GAN-Based Data Augmentation for Visual Finger Spelling Recognition

Bogdan Kwolek

AGH University of Science and Technology
30 Mickiewicza Av., 30-059 Krakow, Poland
bkw@agh.edu.pl

ABSTRACT

In this work we extend WGAN-GP in order to achieve better generation of synthesized images for finger spelling classification. The main difference between the ordinary WGAN-GP and the proposed algorithm is that in the training we employ both training samples and training labels. These training labels are fed to the generator, that generates the synthetic images using both the randomized latent input and the input label. In ordinary WGAN-GP, latent input variables are usually sampled from an unconditional prior. In the proposed algorithm the latent input vector is a concatenation of random part, the class labels and additional variables that are drawn from Gaussian distributions representing hand poses or gesture attributes. The JSL dataset for Hiragana sign recognition has been balanced using the rendered samples on the basis of a 3D hand model as well as the extended WGAN-GP.

Keywords: Generative Adversarial Networks, CNNs, Finger Spelling Recognition

1. INTRODUCTION

Hand gesture is one of the most intuitive and versatile methods to achieve user-friendly communication between people and machines [1]. Gesture recognition is becoming one of the key technologies in the area of human-machine communication, including virtual reality [2], game control, robotics [3] and assistive technologies for the handicapped and the elderly [4]. Touch-less gesture recognition systems are introduced in automotive user interfaces as they have strong potential to improve safety and comfort of travel [5]. Gesture recognition on videos from a single color camera is very useful, yet difficult task, due to occlusions, differences in hand anatomy, variations of posture appearance, etc. In real scenarios, many challenging sub-tasks such as detection of hands and other body parts, motion modeling and tracking, pattern recognition and classification should be executed to perform on-line gesture recognition. Despite considerable research [6], there are still challenges in designing hand gesture recognition systems for real-life applications.

While gesture communication may involve motion of all parts of the body, most of the research focuses on arms and hands, which are the most essential limbs in nonverbal communication. Hand gestures can be classified into static hand gestures that are characterized by hand posture with a particular finger-thumb-palm configuration, and dynamic hand gestures, which can be characterized by their shape, position and movement. Depending on the type of the input data, interpreting hand gestures could be achieved in several ways. In the last decade, several approaches to static gesture recognition on color images were proposed [7]. Despite high potential of convolutional neural networks (CNNs), a recent survey on hand gesture recognition [7] reports only one significant approach, i.e. [8]. In [9], a CNN has been utilized in classification of six hand gestures to control robots using colored gloves. In a more recent work [10], a CNN has been implemented using Theano framework and then executed on the Nao humanoid robot. In recently published work [11], a CNN has been trained on one million of exemplars. However, only a subset of data, namely 3361 manually labeled frames into 45 classes of real sign language has been made publicly available. One of the obstacles to broader use of CNNs for gesture recognition is lack of properly aligned datasets of sufficient size as well as shortage of robust real-time hand detectors.

In this work, a Generative Adversarial Network (GAN) has been used to improve the accuracy of classification of finger spellings from Hiragana sign language. It has been used to balance and augment the training part of JSL dataset consisting of 5890 samples, which were recorded by a single RGB camera [12]. We demonstrate that better classification rates can be obtained by convolutional neural networks learned on data balanced and augmented by the GAN as well as images rendered on the basis of 3D articulated hand models that were configured to shape the finger spellings.

Finger spelling is a form of sign language, where each sign corresponds to a letter of the alphabet. Tabata and Kuroda [13] proposed a system to recognize hand shapes for finger spelling in Japanese Sign Language (JSL), which they call

”Stringlove”. Their system uses a custom-made glove, equipped with sensors to capture finger features. The glove contains nine contact sensors and 24 inductcoders that jointly estimate several descriptors such as: adduction/abduction angles of fingers, thumb and wrist rotations, the joint flexion/extension of the fingers and the contact positions among the fingertips of the fingers. In [14], a modified form of the shape matrix that is capable of capturing salience of the finger spelling postures on the basis of precise sampling of contours and regions has been proposed. In [15], recognition of JSL finger spellings is achieved by a CNN, which has been trained on a dataset consisting of 5000 real images, 1000 augmented images and 2000 synthetic images, which were generated on the basis of a 3D articulated hand model.

