Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

Gesture Recognition using Conditional Generative Adversarial Networks

Gladys Indri Putri[1)*], Handri Santoso[2)]
[1)2)] Universitas Pradita, Tangerang, Indonesia
[1)]gladys.indri@student.pradita.ac.id, [2)]handri.santoso@pradita.ac.id

Submitted : Apr 20, 2022 | **Accepted** : Apr 25, 2022 | **Published** : Apr 27, 2022

Abstract: Sign language is very useful for giving signs to communication partners, in this case signs are not only for people with disabilities, but can also be used by normal people. Even in children's or adult games, sign language is used as a language or a means of communication with one another. Recognition of sign language using a computer by doing several methods, because the computer does not recognize the image of a sign with a certain meaning. Therefore, it is necessary to train computers to recognize these signs. One of the fields that discusses gesture image recognition is the field of Computer Vision. Where the science of computer vision is able to process the image. In addition, in image processing, it is necessary to carry out deep learning processes such as Convolutional Neural Networks. In the Convolutional Neural Network algorithm, there are also many methods or architectures such as VGG16, VGG19, ResNet-50, DenseNet, Inception_V3 and many more. The use of the architecture is used in accordance with existing needs. Therefore, the choice of architecture will determine the model to be built or not to build from scratch, only transfer learning or pre-train. Pre-training is done by using the initial model and then using it only. Or do some training. The purpose of this study was to detect sign language using the Generative Adversarial Network (GAN). Actually, the Generative Adversarial Network method is widely used in making synthetic images, but this time the Generative Adversarial Network can also detect images from sign language.

Keywords: Sign language; Generative Adversarial Networks; Deep Learning, Convolutional Neural Network; Computer Vision

INTRODUCTION

Communication is the process of delivering information from one party to another. Deaf people communicate using sign language. Sign language that is often used in formal conditions is SIBI sign language (Indonesian Sign Language System). SIBI Sign Language was built by adopting the American Sign Language (ASL) sign language. and arranged by paying attention to the syntax of the Indonesia language. Sign language learning has used manuals, demonstration videos or direct demonstrations by the teacher during the teaching and learning process (Sholawati et al., 2018).

Currently the use of computer vision has been widely used in various fields. In the field of medicine, in the field of transportation, in the field of learning on campus or at school, computer vision has been widely used in helping work. In the medical field, computer vision is used to detect disease. In the field of transportation, computer vision is used to detect vehicle number plates or traffic density and transport management. One of the fields of learning, computer vision is used as a tool to recognize objects or signs or signals.

Sign Language has long been used as learning for the deaf, it is intended so that students can learn with an educator to be able to interact and communicate with others. Actually, the sign language method is better understood by normal people, in order to understand what is meant by deaf people. They normally understand, can think and communicate with others. That's why we should also be able to understand about the signals given to normal humans.

The use of the Deep Learning algorithm (Yu et al., 2022), (Gruschwitz et al., 2021) is able to recognize sign language, using deep learning (Bachute & Subhedar, 2021) and computer vision, then all signals can be detected by the computer. So that if someone who is deaf makes a movement, the computer can recognize what signs will be given to the computer, and the computer is able to translate it, so that normal humans can know what is meant by the deaf. This is still a one-way communication, where the deaf give a signal and normal

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

humans understand the will of the deaf. Two-way communication occurs when both parties understand each other and communicate with each other.

The problem that occurs today is how to detect games that are often played by children in finding or making choices to play games first. Or the game uses sign language to find a winner in the game. Simple sign languages are scissors, paper, and rock. How does the computer recognize the sign language of the game? This study uses the Generative Adversarial Network method. Where the Generative Adversarial Network architecture has two main components, namely Generative and Discriminator. Generative as a synthetic image generate while Discriminator as Classification or detection. This reserach aims to detect the image and determine the classification of the image using the Conditional Generative Adversarial Network.

LITERATURE REVIEW

A literature review used is to look for shortcomings in previous studies as shown in table 1. This is not to find fault, the goal is to complement the shortcomings of previous studies..

Table 1, Previous research.

