УДК 04.31

Л. С. Ясенко, Я. М. Клятченко

ВЛАСТИВОСТІ ЗГОРТКОВІ НЕЙРОННОЇ МЕРЕЖІ НА ОСНОВІ АВТОКОДЕРА

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», Київ

Анотація. Розглянуто згорткові властивості автокодуючої нейронної мережі для задач виявлення об'єктів на зображенні. Для тренувань і тестувань були згенеровані набори даних у вигляді двовимірних зображень з трьома каналами передачі кольору. Зображення згенеровані на основі тривимірної сцени, що складається з таких об'єктів як сфери, куби, циліндри і моделі "мавпочок". Проведено оцінки часу тренування мережі на даних із різними конфігураціями та результату на виході нейронної мережі.
Ключові слова: набір даних, тренування, нейронна мережа, автокодер, згортка.
Аннотация. Рассмотрены сверточные свойства автокодирующей нейронной сети для задач обнаружения объектов на изображении. Для тренировок и тестирований были сгенерированы наборы данных в виде двумерных изображений с тремя цветовыми каналами. Изображения сгенерированы на основе трехмерной сцены, состоящей из таких объектов как сферы, кубы, цилиндры и модели "обезьянок". Проведена оценка времени тренировки сети на данных с различными конфигурациями и результата на выходе нейронной сети.
Ключевые слова: набор данных, тренировка, нейронная сеть, автокодер, свертка.
Abstract. The convolutional properties of the autoencoding neural network for the object detection problem in the image are considered. Data sets in the form of two-dimensional images with three color channels were generated for training and testing. The images are generated based on a three-dimensional scene consisting of objects such as spheres, cubes, cylinders and "monkey" models. The time of network training on the data with different configurations and the result at the output of the neural network were estimated.
Key words: data set, training, neural network, autoencoder, convolution.
DOI: https://doi.org/10.31649/1999-9941-2021-52-3-77-85.

Вступ

В сучасному світі часто використовують нейронні мережі для вирішення задач обробки зображень. Серед цих задач можуть бути такі актуальні задачі як фільтрація, чи виявлення об'єктів. Наприклад, на сьогоднішній день особлива увага приділяється підвищенню ефективності розв'язання задачі обробки потокових даних в процесі безперервного моніторингу навколишнього середовища. Зважаючи на складність формалізації подання потокових даних особлива увага приділяється вирішенню цієї задачі саме в реальному часі з урахуванням інформаційних завад та зашумленості даних.

Актуальність

Для вирішення поставлених задач досліджуються властивості автокодуючої нейронної мережі. Для тренування таких нейронних мереж можна застосовувати як дані з різною розмірністю так і зі сталою. Так в попередній роботі [1], що досліджувала нейронну мережу, в якості знешумлювача зображень було виявлено, що даний автокодер міг не тільки знешумити зображення, але і перетворити зображення з одного виду представлення в інший відповідно до набору тренувальних даних. Актуальність нового дослідження властивостей автокодуючої нейронної мережі обумовлена конфігураційною складністю потокових даних та великою частотою їх зміни. Формуються зображення деякої тривимірної сцени на базі наборів об'єктів-примітивів. Дані набори було отримано в процесі рендерингу з використанням відкритого програмного забезпечення "Blender". Кожен тренувальний набір містить 1000 таких зображень.

Мета

Метою даної роботи є визначення ефективних рішень у сфері задач виявлення об'єктів на зображенні.

Задачі

1. Змоделювати нейронну мережу.
2. Провести конфігурування нейронної мережі.
3. Оцінити результати тренувань і тестів на різних наборах даних.

Розв'язання задач

Розбір зображення потребує використання двовимірного оператора згортки. Це визначається відношенням між $g_i(x, y)$ вхідного зображення, функції $h(\varepsilon, \eta)$ ядра згортки H, та вихідного зображення $g_o(x, y)$:

$$g_0(x,y) = \sum_{\varepsilon=-\infty}^{\infty} \sum_{\eta=-\infty}^{\infty} g_i(\varepsilon,\eta)h(x-\varepsilon, y-\eta), \quad (1)$$

де x, y, ε та η - цілі числа.

Л. С. Ясенко, Я. М. Клятченко, 2021

Можуть існувати прямокутні ядра згортки, в яких $k_x \neq k_y$, проте більшість ядер згортки мають квадратну форму, $k_x = k_y = k$, де k - це не парне число і значно меньше від лінійної розмірності зображення N. Тож формулу 1 можна переписати наступним чином [2]:

$$g_o(x,y) = \sum_{\varepsilon=-(k-1)/2}^{(k-1)/2} \sum_{\eta=-(k-1)/2}^{(k-1)/2} g_i(\varepsilon,\eta)h(x-\varepsilon,y-\eta) \tag{2}$$

В процесі згортки (рис.1) використовується маска (ядро), що рухається від елемента до елемента і обчислює для кожного елемента деяку визначену величину [3].