2. DATA AUGMENTATION

Data augmentation refers to generating additional training data for learning a model by transforming the given input data. Typically, data augmentation is performed online during the learning process. Even for learning deep models using very big amounts of labeled data it can be an useful data regularizer [16]. Classical deep augmentation techniques of images comprise augmenting by adding additive noise, augmenting by intensity scaling/shift, random flipping, random translation and/or rotation, warping. Another group of techniques is based on synthetic data generation. In such an approach, data is generated on the basis of a generative model of some kind that is capable of resembling the real data. An example of such technique is rendering on the basis of 3D models. In a recent work [17], 3D models have been employed to render highly realistic images in order to increase the amount of training data and to improve the performance of object detection. In [15], 3D articulated models of the hand were used to improve the classification of Hiragana finger spelling.

Very recently, it has been demonstrated that GAN networks can enhance the synthetically generated data using information from real data [64]. In order to refine the synthetic image data, for which the annotation is given, a GAN-based network called SimGAN has been introduced. The goal of the refinement process is to use real, unlabelled data to enhance the realism of the rendered images, while preserving the annotation information.

3. GENERATIVE ADVERSARIAL NETWORKS

Generative Adversarial Networks (GANs) are a class of algorithms utilized in unsupervised learning, implemented by two differentiable submodels contesting with each other in a zero-sum game framework. They were introduced by Ian Goodfellow et al. in 2014 [18]. The first submodel is the generator G with parameters θ_G, whereas the second submodel is the discriminator D with parameters θ_D. The trainable generator is responsible for generating images $G(z)$ that resemble the training data distribution p_R, using a latent noise vector z from an input distribution $p(z)$. The aim of the trainable discriminator, which operates on the generated images $G(z)$ as well as the real training data x is to discriminate between generated images and real images. The discriminator and the generator compete against each other. The generator G is learned to minimize the likelihood of D to recognize the generated images as synthetic, i.e. $(D(G(z))) \approx 0$, while the discriminator D is trained to maximize the likelihood of correct decisions for differentiating between real and synthetic images, i.e. $(D(x) \approx 1)$, $(D(G(z))) \approx 0$. This leads to the following minimax function:

$$\begin{aligned} \min_G \max_D V(G,D) &= \mathbb{E}_{x\sim p_R(x)}[\log(D(x))] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))] \\ &= \mathbb{E}_{x\sim p_R(x)}[\log(D(x))] + \mathbb{E}_{x\sim p_G(x)}[\log(1-D(x))] \end{aligned} \tag{1}$$

where $p_G(x)$ stands for generator’s distribution over data x, $p_z(z)$ is data distribution over noise input z, whereas p_R is the distribution of the real images, and $\mathbb{E}_{x\sim p_R(x)}[\log(D(x))]$ has no impact on G during gradient descent updates.

For fixed G, one of the optimal discriminator D is as follows:

$$D^*_G = \frac{p_R}{p_R + p_G} \in [0,1] \tag{2}$$

The optimum of such a minimax game is for $p_R = p_G$, when the generator is perfect at replicating the distribution of the training data. If $p_R = p_G$, D^*_G becomes $\frac{1}{2}$, and the discriminator is unable to differentiate between the real images and synthesized images by the generator. The minimax objective (1) is optimized by alternate optimizations of G and D. The GANs can be trained by using well known techniques, such as backpropagation and Stochastic Gradient Descent.

The training of GANs is not an easy task. One of the undesirable effects that can take place during training of GANs is mode collapse [19]. It is a common problem for GANs. If this undesirable effect occurs, the generator learns to map multiple different input vectors z to identical output $G(z)$. The reason for this is that the gradients of the discriminator are calculated without regard to each other, i.e. without considering any kind of similarity measure for comparison of samples from a given minibatch [19]. In consequence, the generator tends to learn a single (to create a single image) with the biggest response of the discriminator. When this happens, the discriminator learns that this image is synthetic one, i.e. that it was generated by the generator, and in the response the generator shifts this mode a little bit. Because all of the synthesized images in a mode-collapsed minibatch are very similar, there is not enough difference in gradients to coerce the generator to create multiple unlike images. Instead, it will generate only slightly different images, leading to some oscillatory stagnation resulting on creating only vastly small amount of different images, or images that share identical features. Techniques like minibatch discrimination [19] or Wasserstein loss [20] can be used to cope with such undesirable effects.