Author	Topic	Advantage	Disadvantage
(Sholawati et al., 2018)	Development of Sibi Alphabet Sign Language Recognition Application Using Convolutional Neural Network (Cnn) Method	The discussion about SIBI uses Convolutional Neuran Networks, and the resulting model is quite good.	We recommend comparing using at least 2 architectures, and will be able to see the comparison if done with 2 architectures. Unless you use a Generative Adversarial Network, you only use 1 architecture, because in Generative Adversarial Networks there are at least 2 architectures.
(Fadillah et al., 2021)	Augmentation Data to Overcome Data Limitations in the Indonesian Sign Language Translator Model (BISINDO)	Discussion of data augmentation to overcome the problem of lack of datasets. This is very nice to do. So that the resulting model is not overfitting or lacking the image dataset for the SIBI dataset	The drawback for making datasets is also using a deep learning algorithm using a Generative Adversarial Network, because this algorithm is very good at generating synthetic image datasets.
(Susanty et al., 2021)	Indonesian Sign Language Translator Model (BISINDO) Using Transfer Learning Approach	Discussion BISINDO uses transfer learning using Convolutional Neural Network.	Disadvantages of not using other algorithms such as Generative Adversarial Networks. This is very useful because it will produce a pretty good model if you use GAN.
(Damatraseta et al., 2021)	Real-time BISINDO Hand Gesture Detection and Recognition With Deep Learning CNN	The discussion about BISINDO with Hand Gesture Detection, using Convolutional Neural Networks is good	Disadvantages of not using other algorithms such as Generative Adversarial Networks. This is very useful because it will produce a pretty good model if you use GAN.
(Sindarto et al., 2022)	Indonesian Language Sign System Image Classification (SIBI) with Convolutional	Discussion of the classification of Indonesian Language Sign System Image,	Disadvantages of not using other algorithms such as Generative Adversarial Networks.

*name of corresponding author

	Neural Network Method on Android-based Software	using the Convolutional Neural Network Method is quite good	This is very useful because it will produce a pretty good model if you use GAN

Previous studies have been very good, but there is still a possibility of a gap, namely not yet trying to use the Generative Adversarial Network algorithm. State-of-the-art, in this study using the Generative Adversarial Network algorithm as a classification or detection of objects. Because the GAN algorithm is good for detecting or classifying images or objects.

METHOD

This study uses Conditional Generative Adversarial Networks (Liu et al., 2021), where this algorithm can solve the problem of detecting the sign language used. In general, the Generative Adversarial Network algorithm consists of Generator and Descriminator components. The generator functions to generate or generate a synthetic image. While the Discriminator performs a classification, in this case a fake image or a real image, which means that it is similar to a real image dataset.

CGAN, is a Generative Adversarial Network which is done conditioning for the output, which is given a label inside. This label aims to direct the resulting output. Different from GAN in general. The first proposed GAN uses random input, so that the synthetic image generated is random. This conditioning causes the Conditional Generative Adversarial Network (Awan et al., 2021) to produce output according to the label.

Dataset

The dataset used is a public dataset, this dataset is commonly used for deep learning algorithms such as the Convolutional Neural Network. This dataset was developed by Laurence Moroney, who has been working in Artificial Intelligent Advocacy at Google. As seen in figure 1, the Gesture Hand dataset is a public dataset. This dataset uses the CC license version 2.0, free and can be shared according to user needs, both for commercial and non-commercial. This dataset is divided into three parts, a dataset for training, a dataset for validation and a dataset for testing. For the dataset, the training dataset used is 372 images, with the distribution of 124 scissors images, 124 rock images, and 124 paper images. The dataset for validation is 33 images, namely rock, scissors and image paper.

