Biologically inspired artificial intelligence

Project report from topic:

Cyclic GAN - How to travel throughout time.

Lecturer: Dr inż. Grzegorz Baron

Date: 10.06.2023

Section:

Michał Ławinski
Aleksandra Lewandowska

Short introduction presenting the project topic:

The goal of our project is to create a solution that allows us to convert images of people. The targeted transformations are from youngster to adult and vice versa (from adult to youngster).

Four models are trained: Two Generator and two Discriminator:

1. Generator is used for image generation which depicts the target object based on the reference image. For instance, image including an apple is a reference image, and with use of the trained generator will be converted to an orange or a car of one brand will be converted to a car of the other brand.

2. Discriminator is used as a classifier, outputting whether an image is a fake or a real example. Afterwards based on the results we can backpropagate the errors and update weights of our models.

3. When Nash equilibrium is reached, we define it to be the place in which both models converged and we say that training process of Generator and Discriminator are over.

Analysis of the task:

1. Possible approaches to solve the problem with its pros/cons, detailed description of selected methodology:

   Our problem is based on image-to-image transformation. A generally accepted solution to this problem is a GAN (Generative Adversarial Neural Network) which in our case is the only logical approach to object to object (within the image) translation. What is more, GANs achieve extraordinary results, therefore that was an obvious choice for us to make.

   During research we have not found any reasonable alternatives, which could possibly solve our problem. The only thing we came across are Autoencoder, which can be used for data summarizing and data/image generation, however that was not a good approach for our project.

2. Presentation of possible/available datasets and detailed description of the chosen ones:

   For our project we have used a dataset available at:
   http://www.ivl.disco.unimib.it/activities/large-age-gap-face-verification/

   That is a large age-gap face verification, which contains images of young actors, and their equivalence at the older age.

   Alternatively, we wanted to take advantage of Kaggle dataset:
   https://www.kaggle.com/datasets/abhishekyana/young2old-dataset

   However, the issue with this dataset, was the fact, that only 200 images per age category (meaning young and old) were available. Therefore, the model has not been performing as expected.

3. Analysis of available tools/frameworks/libraries suitable for task solution; detailed presentation of selected tools/approach:

   In our case, we could use either PyTorch or Keras which are very popular choices for neural network development. However, programming API of Keras allowed us to create GANs with ease, as it allows for simple and complex neural network data flow graph generation.
   What is more, Keras with TensorFlow are more commonly commercially accepted, in comparison to PyTorch which usually is used in scientific research.

Internal and external specification of the software solution:

1. Classes/objects/methods/scripts/functions etc. – depending on the approach to the project solution + data structures (a, b):

   def prepare_daset():
   pass # prepares dataset structure for batch loading

   First main method that we had to implement was the method which would allow us for creation of such a file structure which could easily allow us for domain to domain image batch generation. By domain to domain, we mean, generation of two sets domain A – youngsters, domain B – older people.

   Then we had to define a **DataLoader** class which allows us for loading image batches during training. There is an additional method defined called **load_data** which is used to results depiction during different stages of the training.

   Then we defined a **CycleGAN** class, which represents our complex cycle GAN model. There we create two instances of generators and discriminators with their structure. What is more, we define the transformations which are used for our network's training:

   a) domain A to domain B translation (youngster to adult)
   b) domain B to domain A translation (adult to youngster)
   c) domain A to domain B to domain A translation (identity A translation)
   d) domain B to domain A to domain B translation (identity B translation)
   e) reconstructed domain A from domain B
   f) reconstructed domain B from domain A

   And compilation of loss functions. All those functionalities are implemented in __init__ method.