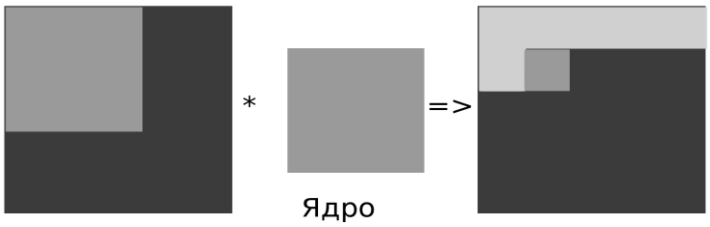

Рисунок 1 – Умовний схематичний вигляд операції згортки

Згорткова нейронна мережа

Архітектура згорткових нейронних мереж (ЗНМ) аналогічна структурі з’єднання нейронів у людському мозку і реалізувала організацією зорової кори. Окремі нейрони реагують на подразники лише в обмеженій зоні поля зору, відомої як Поля сприйняття. Колекція таких полів перекривається, щоб охопити всю зону зору [4]. Для дослідження було взято автокодер конфігурації, що показана на рис.2.

Рисунок 2 – Схематичне зображення згорткової нейронної мережі - автокодера

ЗНМ покращують свої можливості виявлення для незвично розміщених об'єктів, використовуючи шари проріджения (pool) для обмеженої інваріантності трансформації та обертання. Проріджения також дозволяє використовувати більш згортливі шари, зменшуючи споживання пам'яті [5].

Для моделювання нейронної мережі застосовувався популярний фреймворк Pytorch. В процесі тренування використовувалися 3 набори даних (вхідні зображення і перевірочні, на яких були виділені “мавпочки” окремим кольором): перший набір складався з зображень 128 на 128 пікселів, другий − 512 на 512, і третій − зображення різного прямокутного розміру − від 4 до 512 пікселів кратних 4.

Кратність розмірів вхідних даних зумовлена шарами проріджения, після проходження яких розмірність представлення даних змінюється в 2 рази.

Для розрахунку розмірності після проходження згорткових шарів в кодері використовуються наступні формули:

$$H_{out} = \left[\frac{H \ +2\times Py-Dy\times(Ky-1)-1}{Sy}+1\right], \tag{3}$$

$$W_{out} = \left[\frac{W \ +2\times Px-Dx\times(Kx-1)-1}{Sx}+1\right], \tag{4}$$

де H_{out} − висота вихідного шару, H_{in} − висота вхідного шару, Py − висота відступу, Dy − відстань між елементами ядра по вертикалі, Ky − розмір ядра по вертикалі, Sy − крок згортки по вертикалі, W_{out} − ширина вихідного шару, W_{in} − ширина вхідного шару, Px − ширина відступу, Dx відстань між елементами ядра по горизонталі, Kx − розмір ядра по горизонталі, Sx − крок згортки по горизонталі[6].

Для декодера використовують формули для шарів зворотної згортки:

$$H_{out} = (H| \quad |-1) \times Sy - 2 \times Py + Dy \times (Ky - 1) + \text{Pyo} + 1, \tag{5}$$

$$W_{out} = (W| \quad |-1) \times Sx - 2 \times Px + Dx \times (Kx - 1) + \text{Pxo} + 1, \tag{6}$$

де Pyo та Pxo - відступи в вихідному шарі, а решта змінних позначають аналогічно до формул 3, 4[7].

Проріджуючі шари в цій нейронній мережі використовують фільтр марсимуму з ядром 2 на 2.

Конфігурація нейронної мережі (автокодера) має такий вигляд, відповідно до фреймворку Pytorch:

```
Model(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv2): Conv2d(16, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (t_conv1): ConvTranspose2d(64, 64, kernel_size=(2, 2), stride=(2, 2))
  (t_conv2): ConvTranspose2d(64, 16, kernel_size=(2, 2), stride=(2, 2))
  (conv_out): Conv2d(16, 3, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
)
```

Результати

Розглянемо результати тренувань і тестів на різних наборах даних. Усі набори даних тренувалися по 64 епохи з використанням 75% зображень.

Ця нейронна мережа застосовувалася на наборах з зображеннями 128 на 128 пікселів і тренувалася 479.63 одиниці часу. Після тренування середньоквадратичне відхилення перевірочних зображень від зображень на виході становило 0.00267.