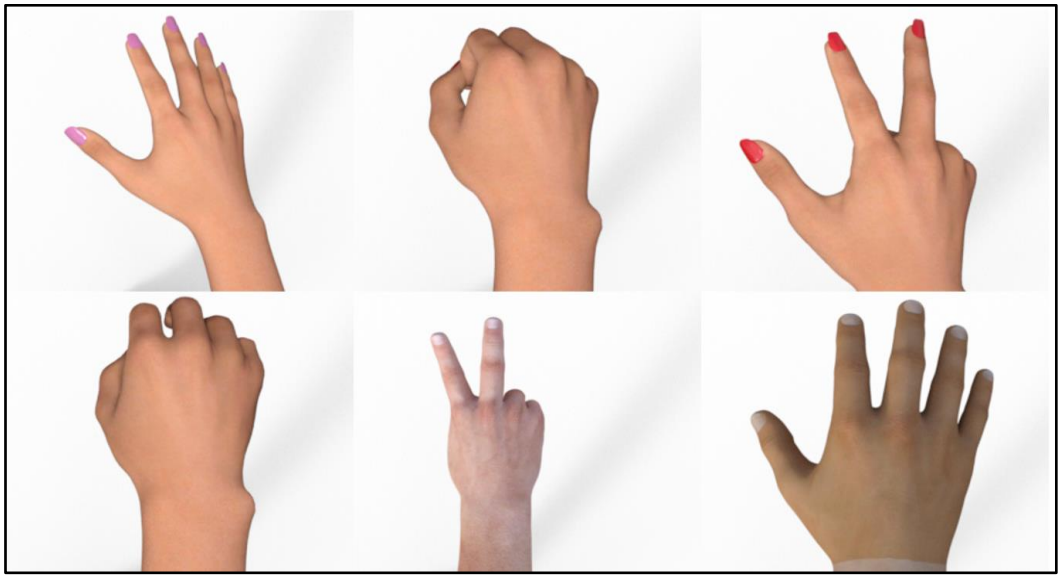

Fig 1. Dataset Gesture Hand
Source : https://laurencemoroney.com/

Conditional Generative Adversarial Networks

Conditional Generative Adversarial Networks (CGAN) in figure 2, shown to have two inputs, namely z as latent space and y fake as label, in the input there is a difference where GAN only has one input, namely latent space (z). Next z and y fake enter the Generator and produce X fake. In the Discriminator there are three inputs,

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

namely X real as Real image data, y real as Label and X fake from the generator. Furthermore, the three inputs enter the Discriminator and produce y as output.

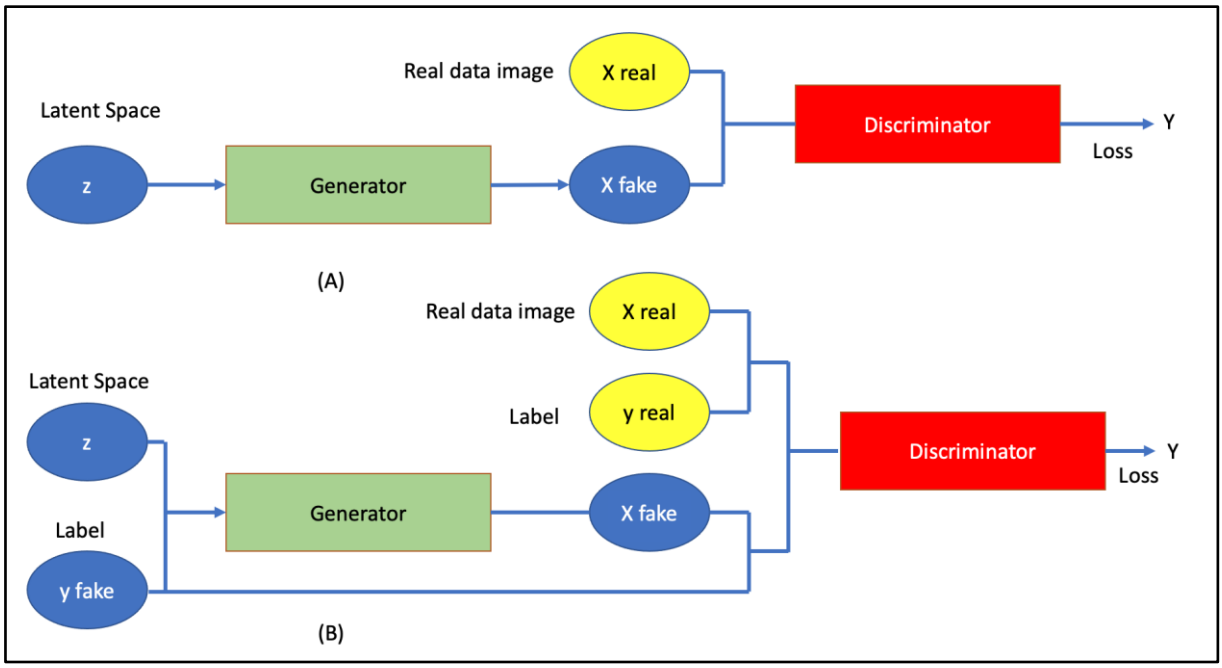

Fig. 2 Architecture Generator Adversarial Networks (A), Conditional Generative Adversarial Network (B)
Source : (Mirza & Osindero, 2014)

On the figure 2, The generative adversarial net can be extended into a conditional model if the generator and discriminator are conditioned on some additional information y. Where the value of y can be any additional information, such as class labels or data from other modalities. We can do the conditioning by inputting the data y into the discriminator function and into the generator function as an additional input layer. In the generator function, the previous input noise pz(z), and y are combined into a shared hidden representation, and the adversarial training framework allows considerable flexibility in the In discriminator x and y are presented as inputs and the discriminatory function is realized again by the Multi Layer Perceptron (Mirza & Osindero, 2014).

