UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

Efficient Generation of Complex Data Distributions

Author:
Philipp KOFMAN

Supervisor:
Oles DOBOSEVYCH

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2020

Declaration of Authorship

I, Philipp KOFMAN, declare that this thesis titled, "Efficient Generation of Complex Data Distributions" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

Efficient Generation of Complex Data Distributions

by Philipp KOFMAN

Abstract

Currently, the active development of image processing methods requires large amounts of correctly labeled data. The lack of quality data makes it impossible to use various machine learning methods.

In case of limited possibilities for collecting real data, used methods for their synthetic generation. In practice, we can formulate the task of the high-quality generation of synthetic images as an efficient generation of complex data distributions, which is the object of study of this work.

Generating high-quality synthetic data is an expensive and complicated process in terms of existing methods. We can distinguish two main approaches that are used to generate synthetic data: image generation based on rendered 3-D scenes and the use of GANs for simple images. These methods have some drawbacks, such as a narrow range of applicability and insufficient distribution complexity of the obtained data. When using GANs to generate complex distributions, in practice, we face a visible increase in the complexity of the model architecture and training procedure.

A deep understanding of the real data complex distributions can be used to improve the quality of synthetic generation. Minimizing the differences in the real and synthetic data distributions can improve not only the generation process but also develop tools for solving the problem of data lack in the field of image processing.

Contents

List of Figures

List of Tables

List of Abbreviations

VAE	**V**ariational **A**utoencoder
GAN	**G**enerative **A**dversarial **N**etwork
Conv	**Conv**olution layer
FC	**F**ully **C**onnected
ROC	**R**eceiver **O**perating **C**haracteristic
PCA	**P**rincipal **C**omponent **A**nalysis
LR	**L**earning **R**ate
BN	**B**atch **N**ormalization

List of Symbols

μ	mean of a sample
σ	standard deviation of a sample
$N(\mu, \sigma)$	normal distribution

Chapter 1

Introduction

Expanding the capabilities of computer vision and deep learning opens up opportunities and approaches to solving many problems that previously remained unresolved. Many tasks that need to be solved remain beyond the reach of modern deep learning technologies - even though there is a large amount of manually annotated data.

Deep learning models do not have an understanding of the input, at least not in the human sense. People understand images based on their experience. Machine learning models do not have access to such experience, and therefore they cannot "understand" the input data in this way. By annotating a large number of training examples for models, we force them to learn a geometric transformation that brings data to human concepts for a specific set of examples, but this transformation is just a simplified outline of the original object model.

Deep learning models do not currently have a mechanism for learning abstractions through the direct definition of an object, but working with thousands, millions, or even billions of training examples solves this problem only partially (Marcus, 2018).

Data collection for such tasks is essential, but sometimes very difficult, especially in the case of rare classes of objects. We should note that for such amount of data, manual annotation is not the best decision since it requires a lot of resources and well-established markup strategies.

One way to solve this problem is to use artificially generated data. However, when using synthetic data, we may face the problem of a big jump in the complexity of choosing the architecture and methods for training the model. We can assume that for the model, there is a fundamental difference between real and generated data.

This study aims to compare the distributions of real and synthetic data, study the reasons for the increase of work complexity when using synthetics and how to eliminate it.

The main problem considered in this paper is the difficulty of generating high-quality synthetic data for their further use in deep learning models for image processing. So, the central objective is to identify the hidden differences between real and synthetic data for their high-quality generation. We highlight related objectives:

- Hypothesis confirmation of the presence of a statistically significant difference in the distributions of real and synthetic data
- Building a pipeline for image conversion
- Quality criterion selection for assessing the generated data

The objects of the study are four primary datasets: real photos collected from auto-recorders (Yu et al., 2018), generated pictures "SYNTHIA" transport routes (Hernandez-Juarez et al., 2017), real photos of dogs (Parkhi et al., 2012) and generated images of dogs using GANs (Goodfellow et al., 2014).

We assume that the identification of distinctive features in the distributions of real and synthetic data will help to avoid the difficulty of transferring the machine learning model between them.

The formal statement of the problem:

1. Conversion of images and their transformation into vector space using neural network methods
2. Construction of space and two presentations: from images to hidden space and vice versa
3. Analysis of distributions in a new hidden space and their investigation using statistical methods
4. Conducting transformations on data in hidden space to minimize differences
5. Display modified synthetic data into the image space
6. Selection of a formal criterion for assessing the quality of artificially generated data so that machine learning models in the field of computer vision containing synthetics in the training dataset show high quality working with test and validation samples of real data.

Chapter 2

Related Work

Since 2010, research has been conducted in the field of visual domain adaptation (Saenko et al., 2010), where the first approach to the problem was statistical methods. However, since 2014, neural network methods have gained considerable popularity (Torralba and Efros, 2011). Soon the lack of data became a related problem, which led to the growth of synthetics generation methods (Tzeng et al., 2014).

The need for synthetic data often arises in many tasks. An outstanding representative of such tasks is autonomous driving. In order to make high-precision classifiers of road markings, signs, cars and other vast volumes of qualitatively marked data are needed.

2.1 3D-based methods

To solve this problem, in 2016 was proposed the idea of generating a dataset based on the gaming world. The article Playing for Data: Ground Truth from Computer Games (Richter et al., 2016) used the GTA5 game world. The purpose of the work was to get markup from screenshots of the game. The main idea is to use an existing virtual 3-D world. However, the limitations of the game did not allow to obtain the complete markup necessary to solve the problem of autonomous driving.

At the same time, in 2016, using the same idea of the virtual world, the SYNTHIA dataset was generated in order to assist in semantic segmentation and the problems of understanding related scenes in the context of driving scenarios (Ros et al., 2016). The authors a bit changed the approach and created their virtual world using the Unity development platform. They built their virtual cities based on real city prototypes.

One of the main advantages was the ability to add natural events, such as time of day, rain, snow, fog and other. These methods are automatic from generating datasets point of view but challenging at the design stage of the virtual world.

Another remarkable example of using synthetic data is the task of the of a person's gaze direction recognition. In 2016, an article Learning an appearance-based gaze estimator from one million synthesized images was published (Wood et al., 2016). There was a method for generating eyes proposed taking into account the eyeball biological features, as well as the skin around it. The work pays great attention to the light characteristics of the eye surface. This work uses the same idea of generating 3-D models of objects, but taking into account their physical characteristics for higher realism.

2.2 NN-based methods

Often there are also problems in which the original dataset contains data of a different nature. From here naturally, arise the tasks of Domain adaptation (Su et al., 2019) and Style transfer. In 2018, an article A Literature Review of Neural Style Transfer was published (Jing et al., 2017). It discusses methods of transferring style from one image to another using neural network methods. Ideas based on the principle that neural networks highlight features of style. The first articles on this topic used features obtained using neural networks VGG (Simonyan and Zisserman, 2014), as well as the principles of autoencoders (Kingma and Welling, 2013). Style transfers are carried out due to tricks with intermediate outputs of neural networks, as well as in various ways of constructing a loss function. In 2017, Judy Hoffman introduced a domain adaptation method called CYCADA (Hoffman et al., 2017). Its essence was the use of a complex architecture consisting of two generators, two discriminators and four auxiliary decision networks(Karacan et al., 2016). The method showed good results; however, for training, it is necessary to have labelled semantic segmentation data (Long, Shelhamer, and Darrell, 2015).

In July 2018, the team of Ming-Yu Liu, Thomas Breuel, Jan Kautz from NVIDIA proposed a method called UNIT (Unsupervised image-to-image Translation) combining the ideas of VAE and GANs (Liu, Breuel, and Kautz, 2017a). The idea of the proposed method is to build neural networks based on GANs and VAEs for style transferring tasks. The proposed method constructs the mapping of source images into hidden space and makes transformations using neural networks in these spaces. This approach transfers styles well and as a result, makes a good conversion of their synthetics into real data. However, this method does not set the primary goal of generating high-quality synthetics.

Recently, a large number of approaches, methods, and architectures have been developed to solve this and similar problems. However, analyzing the work in this area, we can say that insufficient attention was paid to the problem consideration of generating synthetic data precisely from the statistical methods point of view.

Chapter 3

Proposed approach

3.1 Dataset collecting

For experiments, we were selecting data according to two criteria: the relevance of the task for which they can be used and the simplicity of objects for human perception. The principal requirement was the existence of a pair "real data - synthetic data" since the generation of large amounts of synthetic data from scratch is a costly and time-consuming process.

