



**AKADEMIA GÓRNICZO – HUTNICZA  
IM. STANISŁAWA STASZICA W KRAKOWIE  
WYDZIAŁ INŻYNIERII MECHANICZNEJ I ROBOTYKI**

**PRACA DYPLOMOWA  
inżynierska**

**Aplikacja automatycznie wykrywająca naruszenia  
regulaminu portalu społecznościowego z wykorzystaniem  
sieci neuronowych oraz biblioteki PyTorch**

*Automated detection of regulation violation of the social media platform using  
neural network and PyTorch library*

*Autor:*

*Kierunek studiów:*

*Opiekun pracy:*

**Szymon Maksymilian Maroszek**

Inżynieria Mechatroniczna

**dr inż. Mariusz Gibiec**

.....

*podpis*

Kraków, rok 2020

Kraków, dnia 18.01.2021r.

Szymon Maksymilian Maroszek  
*Imiona i nazwisko studenta*

Inżynieria Mechatroniczna, pierwszego stopnia, stacjonarne, ogólnoakademicki  
*Kierunek, poziom, forma studiów*

Wydział Inżynierii Mechanicznej i Robotyki  
*Nazwa Wydziału*

dr inż. Mariusz Gibiec  
*Imiona i nazwisko opiekuna pracy dyplomowej*

**ja niżej podpisany(-a) oświadczam, że:**

jako twórca / współtwórca\* pracy dyplomowej inżynierskiej / licencjackiej / magisterskiej\* pt.  
**Aplikacja automatycznie wykrywająca naruszenia regulaminu portalu społecznościowego z wykorzystaniem sieci neuronowych oraz biblioteki PyTorch**

1. **uprzedzony(-a) o odpowiedzialności karnej** na podstawie art. 115 ust. 1 i 2 ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (t.j. Dz.U. z 2018 r. poz. 1191, z późn. zm.): „Kto przywłaszcza sobie autorstwo albo wprowadza w błąd co do autorstwa całości lub części cudzego utworu albo artystycznego wykonania, podlega grzywnie, karze ograniczenia wolności albo pozbawienia wolności do lat 3. Tej samej karze podlega, kto rozpowszechnia bez podania nazwiska lub pseudonimu twórcy cudzy utwór w wersji oryginalnej albo w postaci opracowania, artystyczne wykonanie albo publicznie zniekształca taki utwór, artystyczne wykonanie, fonogram, wideogram lub nadanie.”, **a także uprzedzony(-a) o odpowiedzialności dyscyplinarnej** na podstawie art. 307 ust. 1 ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668, z późn. zm.) „Student podlega odpowiedzialności dyscyplinarnej za naruszenie przepisów obowiązujących w uczelni oraz za czyn uchybiający godności studenta.” **niniejszą pracę dyplomową wykonałem(-am) osobiście i samodzielnie i nie korzystałem(-am) ze źródeł innych niż wymienione w pracy;**
2. praca dyplomowa jest wynikiem mojej twórczości i nie narusza praw autorskich innych osób;
3. wersja elektroniczna przedłożonej w wersji papierowej pracy dyplomowej jest wersją ostateczną, która będzie przedstawiona komisji przeprowadzającej egzamin dyplomowy;
4. praca dyplomowa nie zawiera informacji podlegających ochronie na podstawie przepisów o ochronie informacji niejawnych ani nie jest pracą dyplomową, której przedmiot jest objęty tajemnicą prawnie chronioną;
5. [ ]\*\* udzielam nieodpłatnie Akademii Górniczo-Hutniczej im. Stanisława Staszica w Krakowie licencji niewyłącznej, bez ograniczeń czasowych, terytorialnych i ilościowych na udostępnienie mojej pracy dyplomowej w sieci Internet za pośrednictwem Repozytorium AGH.

.....  
*czytelny podpis studenta*

**Jednocześnie Uczelnia informuje, że:**

1. zgodnie z art. 15a ww. ustawy o prawie autorskim i prawach pokrewnych uczelni przysługuje pierwszeństwo w opublikowaniu pracy dyplomowej studenta. Jeżeli uczelnia nie opublikowała pracy dyplomowej w terminie 6 miesięcy od dnia jej obrony, autor może ją opublikować, chyba że praca jest częścią utworu zbiorowego. Ponadto uczelnia jako podmiot, o którym mowa w art. 7 ust. 1 pkt 1 ustawy z dnia 20 lipca 2018 r. – Prawo o szkolnictwie wyższym i nauce (Dz. U. z 2018 r. poz. 1668, z późn. zm.), może korzystać bez wynagrodzenia i bez konieczności uzyskania zgody autora z utworu stworzonego przez studenta w wyniku wykonywania obowiązków związanych z odbywaniem studiów, udostępniać utwór ministrowi właściwemu do spraw szkolnictwa wyższego i nauki oraz korzystać z utworów znajdujących się w prowadzonych przez niego bazach danych, w celu sprawdzania z wykorzystaniem Jednolitego Systemu Antyplagiatowego. Minister właściwy do spraw szkolnictwa wyższego i nauki może korzystać z prac dyplomowych znajdujących się w prowadzonych przez niego bazach danych w zakresie niezbędnym do zapewnienia prawidłowego utrzymania i rozwoju tych baz oraz współpracujących z nimi systemów informatycznych;
2. w świetle art. 342 ust. 3 pkt 5 i art. 347 ust. 1 ustawy Prawo o szkolnictwie wyższym i nauce minister właściwy do spraw szkolnictwa wyższego i nauki prowadzi bazę danych zwaną repozytorium pisemnych prac dyplomowych, która obejmuje: tytuł i treść pracy dyplomowej; imiona i nazwisko autora pracy dyplomowej; numer PESEL autora pracy dyplomowej, a w przypadku jego braku – numer dokumentu potwierdzającego tożsamość oraz nazwę państwa, które go wydało; imiona i nazwisko promotora pracy dyplomowej, numer PESEL, a w przypadku jego braku – numer dokumentu potwierdzającego tożsamość oraz nazwę państwa, które go wydało; imiona i nazwisko recenzenta pracy dyplomowej, numer PESEL, a w przypadku jego braku – numer dokumentu potwierdzającego tożsamość oraz nazwę państwa, które go wydało; nazwę uczelni; datę zdania egzaminu dyplomowego; kierunek, poziom i profil studiów. Ponadto, zgodnie z art. 347 ust. 2-5 ustawy Prawo o szkolnictwie wyższym i nauce ww. dane wprowadzają do Zintegrowanego Systemu Informacji o Szkolnictwie Wyższym i Nauce POL-on (System POL-on) rektorzy. Dostęp do danych przysługuje promotorowi pracy dyplomowej oraz Polskiej Komisji Akredytacyjnej, a także ministrowi w zakresie niezbędnym do prawidłowego utrzymania i rozwoju repozytorium oraz systemów informatycznych współpracujących z tym repozytorium. Rektor wprowadza treść pracy dyplomowej do repozytorium niezwłocznie po zdaniu przez studenta egzaminu dyplomowego. W repozytorium nie zamieszcza się prac zawierających informacje podlegające ochronie na podstawie przepisów o ochronie informacji niejawnych.