   The class is equipped into a few more methods. Although the key methods are:

   a) **build_generator** which creates the model representing our generator
   b) **build_discriminator**
   c) **train**

   In the train method we go through all iterations. In each iteration we draw batches on images (going through all images). So basically, we start creating fake images, meaning generating image from domain B based on domain A, and vice versa, image from domain A based on domain B. Then based on our discriminators we calculate how well

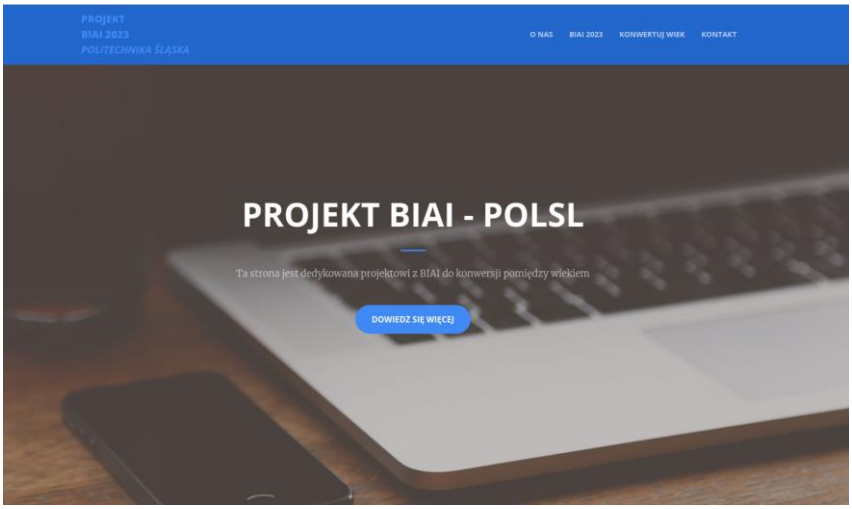
generators fooled the discriminators. And then we calculate losses on real images, and take the average, for each domain A to B and domain B to A, and then we take the average of both. So, basically, now we trained the discriminators, A and B.

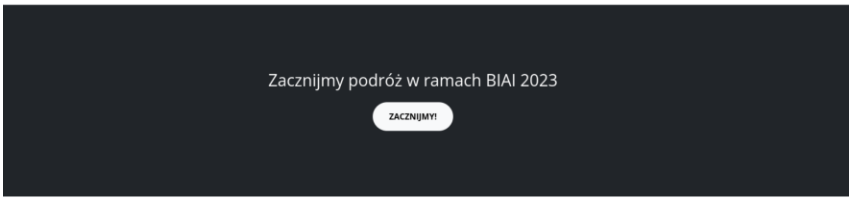
Then we proceed to training the generators, and for such purpose we use the combined model, which takes advantage of 6 transformations defined above, for which loss functions have been defined.

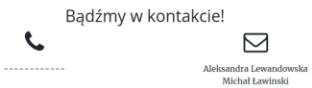
And at the end we simply display result images, showing how well the network generated following transformations domain A -> domain B -> domain A, and domain B, domain A -> domain B, where domain A – youngster, domain B – adult.

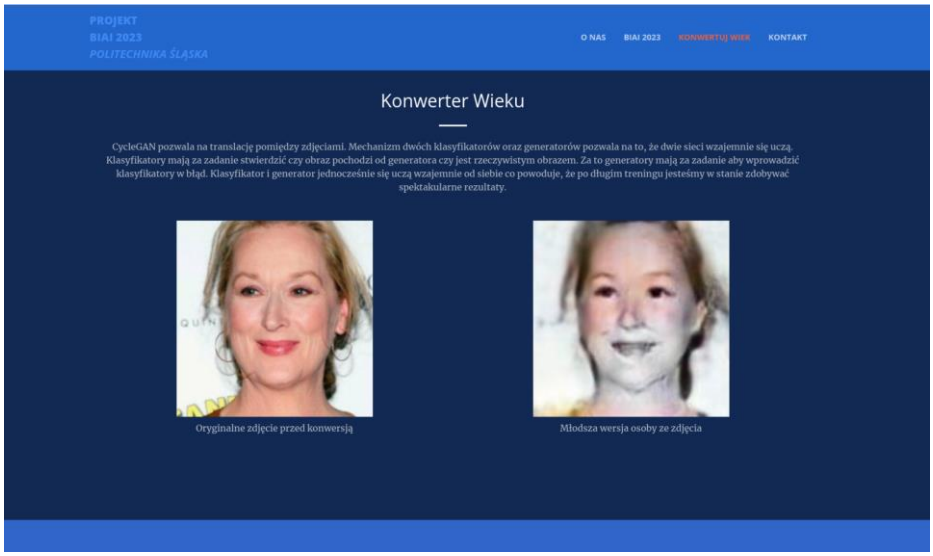
At the end we save our models, both discriminators and generators, which can be used in practice, in our web app, which allows to take selfie, and make ourselves older or younger.