In [21], it has been demonstrated that the changes of input vector z can lead to representing the training data by a submanifold spanned by the input images. In particular, it has been demonstrated that on the basis of two noise vectors z_1 and z_2 from a uniform distribution and a linear interpolation between them, the generated images between z_1 and z_2 compose a smooth transition. Thus, it has been shown that the generator just does not learn to recreate the training images, but it is also capable of establishing a submanifold spanned by the input images. It has been also demonstrated that such latent spaces allow manipulation on the generated images. For instance, by calculating the average of latent vector corresponding to output images with people wearing glasses, such an average can be employed to add glasses to a person without glasses. This can be completed by adding the average of latent vector for images with people with glasses to the vector of a person without glasses and then generating the images using the updated vector. The discussed work demonstrated also that using the discriminator as a feature extractor in the image classification can lead to competitive results on SVHN (Street View House Numbers) and CIFAR datasets.

In the original formulation for GANs introduced by Goodfellow et al. [18], the objective function was the Jensen-Shannon (JS) divergence. In the recently introduced Wasserstein Generative Adversarial Networks (WGANs) by Arjovsky et al. [20], the loss function uses the Earth-Mover (EM) distance or Wasserstein distance, which expresses the optimal cost of transferring the distribution p_R into p_G. It has been proved that a small EM distance corresponds to a small difference in distributions. When optimized, the Wasserstein distance has better properties in comparison to JS or KL (Kullback-Leibler) divergence, including more reasonable behavior during learning distributions supported by low dimensional manifolds. Out of JS, KL, and Wasserstein distance, only the Wasserstein distance guarantees the continuity and differentiability. As a result, the WGANs are being able to model probability distributions, where other algorithms with JS- or KL-based objectives fail. Thus, the Wasserstein distance allows us to construct a compelling loss function for generative models. Unfortunately, calculating the Wasserstein distance

$$W(p_R, p_G) = \inf_{\lambda \in \Pi(p_R, p_G)} \mathbb{E}_{(x,y)\sim\gamma}[||x - y||] \tag{3}$$

exactly is intractable. The discussed paper demonstrates how we can calculate an approximation of W using results from Kantorovich-Rubinstein duality, which shows that W is equivalent to

$$W(p_R, p_\theta) = \sup_{||f||_L \leq 1} \mathbb{E}_{x\sim p_R}[f(x)] - \mathbb{E}_{x\sim p_\theta}[f(x)] \tag{4}$$

where $\sup$ is the least upper bound and f is a $1-$ Lipschitz function following the constraint:

$$|f(x_1) - f(x_2)| \leq |x_1 - x_2| \tag{5}$$

In order to compute the Wasserstein distance, we just need to find a $1-$ Lipschitz function. So, we can construct a deep neural network to learn it. Such a network is quite similar to the discriminator, just with no sigmoid function, with a scalar score as output rather than a probability. This scalar score reflects how real the input images are and such a discriminator is called a critic to reflect its new role. In the WGAN neural networks when the critic operates well the generator can still learn. In GAN, the loss is a measure how well the discriminator is fooled rather than a measure of quality of the image. On the other hand, the WGAN loss function reflects the image quality that is far more desirable. One of the difficulties in learning the WGAN is to enforce the Lipschitz constraint. Clipping permits enforcing the Lipschitz constraint on the critic's model to compute the Wasserstein distance:

$$|f(x_1) - f(x_2)| \leq K|x_1 - x_2| \tag{6}$$

However, the use of clipping to restrict the maximum weight value in f can result in a model that does not converge and poor quality of the rendered images, particularly if the hyperparameter c that controls the range in which the values assumed by the discriminator should be: $w \leftarrow \text{clip}(w, -c, c)$, is not tuned properly. In practice, the performance of WGAN can be very sensitive to hyperparameter c. Moreover, the weight clipping reduces the capacity of the model as well as constraints the ability of modeling complex functions. Additionally, the weight clipping can lead to vanishing or exploding gradient norms, which in turn result in unstable gradients, and thus, unstable training [22]. The recently introduced Wasserstein GAN with gradient penalty (WGAN-GP) [22] employs gradient penalty instead of the weight clipping in order to enforce the Lipschitz constraint. The discussed work proved that the optimal critic in the WGAN neural networks has unit gradient norm almost everywhere under p_R and p_G. By using a soft penalty on the gradient norm of the critic to enforce the 1-Lipschitz constraint of the original WGAN, the gradient norm can be encouraged towards 1. In the proposed approach, Gulrajani et al. [22] sample uniformly from a distribution $p_{\hat{x}}$ that is defined along lines among pairs of samples coming from the model distribution p_G and the real data distribution p_R. The WGAN-GP loss function can be expressed as follows:

$$L = \underbrace{\mathbb{E}_{x\sim p_R(x)}[(D(x))] - \mathbb{E}_{z\sim p_z(z)}[D(G(z))]}_{\text{WGAN critic loss}} + \underbrace{\lambda \mathbb{E}_{\hat{x}\sim p_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2)^2]}_{\text{WGAN-GP gradient penalty}} \tag{7}$$

where λ stands for a penalty coefficient. Compared to the WGAN neural network, which should use batch normalization in both the generator and the discriminator, the WGAN-GP does not utilize the batch normalization in the critic, as the gradient penalty term in the objective function would not be valid [22]. Instead, it is recommended to utilize the layer normalization [23].

4. THE ALGORITHM

We extended WGAN-GP in order to achieve better generation of synthesized images for data augmentation and learning CNNs for finger spelling classification on such augmented datasets. The proposed algorithm for generation of synthetic images of finger spelling employs label conditioning, which has previously been applied to ordinary GANs [24]. By adding an auxiliary classifier to the discriminator of WGAN-GP, the modified discriminator produces not only a probability distribution over sources but also probability distribution over the class labels. The main difference between the ordinary WGAN-GP and the proposed algorithm is that to train it we employ both training samples and training labels. These training labels are fed to the generator, that generates the synthetic images using both the random noise and the input label. The discriminator also has to predict the source of the image (i.e. if it is a synthesized image or a real one), and it also has to predict the label of the image.

In the ordinary GANs, latent input variables are sampled from an unconditional prior. As a result, it is not easy to interpret the contribution of the individual variables and to steer the synthesis of the images. As shown in [21], by performing some operations on latent input samples we can steer the image synthesis, for instance, we can add some attributes to the synthesized objects. This observation motivated us to include additional variables in the latent input vector. In the proposed algorithm the latent input vector is a concatenation or random part, the class labels and additional variables that are drawn from some number of Gaussian distributions, and represent (characteristic) hand poses or gesture attributes. They were sampled from Gaussian distributions representing classes sharing common poses or hand appearances.

5. GESTURE DATASET

Experimental evaluations have been performed on JSL finger spelling dataset [15]. The JSL words are articulated by five fingers of the single hand and the direction of the hand. Most of finger spellings are expressed through static postures. In addition, dullness, half dullness, and long sound are expressed by dynamic hand postures. In this work we considered only static Hiragana signs and classified 41 static finger spellings from the JSL. Figure 1 depicts sample images from the JSL dataset [15]. In the following columns and rows (from left to right and from top to bottom), the words that make up the Hiragana alphabet are presented.

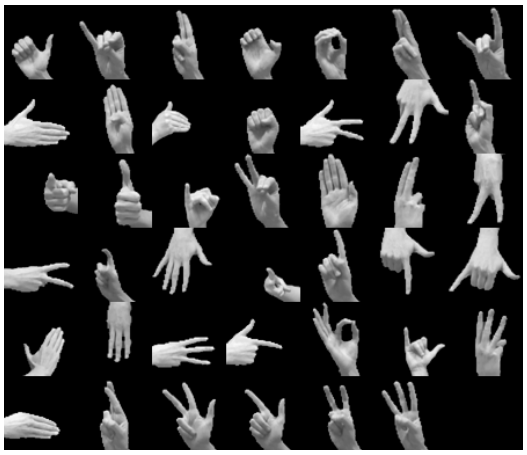


Figure 1. Sample images from JSL dataset.

Training and evaluation of CNN-based classifiers as well as GANs that were used for data augmentation has been performed on 5890 real images from the JSL dataset. The evaluations were performed in a person independent scenario, wherein gestures performed by some subject have been used for testing, whereas gestures performed by remaining subjects have been utilized for training. The test data consists of 579 images with the finger spellings, which were expressed by person #10. All experiments were performed on gray images of size 64×64.

Figure 2 contains a plot of number of training images versus Hiragana sign in the training part of the JSL dataset. As we can notice on the discussed plot, the JSL dataset is somewhat unbalanced. The Hiragana sign 'chi', see third sign in the third row from top on Fig. 1, is represented by the smallest number of images (45 images).