\* - niepotrzebne skreślić;

\*\* - należy wpisać TAK w przypadku wyrażenia zgody na udostępnienie pracy dyplomowej, NIE – w przypadku braku zgody; nieuzupełnione pole oznacza brak zgody na udostępnienie pracy.

**Imię i nazwisko studenta:** Szymon Maroszek

**Numer albumu:** 300061

*Nazwa Wydziału: Wydział Inżynierii Mechanicznej i Robotyki*

*Kierunek studiów: Inżynieria Mechatroniczna*

*Poziom studiów: I stopień*

*Profil studiów: -----*

*Forma studiów: stacjonarne*

*Imię i nazwisko opiekuna pracy dyplomowej: dr inż. Mariusz Gibiec*

## **OŚWIADCZENIA STUDENTA**

### **Niniejszym oświadczam, że:**

- jestem gotowy/a przystąpić do egzaminu dyplomowego przeprowadzanego w trybie zdalnym z wykorzystaniem technologii informatycznych zapewniających kontrolę jego przebiegu i rejestrację,
- posiadam dostęp do łącza internetowego o przepustowości wystarczającej do przesyłania transmisji dźwięku i obrazu,
- posiadam odpowiedni sprzęt (komputer lub inne urządzenie) wyposażony w mikrofon, głośniki oraz kamerę, umożliwiający przesyłanie dźwięku i obrazu,
- jestem świadomy, że egzamin dyplomowy przeprowadzany w trybie zdalnym może być rejestrowany<sup>1</sup>, w związku z tym wyrażam zgodę na jego rejestrację,
- akceptuję zasady organizacji i przeprowadzania egzaminu dyplomowego przeprowadzanego w trybie zdalnym ustalone w Akademii Górniczo-Hutniczej im. Stanisława Staszica w Krakowie.

Kraków, 18.01.2021  
*miejsce i data*

.....  
*podpis studenta*

---

<sup>1</sup> Zgodnie z art. 76a ustawy z dnia 20 lipca 2018 r. Prawo o szkolnictwie wyższym i nauce (t.j. Dz. U. z 2020 r. poz. 85, z późn. zm.) uczelnia może zorganizować weryfikację osiągniętych efektów uczenia się określonych w programie studiów, w szczególności przeprowadzać zaliczenia i egzaminy kończące określone zajęcia oraz egzaminy dyplomowe, poza siedzibą uczelni lub poza jej filią z wykorzystaniem technologii informatycznych zapewniających kontrolę ich przebiegu i rejestrację. Zasady weryfikacji osiągniętych efektów uczenia się w ww. sposób Uczelnia ma obowiązek udostępnić w Biuletynie Informacji Publicznej na swojej stronie podmiotowej.

Kierunek studiów: Inżynieria Mechatroniczna

Specjalność: -----

*Imię i nazwisko studenta: Szymon Maroszek*

**Praca dyplomowa inżynierska**

Aplikacja automatycznie wykrywająca naruszenia regulaminu portalu społecznościowego z wykorzystaniem sieci neuronowych oraz biblioteki PyTorch

Opiekun: *dr inż. Mariusz Gibiec*

**STRESZCZENIE**

Praca inżynierska opisuje projekt oraz implementację inteligentnych rozwiązań monitorujących treści udostępniane na portalu społecznościowym. W pracy zawarty został wstęp wyjaśniający motywację podjęcia tematu wraz z przeglądem istniejących rozwiązań. Następnie zdefiniowany został cel pracy oraz założenia projektowe. Praca przeprowadza czytelnika przez proces implementacji inteligentnych rozwiązań na stronie internetowej. Pierwszym etapem opisanym w pracy jest kolekcjonowanie oraz dostosowywanie zestawu danych uczących. Następnie przedstawiony został proces projektowania i optymalizacji własnej sieci neuronowej oraz sposób wykorzystania sieci wstępnie przetrenowanych. Kolejnym krokiem projektu było stworzenie od zera portalu społecznościowego z wykorzystaniem biblioteki Django oraz przygotowanie i implementacja rozwiązań zabezpieczających portal przed niepożądanymi treściami na podstawie stworzonej sieci neuronowej. Ostatnim etapem pracy jest rozwinięcie tematu możliwych rozwiązań oraz skalowalności aplikacji. Podsumowanie pracy rozwija wnioski wyciągnięte podczas projektowania oraz testów rozwiązań.

Field of Study: Mechatronic Engineering

Profile/speciality: -----

*First name and family name of the student: Szymon Maroszek*

**Engineer Diploma Thesis**

*Automated detection of regulation violation of the social media platform using neural network and PyTorch library*

Supervisor: *dr inż. Mariusz Gibiec*

**SUMMARY**

The following thesis describes the design and implementation of intelligent content monitoring solutions on a social media site. The thesis includes an introduction explaining the motivation to take up the topic along with an overview of existing solutions. The thesis also includes its purpose and design assumptions defined. The reader is guided through the process of implementing intelligent solutions on the website. The first stage described in the thesis is collecting and adjusting the training data set. Then, the process of designing and optimizing one's own neural network and the way of using pre-trained networks is presented. The next step of the project is to create a social media website from scratch using the Django library, and to prepare and implement solutions protecting the site against unwanted content based on the created neural network. The last step is to develop the topic of possible solutions and application scalability. The work summary develops the conclusions drawn during the design and testing of the application.

Kraków, dnia 18.01.2021r.

**AKADEMIA GÓRNICZO-HUTNICZA  
WYDZIAŁ INŻYNIERII MECHANICZNEJ I ROBOTYKI**

**TEMATYKA PRACY DYPLOMOWEJ**

Szymon Maroszek

**Tytuł pracy dyplomowej:**

Aplikacja automatycznie wykrywająca naruszenia regulaminu portalu społecznościowego z wykorzystaniem sieci neuronowych oraz biblioteki PyTorch

Promotor pracy: *dr inż. Mariusz Gibiec*

Recenzent pracy: *dr hab. inż. Joanna Iwaniec, prof. AGH*

.....  
*Podpis dziekana*

**PLAN PRACY DYPLOMOWEJ:**

1. Wstęp
2. Cel pracy
3. Kolekcjonowanie danych
4. Projektowanie oraz optymalizacja sieci neuronowej
5. Implementacja sieci na stronie internetowej
6. Kierunek rozwoju
7. Wyniki i podsumowanie
8. Literatura

**Praktyka (dyplomowa):**

ING Business Shared Services B.V. spółka z ograniczoną odpowiedzialnością  
Oddział w Polsce  
ul. Konduktorska 35, 40-155 Katowice

Kraków, 18.01.2021 .....  
*data                      podpis dyplomanta*

.....  
*podpis promotora*

Termin oddania do dziekanatu: \_\_\_\_\_ 20\_\_ r.