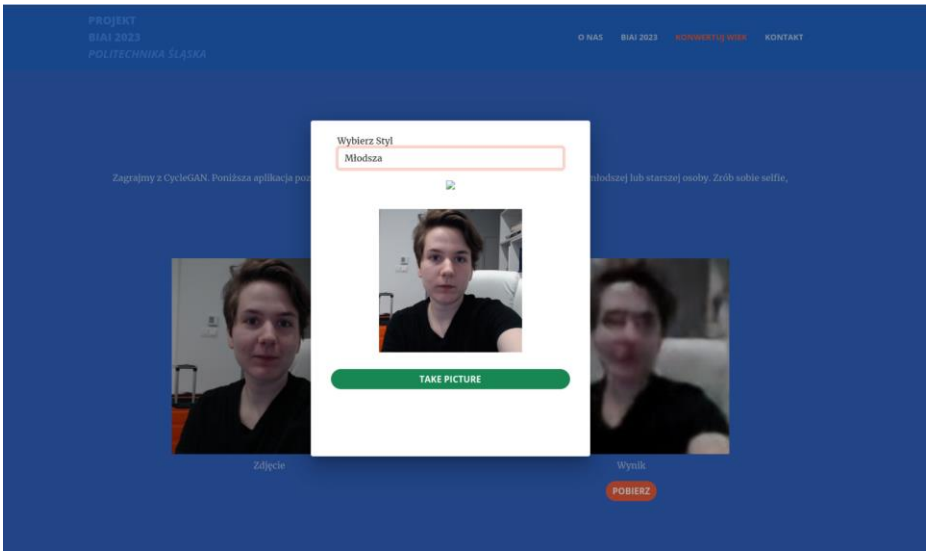
2. User interface / GUI/console

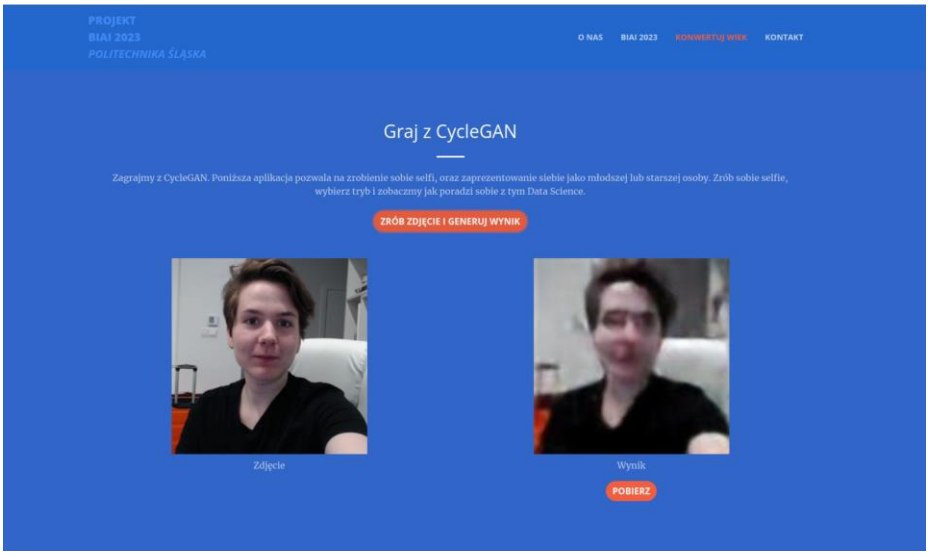







PROJEKT
BIAI 2023
POLITECHNIKA ŚLĄSKA
O NAS
BIAI 2023
KONWERTUJ WIEK
KONTAKT
Konwerter Wieku
CycleGAN pozwala na translację pomiędzy zdjęciami. Mechanizm dwóch klasyfikatorów oraz generatorów pozwala na to, że dwie sieci wzajemnie się uczą. Klasyfikatory mają za zadanie stwierdzić czy obraz pochodzi od generatora czy jest rzeczywistym obrazem. Za to generatory mają za zadanie aby wprowadzić klasyfikatory w błąd. Klasyfikator i generator jednocześnie się uczą wzajemnie od siebie co powoduje, że po długim treningu jesteśmy w stanie zdobywać spektakularne rezultaty.
Oryginalne zdjęcie przed konwersją
Młodsza wersja osoby ze zdjęcia


PROJEKT
BIAI 2023
POLITECHNIKA ŚLĄSKA
O NAS
BIAI 2023
KONWERTUJ WIEK
KONTAKT
Wybierz Styl
Młodsza
TAKE PICTURE
Zdjęcie
Wynik
POBIERZ


PROJEKT
BIAI 2023
POLITECHNIKA ŚLĄSKA
O NAS
BIAI 2023
KONWERTUJ WIEK
KONTAKT
Graj z CycleGAN
Zagrajmy z CycleGAN. Poniższa aplikacja pozwala na zrobienie sobie selfi, oraz zaprezentowanie siebie jako młodszej lub starszej osoby. Zrób sobie selfie, wybierz tryb i zobaczmy jak poradzi sobie z tym Data Science.
ZRÓB ZDJĘCIE I GENERUJ WYNIK
Zdjęcie
Wynik
POBIERZ

Experiments:

1. Experimental background – description of experiments, what parameters/conditions were the subject of change and why, what results were observed and why; method of results presentation – description of diagrams/axes/values; if some raw results were processed, any calculations were performed, it must be explained:

   For our project we took advantage of well described cycle-GAN architectures which allow for object-to-object translation. Most of our work was focused on the fact of how well the model can be trained depending on the dataset that is used. Primary results achieved with the Kaggle dataset were quite disappointing and the model was not learning as good as expected.

   However, when we took advantage of a larger dataset, the results were significantly better. Other metrics that we have played with, were mostly epoch and batch count. During training, reading the image results after every few epochs we could notice with the dataset that was at hand, that the results were not getting better after about 400 epochs.

2. Presentation of experiments with detailed analysis and description of the results with partial conclusions possible to be drawn based on the presented part of experiments:

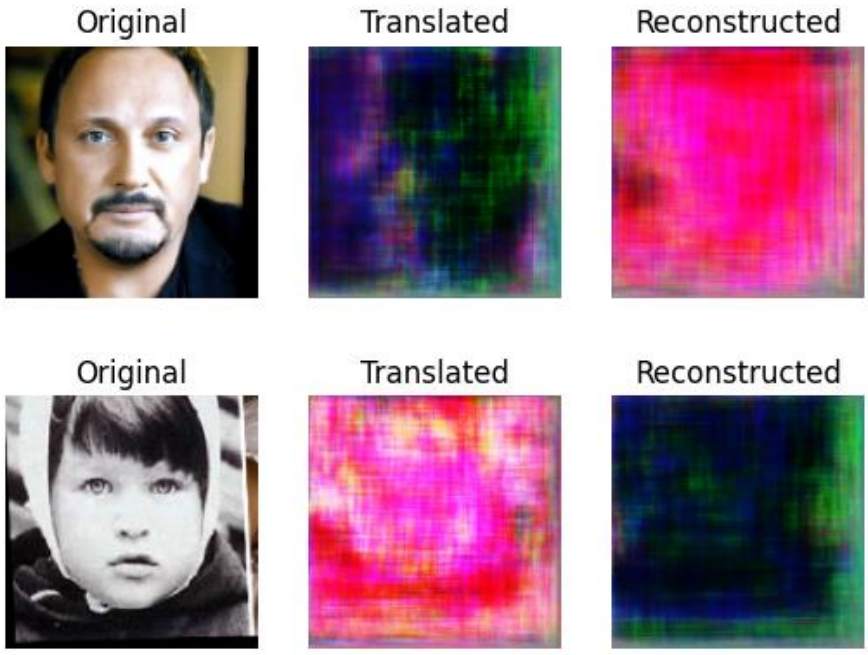


Fig. 1 Training results after few first epochs.