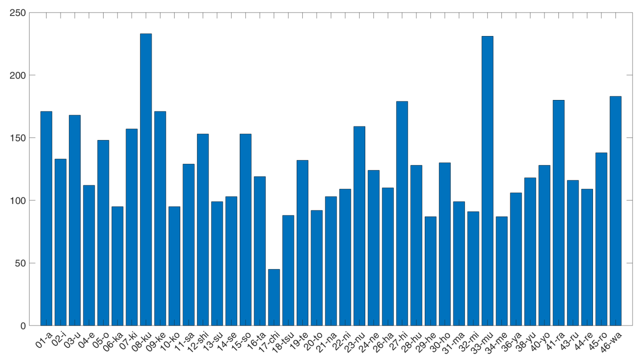


Figure 2. Number of images vs. finger spelling in training part of the JSL dataset.

6. EXPERIMENTS

At the beginning we experimented with various architectures of GANs. However, due to specificity of JSL finger spelling dataset, the synthesized images were not suitable for augmenting or even balancing the JSL dataset. It turned out that existing implementations of GANs, and even WGANs achieve unsatisfactory results. The main reason is that the parameters used in the models of the existing implementations were tuned for widely used datasets like MNIST, FMNIST, etc. What is more, it turned out that after manual selection of the images from the synthesized image pools and using them to augment the JSL training data the classification results were even worse.

Next, we employed the 3D articulated hand model from libHand library [25] to synthesize Hiragana finger spellings. A program in C/C++ language with OpenGL/Glut graphics engine has been developed and implemented to configure the 3D hand model in the requested poses and then to render them on images of size 64×64. Posing the articulated hand model in the desired poses has been done using a slider for selecting a joint on the hand skeleton and three sliders to change the rotation angles of the selected joint. By observing the reference image with a model Hiragana sign (pattern-sign), changing values of the sliders, and eventually rotating the model to observe the rendered hand from a different view point, the fingers could easily be bent to the desired shapes. For each Hiragana sign the 3D hand model has been manually configured in thirty different poses to represent possible variations in expressing the given Hiragana sign by different performers. Posing of the 3D hand models has been done by thirty modelers/students. Each of 1230 hand signs has been rotated about x, y and $-z$ axes. For each axis two random numbers were generated to rotate the 3D hand. On the basis of such set of models configured in such a way we rendered gray images of size 64×64, i.e. $41 \times 30 \times 7 = 8610$ images. The JSL-rend dataset is available at: `http://home.agh.edu.pl/~bkw/research/data/icmv`. Figure 1a) illustrates example images that were sampled from the dataset (model not rotated). Given the collection of rendered finger spellings we sampled images to balance the JSL dataset such that the minimal number of training images in each class (JSL+JSL-rend) was not smaller than one hundred and forty. Thanks to balancing the dataset and then using it to synthesize the images on the basis of WGANs, perceptually better Hiragana signs were generated in comparison to experiments in which only real images from JSL dataset were employed.

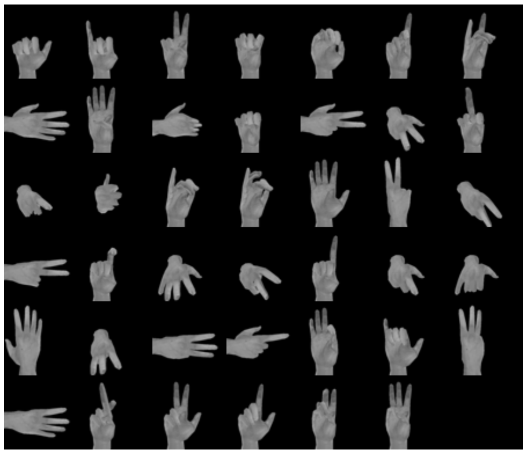
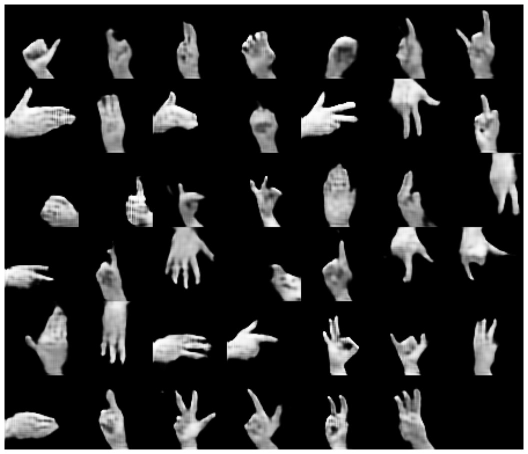
a) 
b) 

Figure 3. Synthesized Hiragana signs using libHand a), synthesized Hiragana signs using GAN b).