# Spis treści

<b>1. Wstęp</b>	<b>9</b>
<b>2. Cel pracy</b>	<b>10</b>
2.1 Założenia projektowe	10
<b>3. Kolekcjonowanie danych</b>	<b>11</b>
3.1 Określenie wymagań	11
3.2 Dostępne narzędzia oraz źródła	12
3.3 Selekcja oraz preparacja danych	14
<b>4. Projektowanie oraz optymalizacja sieci neuronowej</b>	<b>21</b>
4.1 Projektowanie konwolucyjnej sieci neuronowej	21
4.2 Optymalizacja oraz uczenie modelu	34
4.3 Sieci wstępnie przetrenowane	37
4.4 Podsumowanie	39
<b>5. Implementacja sieci na stronie internetowej</b>	<b>39</b>
5.1 Tworzenie strony	39
5.2 Dodawanie tagów do rozpoznanych obiektów	49
5.3 Shadow banning	50
5.4 Rozpoznawanie obiektów podczas transmisji na żywo	57
5.5 Automatyczne wyłączanie transmisji	58
5.6 Automatyczne ukrywanie elementów odwracających uwagę	60
<b>6. Kierunek rozwoju</b>	<b>62</b>
6.1 Możliwe rozwiązania	62
6.2 Skalowalność aplikacji	63
<b>7. Wyniki i podsumowanie</b>	<b>64</b>
<b>8. Literatura</b>	<b>65</b>



# 1. Wstęp

Wraz z pędzącym postępem technologicznym coraz większa część naszego życia przenosi się do sieci. Zdobywanie wiedzy, rozrywka czy życie towarzyskie nierozłącznie połączone są z wachlarzem dostępnych stron internetowych.

Ostatnie globalne wydarzenia całkowicie odmieniły życie, które znamy. Zamknięcie restauracji, kin czy siłowni oraz globalny lock-down zmusiły ludzi z każdego zakątka świata do zmiany swoich przyzwyczajeń i dostosowania się do nowej rzeczywistości.

Praca zdalna, zdalne nauczanie czy zakupy online to tylko wierzchołek góry lodowej czynności, bez których świat stanąłby w miejscu. Rosnące zapotrzebowanie na możliwość wykonania podstawowych zadań bez wychodzenia z domu sprawiła, że usługi internetowe zaczęły rozwijać się szybciej niż kiedykolwiek.

W czasach, w których nawet mały osiedlowy sklep ma swoją stronę internetową czy konto na Instagramie, bezpieczeństwo w sieci nie może być pomijane. Większość dużych stron internetowych posiada pewnego rodzaju regulamin, który w jasny sposób przedstawia czego nie wolno robić w obrębie danego portalu. Regulamin w pewnym stopniu chroni użytkowników oraz sam portal przed niechcianą treścią pod warunkiem, że treści łamiące regulamin są na bieżąco usuwane przez administratorów. Niestety wraz z rozwojem portalu oraz zwiększeniem ilości udostępnianych przez użytkowników treści, egzekwowanie regulaminu staje się coraz cięższe.

Dlatego właśnie branżowi giganci tacy jak YouTube czy Facebook implementują na swoich portalach narzędzia, które znacznie ułatwiają usuwanie treści niezgodnych z regulaminem portalu.

Tylko w pierwszym kwartale 2019 roku z portalu YouTube usunięte zostało ponad 8 milionów filmów, z czego ponad 75% z nich została automatycznie oznaczona jako niezgodna z regulaminem portalu przez sztuczną inteligencję.[25] Większość z tych zagrożeń została zidentyfikowana jeszcze zanim ktokolwiek je wyświetlił. Pokazuje to, że zastosowanie sztucznej inteligencji na portalach społecznościowych może mieć świetne rezultaty.

Do podobnego wniosku doszedł Facebook, na którym również doświadczyć możemy działania sztucznej inteligencji. Prawdopodobnie Facebook korzysta z wielu metod, najłatwiejszymi do zauważenia są jednak rozwiązania odpowiedzialne za monitorowanie

udostępnianych treści. Portal filtruje zawartość zaczynając od treści najbardziej oczywistych takich jak przemoc i nienawiść kończąc na usuwaniu ogłoszeń sprzedażowych ze zwierzętami.

Przykłady te pokazują skuteczność sztucznej inteligencji w monitorowaniu zawartości portalu, są to jednak przykłady rozwiązań zaimplementowanych przez wielomiliardowe przedsiębiorstwa z ogromnymi zespołami inżynierów.

Co zatem jeśli chcielibyśmy zaimplementować podobne rozwiązania na mniejszym portalu bez wydawania milionów dolarów?

Niniejsza praca skupia się na przeprowadzeniu czytelnika przez proces przygotowania podobnych rozwiązań na własnym portalu społecznościowym.

## **2. Cel pracy**

Cel niniejszej pracy podzielić można na dwa etapy. Pierwszym z nich jest przygotowanie oraz optymalizacja konwolucyjnej sieci neuronowej rozpoznającej obiekty najczęściej udostępniane na portalach społecznościowych.

Drugim etapem jest zaprojektowanie inteligentnych rozwiązań monitorujących przepływ treści udostępnianych na portalu społecznościowym oraz weryfikacja ich zgodności z regulaminem wykorzystując przygotowaną wcześniej sieć neuronową.

Zaprezentowane w pracy rozwiązania są wynikiem połączenia wiedzy pochodzącej z kilku dziedzin branży IT, takich jak Data Science, Machine Learning czy Frontend oraz Backend Development.

Praca szczegółowo pokazuje proces projektowania oraz implementacji wspomnianych rozwiązań, zaczynając od doboru odpowiednich technologii, przez kolekcjonowanie danych niezbędnych do trenowania sieci, kończąc na zbudowaniu prostego portalu społecznościowego.

### **2.1. ZAŁOŻENIA PROJEKTOWE**

Przed przystąpieniem do pracy dobrane zostały założenia projektowe:

- Algorytm działający na stronie rozpoznaje osiem obiektów najczęściej udostępnianych na portalach społecznościowych. Dwa z nich są obiektami zabronionymi. Kategorie wybranych obiektów prezentują się następująco:
  - Zdjęcia przedstawiające **dzieci**
  - Zdjęcia przedstawiające **samochody**
  - Zdjęcia przedstawiające **koty**
  - Zdjęcia przedstawiające **psy**
  - Zdjęcia przedstawiające **jedzenie**
  - Zdjęcia przedstawiające **buty**
  - Zdjęcia przedstawiające **broń - niezgodne z regulaminem portalu**
  - Zdjęcia będące **reklamą** jakiegoś **produktu - niezgodne z regulaminem portalu**
- Portal jest w stanie obronić się przed treściami łamiącymi regulamin bez ingerencji administratora, zarówno w przypadku udostępnionych zdjęć jak i w czasie transmisji wideo na żywo.

Sprecyzowane założenia projektowe pozwoliły na dobranie odpowiednich technologii:

- Projektowanie sieci neuronowych oraz ich trening wykonane zostaną z wykorzystaniem biblioteki PyTorch.
- Struktura strony zbudowana zostanie w środowisku Django.

### 3. Kolekcjonowanie danych

#### 3.1. OKREŚLENIE WYMAGAŃ

Uwzględniając założenia projektowe oraz aktualne trendy mediów społecznościowych określone zostały wymagania dotyczące danych uczących.