By the first criterion, we selected the SYNTHIA dataset. SYNTHIA is a dataset that has been generated to aid semantic segmentation and related scene understanding problems in the context of driving scenarios. SYNTHIA consists of a collection of photo-realistic frames rendered from a virtual city and comes with precise pixel-level semantic annotations for 13 classes: misc, sky, building, road, sidewalk, fence, vegetation, pole, car, sign, pedestrian, cyclist, lane-marking (Hernandez-Juarez et al., 2017).

Since the volume of SYNTHIA dataset is not large enough for the experiments, it was decided to add a part of dataset extracted from the game Grand Theft Auto V (Martinez et al., 2017). Dataset contains semantic annotations for 19 classes.

The self-driving task requires maximum accuracy in its solution, and therefore large high-quality datasets, which is consistent with the relevance of our work. In this case, we chose the Berkeley DeepDrive dataset as real data on which three complex tasks for the CVPR 2018 Autonomous driving workshop were conducted: detection of road objects, segmentation of the driving region and adaptation of semantic segmentation domains (Yu et al., 2018) (fig. 3.1).

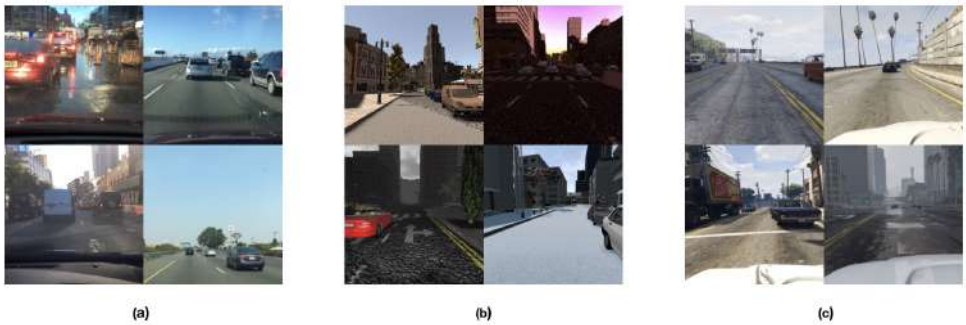

FIGURE 3.1: Examples of city datasets: (a) DeepDrive dataset, (b) SYNTHIA dataset, (c) GTA V dataset

According to the second criterion, we took dogs images dataset because they are easy for human perception, but challenging to formalize for a computer. It follows that the distribution is complex, and this is a vital aspect to consider in our study.

As a real dataset, we selected Stanford Dogs Dataset (Parkhi et al., 2012), which contains images of 120 dog breeds from around the world. This dataset was created using images and annotations from ImageNet for the task of detailed categorization of images. As its synthetic analogue, we chose images of dogs generated using GAN (Goodfellow et al., 2014) method from the Kaggle Generative Dog Images competition. It contains 10,000 examples of synthetically generated dogs without markup (fig. 3.2).

(a) (b)

FIGURE 3.2: Examples of dogs datasets: (a) dogs images generated using GAN, (b) Stanford Dogs Dataset

3.2 Problem solution

Before the experiment starts, we converted our data to a single image of 224x224 size and three colour channels format.

Our hypothesis assumes that the distributions of real and synthetic data have statistically significant differences. For humans, the difference between synthetic images and real data is intuitive, but like many similar processes, hard to formalize. Based on this statement, our approach attempts to formalize these differences.

The approach we chose for the first iteration of the experiment involves using trained neural networks to extract image information in vector form. The figure 3.3 shows the visualization of the first iteration of the experiment.

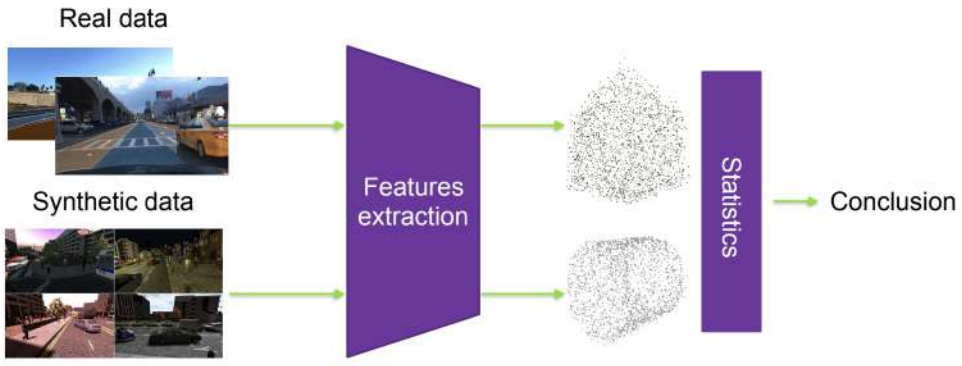

FIGURE 3.3: First iteration: hypothesis about the distributions of real and synthetic data

We will use several networks, such as AlexNet (Krizhevsky, Sutskever, and Hinton, 2012), MobileNetV2 (Sandler et al., 2018), VGG13/16 with batch normalization (Simonyan and Zisserman, 2014), MNASnet, MNASnet0.5 (Zoph et al., 2017), ResNet152 (He et al., 2015), ResNeXt101 (Xie et al., 2016), pre-trained on Imagenet (Deng et al., 2009), as feature extractors (Ioffe and Szegedy, 2015). Using a statistical test, Student's T-criterion (Press et al., 1992), we can test our assumption about the distinguishability of synthetic and real data with a certain level of confidence. Using the Kullback – Leibler divergence (relative entropy) (Kullback and Leibler, 1951), we can verify our assumptions about the difference in the distributions of real and synthetic data.

The next step is to train the variational autoencoder (Kingma and Welling, 2013) on real and synthetic data, thereby constructing a hidden space and two mappings: from pictures to hidden space and vice versa. We will analyze and compare the basic statistical characteristics of real and synthetic data in a hidden space. We will use simple mathematical operations in order to approximate the statistical characteristics of synthetic data to real ones.

Then we pass the converted hidden representations of the synthetic data through the decoder. At the output, we expect to get images close to real. The figure 3.4 shows the visualization of the second iteration of the experiment.

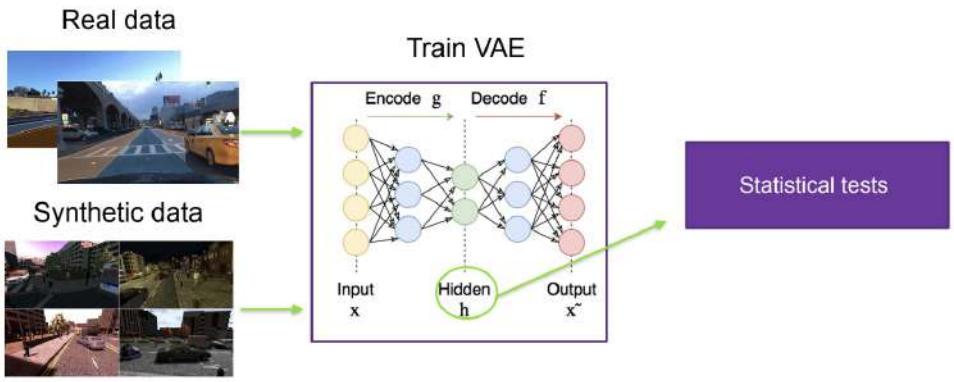

FIGURE 3.4: Second iteration: hypothesis about the hidden space and mappings

3.3 Hypothesis verification

Two experiments can serve as verification of our main hypothesis.

First, we can re-pass the generated data through the trained nets from the firs iteration. Then, a measure of quality will be a statistically insignificant difference in data distributions.

As a second experiment, let us pass the transformed synthetic and initial real data through a simple neural network, which will solve the binary classification problem, i.e., determine the nature of the image.

After that, we will use the validation dataset to predict the binary classification label. If a neural network cannot accurately predict the correct label, then the conversion quality of synthetic data can be considered high. We assume that a neural network cannot distinguish class labels if the ROC-AUC (Davis and Goadrich, 2006) value is about 0.5 on the validation dataset.

Chapter 4

Experiments

4.1 Data dissimilarity hypothesis verification

The data presented in the form of colour images in the RGB format, the image sizes with which we work are 224x224, 128x128, 64x64, 32x32 pixels.

Consequently, the dimension of the space we are working with is $[0..1]^{NxNx3}$, where the colour intensity is determined by the interval from zero to one. Vectors have a very high dimension - this greatly complicates the work with images and their analysis. It is also essential that the vectors are not random. In other words, the intensity of a particular pixel is highly dependent on the pixel values that are next to it. Therefore, the pixel intensity values strongly depend on the spatial position in the picture, as the pixel values in the local area of the image.

These relationships and dependencies determine the statistical distribution of images.