На наступному графіку наведено залежність втрат (відхилення/losses) від кількості епох тренування (epochs).

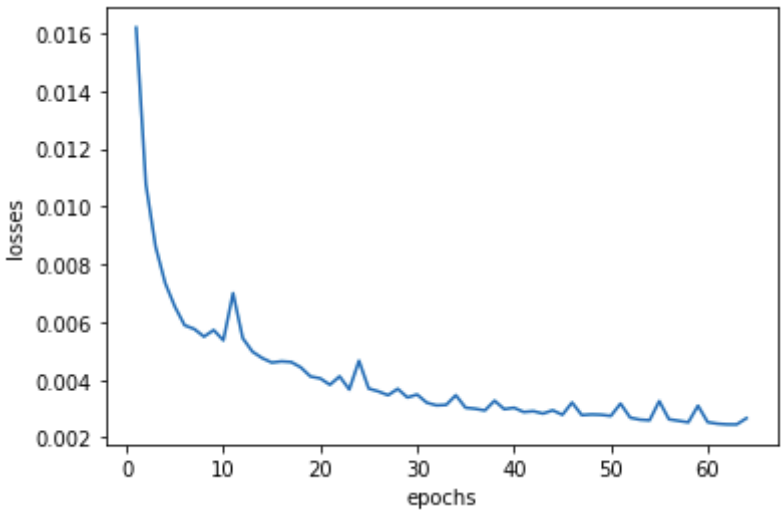

Рисунок 3 – Графік залежності під час тренування на першому наборі даних

Для тестувань використовувалося 25 % зображень. З усіх наборів даних. Отримано наступні дані. Найгірший результат для тестування на наборі зображень 128 на 128 пікселів становить 0.0186, найкращий − 0.00067, середній 0.044. Для тестування на наборі зображень 512 на 512 пікселів найгіршим є ре-

зультат 0.0327, найкрашим − 0.00192, середнім − 0.0126. Для тестування на наборі зображень від 4 до 512 пікселів найгіршим є результат 0.0382, найкрашим − 0.00018, середнім − 0.0091. Згідно з цими даними найкраще, дана мережа підходить для виявлення об’єктів на тому типі набору даних, на якому її було натреновано.

Для певності розглянемо зображення на виході нейронної мережі.