Dane podzielić można na osiem kategorii, każda z nich reprezentuje jeden obiekt. Aby zapewnić najbardziej optymalny proces uczenia, przygotowane kategorie powinny zawierać zbliżoną ilość zdjęć, a obrazy nie powinny zawierać szumów i znaków wodnych. Zestaw

danych uczących nie może także być zbyt mały, ilość zdjęć bezpośrednio wpływa na jakość nauki.

Przy dużych zestawach danych ich ogólna jakość spada, dlatego ostatnim założeniem przy kolekcjonowaniu danych było utrzymanie wysokiej jakości zdjęć nawet kosztem zmniejszenia liczności zestawu.

### 3.2. DOSTĘPNE NARZĘDZIA ORAZ ŹRÓDŁA

Kolekcjonowanie danych najlepiej rozpocząć od zebrania możliwie jak największej ilości zdjęć z gotowych już zestawów danych uczących.

Jednym z najlepszych miejsc do znalezienia interesujących nas danych jest strona **Kaggle.com**. Po przejrzaniu dostępnych na wyżej wymienionym serwisie zestawów, pobrane zostały trzy:

- Zdjęcia przedstawiające koty - 5 000 zdjęć
- Zdjęcia przedstawiające psy - 5 000 zdjęć
- Zdjęcia przedstawiające jedzenie - 100 000 zdjęć, zredukowane następnie do 5 000 zdjęć

Niestety, zasoby portalu są ograniczone, na dzień 1 grudnia 2020 r. nie było możliwe znalezienie przygotowanych zestawów zdjęć dla pozostałych kategorii obiektów.

Po wyczerpaniu gotowych rozwiązań konieczne było przygotowanie własnych zestawów danych uczących. Internet jest pełen zdjęć - skompletowanie niezbędnych danych nie powinno zatem sprawić kłopotów. Możliwości jest wiele, problemem jest skorzystanie z nich w sposób efektywny.

Pierwszą stroną odwiedzoną w celu znalezienia zdjęć był oczywiście portal Google.com. W zakładce grafika czeka niezliczona ilość obrazów, jednakże portal jak większość podobnych stron, nie udostępnia funkcji pobierania wielu zdjęć jednocześnie, oznacza to, że pobranie każdego zdjęcia wiąże się z osobnym jego otwarciem a następnie dopiero pobraniem.

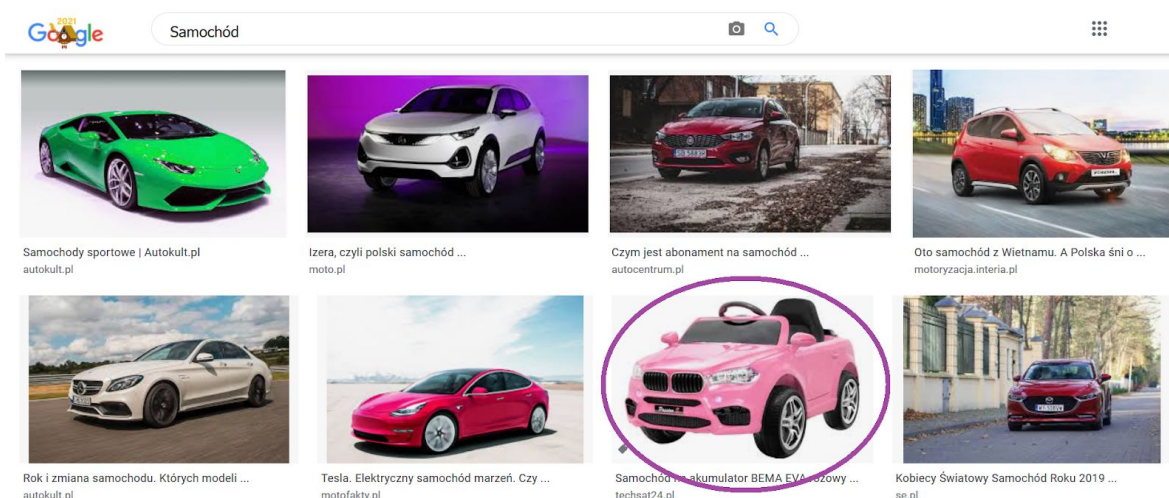
Konieczność otwarcia każdego pliku z osobna w perspektywie pobrania ok. 20 000 - 30 000 zdjęć dyskwalifikuje pomysł "manualnego" uzyskania danych.

Lepszą metodą na pozyskanie danych jest **web scraping**, czyli wyodrębnianie interesujących nas elementów strony internetowej a następnie wykonywanie operacji tylko na nich.

Pozyskanie zdjęć metodą web scraping'u możliwe jest np. dzięki użyciu biblioteki **Selenium**. To właśnie w oparciu o tę bibliotekę przygotowany został skrypt wyodrębniający i pobierający interesujące nas zdjęcia z portalu Google Grafika. Ponieważ zasoby portalu Google również są ograniczone, konieczne było znalezienie dodatkowej metody na zdobycie danych.

Duże biblioteki zdjęć można znaleźć na wielu stronach, jedną z nich jest **Flickr.com**. Dodatkowo Flickr poza darmowymi do pobrania zdjęciami, udostępnia użytkownikom strony narzędzie podobne do poprzednio użytego skryptu. **Flickr API** nie wykorzystuje jednak web scraping'u, a pozwala na wykonywanie zapytań bezpośrednio do backendu oraz bazy danych aplikacji. Przy użyciu narzędzia udostępnionego przez aplikację Flickr, napisany został skrypt, który pozwala na wyszukanie oraz pobranie zdjęć zgodnych z interesującą nas frazą. Flickr API pozwala także na tworzenie zapytań filtrujących zdjęcia na podstawie ich licencji dotyczącej praw autorskich, co może być bardzo pomocne w przypadku komercyjnych projektów.

Użycie skryptów jest stosunkowo proste, w obu przypadkach należy podać interesujące nas frazy oraz ilość zdjęć, które chcemy pobrać, cała reszta dzieje się automatycznie. Problemатyczne natomiast, okazało się samo wyszukiwanie oraz pozycjonowanie obrazów przez portale. Wpisanie niesprecyzowanej ogólnej frazy takiej jak np. "Samochód" powoduje wyszukanie wielu fotografii, które nie spełniają przyjętych wcześniej wymagań. Wśród wyników znaleźć możemy duplikaty zdjęć, zdjęcia niskiej jakości oraz zdjęcia, które przedstawiają zupełnie inne obiekty np. zabawkowe samochody.



*Zdjęcie 1 - wyniki wyszukiwania*

Testy skryptów pozwoliły na wyklarowanie sposobu działania wyszukiwarek portali.

Wyszukiwanie sprecyzowanych fraz, takich jak np. "Samochód Volvo" zamiast ogólnego "Samochód", okazało się znacznie bardziej skuteczne, ograniczyło duplikaty zdjęć oraz występowanie zdjęć niepożądanych obiektów.

Niestety wyszukiwanie sprecyzowanych fraz zmniejsza różnorodność zestawu danych, a także zmniejsza ilość wyszukanych zdjęć. Stąd aby zapewnić różnorodność oraz nie wyczerpać dostępnych obrazów, zdjęcia pobierane były w zestawach liczących od 100 do 200 sztuk.