Real and synthetic images contain the same essence, and the person quickly determines what represented in those and other images, however, he just as easily distinguishes them. The pixel dependencies that generate our image distributions are very complex. They are almost impossible to formalize.

However, we can say that part of the distribution is responsible for determining the meaning shown in the picture. The other part determines whether the image is real or synthetic. Most likely, most of the dependencies are inherent in the first and second reasons.

4.1.1 Classifier construction

Neural networks can build maps from the space of inputs to spaces with specific properties, for example, linear separability of classes.

Neural networks learn distributions based on the laws of pixel relationships and find the correct mapping of spaces. Consider neural networks separating real and synthetic data. Neural networks have poor interpretability.

However, for networks with a small number of neurons, it is possible to visualize layers and understand on what basis the network decides whether the picture is real or synthetic. The general network architecture presented in the figure 4.1.

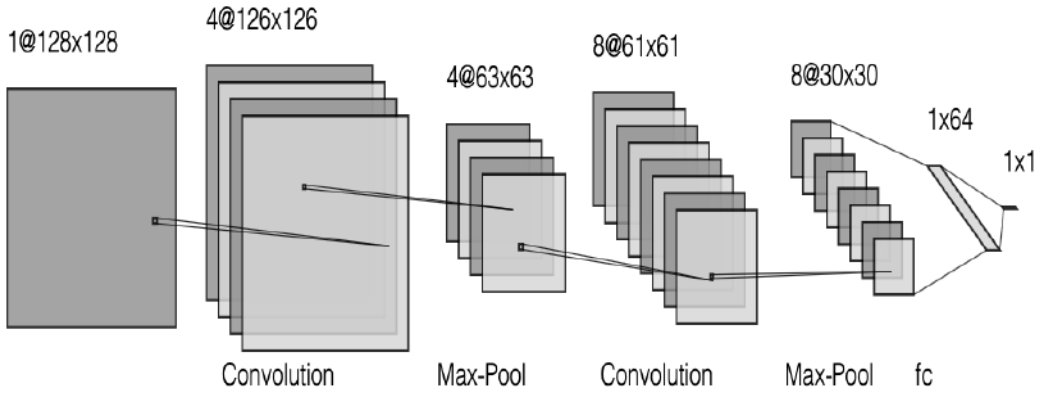

FIGURE 4.1: Network architecture for the 128x128 image

The network architecture described as follows:

$$[4-Conv3x3]-[MaxPool2x2]-[8-Conv3x3]-$$
$$-[MaxPool2x2]-[Dropout-0.2]-[FC-64]-[FC-1]$$

ReLU used as a nonlinearity function. The last nonlinearity is the sigmoid function. The training was conducted by the Adam optimizer (Kingma and Ba, 2014), with a learning rate of 0.0001. The ultimate optimization function is binary cross-entropy. There were ten eras of training with a batch size of sixteen.

During the experiments, four models trained for different image sizes, particularly 224x224, 128x128, 64x64, 32x32. ROC curves show the simulation results on the test data. Descending image size from left to right (fig. 4.2).

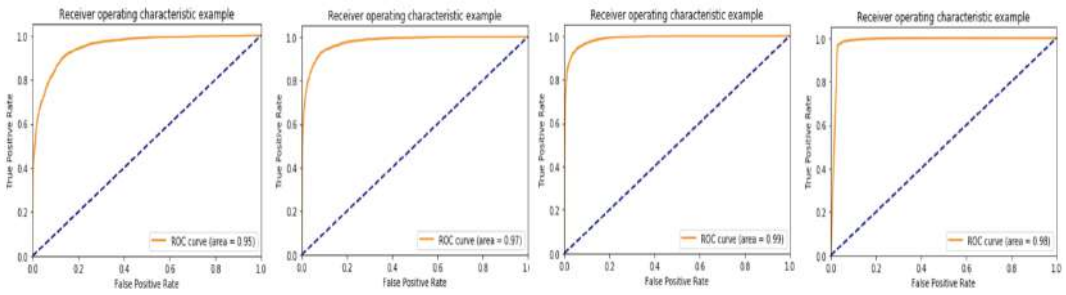

FIGURE 4.2: ROC curves for test data

All networks differentiate real data from synthetic very well. It means that networks build proper spaces where there is a linear separability.

The table 4.1 shows the results of the Average precision-recall score quality metric for trained networks with different image size.

TABLE 4.1: Average precision-recall score quality metric for trained networks

Image size	$224x224$	$128x128$	$64x64$	$32x32$
Average precision-recall score	0.90	0.96	0.93	0.87

Figure 4.3 shows the results of a trained network on test data.

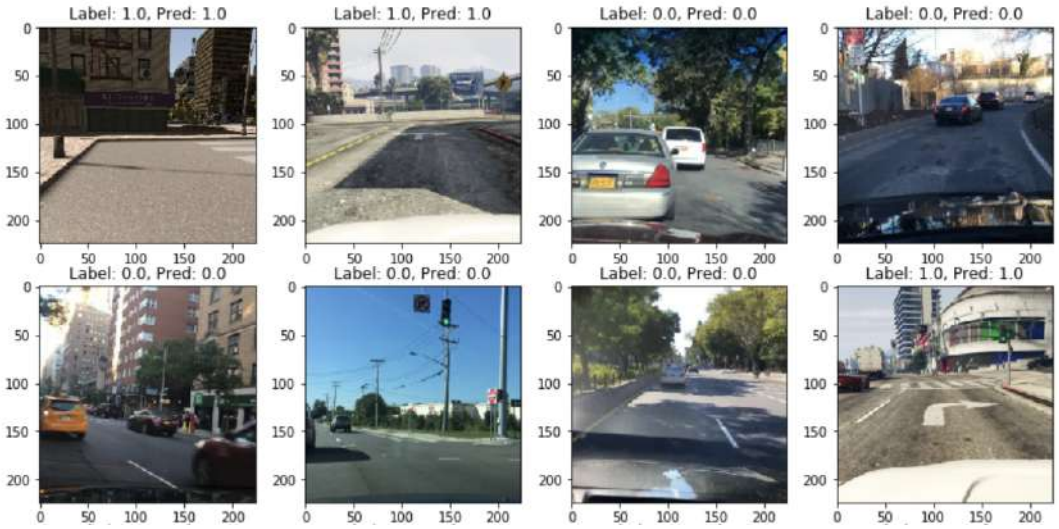

FIGURE 4.3: Results of trained network on test data

4.1.2 Model interpretation

Based on the activation maps after the convolutional layers (fig. 4.4), we can conclude that the neural network looks at the picture as a whole, and not at its specific sections. The neural network responds to the style of the picture, and not to specific objects on it.

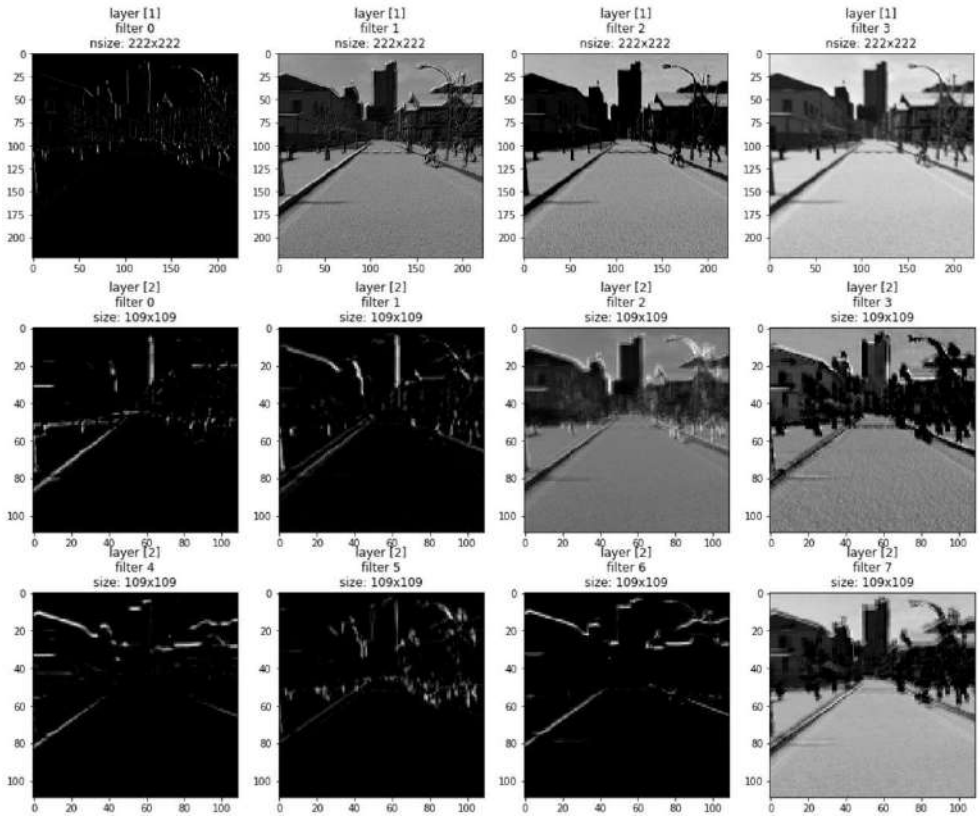

FIGURE 4.4: Transformation after the first and second convolutional layers of real and synthetic images.