We can see that after few first epoch the network was not performing great, during new image generation we can see that although patter occur, the result image is far from expected output.

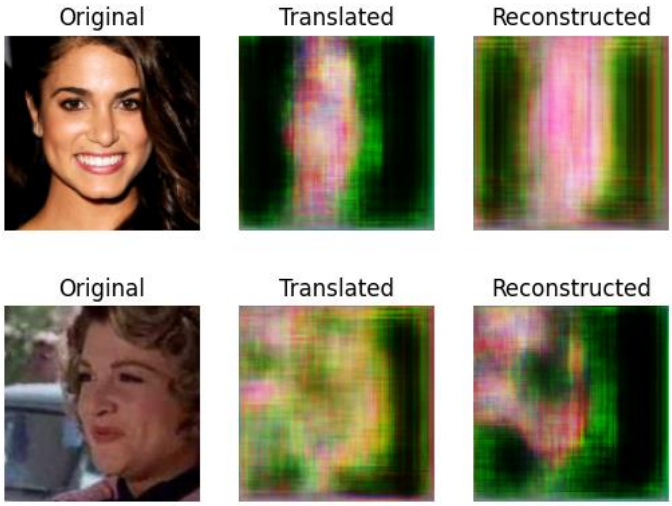


Fig. 2 Following next iterations results.

After a few more iterations we can see that faces and human shape emerge out of the images, and the neural network generates better results.

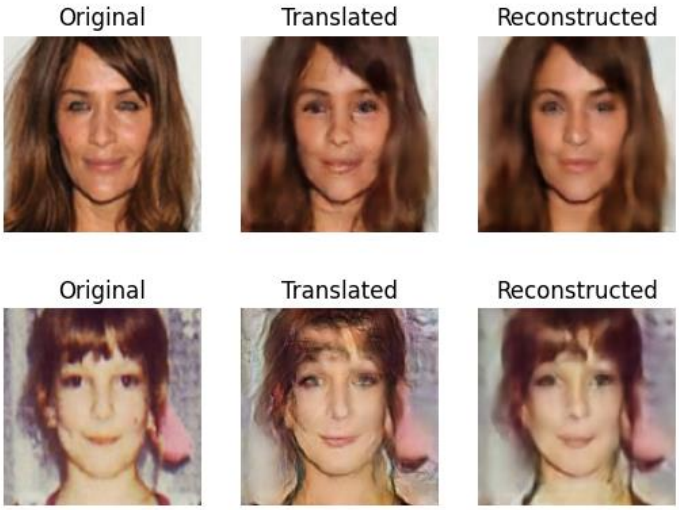


Fig. 3 The results after the full training

We can notice that after 400 epochs of training, the neural network generated acceptable results, although those results look like this on the training dataset. In the case of images not included in the training dataset, the results are very interesting, although not making as good an impression as the results above.

It is also worth noticing that the images are of the input/output sizes of (128, 128). Therefore, we can notice that images are very small.

Summary, overall conclusions, possible improvements, future work:

Summing up all our research and results, we can conclude that Generative Adversarial Neural Networks are a powerful tool, which makes a huge impression. Especially as a result can be extraordinary. Although the images that our network learned to generate are not perfect, and the results generated on the test images are significantly less impressive than images generated on the training dataset, we can see that these algorithms are powerful.