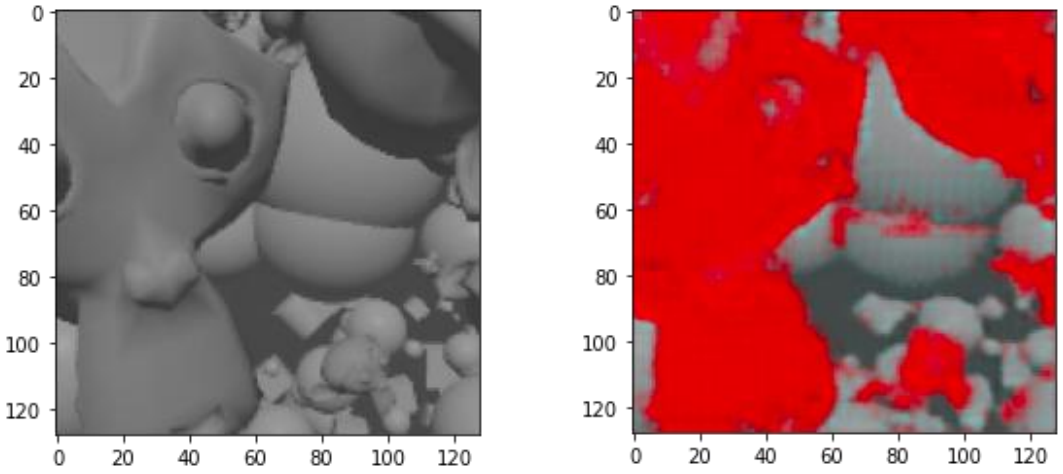

Рисунок 4 – Вхідне та вихідне зображення на наборі 128 на 128

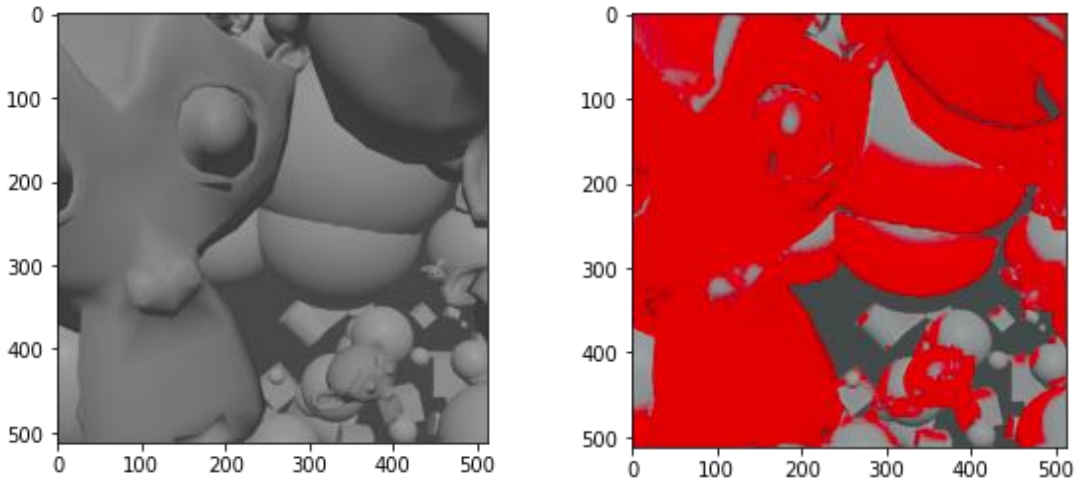

Рисунок 5 – Вхідне та вихідне зображення на наборі 512 на 512

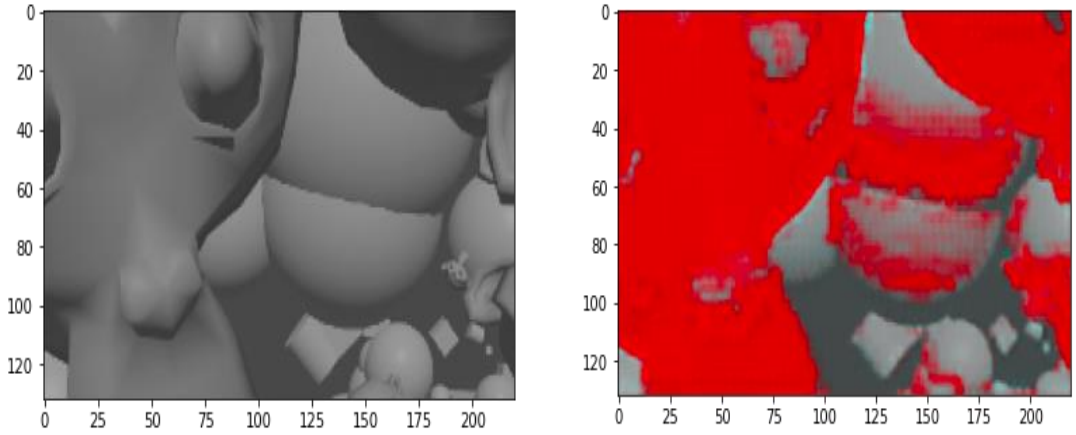

Рисунок 6 – Вхідне та вихідне зображення на наборі від 4 до 512

Із рисунків 4 − 6 видно, що на двох останніх наборах нейронна мережа набагато гірше обробляє тіні об’єктів і позначає їх як “мавпочок” − червоним.

Наступна мережа застосовувалася на наборах з зображеннями 512 на 512 пікселів, вона тренувалася 3910.46 одиниці часу. Після тренування середньоквадратичне відхилення перевірочних зображень від зображень на виході становило 0.00479.

На наступному графіку наведено залежність втрат (відхилення/losses) від кількості епох тренування (epochs).