The weights of the first fully-connected layer look comprehensively at the whole picture (fig. 4.5), some at minor artifacts, such as a car torpedo.

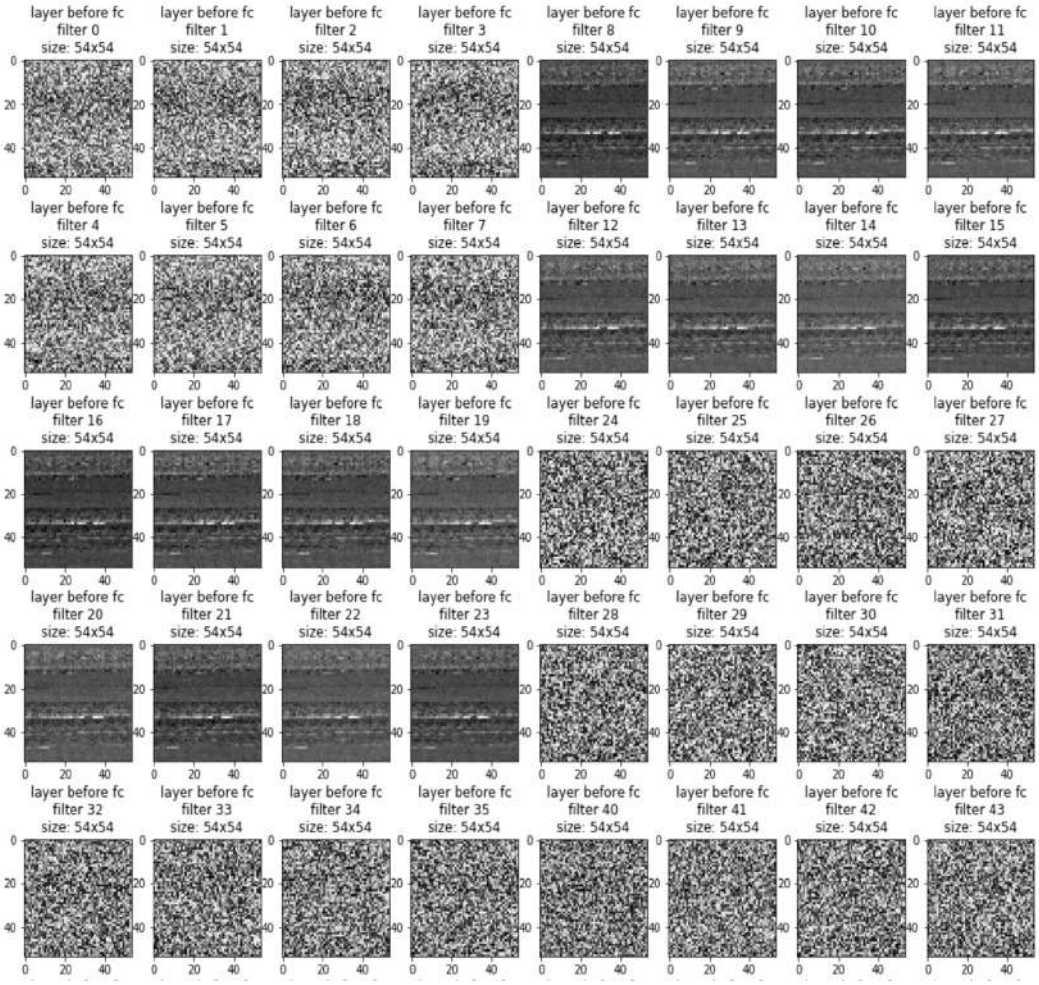

FIGURE 4.5: Some activation maps of fully connected layer

Constructed neural networks entirely separate real and synthetic data focusing on the style of the image, not the semantic meaning of facilities. A space with linear separability is essential for us in terms of understanding the formal differences between real and synthetic data.

4.1.3 Style space analysis

Consider the penultimate layer of the constructed neural network. Its dimension is 64. Further, on the network, we use one linear layer with a sigmoidal activation function.

We can describe the prediction work of our neural network can as follows: constructing a map into the space R^{64} and logistic regression in this space.

High-quality metrics in test data indicate good linear separability in the intermediate space. We visualize this space using the PCA (fig. 4.6) algorithm (Jolliffe, 1986).

The algorithm can help us to find such linear mapping from 64-dimensional vectors to 2-dimensional - maximizing the percentage of explanations for a variance.

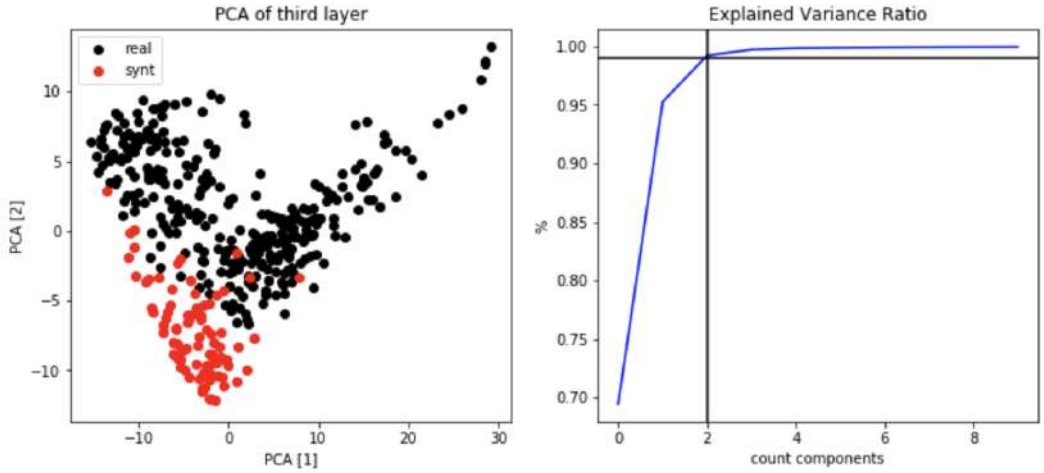

FIGURE 4.6: Visualization of hidden space

Visualization of the two main components of PCA decomposition makes clear that the points grouped in two clusters. At the same time, the second graph shows that already for 2 components, almost all the variability of data in hidden space is explained. Therefore visualization reflects reality.

A similar picture observed for all 4 neural networks. Thus, real and synthetic data are separable in terms of style.

4.1.4 Content space analysis

The ImageNet (Deng et al., 2009) competition has contributed to the appearance of such a direction as transfer leering (Zoph et al., 2017). The main idea is that having trained a neural network on a large data set, in future, can be used as an extractor of high-level features.

Further use means replacing the last layers of the network with new ones and additional training for your data set. The concept creators of style transferring (Jing et al., 2017) argue that the first layers of neural networks are responsible for the style of the image, and the final ones for its content.

Images of real and synthetic data taken from the same domain, so they contain the same content. The use of high-level features assumes that features store information about objects in the image, but not about style.

As a feature of extractors, we use the following neural networks trained on ImageNet (Deng et al., 2009): AlexNet (Krizhevsky, Sutskever, and Hinton, 2012), MobileNetV2 (Sandler et al., 2018), VGG13 BN, VGG16 BN (Simonyan and Zisserman, 2014), MNasNet, MNasNet0.5 (Zoph et al., 2017), ResNet152 (He et al., 2015), ResNeXt101 (Xie et al., 2016).

These networks were chosen because of their unspoken popularity during transfer learning. So the MobileNetV2, MNasNet, MNasNet0.5 networks are small and lightweight, AlexNet is the first of its kind. VGG13 BN, VGG16 BN, ResNet152, ResNeXt101 - deep networks with good and proven in practice results.

All networks have cut off the last two classification layers. Passing real and synthetic sets of images through the network, we get two point clouds for each network.

The statistical characteristics of point clouds should be the same within the synthetic and real data for each group. For simplicity, compare the average values.