Owing to expanding the JSL dataset about 3D-model based rendered Hiragana signs, the WGAN-GP achieved better results, but the quality of most synthesized images was still unsatisfactory. Extending WGAN-GP about class information and adding attribute dependent latent variables improves the quality of synthesized Hiragana signs. The experimental results are presented in Tab 1. As we can notice, extending the JSL dataset about JSL-rend data allows us to improve the classification performance, cf. first and second row in Tab 1. Balancing the JSL dataset with manually selected samples from pool of images generated by WGAN-GP did not improve the classification results, despite numerous training sessions with unlike parameters, see also 3rd row in Tab 1. In the discussed experiment the JSL+JSL-rend dataset has been further balanced so that the minimal number of samples in the considered classes is equal to one hundred and fifty.

As we can notice in Tab 1, the best results were achieved using the proposed extension of WGAN-GP. In the presented experiment the length of latent vector with samples from uniform distribution has been set to 60. It has been extended about 41 element vector with labels representing classes and 20 element vector with samples drawn from five Gaussian distributions, representing three hand poses (vertical/horizontal/neutral) and frontal/backward hand sides, see also Fig. 1. Given the trained WGAN-GP on JSL dataset augmented with 3D-model based rendered Hiragana words, the generator has been used to generate synthetic images. The classification has been performed on JSL data, augmented by samples rendered on the basis of 3D hand models and data generated by WGAN-GP. For gestures having the number of training images smaller than 150 we manually selected the images from the synthesized pool to achieve the mentioned above number of training samples. This means, that several rendered Hiragana words ('a', 'u', 'o', 'ki', 'ku', 'ke', 'shi', 'so', 'nu', 'hi', 'mu', 'ra', 'wa'), were not included in the dataset, see also poorly rendered sign 'ra' on Fig. 1b, which is indistinguishable from 'nu' and 'ha', which in turn are similar to each other. Samples selected in such a way were concatenated with real images from Hiragana JSL dataset as well as images rendered on the basis of the 3D hand models. The classification has been performed using CNNs that were used in our previous work [12]. The WGAN-GP has been built on ResNet and trained on TitanX GPU with epoch size set to 32 and number of epochs set to 300. The time for training the WGAN-GP on the Titan X is about one day. The WGAN-GP neural network has been implemented in Python using TensorFlow/Keras frameworks.

Table 1. Classification performance in performer independent experiment: first row - performance measures obtained by CNN on JSL dataset, second row - performance attained by CNN on JSL + JSL-rend data, third row - performance achieved by CNN on JSL + JSL-rend and data augmented by WGAN-GP, fourth row - performance obtained by CNN on JSL + JSL-rend and data augmented by proposed WGAN-GP.

	Accuracy	Precision	Recall	F1-score
CNN - JSL	0.753	0.792	0.754	0.730
CNN - JSL + JSL-rend	0.769	0.770	0.771	0.742
CNN - JSL + JSL-rend + WGAN-GP	0.734	0.738	0.734	0.705
CNN - JSL + JSL-rend + WGAN-GP prop.	0.788	0.820	0.788	0.777

7. CONCLUSION

In this work we proposed an approach for finger spelling recognition using data augmented by GANs as well as data rendered on the basis of 3D hand model. The experimental results show that owing to proposed WGAN-GP extensions we can synthesize additional data, which allows us balance JSL dataset and to improve the classification performance of Hiragana fingerspellings. The samples generated on the basis of the 3D hand models relatively well reflect the shapes of the gestures as well as the position of the fingers within hand expressing the given gesture, but they look as gestures performed by flat hands. Future work will focus on extending the WGAN-GP to refine, i.e. to make more realistic the synthesized images on the basis of 3D models of the hands. We expect that thanks to further extensions of our GAN it will be possible to achieve more realistic data, as well as to generate more data, thanks to which rational premises for classifying color images will arise.