$$\min_G \max_D V(D, G) = Ex{\sim}pdata(x)\,[\log D(x|y)] + Ez{\sim}pz(z)\,[\log(1 - D(G(z|y)))] \quad (1)$$

Adversarial nets have the advantage that they use Markov chains and are never required, only back propagation is used to obtain gradients and no inference is required during learning, and a wide variety of factors and interactions can be easily entered into the model. Furthermore, as shown in (I. Goodfellow et al., 2014), it can yield sophisticated log likelihood estimates on realistic samples. In an unconditioned generative model, there is no control on the mode of the data generated. However, by conditioning the model on additional information, it is possible to direct the data creation process. The conditioning can be based on the class label, on some part of the data for inpainting such as (I. J. Goodfellow et al., 2013), or even on data from different modalities.

To explore the model generated by CGAN and expand the range of work carried out by (Isola et al., 2019), where the loss function L1 loss 1 is added to the CGAN (Mishra & Herrmann, 2021), (Xia et al., 2021) hostility loss for the network generator. Replaces the CGAN hostility loss with the sarial loss function used by one of the advanced GANs on models such as the WGAN, WGAN-GP, and lsGAN. Use a variety of traditional loss functions and add them to the loss adversary for the generator network such as L1, structure, gradient, KLD, content, and softmax loss functions. These models have been evaluated on a “facade” dataset trained over 200 epochs. Peak-signal-to-noise-ratio (PSNR), Structural Similarity Index (SSIM), Universal Quality Index (UQI), and Visual Fidelity Information. (VIF) was used to evaluate the results to decide the best combination of the adversarial loss and traditional loss function according to the CGAN architecture.