To compare the average of the two groups, we use the two-sided t-test (Press et al., 1992). With the null hypothesis that two independent samples have identical

mean values. Since vectors are multidimensional, we make a rough assumption that the components are independent.

This assumption is not wholly valid at least because of the value of the components generated from one hidden variable.

To obtain a measure of means equality, we consider the percentage of components for which we reject the null hypothesis. The smaller the percentage, the more features with the same average.

To check for equality of means within the real and synthetic data, for each group, we take two random subsamples without intersections and conduct tests comparing the means of these 2 subsamples.

Consequently, we get 2 numbers - the percentage of features with a different expectation for real and for synthetic data. This experiment is carried out independently for all neurons of networks. Results presented in Table 4.2

TABLE 4.2: Comparing expectations within single data group

Model	Real data	Synthetic data
AlexNet	7.59%	3.76%
MobileNetV2	1.47%	2.39%
VGG13_bn	0.24%	0.85%
VGG16_bn	0.24%	0.32%
MNasNet	1.54%	1.89%
MNasNet0.5	2.69%	3.57%
ResNet152	0.0%	0.0%
ResNeXt101	0.0%	0.0%

For almost all networks, the percentage of components with different mat expectations does not exceed three per cent. This experiment shows us that with high probability, the data within the same group have the same nature.

Since high-level features collected from the last layers of the network, they contain information about the content. Therefore, the statistical characteristics for the cloud of synthetic data should match with the statistical characteristics of the real cloud. In practice, this does not happen and we can observe it in table 4.3.

TABLE 4.3: Comparing expectations between real and synthetic data

Model	Mean by components NOT equal
AlexNet	96.34%
MobileNetV2	88.57%
VGG13_bn	91.94%
VGG16_bn	86.18%
MNasNet	86.47%
MNasNet0.5	89.96 %
ResNet152	26.37%
ResNeXt101	32.76%

For most networks, many features have a different mathematical expectation. The ResNet152 and ResNext101 networks deserve special attention, in case of comparing equalities of the means within each of the groups for these 2 networks, all features had the same means.

In case of comparisons of means for two different groups, these 2 networks have the smallest percentages of various math expectations. As one of the interpretations, there was an assumption that those features for which expectations matched were content-rich, and those that did not match were stylistic, but looking at the types of feature distribution, we can see that for all networks except these two, the distributions are normal, for ResNet152 and ResNext101 it exponential (fig. 4.7).

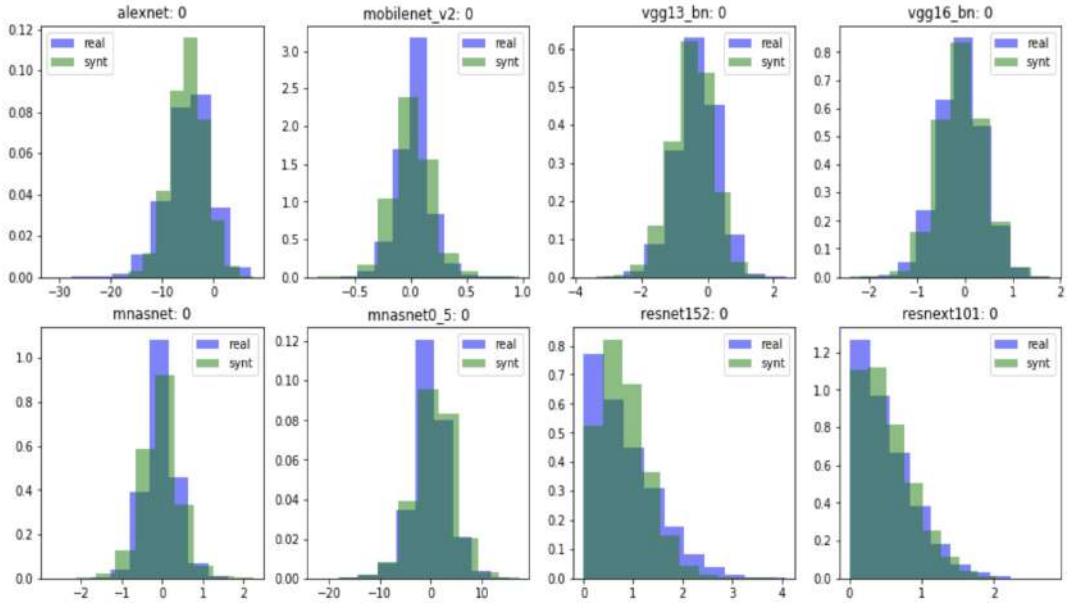

FIGURE 4.7: Features distribution

It is known that with large sample values, t-test also works for abnormal distributions [1]. Most likely, the difference in percentage order for these two networks is explained accurately by the distribution family of their features.

Real and synthetic data are different if: the neural network divides these two classes based on the style of the image, and not on its content. Distributions of high-level representations have statistically significant differences.

We conducted similar experiments on the dataset with dogs. A small neural network, could not distinguish the real dogs. We assumed that real and GAN-generated images had a very similar style. The trained networks inefficiently allocate content and as a consequence, contain more information about the style, which in this case, matches.

Unfortunately, no statistical differences were revealed by the tests. The analysis of formal differences for datasets with dogs no longer carried out, and further experiments with it stopped.

4.2 Construction representative hidden space

It is quite challenging to work directly with images. It is much more convenient to work with numerical vectors. Constructing a mapping function from image space

[1] https://stats.stackexchange.com/questions/9573/t-test-for-non-normal-when-n50

to a hidden space is quite simple; however, constructing a reconstruction function is much more difficult. Hidden space must have the property of compactness. Compactness implies that a small change in the image leads to a small change in the hidden space and, on the converse, small changes in the hidden constancy should not actively change the reverse presentation. Construction of space with such properties, and mapping functions as well, is carried out by constructing a variational autoencoder (Kingma and Welling, 2013).

4.2.1 Building VAE

Variational autoencoder consists of two networks - encoder and decoder. The architectures of these 2 networks determine the complexity of the mappings. In the classic VAE (fig. 4.8), the encoder is a 3-layer neural network that narrows the data. The encoder also represents a 3-layer neural network.

The peculiarity is that the encoder generates two vectors - the average and the variance. Then the reparametrization trick (Kingma, Salimans, and Welling, 2015) is applied. During reparameterization, a random value generated from the normal distribution with the encoder output parameters and this vector transferred to the decoder input. All layers are fully connected.

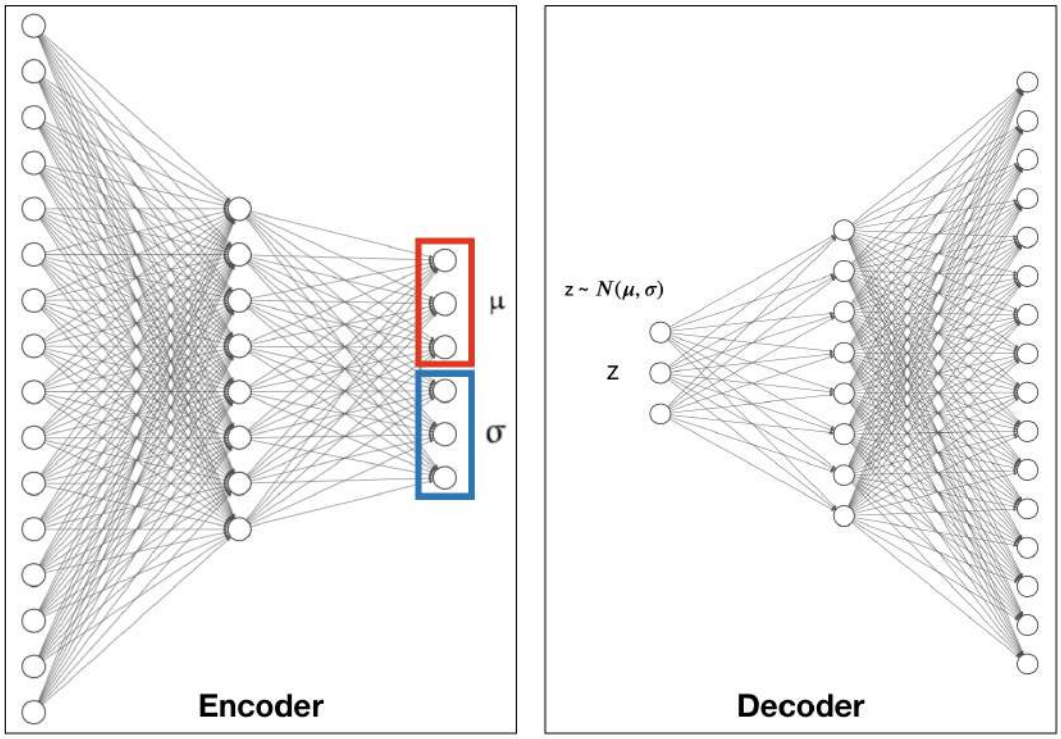

FIGURE 4.8: Variational autoencoder architecture

Still, it is not possible to apply this model in its pure form. The total size of the weights of the neural network grows quadratically from the number of neurons. The resulting network will be enormous.