Wykorzystane skrypty pozwoliły na pobranie ok. 30 000 zdjęć, dla każdego z pozostałych pięciu obiektów zebrano ok. 6 000 zdjęć.

### 3.3. SELEKCJA ORAZ PREPARACJA DANYCH

Selekcja jest ostatnim krokiem procesu przygotowania zestawu danych.

Po zebraniu satysfakcjonującej liczby zdjęć, należy usunąć z zestawu fotografie, które nie nadają się do uczenia sieci.

W przypadku zdjęć, preparacje zestawu najlepiej rozpocząć od usunięcia uszkodzonych plików. Niektóre z pobranych obrazów mogą być uszkodzone, wynika to z faktu, iż ich miniaturki wyświetlane na stronach, czasami prowadzą do uszkodzonych plików. Uszkodzone pliki mogą powstać także przez nieprawidłowe działanie skryptu. Najlepszym sposobem pozbycia się uszkodzonych zdjęć jest napisanie prostego skryptu, który usunie uszkodzone zdjęcia. Skrypt używając biblioteki **Pillow** otwiera po kolei pobrane wcześniej obrazy, następnie usuwając te z nich, które są uszkodzone. Pozwala to na usunięcie uszkodzonych zdjęć bez przeglądania i filtrowania pobranych zestawów danych.

Dodatkowo skrypt nie wyświetla otwieranych obrazów, dzięki temu jego działanie jest znacznie szybsze od konwencjonalnych metod filtrowania danych.

Kolejnym problemem, który należy zaadresować są duplikaty zdjęć, podobnie jak w przypadku poprzednich zadań został on rozwiązany dzięki użyciu odpowiedniego skryptu. Napisany skrypt otwiera zdjęcia, które następnie stają się argumentami odpowiedniej funkcji. Funkcja zamienia każde zdjęcie na odpowiednią **wartość hash**, jest ona zależna od zawartości obrazu. Następnie wartości hash porównywane są ze sobą, jeżeli któraś wartość hash

występuje więcej niż jeden raz, oznacza to, że w naszym zestawie danych znajduje się duplikat. Po porównaniu wartości duplikaty zostają automatycznie usunięte.

Po usunięciu duplikatów oraz uszkodzonych zdjęć należy rozwiązać ostatni już problem, tj. usunięcie zdjęć niskiej jakości. Niska jakość zdjęć nie musi oznaczać ich niskiej rozdzielczości, zdjęcie dobrej jakości w kontekście maszynowego uczenia powinno przede wszystkim zawierać uwydatnione cechy obiektu, który przedstawia. Biorąc pod uwagę poprzednie przeszkody, użycie skryptu wydaje się być najlepszym rozwiązaniem, niestety w tym przypadku jego przygotowanie prawdopodobnie przerosłoby swoją złożonością całą przygotowywaną aplikację.

Filtrowanie zdjęć pod kątem ich przydatności podczas nauki sieci neuronowej jest cięższe, ponieważ wiele cech obrazów, których chcemy się pozbyć nie ma reprezentacji matematycznej. Filtracja będzie opierać się zatem na porównywaniu i selekcjonowaniu ich “manualnie”. Aby dobrze zobrazować problem preparacji zestawu danych, wszystkie przedstawione przykłady opierać się będą o zdjęcia samochodów. Należy jednak pamiętać, iż poza cechami wspólnymi dla wszystkich kategorii obiektów takimi jak np. znaki wodne, filtracja innych kategorii może polegać na selekcjonowaniu zupełnie innych właściwości.

Końcowa preparacja danych jest etapem, który zajmuje najwięcej czasu, dlatego warto zacząć od wyznaczenia cech odpowiedniego zdjęcia:

- **Zdjęcie przedstawia odpowiedni obiekt**

Pobierając zdjęcia w dużych zestawach nie jesteśmy w stanie zapobiec pobieraniu obrazów niezgodnych z opisem. Może także zdarzyć się, że pobrany obraz przedstawia poszukiwany przez nas obiekt natomiast jego inne właściwości nie spełniają naszych wymagań. Podobna sytuacja ma miejsce na zdjęciu nr 2 oraz nr 3, oba zdjęcia przedstawiają ten sam model samochodu, niestety jedno z nich jest namalowanym obrazem i nie przedstawia realnego obiektu. Z oczywistych względów obraz nr 2 oraz inne zdjęcia przedstawiające obiekty niezgodne z frazą wyszukiwania powinny być usunięte z zestawu danych jako pierwsze.



Zdjęcie 2 - pobrany obraz, źródło:  
<https://www.artstation.com/artwork/e08a9P>



Zdjęcie 3 - pobrany obraz, źródło:  
<https://wallpaperaccess.com/rwb-porsche>

- Przedstawiony obiekt jest dobrze widoczny oraz uwydatnione są jego kluczowe cechy



Zdjęcie 4 - pobrany obraz, źródło:  
<https://moto.rp.pl/parking/41676-nowe-porsche-911-targa-lato-moze-sie-rozpoczac>



Zdjęcie 5 - pobrany obraz, źródło:  
<https://moto.rp.pl/parking/41676-nowe-porsche-911-targa-lato-moze-sie-rozpoczac>

Patrząc na zdjęcie nr 4 oraz nr 5, natychmiast zauważamy znaczące różnice między nimi. Różnice są bardzo widoczne pomimo faktu, iż oba zdjęcia przedstawiają dokładnie ten sam samochód oraz pochodzą z tej samej sesji zdjęciowej. Główną a zarazem jedyną znaczącą różnicą jest kadr zdjęcia. Zdjęcie nr 4 przedstawia bardzo dokładne zbliżenie na koła samochodu, zdjęcie nr 5 również przedstawia koła, są one jednak jedynie komponentem głównego obiektu jakim jest samochód. Filtrując obiekty pod kątem uwydatnionych cech nie jest możliwe wyznaczenie sztywnych ram selekcji.



Dobrze obrazuje to zdjęcie nr 6, fotografia przedstawia ten sam model samochodu, jest dobrej jakości a obiekt jest dobrze widoczny. Niestety kadr zdjęcia powoduje, że widoczna jest tylko połowa samochodu, dodatkowo w samochodzie siedzi człowiek, co również możemy uznać za pewne utrudnienie.



*Zdjęcie 6 - pobrany obraz, źródło: <https://www.pinterest.co.uk/amp/pin/542331980102757760/>*

Podobnie jak w przypadku zdjęcia nr 6, większość pobranych obrazów nie eksponuje przedstawionego obiektu w sposób idealny. Dlatego na każdą filtrowaną fotografię należy spojrzeć indywidualnie i ocenić jej przydatność w procesie uczenia sieci.

- **Zdjęcie nie zawiera znaków wodnych oraz zbędnych napisów**

Duża część pobranych zdjęć pochodzi ze stron internetowych oferujących zdjęcia stockowe. Praktycznie wszystkie z tych stron dodają do swoich obrazów jakiś rodzaj znaku wodnego. Znaki wodne poza skutecznym zabezpieczeniem fotografii, powodują także mniejsze uwydatnienie cech przedstawionego obiektu, co zmniejsza efektywność treningu sieci.