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

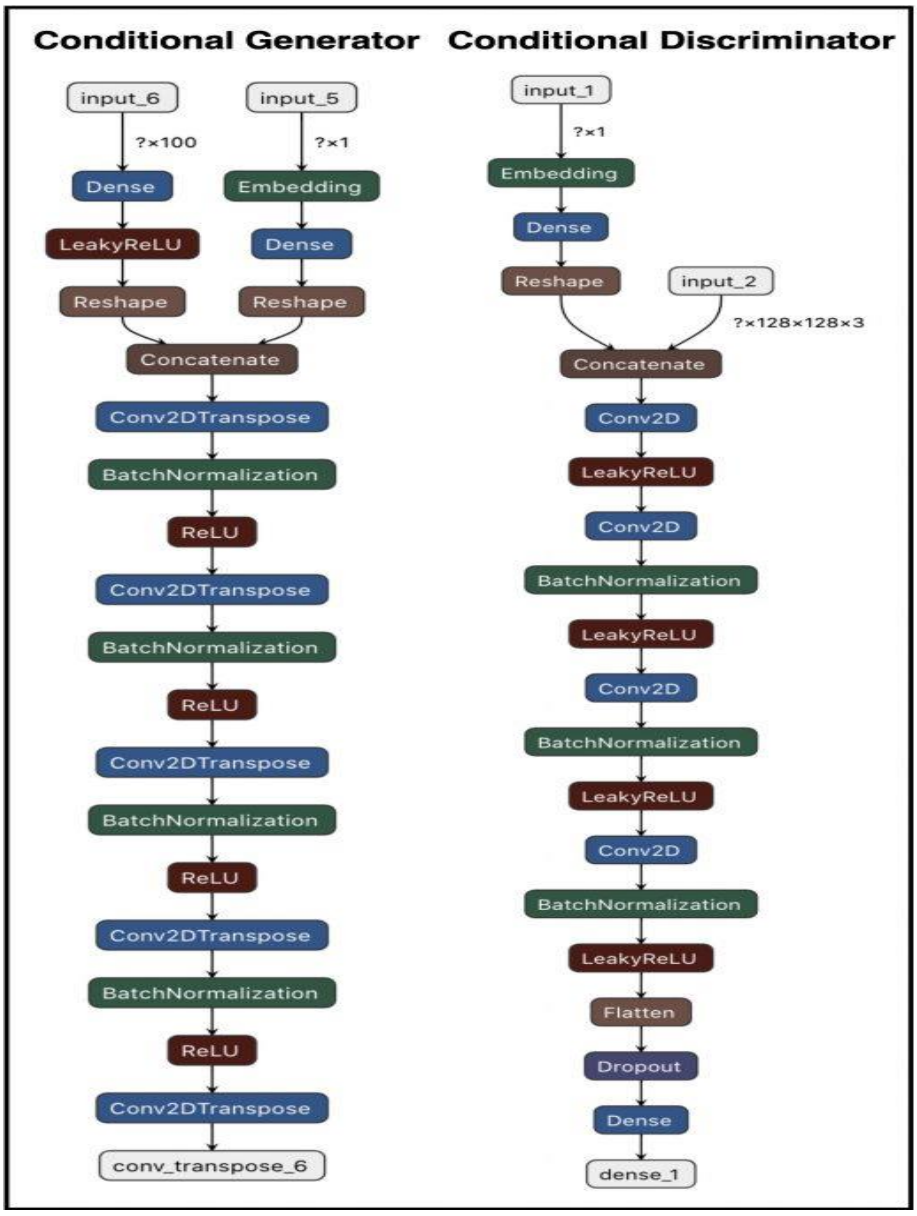

Fig. 3 Architecture Conditional Generator and Condotional Discriminator
Source : (Mirza & Osindero, 2014)

The Conditional Generator, in figure 3, performs Dense, leakyReLU, Reshape. Then merge or concate from the Embeeding, Dense, Reshape process. The result is Conv2DTranspose, then BatchNormalization and ReLU are performed. And so on, done up to 3 times, then do Conv2DTranspose, Conv_transpose. Conditional Discriminator, embedding, dense, reshape is done concate from input_2. After that, Conv2D, LeakyReLU, Conv2D, BatchNormalization were done. After that LeakyReLU, Conv2D, BatchNormalization, LeakyReLU, Conv2D, BatchNormalization, Leaky ReLU, Flatten, dropout, dense after that last dense_1.

The generator has two inputs, namely image data which is inputted as real data, while label input which performs conditions to perform useful conditioning as output. Therefore the Conditional Generative Adversarian Network is able to control the output. While the Generative Adversarial Network cannot condition the output. It is this factor that distinguishes the Conditional Generative Adversarial Network, which differs in terms of output, and the Generative Adversarial Network.

RESULT

The following are the detection results using Conditional Generative Adversarial Networks. In figure 4, the implementation of Conditional GAN uses Pytorch. The result used looks darker than the dataset, while in figure

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

5, using tensorfloe, the result is lighter. In the results there are differences, where tensorflow results are better than using Pytorch.

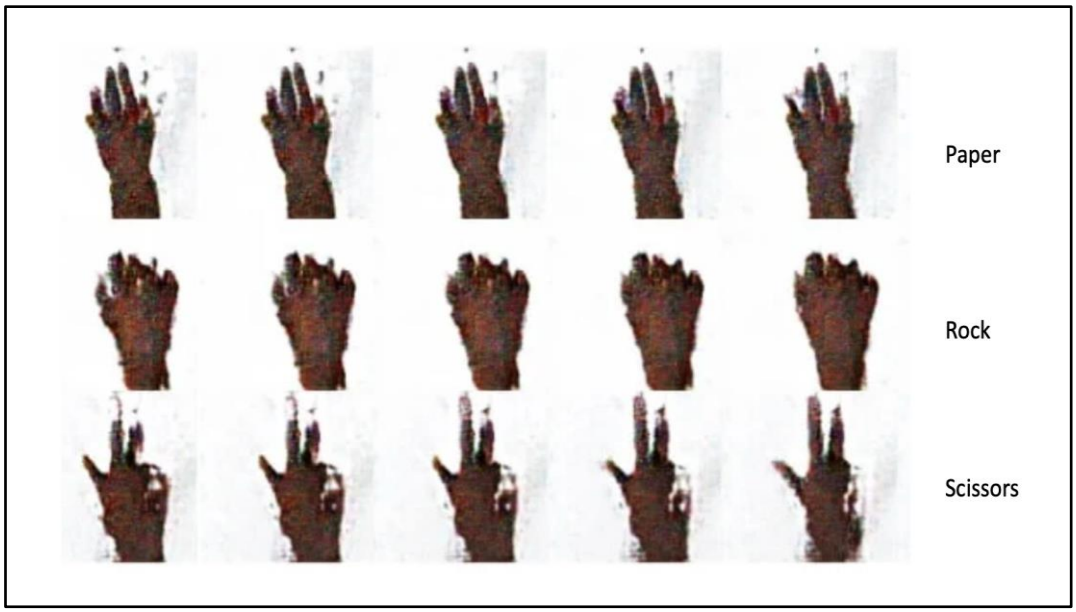

Fig. 4 Result Conditional Generative Adversarial Networks menggunakan Pytorch
Source : researcher property