VAE architecture is changing to reduce network size. The internal structure and idea with reparametrization remain the same. To make it possible to apply VAE of an acceptable size at the beginning of the encoder and the end of the decoder, we add layers of convolutions that are classic for computer vision. Only those features that display the contents of the image make sense to transmit to the encoder input.

In our experiments, we used ResNet152 and VGG11 BN (Simonyan and Zisserman, 2014) networks trained on ImageNet. The choice focused on these 2 models since it got out to choose the optimizer parameters and the learning speed for them. After the decoder, three convolution layers added with batch normalization, the parameters of the layers selected in such a way that output would be the image.

To fit the size of the input image, it stretched using bilinear interpolation. Each model trained in 200 eras with the Adam optimizer. Lr - changed according to CiclicLR (Smith, 2015) with an interval of steps from $1e-5$ to $1e-2$. The full cycle of LR changes - takes 1000 steps of the optimizer.

Internal VAE architecture described as follows (fig. 4.9):

$$[fc-4096]-[fc-2048]-[2xfc-512]-[fc-2048]-fc-4096]$$

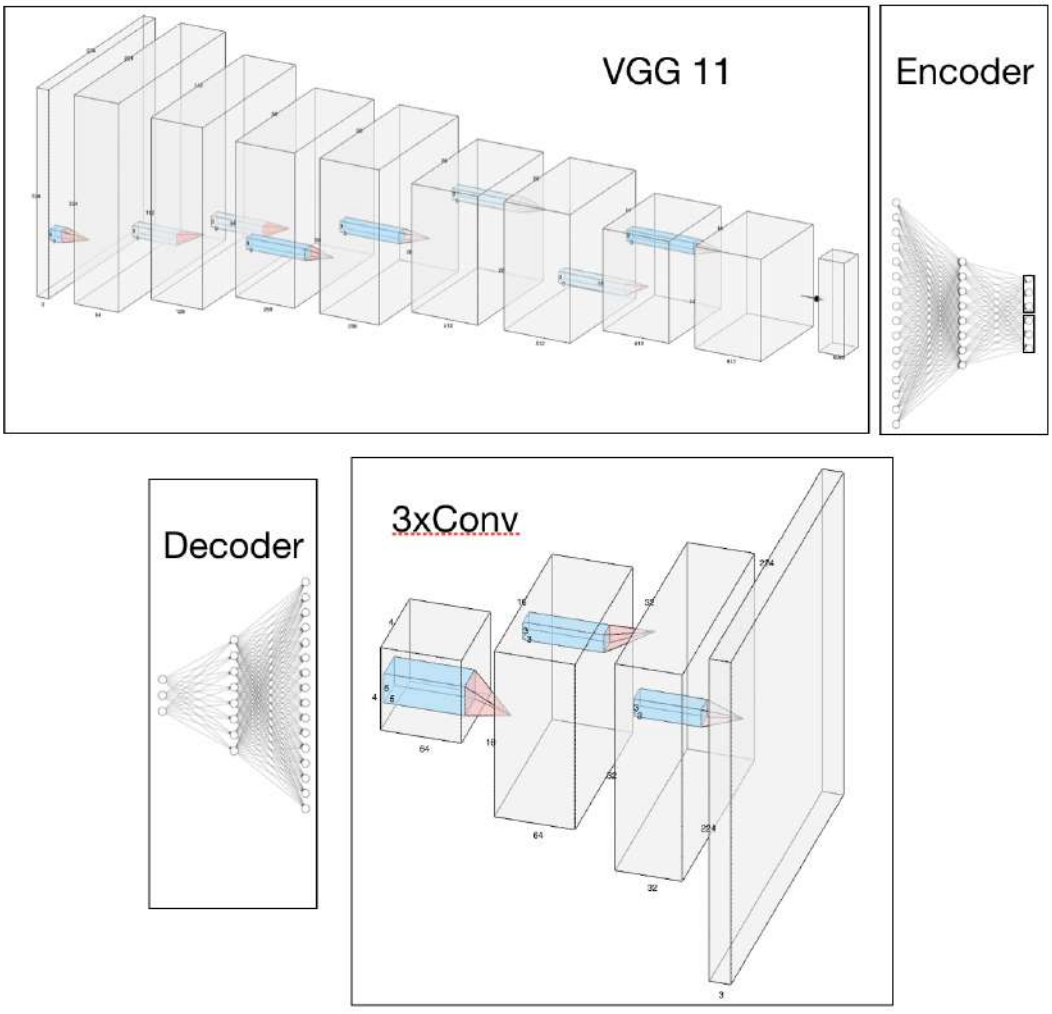

FIGURE 4.9: Modified variational autoencoder

The training took place on a mixed dataset of real and synthetic data. The following results obtained for ResNet based network(fig. 4.10):

FIGURE 4.10: ResNet based network results

And for VGG based architecture results presented on figure 4.11.

FIGURE 4.11: VGG11 based network results

The image quality is very blurry, and the network restores the colour range of the image, but substantially does not restore the details. Due to the great depth of the network, the gradients practically do not reach initial layers. As a result, the initial filters almost do not change, and templates typical of our data cannot form.

4.2.2 Building VAE with residual connections

The problem of gradient attenuation in the network solved by changing the architecture, namely by adding residual connections (He et al., 2015). Changes to the reparametrization scheme also contributed to results improvement.

In the standard reparametrization scheme, the following scheme used: $z_i\ N(\mu, \sigma)$, where z_i is the decoder input, and μ, σ are generated by the encoder. Also, during the optimization of a variational autoencoder, the loss function includes the Kullback-Leibner distance (Kullback and Leibler, 1951) between the distributions $N(\mu, \sigma)$ and $N(0, I)$.

Using the trick with reparametrization is explained by the following idea. An encoder builds a mapping from image space to hidden space. After reparametrization (fig. 4.12 (a)), a random point taken in the vicinity of the mathematical expectation generated by encoder, and then the decoder trains to restore the initial image from the sampled point. The random vector multiplied by the sigma generated by the encoder and added to the expectation. Such change in hidden space leads to a slight distortion of the initial image. This idea ensures that space has the property of compactness.

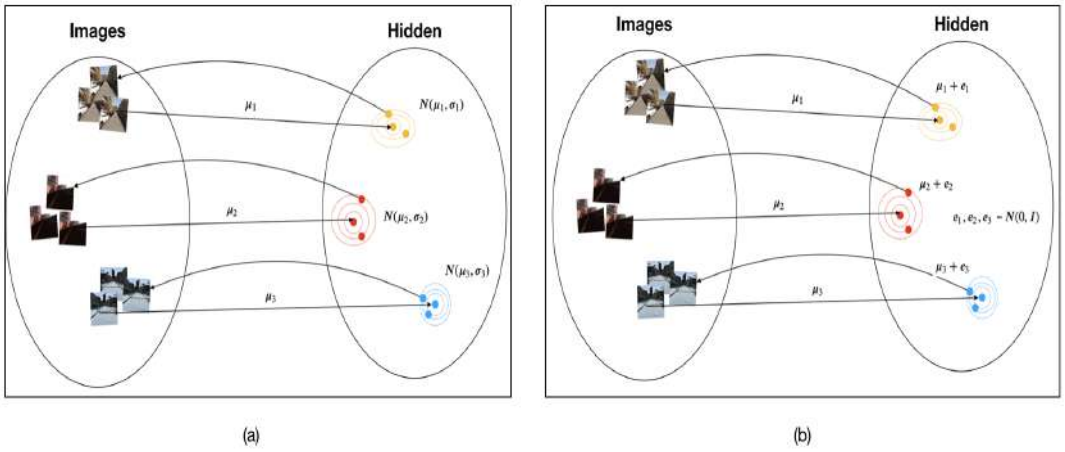

FIGURE 4.12: Reparametrization schemes: (a) - classic, (b) - modified

An alternative is to get rid of the sigma vector generation from the encoder and to sample a new point immediately from the distribution $N(\ mu_1, I)$(fig. 4.12 (b)) It reduces the number of network parameters and also improves convergence. Such a scheme proposed in the work of UNIT (Liu, Breuel, and Kautz, 2017b), but there it was used without much motivation.