Ponieważ chcemy możliwie jak najbardziej ograniczyć szumy oraz nie chcemy aby nasza sieć rozpoznawała znaki wodne, zdjęcia te należy usunąć z przygotowanego zestawu.



*Zdjęcie 7 - obraz ze znakiem wodnym*



*Zdjęcie 8 - pobrany obraz, źródło:*

*<https://www.autocar.co.uk/car-review/porsche/911-gt>*

3

Przeglądając tysiące plików pomyłka jest bardzo prawdopodobna, dlatego zestaw danych został przefiltrowany kilkakrotnie.

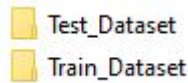
Po zakończeniu procesu filtracji, zestaw danych zawiera ok. 25 000 zdjęć.

- **Podział na dane uczące oraz dane testowe**

Jednym z ostatnich etapów przygotowywania zestawu danych, jest podzielenie zestawu na część uczącą oraz część testową. Rozdzielność zestawu uczącego od testowego jest jedną z najważniejszych zasad maszynowego uczenia, operacja ta może zostać wykonana na dwa sposoby. Pierwszym z nich jest podzielenie zestawu zdjęć bezpośrednio przed nauką sieci, dzięki zastosowaniu odpowiednich warunków w kodzie skryptu uczącego. Drugim rozwiązaniem jest rozdzielanie obrazów do dwóch osobnych folderów na dysku.

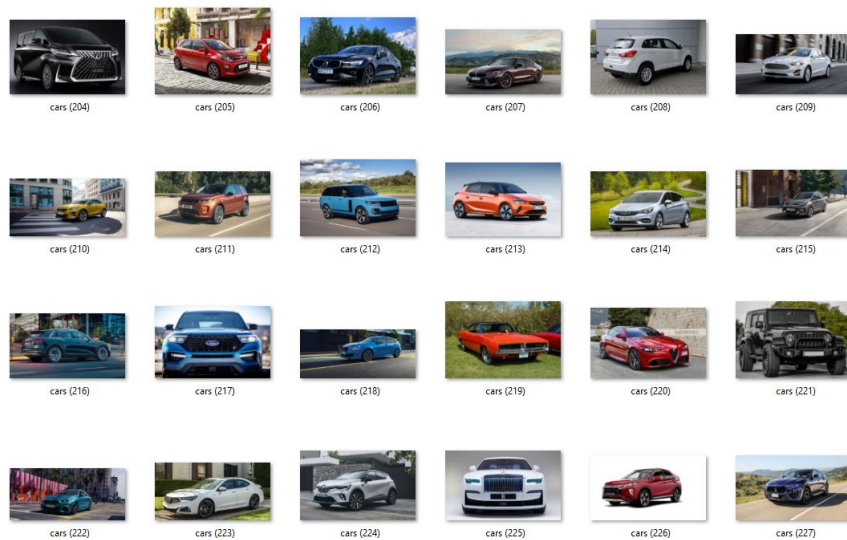
Oba rozwiązania pozwalają na przygotowanie konkretnych, rozdzielonych zestawów danych i są równie skutecznie w procesie uczenia sieci. Przewagą drugiego rozwiązania jest jego użyteczność w testach sieci po zakończeniu procesu uczenia. Dzięki faktycznemu rozdzielaniu danych do dwóch osobnych folderów na dysku, zdjęcia z zestawu testowego są zawsze gotowe do testów w dalszej części rozwoju projektu. Jest to szczególnie ważne, ponieważ wyodrębnianie zdjęć testowych z folderu głównego przed każdym przeprowadzonym testem jest skrajnie nieefektywne i niesie za sobą duże ryzyko przeprowadzenia testu na zdjęciu, które użyte zostało do uczenia sieci, co może przyczynić się do błędnej oceny skuteczności algorytmu w dalszej części pracy. Dlatego, to właśnie

drugie rozwiązanie zostało zastosowane, zdjęcia zostały podzielone na 2 zestawy, zestaw testowy zawiera po 100 zdjęć z każdej kategorii.



*Zdjęcie 9 - przygotowane zestawy danych*

- **Ujednolicenie nazw plików**



*Zdjęcie 10 - ujednolicone nazwy plików*

Na zakończenie procesu przygotowania zestawu danych, dobrą praktyką jest ujednolicenie nazw plików, pomoże to uniknąć potencjalnych problemów z wczytywaniem zdjęć podczas uczenia sieci i ułatwi segregację.

- **Normalizacja zdjęć**

Normalizacja to proces, którego zadaniem jest zmniejszenie różnorodności zestawu zdjęć. Technika polega na zmianie zakresu wartości poszczególnych pikseli na fotografii w taki sposób, aby zbliżyć wszystkie zdjęcia do średniej wartości pikseli obliczonej dla tego właśnie zestawu. Normalizacja zdjęć pomaga w przyspieszeniu oraz zwiększeniu skuteczności procesu uczenia.

Aby prawidłowo wykonać normalizację dużego zestawu zdjęć, należy obliczyć średnią wartość oraz odchylenie standardowe pikseli. Jako, iż przygotowany zestaw danych liczy ponad 25 000 zdjęć, najlepszym rozwiązaniem będzie przygotowanie odpowiedniego skryptu. Przygotowany skrypt zwraca średnią wartość dla każdego z trzech kanałów RGB oraz kolejno odchylenie standardowe dla każdego z nich. Obliczona wartość średnia dla kanałów RGB przygotowanego zestawu danych wynosi kolejno: **0.54875654**, **0.51137114**, **0.47434878**, podczas gdy wartość odchylenia standardowego wynosi: **0.28420544**, **0.2819345**, **0.28757098**. Wartości te pozwolą na prawidłowe przeprowadzenie normalizacji zestawu danych.

Normalizacja przygotowanego zestawu danych nie zostanie jednak przeprowadzona na zebranych plikach. Biblioteka PyTorch udostępnia wiele funkcji, które pomagają w procesie uczenia sieci, jedną z nich jest funkcja normalizująca zestaw danych. Przy jej użyciu, możliwe jest przeprowadzenie procesu normalizacji podczas ładowania plików do pamięci komputera, oznacza to, że przygotowane pliki mogą pozostać niezmienione. Normalizacja plików sposobem zaproponowanym przez inżynierów biblioteki PyTorch odbywa się bezpośrednio przed procesem uczenia sieci, dlatego paragraf jej dotyczący został umieszczony na końcu procesu dostosowywania zestawu danych.

Aby przeprowadzić normalizację zestawu danych przed procesem nauki, wystarczy dodać funkcję **Normalize** wraz z obliczonymi wcześniej wartościami, do funkcji ładującej zestaw do pamięci podręcznej komputera. Przeprowadzenie normalizacji powinno usprawnić proces uczenia.