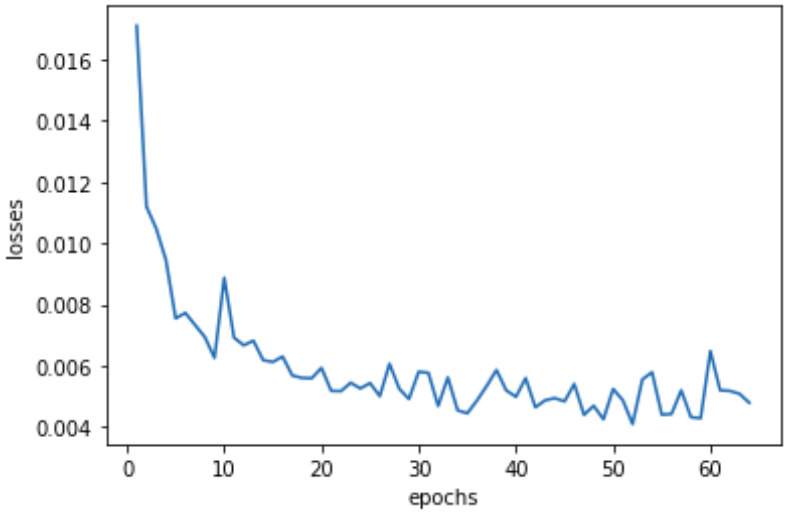

Рисунок 7 –Графік залежності під час тренування на другому наборі даних

Тестування відбувалося аналогічно. Після тренування були отримані такі дані. Найгірший результат для тестування на наборі зображень 128 на 128 пікселів становить 0.0182, найкращий − 0.00183, середній 0.0075. Для тестування на наборі зображень 512 на 512 пікселів найгіршим є результат 0.0167, найкращим − 0.00138, середнім − 0.0060. Для тестування на наборі зображень від 4 до 512 пікселів найгіршим є результат 0.0217, найкрашим − 0.00033, середнім − 0.0062. Згідно з цими даними найкраще, дана мережа, так само, підходить для виявлення об’єктів на тому типі набору даних, на якому її було натреновано.

Розглянемо зображення на виході нейронної мережі.