The following formulas describe the architecture of the encoder and decoder for VAE with residual connections:

Encoder:

$$[32Conv7x7] - [64Conv4x4] - [128Conv4x4] - 4x[64 - ResBlock]$$

Decoder:

$$4x[64 - ResBlock] - [Upsample] - [128Conv5x5] -$$
$$-[Upsample] - [64Conv5x5] - [Upsample] - [3Conv7x7]$$

ResBlock - represents two layers of convolutions, as well as residual connection. We trained a total of 3 networks, and the training took place for networks with different residual block sizes: 16, 32, and 64 (fig. 4.13).

Reconstructions for these networks are presented in the image below, from top to bottom in increasing residual block sizes. The last network better displays the colour characteristics of the image, most likely this effect occurred due to an increase in the weight of the network, and the last network studied taking into account more stringent colour augmentations.

All networks learned how to identify the features of the relief and the interior of the city. The training lasted for 100 eras with the Adam optimizer. Lr - changed according to CiclicLR with an interval of steps from $1e-7$ to $1e-4$. Full LR change cycle - takes 2000 optimizer steps. The hidden space of all 3 networks is much larger than that of previous models.

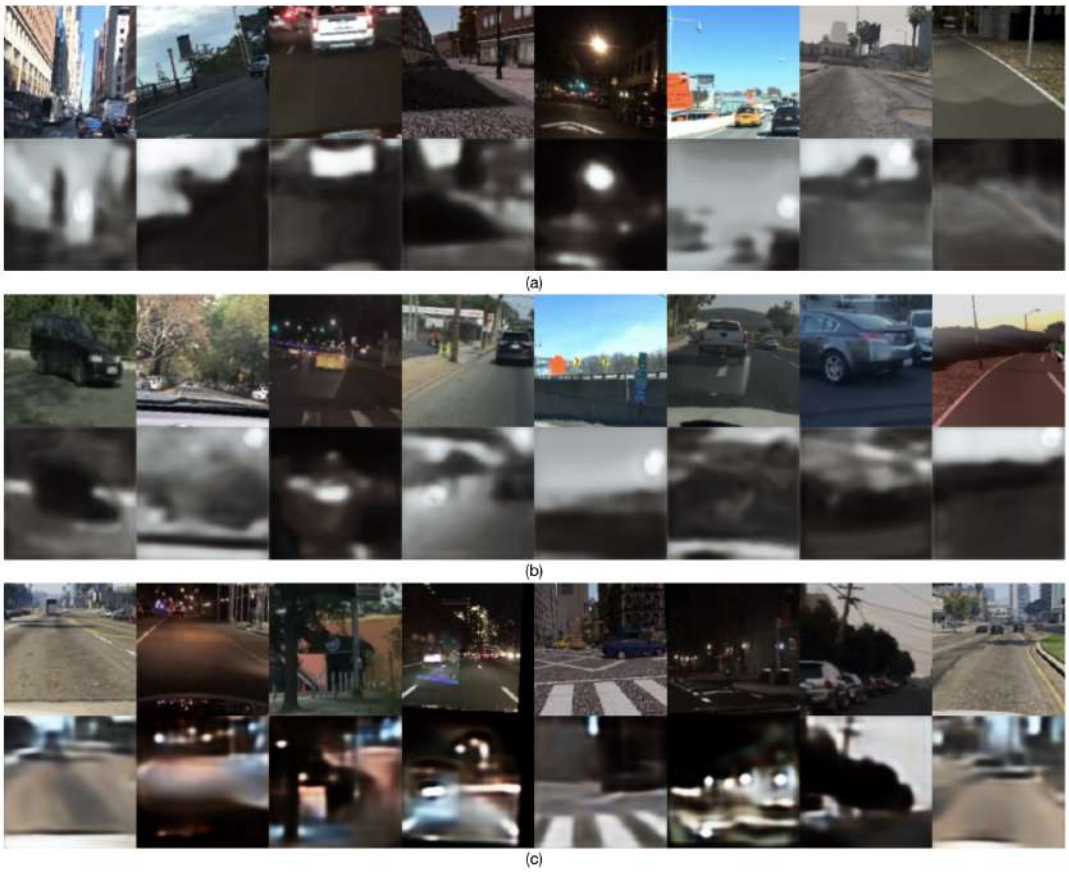

FIGURE 4.13: Results for networks with different residual block sizes:
(a) - 16, (b) - 32, (c) - 64

4.2.3 The problem of data multimodality

All reconstructed images show a significant decrease in quality. Such an effect is inherent in almost all variational autoencoders. However, this feature of the reconstructed images also explained by another fact.

During optimization, the distribution of features in a hidden space tends to approximate normal, and the encoder can quite well approach the initial distribution if it is initially close to normal.

The task posed to the encoder is to perform coding from a high-dimensional space to a lower one while preserving the family of distributions.

The dataset with which we work has at least two modes: the first corresponds to real data, the second to synthetic. Also, in the group of synthetic data, modes can be formed according to the conditions under which the generation took place, for example, it can be a time of the day, weather or season. In the part of the dataset with real data, we can also find other modes. For simplicity, we restrict ourselves to two - the mode of real data and synthetic. Thus, the initial data has a bimodal distribution.

The minimized functional of a variational autoencoder includes a measure of proximity between the generated distribution and the normal distribution $N(0, I)$. The optimal solution would be to reduce both modes to a normal distribution - $N(0, I)$. In practice, the distribution generated by the encoder is bimodal, and we can't reach the optimum for the measure of proximity.

4.2.4 VAE hidden space tests

Let's conduct tests for the equality of the mean within the hidden space generated by VAE. The table 4.4 presents the percentage of features with unequal expectation within one data type.

TABLE 4.4: Comparing expectations within single data group

Model	Real	Synthetic
VGG11VAE	2.54%	18.75%
ResNet152 VAE	1.77%	18.13%
Residual VAE Hidden-4096	3.45%	2.61%
Residual VAE Hidden-8192	2.71%	2.66%
Residual VAE Hidden-16384	3.85%	2.5%

The table 4.5 shows the percentage of unequal math expectations when comparing clouds of real and synthetic data.

TABLE 4.5: Comparing expectations between real and synthetic data

Model	Real vs Synthetic
VGG11VAE	72.85%
ResNet152 VAE	73.91%
Residual VAE Hidden-4096	75.82%
Residual VAE Hidden-8192	76.28%
Residual VAE Hidden-16384	74.64%

Based on the results obtained, we can say that the variable autoencoders have managed to build a hidden space in which real and synthetic data form clusters.

4.3 Proof of decoders weakness

When converting the average value of synthetic data to the real average and further feeding these points to the input of the decoder, at the output, we got noise presented on figure 4.14, not images that are very similar to the real ones.

FIGURE 4.14: Generated data

Looking more closely at the points in the hidden space, it turns out that it is quite sparse. The bulk of the points from which the images recovered are in a multidimensional cube with value limits from -1.5 to 1.5.

The number of points concentrated in this area is about eighty thousand. The total density is about twenty-six thousand images per one conventional unit of space measure.

With a sufficient density of points in the hidden space, it would have content-containing images in each of the points (fig. 4.15 (a)). However, the real situation corresponds to a sparse image (fig. 4.15 (b)), and a randomly taken point is unlikely to belong to an area with a proper reconstruction.

In order to generate quality synthetics, it is at least necessary the space be dense. There is no point in comparing real data and sampled data because even without tests, you can see that they do not show the essence. Consequently, having a variational autoencoder fitted building dense hidden spaces, the subsequent equating of the average value of synthetic representations to the required average of real data can lead to the generation of useful synthetic data.

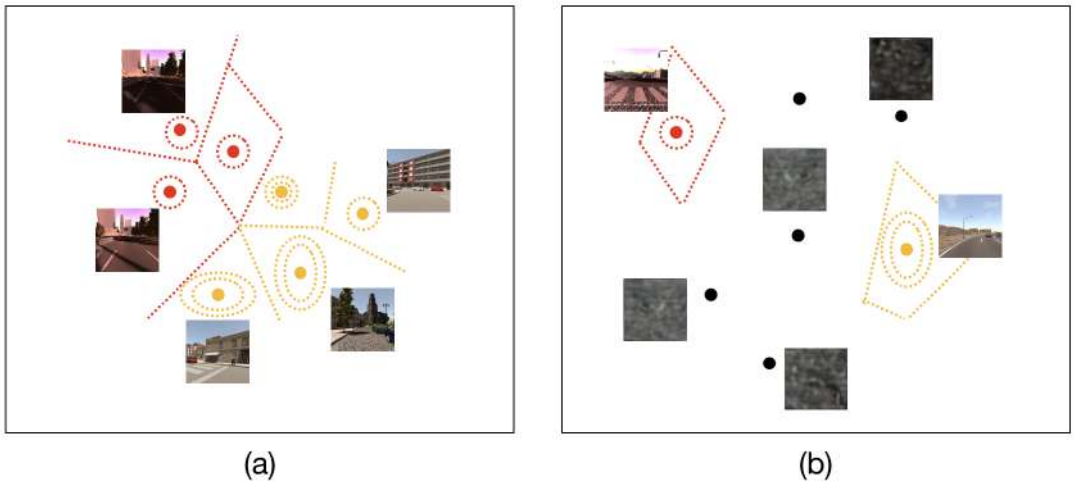

FIGURE 4.15: VAE hidden space: (a) - dense space, (b) - sparsed space

4.4 Implementation details

For the realization of experiments, we used the framework of deep learning - PyTorch (Paszke et al., 2017) with the integration of python 3.7. PyTorch provides the ability to operate with internal neurons of the network, which allowed to carry out a detailed analysis of networks, as well as effective training. The training models were taken from the torchvision library.