*Zdjęcie 11 - obraz przed normalizacją, źródło:  
<https://www.12minutos.com/5f113751ce79d/joven-empresario-con-gran-demanda-suele-pasearse-en-sus-carros-de-lujo.html>*



*Zdjęcie 12 - obraz po normalizacji*



*Zdjęcie 13 - obraz przed normalizacją, źródło:  
<https://newsroom.porsche.com/de/2019/produkte/porsche-911-carrera-s-features-mark-webber-markenbotschafter-18373.html>*



*Zdjęcie 14 - obraz po normalizacji*

## **4. Projektowanie oraz optymalizacja sieci neuronowej**

### **4.1. PROJEKTOWANIE KONWOLUCYJNEJ SIECI NEURONOWEJ**

Projektowanie architektury sieci neuronowej, podobnie jak w przypadku poprzednich problemów rozpocząć należy od analizy problemu i zadaniu kilku kluczowych pytań:

- **Jakie będzie przeznaczenie sieci?**

Sieć będzie wykorzystywana do identyfikacji obrazów na portalu społecznościowym.

- **Jak bardzo skomplikowane będą dane przetwarzane przez sieć?**

Przygotowany zestaw danych w kontekście sieci neuronowych oraz rozpoznawania obiektów jest stosunkowo prosty. Dla najbardziej zaawansowanych architektur, zadanie rozpoznania ośmiu obiektów jest trywialne. Ilość zebranych zdjęć także nie jest imponująca mając na uwadze, iż sieci przygotowywane przez duże firmy uczone są przy pomocy zestawów liczących setki tysięcy a nawet miliony zdjęć. Fakt ten wykorzystany zostanie na naszą korzyść. Skomplikowane zestawy danych wymagają skomplikowanych rozwiązań, w



przypadku przygotowanego zestawu danych najskuteczniejszym rozwiązaniem okazać może się najprostsze z nich.

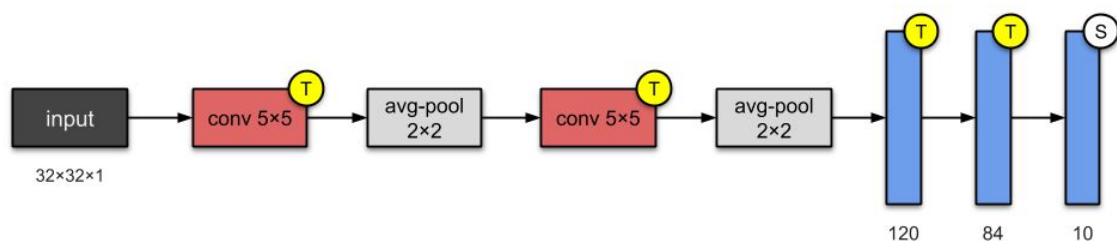
- **Jakie będą parametry uczące?**

Ze względu na dużą liczbę parametrów uczących oraz fakt, że zmiana jednego parametru ma wpływ na wszystkie pozostałe, na czas projektowania sieci neuronowej, dobrane zostały parametry domyślne. Zabieg ten ma na celu zminimalizowanie błędu podczas szacowania skuteczności sieci. Dodatkowo, ponieważ różne sieci reagują w różny sposób na zmianę parametrów uczących, dobranie popularnych parametrów pozwoli ograniczyć występowanie wyjątkowo nieskutecznych procesów uczenia, co z kolei pozwoli na zaoszczędzenie cennego czasu. **Dobre parametry:**

- Krok uczenia - **0.001**
- Liczba epok - **10** lub **50** przy drugiej próbie skuteczności
- Momentum - **0.9**
- Spadek wagi (weight decay) - **0**
- Planista (scheduler) - **brak**
- Liczność zestawu weryfikującego - **15%** zestawu uczącego (ok. 4 300 zdjęć)

Trudno jednak rozpocząć projektowanie architektury bez żadnej inspiracji.

Z uwagi na fakt, iż zgodnie z poczynionymi założeniami, przygotowywana sieć ma być możliwie najprostsza, jako wyjściową sieć w procesie projektowania wybrana została sieć **LeNet-5**.



*Zdjęcie 15 - architektura LeNet-5, źródło:*

<https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d>

Przedstawiona w roku 1994 przez Yann LeCun, sieć LeNet-5 była jedną z pierwszych konwolucyjnych sieci neuronowych. Znakomita skuteczność w obszarze klasyfikacji obrazów idąca w parze z redukcją przetwarzanych parametrów przypieczętowała wysoką pozycję architektury w dziedzinie rozpoznawania obiektów. Architektura LeNet-5 była inspiracją dla wielu znacznie bardziej zaawansowanych, współczesnych architektur - dobrze rokuje to dla przygotowywanej w pracy architektury.

Architektura LeNet-5 składa się z trzech powłok konwolucyjnych, dwóch powłok odpowiedzialnych za pooling oraz trzech w pełni połączonych powłok, przygotowana została z myślą o czarno białych obrazach o wielkości 32x32 pikseli. Ponieważ zdjęcia, przetwarzane na stronie będą kolorowe, pierwszą adaptacją sieci będzie dodanie dwóch dodatkowych kanałów do warstwy wejściowej. Warstwa wejściowa po przekształceniu z wymiarów 32x32x1 do 32x32x3 poza przetwarzaniem pojedynczych pikseli jest w stanie każdemu z nich przypisać także kolor w skali RGB.

Kolejną adaptacją będzie zwiększenie rozmiaru zdjęcia wejściowego. Rozmiar 32x32 piksele jest znacznie za mały, wzorując się najskuteczniejszymi architekturami dobrany został rozmiar 256x256 pikseli.

W tym momencie pracy nad aplikacją, możliwe jest przygotowanie skryptu uczącego w celu przeprowadzenia pierwszych testów.

Przygotowany skrypt składa się z kilku części, do najważniejszych możemy zaliczyć funkcję wczytującą dane, deklarację architektury, oraz funkcję uczącą oraz testującą gotową już sieć.