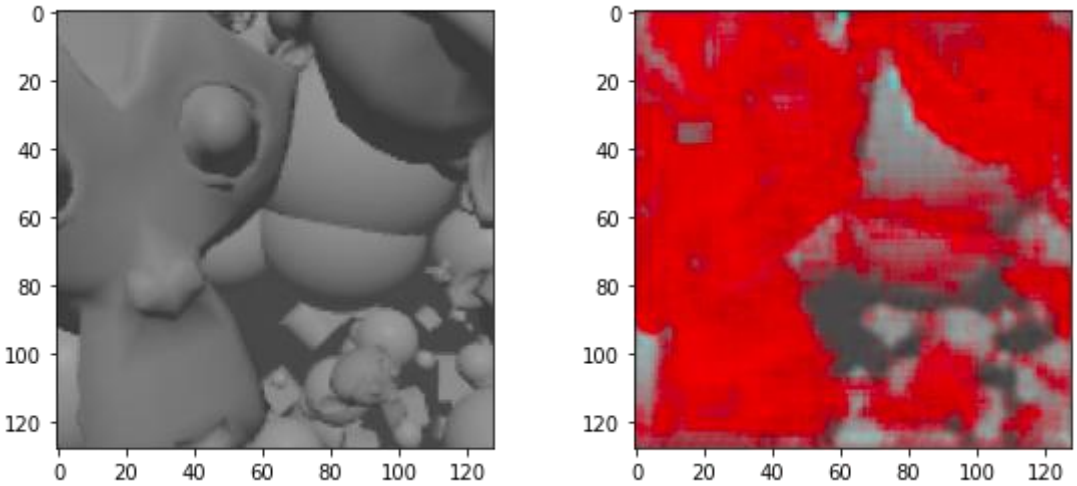

Рисунок 8 – Вхідне та вихідне зображення на наборі 128 на 128

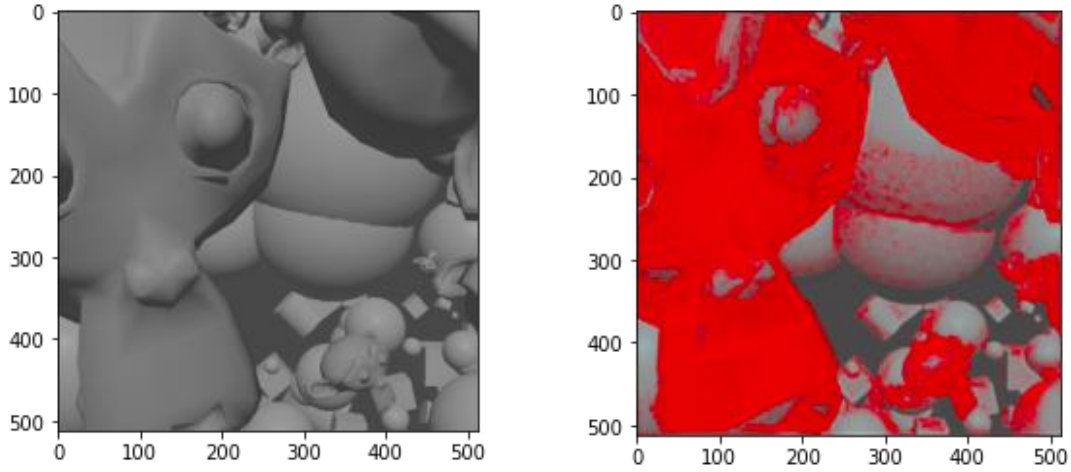

Рисунок 9 – Вхідне та вихідне зображення на наборі 512 на 512

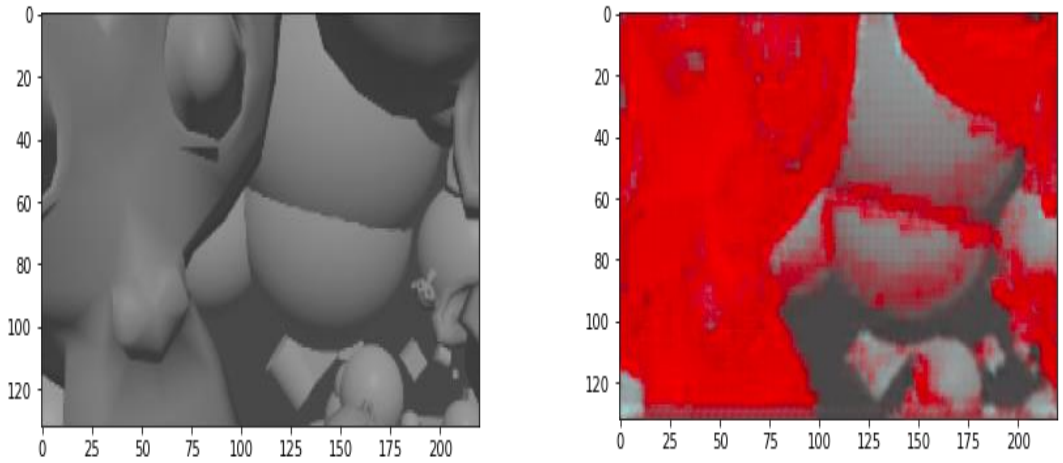

Рисунок 10 – Вхідне та вихідне зображення на наборі від 4 до 512

Вдповідно до зображень на рисунках 14-16 можна сказати, що дана нейронна мережа, так само, не може розрізняти тіні і значно гірше працює на інших наборах даних.

Тож розглянемо третій варіант, де нейронна мережа тренувалася на наборі різних прямокутних зображень. Час її тренування тривав 963.08 одиниці часу, і по його закінченню, середньоквадратичне відхилення становило 0.00561. Графік тренування представлено рисунком 11.

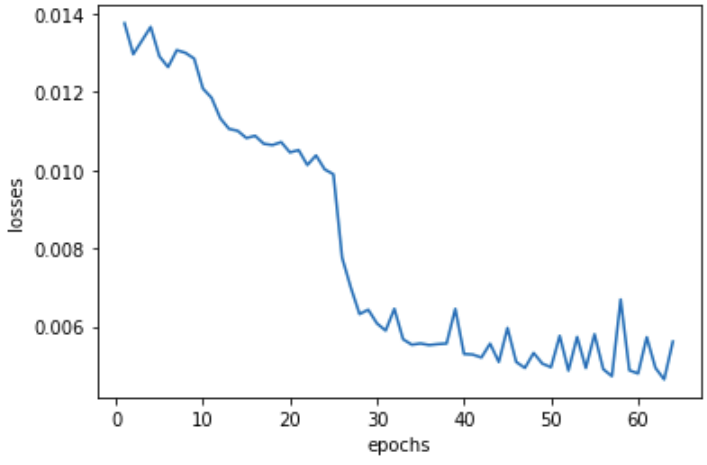

Рисунок 11 – Графік залежності під час тренування на третьому наборі даних

Тестування проводилося аналогічно до попередніх двох прикладів. Найгірший результат для тестування на наборі зображень 128 на 128 пікселів становить 0.0340, найкращий − 0.00378, середній 0.0154. Для естування на наборі зображень 512 на 512 пікселів найгіршим є результат 0.0443, найкращим − 0.00624, середнім − 0.0212. Для естування на наборі зображень від 4 до 512 пікселів найгіршим є результат 0.0647, найкрашим − 0.00180, середнім − 0.0186. Згідно з цими даними найкраще, дана мережа, так само, підходить для виявлення об’єктів на тому типі набору даних, на якому її було натреновано.

Розглянемо зображення на виході нейронної мережі.