All networks were trained on a GeForce GTX 1080 TI GPU with 11.264 MB GDDR5X memory. Training of variation auto-encoder for one epoch lasted about 50 minutes. General training of networks in total took from 4 to 8 days. Training of classification network for 20 epochs lasted about one hour.

Chapter 5

Conclusion

In this paper, we considered the actual problem of efficient generation of synthetic data. After the analysis of existing approaches, which were surrounding its solution, the method, which combined classical statistics with modern neural network methods, was applied.

Initially, we set a goal to prove that there is a statistically significant difference in the distributions of real and artificially generated data and during the first iteration of experiments we successfully confirmed this hypothesis by proving it based on content-containing features obtained from pre-trained neural networks. Stylistic differences were also confirmed using the classifier.

At this stage, we also had to correct the direction of our experiments by excluding from consideration not quite successfully selected data about dogs, as it turned out that the style of synthetic images is too close to the style of real ones.

The second task was to work with autoencoders in order to obtain a hidden space, and we also found out stylistic differences there. After analyzing all the obtained results based on two iterations of experiments, we tried to generate synthetic data according to the justification in the pipeline of experiments. But the result was unsatisfactory, after which the reason of such generation explained.

The described problems interfering generation are mostly of technical nature and do not prove the inoperability of the idea. The given work might have proceeded, and it is necessary to develop it deepening in feature of used network architectures, increasing in volumes of the training data and approaches of the training algorithms. From the statistical side it would be rational to consider other statistical characteristics in addition.

Bibliography

Davis, Jesse and Mark Goadrich (2006). "The relationship between Precision-Recall and ROC curves". In: *ICML '06*.

Deng, J. et al. (2009). "ImageNet: A Large-Scale Hierarchical Image Database". In: *CVPR09*.

Goodfellow, Ian J. et al. (2014). "Generative Adversarial Nets". In: *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'14. Cambridge, MA, USA: MIT Press, pp. 2672–2680. URL: http://dl.acm.org/citation.cfm?id=2969033.2969125.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385. arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

Hernandez-Juarez, Daniel et al. (2017). "Slanted Stixels: Representing San Francisco's Steepest Streets". In: *British Machine Vision Conference (BMVC), 2017*.

Hoffman, Judy et al. (2017). "CyCADA: Cycle-Consistent Adversarial Domain Adaptation". In: *ICML*.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *ArXiv* abs/1502.03167.

Jing, Yongcheng et al. (2017). "Neural Style Transfer: A Review". In: *IEEE transactions on visualization and computer graphics*.

Jolliffe, I.T. (1986). *Principal Component Analysis*. Springer Verlag.

Karacan, Levent et al. (2016). "Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts". In: *ArXiv* abs/1612.00215.

Kingma, Diederik P. and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. URL: http://arxiv.org/abs/1412.6980.

Kingma, Diederik P. and Max Welling (2013). "Auto-Encoding Variational Bayes". In: *CoRR* abs/1312.6114.

Kingma, Durk P, Tim Salimans, and Max Welling (2015). "Variational Dropout and the Local Reparameterization Trick". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., pp. 2575–2583. URL: http://papers.nips.cc/paper/5666-variational-dropout-and-the-local-reparameterization-trick.pdf.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Kullback, Solomon and R. A. Leibler (1951). *ON INFORMATION AND SUFFICIENCY*.

Liu, Ming-Yu, Thomas Breuel, and Jan Kautz (2017a). "Unsupervised Image-to-Image Translation Networks". In: *ArXiv* abs/1703.00848.

Liu, Ming-Yu, Thomas Breuel, and Jan Kautz (2017b). "Unsupervised Image-to-Image Translation Networks". In: *CoRR* abs/1703.00848. arXiv: 1703.00848. URL: http://arxiv.org/abs/1703.00848.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully convolutional networks for semantic segmentation". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440.

Marcus, Gary (2018). "Deep Learning: A Critical Appraisal". In: *CoRR* abs/1801.00631. arXiv: 1801.00631. URL: http://arxiv.org/abs/1801.00631.

Martinez, Mark et al. (2017). "Beyond Grand Theft Auto V for Training, Testing and Enhancing Deep Learning in Self Driving Cars". In: *ArXiv* abs/1712.01397.

Parkhi, Omkar M. et al. (2012). "Cats and Dogs". In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Paszke, Adam et al. (2017). "Automatic Differentiation in PyTorch". In: *NIPS 2017 Workshop on Autodiff*. Long Beach, California, USA. URL: https://openreview.net/forum?id=BJJsrmfCZ.

Press, William H. et al. (1992). "Numerical Recipes in C: The Art of Scientific Computing, Second Edition". In:

Richter, Stephan R. et al. (2016). "Playing for Data: Ground Truth from Computer Games". In: *CoRR* abs/1608.02192. arXiv: 1608.02192. URL: http://arxiv.org/abs/1608.02192.

Ros, Germán et al. (2016). "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3234–3243.

Saenko, Kate et al. (2010). "Adapting Visual Category Models to New Domains". In: *Proceedings of the 11th European Conference on Computer Vision: Part IV*. ECCV'10. Berlin, Heidelberg: Springer-Verlag, pp. 213–226. ISBN: 3-642-15560-X, 978-3-642-15560-4. URL: http://dl.acm.org/citation.cfm?id=1888089.1888106.

Sandler, Mark et al. (2018). "Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation". In: *CoRR* abs/1801.04381. arXiv: 1801.04381. URL: http://arxiv.org/abs/1801.04381.

Simonyan, Karen and Andrew Zisserman (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. cite arxiv:1409.1556. URL: http://arxiv.org/abs/1409.1556.

Smith, Leslie N. (2015). "No More Pesky Learning Rate Guessing Games". In: *CoRR* abs/1506.01186. arXiv: 1506.01186. URL: http://arxiv.org/abs/1506.01186.

Su, Jong-Chyi et al. (2019). "Active Adversarial Domain Adaptation". In: *CVPR Workshops*.

Torralba, A. and A. A. Efros (2011). "Unbiased Look at Dataset Bias". In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR '11. Washington, DC, USA: IEEE Computer Society, pp. 1521–1528. ISBN: 978-1-4577-0394-2. DOI: 10.1109/CVPR.2011.5995347. URL: https://doi.org/10.1109/CVPR.2011.5995347.

Tzeng, Eric et al. (2014). "Deep Domain Confusion: Maximizing for Domain Invariance". In: *CoRR* abs/1412.3474. arXiv: 1412.3474. URL: http://arxiv.org/abs/1412.3474.

Wood, Erroll et al. (2016). "Learning an Appearance-based Gaze Estimator from One Million Synthesised Images". In: *Proceedings of the Ninth Biennial ACM Symposium on Eye Tracking Research & Applications*. ETRA '16. New York, NY, USA: ACM, pp. 131–138. ISBN: 978-1-4503-4125-7. DOI: 10.1145/2857491.2857492. URL: http://doi.acm.org/10.1145/2857491.2857492.

Xie, Saining et al. (2016). "Aggregated Residual Transformations for Deep Neural Networks". In: *CoRR* abs/1611.05431. arXiv: 1611.05431. URL: http://arxiv.org/abs/1611.05431.

Yu, Fisher et al. (2018). "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling". In: *CoRR* abs/1805.04687. arXiv: 1805.04687. URL: http://arxiv.org/abs/1805.04687.

Zoph, Barret et al. (2017). "Learning Transferable Architectures for Scalable Image Recognition". In: *CoRR* abs/1707.07012. arXiv: 1707.07012. URL: http://arxiv.org/abs/1707.07012.