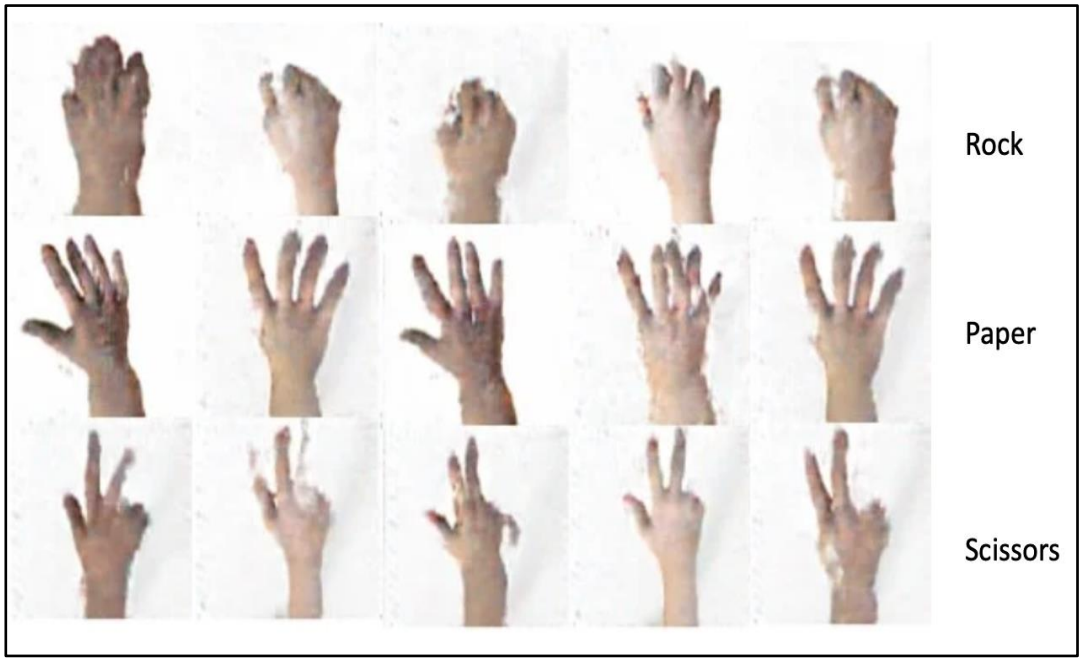

Fig. 5 Result Conditional Generative Adversarial Networks menggunakan Tensorflow
Source : researcher property

DISCUSSIONS

In figure 4 and figure 5, we use a different framework, namely Tensorflow and Pytorch. In theory there is a difference where tensorflow is superior when compared to pytorch. This causes the resulting image to be better. Therefore, the use of Tensorflow is more popular when compared to the use of Pytorch. Here are the appearances of the Conditional Generator and Conditional Discriminator.

Script Conditional Generator

```
class Generator(nn.Module):
```

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 6, Number 2, April 2022
DOI : https://doi.org/10.33395/sinkron.v7i2.11397
e-ISSN : 2541-2019
p-ISSN : 2541-044X

```
  def __init__(self):
    super(Generator, self).__init__()
    self.label_conditioned_generator =
    nn.Sequential(nn.Embedding(n_classes, embedding_dim), nn.Linear(embedding_dim, 16))
    self.latent = nn.Sequential(nn.Linear(latent_dim, 4*4*512), nn.LeakyReLU(0.2, inplace=True))
    self.model = nn.Sequential(nn.ConvTranspose2d(513, 64*8, 4, 2, 1, bias=False),
                nn.BatchNorm2d(64*8, momentum=0.1,  eps=0.8),
                nn.ReLU(True),
                nn.ConvTranspose2d(64*8, 64*4, 4, 2, 1,bias=False),
                nn.BatchNorm2d(64*4, momentum=0.1,  eps=0.8),
                nn.ReLU(True),
                nn.ConvTranspose2d(64*4, 64*2, 4, 2, 1,bias=False),
                nn.BatchNorm2d(64*2, momentum=0.1,  eps=0.8),
                nn.ReLU(True),
                nn.ConvTranspose2d(64*2, 64*1, 4, 2, 1,bias=False),
                nn.BatchNorm2d(64*1, momentum=0.1,  eps=0.8),
                nn.ReLU(True),
                nn.ConvTranspose2d(64*1, 3, 4, 2, 1, bias=False),
                nn.Tanh())

  def forward(self, inputs):
    noise_vector, label = inputs
    label_output = self.label_conditioned_generator(label)
    label_output = label_output.view(-1, 1, 4, 4)
    latent_output = self.latent(noise_vector)
    latent_output = latent_output.view(-1, 512,4,4)
    concat = torch.cat((latent_output, label_output), dim=1)
    image = self.model(concat)
    #print(image.size())
    return image
```