ACKNOWLEDGMENTS

This work was supported by Polish National Science Center (NCN) under research grants 2014/15/B/ST6/02808 and 2017/27/B/ST6/01743.

REFERENCES

[1] S. S. Rautaray and A. Agrawal, “Vision based hand gesture recognition for human computer interaction: A survey,” Artif. Intell. Rev. **43**(1), 1–54 (2015).

[2] K. M. Sagayam and D. J. Hemanth, “Hand posture and gesture recognition techniques for virtual reality applications: a survey,” Virtual Reality **21**(2), 91–107 (2017).

[3] F. Chen, Q. Zhong, F. Cannella, K. Sekiyama, and T. Fukuda, “Hand gesture modeling and recognition for human and robot interactive assembly using hidden markov models,” Int. J. of Advanced Robotic Systems **12**(4), 48 (2015).

[4] S. Patil, D. K. Dennis, C. Pabbaraju, R. Deshmukh, H. Simhadri, M. Varma, and P. Jain, “GesturePod: Programmable gesture recognition for augmenting assistive devices, 2018,” tech. rep. (May 2018).

[5] E. Ohn-Bar and M. M. Trivedi, “Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations,” IEEE Trans. on Intelligent Transportation Systems **15**(6), 2368–2377 (2014).

[6] P. Pisharady and M. Saerbeck, “Recent methods and databases in vision-based hand gesture recognition,” Comput. Vis. Image Underst. **141**, 152–165 (2015).

[7] O. K. Oyedotun and A. Khashman, “Deep learning in vision-based static hand gesture recognition,” Neural Computing and Applications , 1–11 (2016).

[8] J. Tompson, M. Stein, Y. LeCun, and K. Perlin, “Real-time continuous pose recovery of human hands using convolutional networks,” ACM Trans. Graph. **33**(5) (2014).

[9] J. Nagi and F. Ducatelle, et al., “Max-pooling convolutional neural networks for vision-based hand gesture recognition,” in IEEE ICSIP, 342–347 (2011).

[10] P. Barros, S. Magg, C. Weber, and S. Wermter, A Multichannel Convolutional Neural Network for Hand Posture Recognition, 403–410, Springer, Cham (2014).

[11] O. Koller, H. Ney, and R. Bowden, "Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled," in IEEE Conf. on Comp. Vision and Pattern Rec., 3793–3802 (2016).

[12] B. Kwolek and S. Sako, "Learning siamese features for finger spelling recognition," in Advanced Concepts for Intelligent Vision Systems, 225–236, Springer Int. Publ. (2017).

[13] Y. Tabata and T. Kuroda, "Finger spelling recognition using distinctive features of hand shape," in Int. Conf. on Disability, Virtual Reality and Associated Technologies with Art Abilitation, 287–292 (2008).

[14] L. Kane and P. Khanna, "A framework for live and cross platform fingerspelling recognition using modified shape matrix variants on depth silhouettes," Comput. Vis. Image Underst. **141**, 138–151 (2015).

[15] H. Hosoe, S. Sako, and B. Kwolek, "Recognition of JSL finger spelling using convolutional neural networks," in Fifteenth IAPR Int. Conf. on Machine Vision Applications (MVA), 85–88, IEEE, Nagoya, Japan (2017).

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. of the 25th Int. Conf. on Neural Information Processing Systems - Volume 1, NIPS'12, 1097–1105 (2012).

[17] A. Rozantsev, V. Lepetit, and P. Fua, "On rendering synthetic images for training an object detector," Comput. Vis. Image Underst. **137**, 24–37 (Aug. 2015).

[18] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Proc. of the 27th Int. Conf. on Neural Information Processing Systems - Volume 2, NIPS'14, 2672–2680, MIT Press, Cambridge, MA, USA (2014).

[19] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in Proc. of the 30th Int. Conf. on Neural Information Processing Systems, NIPS'16, 2234–2242 (2016).

[20] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in Proc. of the 34th Int. Conf. on Machine Learning, 214–223 (2017).

[21] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," CoRR **abs/1511.06434** (2015).

[22] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in Advances in Neural Information Processing Systems 30, 5767–5777 (2017).

[23] J. Ba, R. Kiros, and G. E. Hinton, "Layer normalization," CoRR **abs/1607.06450** (2016).

[24] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," (2017).

[25] M. Šarić, "Libhand: A library for hand articulation," (2011). Version 0.9.