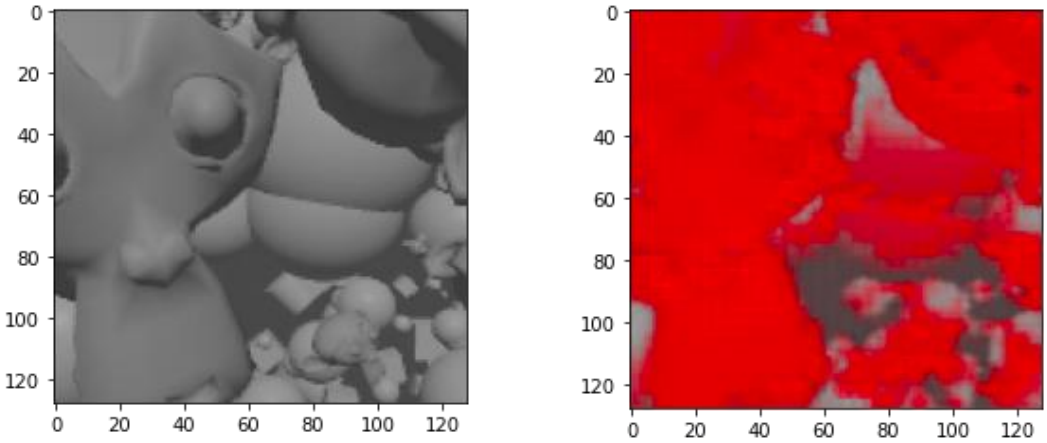

Рисунок 12 – Вхідне та вихідне зображення

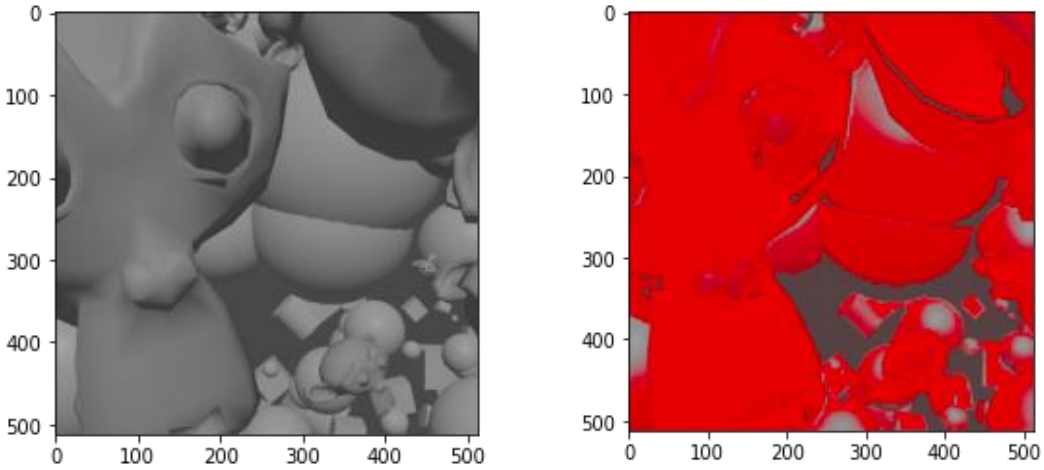

Рисунок 13 – Вхідне та вихідне зображення

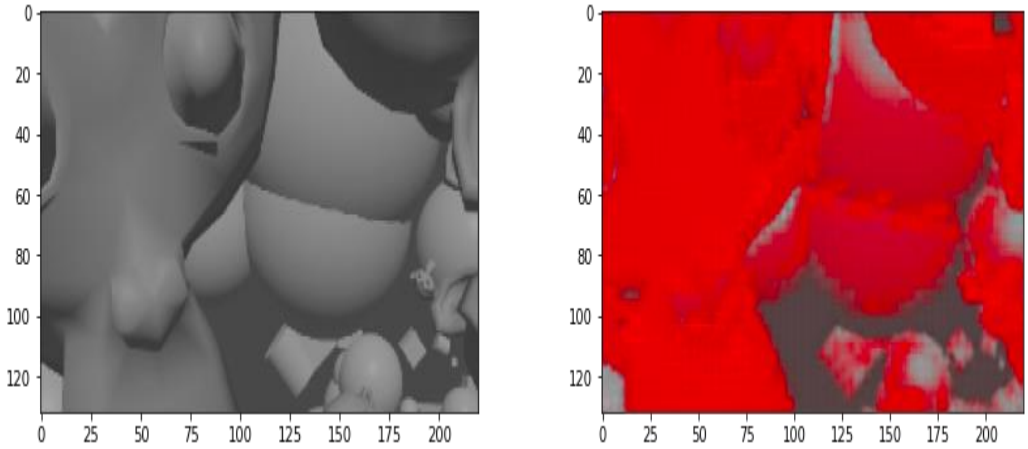

Рисунок 14 – Вхідне та вихідне зображення

За числовими показниками і за рисунками 12-14 ця нейронна мережа була натренована найгірше в порівнянні з попередніми, і тому на всіх вихідних зображеннях позначала всі об’єкти червоним.