Script Conditional Discriminator

```
class Discriminator(nn.Module):
  def __init__(self):
    super(Discriminator, self).__init__()
    self.label_condition_disc =
    nn.Sequential(nn.Embedding(n_classes, embedding_dim),
                nn.Linear(embedding_dim, 3*128*128))

    self.model = nn.Sequential(nn.Conv2d(6, 64, 4, 2, 1, bias=False),
                nn.LeakyReLU(0.2, inplace=True),
                nn.Conv2d(64, 64*2, 4, 3, 2, bias=False),
                nn.BatchNorm2d(64*2, momentum=0.1,  eps=0.8),
                nn.LeakyReLU(0.2, inplace=True),
                nn.Conv2d(64*2, 64*4, 4, 3,2, bias=False),
                nn.BatchNorm2d(64*4, momentum=0.1,  eps=0.8),
                nn.LeakyReLU(0.2, inplace=True),
                nn.Conv2d(64*4, 64*8, 4, 3, 2, bias=False),
                nn.BatchNorm2d(64*8, momentum=0.1,  eps=0.8),
                nn.LeakyReLU(0.2, inplace=True),
                nn.Flatten(),
                nn.Dropout(0.4),
                nn.Linear(4608, 1),
                nn.Sigmoid()
               )

  def forward(self, inputs):
    img, label = inputs
    label_output = self.label_condition_disc(label)
    label_output = label_output.view(-1, 3, 128, 128)
    concat = torch.cat((img, label_output), dim=1)
    #print(concat.size())
    output = self.model(concat)
    return output
```

Script Training Conditional Generative Adversarial Networks

```
num_epochs = 200
for epoch in range(1, num_epochs+1):

  D_loss_list, G_loss_list = [], []
```

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

```
for index, (real_images, labels) in enumerate(train_loader):
    D_optimizer.zero_grad()
    real_images = real_images.to(device)
    labels = labels.to(device)
    labels = labels.unsqueeze(1).long()

    real_target = Variable(torch.ones(real_images.size(0), 1).to(device))
    fake_target = Variable(torch.zeros(real_images.size(0), 1).to(device))

    D_real_loss = discriminator_loss(discriminator((real_images, labels)), real_target)
    # print(discriminator(real_images))
    #D_real_loss.backward()

    noise_vector = torch.randn(real_images.size(0), latent_dim, device=device)
    noise_vector = noise_vector.to(device)

    generated_image = generator((noise_vector, labels))
    output = discriminator((generated_image.detach(), labels))
    D_fake_loss = discriminator_loss(output,  fake_target)

    # train with fake
    #D_fake_loss.backward()

    D_total_loss = (D_real_loss + D_fake_loss) / 2
    D_loss_list.append(D_total_loss)

    D_total_loss.backward()
    D_optimizer.step()

    # Train generator with real labels
    G_optimizer.zero_grad()
    G_loss = generator_loss(discriminator((generated_image, labels)), real_target)
    G_loss_list.append(G_loss)

    G_loss.backward()
    G_optimizer.step()
```

CONCLUSION

From the explanation above, it can be concluded in the Conditional Generative Adversarial research, that this method is very different from the Generative Adversarial Network, where the output of GAN is random, while CGAN, has output that can be conditioned because it has a label on the input. Using the Tensorflow framework with Pytorch there are differences in the results or output, where the output is better. The problem can be answered, that the recognition of sign language is limited to the scissors, paper and stone dataset, that the model generated by the Conditional Generative Adversarial Network can recognize images from sign language. Research using Conditional Generative Adeverasrial Network can be used as a sign language image identifier.

SUGGESTION

This research can still be continued by combining the Conditional Generative Adversarial Network method with the Super Resolution Generative Adversarial Network to produce a clearer and better image. It is hoped that later normal human voice input can be developed to translate into text and translate to image, or text to image translation using the Generative Adversarial Network.

