive-bayes nearest-neighbor (ST-NBNN) for skeleton-based action recognition. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, pp. 445–454 (2017)
52. Weng, J., Liu, M., Jiang, X., Yuan, J.: Deformable pose traversal convolution for 3D action and gesture recognition. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 142–157. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_9

53. Xia, L., Chen, C., Aggarwal, J.: View invariant human action recognition using histograms of 3D joints. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 20–27. IEEE (2012)
54. Yun, K., Honorio, J., Chattopadhyay, D., Berg, T.L., Samaras, D.: Two-person interaction detection using body-pose features and multiple instance learning. In: 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. IEEE (2012)
55. Zhang, P., Lan, C., Xing, J., Zeng, W., Xue, J., Zheng, N.: View adaptive recurrent neural networks for high performance human action recognition from skeleton data. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2136–2145 (2017)
56. Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X.: Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 3697–3703 (2016)