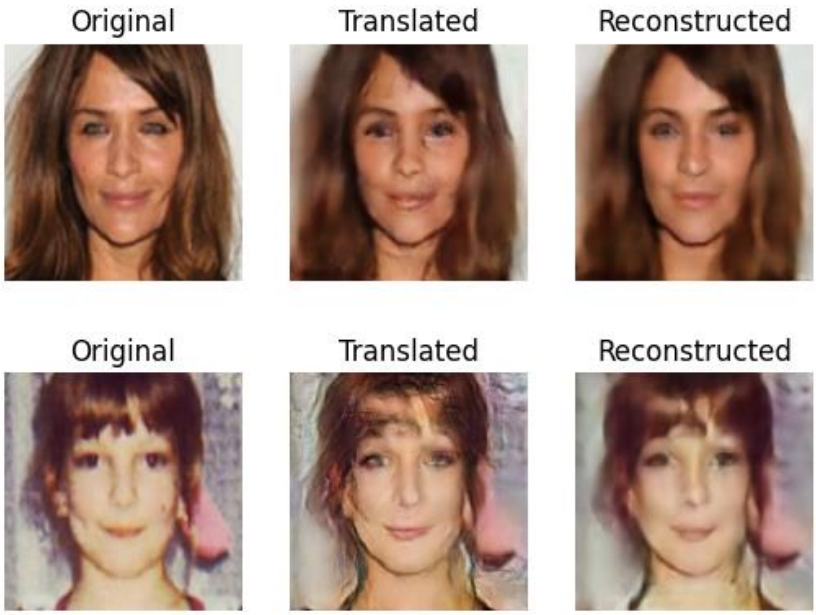


Fig.1 Results generated on the training dataset.

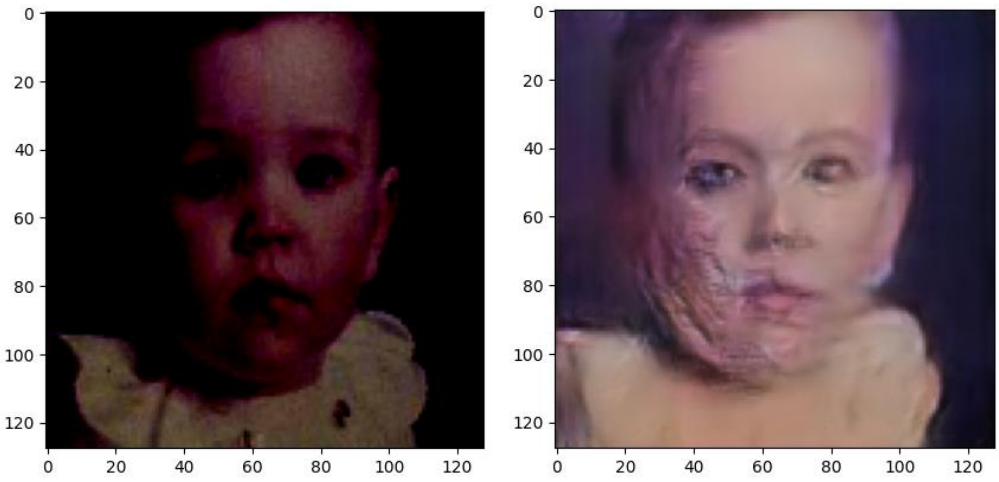


Fig.2 Result generated for image not included in training dataset.

Further work could be focused on creating better images with higher dimensionality.
We can try training the model on the larger datasets and play with the network's architecture and see whether we can achieve better results. The one problematic thing is that, such networks require lots of training and such training requires very good, and fast hardware (GPUs) which makes such research quite problematic, although with the access to such hardware it would be definitely interesting.

References – list of sources used during the work on the project:

- http://www.ivl.disco.unimib.it/activities/large-age-gap-face-verification/
- https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/
- https://www.ibm.com/topics/neural-networks
- https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e
- https://bios691-deep-learning-r.netlify.app/class/04-class/
- GANs in Action DEEP LEARNING WITH GENERATIVE ADVERSARIAL NETWORKS JAKUB LANGR VLADIMIR BOK

Link to the shared cloud/Google Drive/OneDrive/Git/etc. where all the files are placed:

https://github.com/olalew/BIAI