REFERENCES

Awan, S. E., Bennamoun, M., Sohel, F., Sanfilippo, F., & Dwivedi, G. (2021). Imputation of missing data with class imbalance using conditional generative adversarial networks. *Neurocomputing*, *453*, 164–171. https://doi.org/10.1016/j.neucom.2021.04.010

Bachute, M. R., & Subhedar, J. M. (2021). Autonomous Driving Architectures: Insights of Machine Learning and Deep Learning Algorithms. *Machine Learning with Applications*, *6*(March), 100164. https://doi.org/10.1016/j.mlwa.2021.100164

Damatraseta, F., Novariany, R., & Ridhani, M. A. (2021). Real-time BISINDO Hand Gesture Detection and Recognition with Deep Learning CNN. *Jurnal Informatika Kesatuan*, *1*(1), 71–80. https://doi.org/10.37641/jikes.v1i1.774

Fadillah, R. Z., Irawan, A., Susanty, M., & Artikel, I. (2021). Data Augmentasi Untuk Mengatasi Keterbatasan Data Pada Model Penerjemah Bahasa Isyarat Indonesia (BISINDO). *Jurnal Informatika*, *8*(2), 208–214. https://ejournal.bsi.ac.id/ejurnal/index.php/ji/article/view/10768

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Goodfellow, I. J., Mirza, M., Courville, A., & Bengio, Y. (2013). Multi-prediction deep Boltzmann machines. *Advances in Neural Information Processing Systems*, 1–9.

Goodfellow, I., Mehdi, M., Bing, X., & David, W. (2014). *Generative Adversarial Nets*.

Gruschwitz, P., Grunz, J.-P., Kuhl, P. J., Kosmala, A., Bley, T. A., Petritsch, B., & Heidenreich, J. F. (2021). Performance testing of a novel deep learning algorithm for the detection of intracranial hemorrhage and first trial under clinical conditions. *Neuroscience Informatics*, *1*(1–2), 100005. https://doi.org/10.1016/j.neuri.2021.100005

Isola, P., Jun-Yan, Z., Tinghui, Z., & Alexei, E. (2019). Image-to-Image Translation with Conditional Adversarial Networks. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11129 LNCS*, 601–618. https://doi.org/10.1007/978-3-030-11009-3_37

Liu, J., Zhu, G., & Yin, J. (2021). Joint color spectrum and conditional generative adversarial network processing for underwater acoustic source ranging. *Applied Acoustics*, *182*, 108244. https://doi.org/10.1016/j.apacoust.2021.108244

Mirza, M., & Osindero, S. (2014). *Conditional Generative Adversarial Nets*. 1–7. http://arxiv.org/abs/1411.1784

Mishra, P., & Herrmann, I. (2021). GAN meets chemometrics: Segmenting spectral images with pixel2pixel image translation with conditional generative adversarial networks. *Chemometrics and Intelligent Laboratory Systems*, *215*(April), 104362. https://doi.org/10.1016/j.chemolab.2021.104362

Sholawati, M., Auliasari, K., & Ariwibisono, F. X. (2018). *Pengembangan Aplikasi Pengenalan Bahasa Isyarat Abjad Sibi Menggunakan Metode Convolutional Neural Network (Cnn)*.

Sindarto, S. S., Ratnawati, D. E., & Arwani, I. (2022). *Klasifikasi Citra Sistem Isyarat Bahasa Indonesia (SIBI) dengan Metode Convolutional Neural Network pada Perangkat Lunak berbasis Android*. *6*(5), 2129–2138.

Susanty, M., Fadillah, R. Z., & Irawan, A. (2021). Model Penerjemah Bahasa Isyarat Indonesia (BISINDO) Menggunakan Pendekatan Transfer Learning. *Petir*, *15*(1), 1–9. https://doi.org/10.33322/petir.v15i1.1289

Xia, Y., Ravikumar, N., Greenwood, J. P., Neubauer, S., Petersen, S. E., & Frangi, A. F. (2021). Super-Resolution of Cardiac MR Cine Imaging using Conditional GANs and Unsupervised Transfer Learning. *Medical Image Analysis*, *71*, 102037. https://doi.org/10.1016/j.media.2021.102037

Yu, L., Xue, L., Liu, F., Li, Y., Jing, R., & Luo, J. (2022). The applications of deep learning algorithms on in silico druggable proteins identification. *Journal of Advanced Research*, *xxxx*. https://doi.org/10.1016/j.jare.2022.01.009

*name of corresponding author