Висновки

1. Зазвичай дана модель у вигляді автокодера не застосовується для виявлення об’єктів на зображені. Зображення що подавалися на вхід, для тренування нейронної мережі, мали бути кратні 4, щоб під час відновлення за допомогою декодера розміри вхідного і вихідного зображення збігалися. У разі використання даних не кратних 4 необхідно, відповідно до формул 3-6 змінити внутрішні шари нейронної мережі, якщо всі зображення в наборі будуть однакового розміру, або додавати до зображення пусті рядки так колонки пікселів, щоб воно стало кратним 4.

2. Конфігурація нейронної мережі відбувається в процесі її тренування, шляхом порівняння вихідних і очікуваних зображень. Дана конфігурація не є універсальною, оскільки портебує дані специфічної розмірності на вході. Для того, щоб використовувати цю нейронну мережу на будь яких пласких зображеннях з 3 каналами даних необхідно додати в алгоритм їх обробку перед і після проходження автокодера.

3. При тренуванні нейронних мереж виникають такі проблеми як “недотренування”, або “перетренування”. В даному випадку в усіх трьох випадках не вистачило ні часу для тренування, ні даних. Що свідчить про “недотренування”. Для покращення результатів необхідно збільшити час а також збільшити вибірку даних на яких можна натренувати мережу.

Список літератури

[1] L. Yasenko, Y. Klyatchenko and O. Tarasenko-Klyatchenko, «Image noise reduction by denoising autoencoder», *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, 2020, pp. 351-355, doi:10.1109/DESSERT50317.2020.9125027.

[2] Michael Seul, Lawrence O'Gorman, Michael J. Sammon, Practical Algorithms for Image Analysis. [Online]. Available: https://books.google.com.ua/books?id=5xcIErZZIN8C&printsec=frontcover&dq=Practical+Algorithms+for+Image+Analysis&hl=uk&sa=X&redir_esc=y#v=onepage&q=Practical%20Algorithms%20for%20Image%20Analysis&f=false.

[3] Р. Гонсалес, Р. Вудс, *Цифровая обработка изображений*. Москва: Техносфера. 2005, 1072с.

[4] A Comprehensive Guide to Convolutional Neural Networks − the ELI5 way. Sumit Saha. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[5] Convolutional Neural Network (CNN). [Online]. Available: https://developer.nvidia.com/discover/convolutional-neural-network.

[6] Conv2d. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html.

[7] ConvTranspose2d. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html#torch.nn.ConvTranspose2d.

Стаття надійшла: 09.11.2021.

References

[1] L. Yasenko, Y. Klyatchenko and O. Tarasenko-Klyatchenko, «Image noise reduction by denoising autoencoder», *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, 2020, pp. 351-355, doi:10.1109/DESSERT50317.2020.9125027.

[2] Michael Seul, Lawrence O'Gorman, Michael J. Sammon, Practical Algorithms for Image Analysis. [Online]. Available: https://books.google.com.ua/books?id=5xcIErZZIN8C&printsec=frontcover&dq=Practical+Algorithms+for+Image+Analysis&hl=uk&sa=X&redir_esc=y#v=onepage&q=Practical%20Algorithms%20for%20Image%20Analysis&f=false.

[3] R. Gonzalez, R. Woods. Digital image processing. Moscow: Technosphere. 2005, 1072 p. [in Rusian].

[4] A Comprehensive Guide to Convolutional Neural Networks − the ELI5 way. Sumit Saha. [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[5] Convolutional Neural Network (CNN). [Online]. Available: https://developer.nvidia.com/discover/convolutional-neural-network.

[6] Conv2d. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Conv2d.html.

[7] ConvTranspose2d. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.ConvTranspose2d.html#torch.nn.ConvTranspose2d.

Відомості про авторів

Ясенко Лев Сергійович – магістрант кафедри системного програмування і спеціалізованих комп'ютерних систем.

Клятченко Ярослав Михайлович – кандидат технічних наук, доцент кафедри системного програмування і спеціалізованих комп'ютерних систем.

Л. С. Ясенко, Я. М. Клятченко

СВЕРТОЧНЫЕ СВОЙСТВА НЕЙРОННОЙ СЕТИ НА ОСНОВЕ АВТОКОДЕРА

Национальный технический университет Украины «Киевский политехнический институт имени Игоря Сикорского», Киев

L. S. Yasenko, Y. M. Klyatchenko

CONVOLUTIONAL PROPERTIES OF A NEURAL NETWORK BASED ON AUTOENCODERS

National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Kyiv