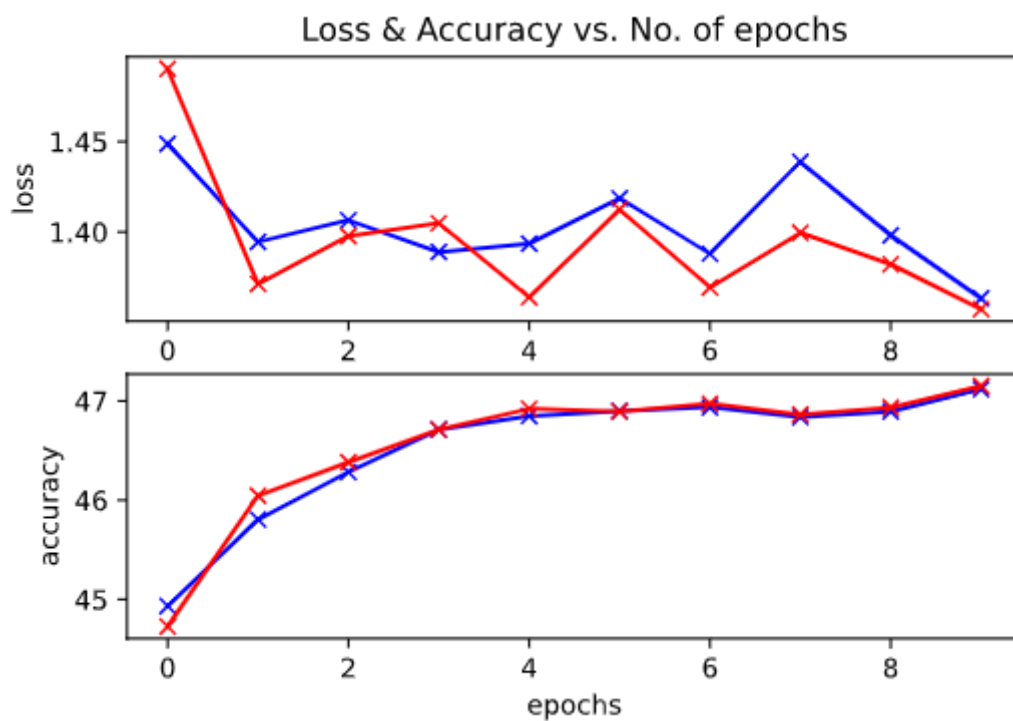
Ocena skuteczności procesu uczenia polegać będzie na porównaniu wykresów **funkcji straty** oraz wykresu **dokładności** klasyfikacji dla danych **uczących** oraz **walidacyjnych**.

- **Model 1**

Jako pierwszą wersję projektowanej architektury przyjęta została podstawowa architektura LeNet-5 dostosowana do pracy z kolorowymi zdjęciami o rozmiarze 256x256 pikseli.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 6, 252, 252]	456
Tanh-2	[-1, 6, 252, 252]	0
AvgPool2d-3	[-1, 6, 126, 126]	0
Conv2d-4	[-1, 16, 122, 122]	2,416
Tanh-5	[-1, 16, 122, 122]	0
AvgPool2d-6	[-1, 16, 61, 61]	0
Conv2d-7	[-1, 120, 57, 57]	48,120
Tanh-8	[-1, 120, 57, 57]	0
Flatten-9	[-1, 389880]	0
Linear-10	[-1, 84]	32,750,004
Tanh-11	[-1, 84]	0
Linear-12	[-1, 8]	680
Total params: 32,801,676		
Trainable params: 32,801,676		
Non-trainable params: 0		

Zdjęcie 16 - architektura LeNet-5 dostosowana do pracy z kolorowymi zdjęciami



Wykres 1 - wykres funkcji straty oraz skuteczności sieci



Z wykresu nr 1 wyczytać można, że skuteczność sieci wynosi ok. 47%, jest to wynik daleki od satysfakcjonującego. Dlatego sieć potrzebuje dalszej adaptacji do rozwiązywanego problemu.

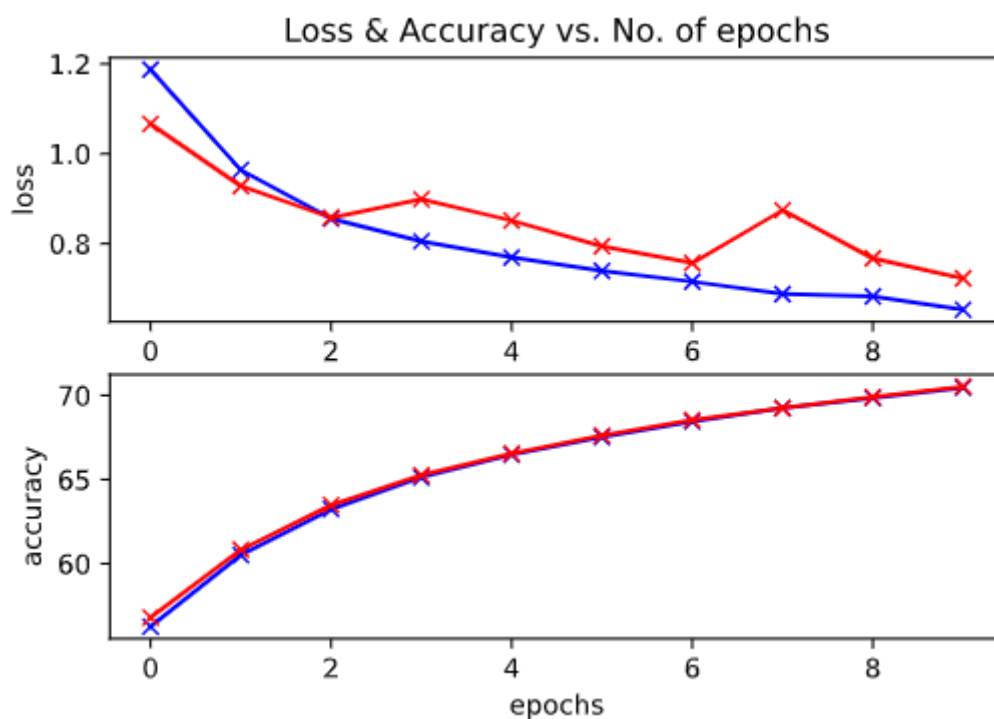
## • Model 2

Ponieważ sieć LeNet-5 stworzona została do klasyfikacji znacznie mniejszych obrazów, ilość parametrów w powłokach liniowych po dostosowaniu sieci do zdjęć kolorowych okazała się bardzo wysoka (wykres nr 1). Ponadto, filtr maski wykonującej pooling znacznie lepiej radzi sobie w przypadku zdjęć czarno-białych.

Dlatego, aby zmniejszyć ilość parametrów sieci w drugim modelu architektury dodana została dodatkowa powłoka wykonująca operacje pooling'u. Teraz powłoka ta występuje po każdej powłoce konwolucyjnej, aby zapewnić jej skuteczniejsze działanie zmieniona została maska funkcji, zamiast średniej wartości funkcja przekazuje wartość maksymalną.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 6, 252, 252]	456
Tanh-2	[-1, 6, 252, 252]	0
MaxPool2d-3	[-1, 6, 126, 126]	0
Conv2d-4	[-1, 16, 122, 122]	2,416
Tanh-5	[-1, 16, 122, 122]	0
MaxPool2d-6	[-1, 16, 61, 61]	0
Conv2d-7	[-1, 120, 57, 57]	48,120
Tanh-8	[-1, 120, 57, 57]	0
MaxPool2d-9	[-1, 120, 28, 28]	0
Flatten-10	[-1, 94080]	0
Linear-11	[-1, 84]	7,902,804
Tanh-12	[-1, 84]	0
Linear-13	[-1, 8]	680
=====		
Total params: 7,954,476		
Trainable params: 7,954,476		
Non-trainable params: 0		

*Zdjęcie 17 - architektura modelu 2*



Wykres 2 - wykres funkcji straty oraz skuteczności sieci

Przedstawiona na wykresie nr 2 skuteczność sieci znacznie wzrosła. Zmiana maski oraz filtra powłoki wykonującej operacje pooling'u poprawiła skuteczność sieci o ponad 20%. Tak duży wzrost budzi nadzieję, iż kolejne adaptacje pozwolą na osiągnięcie naprawdę wysokiej skuteczności modelu.

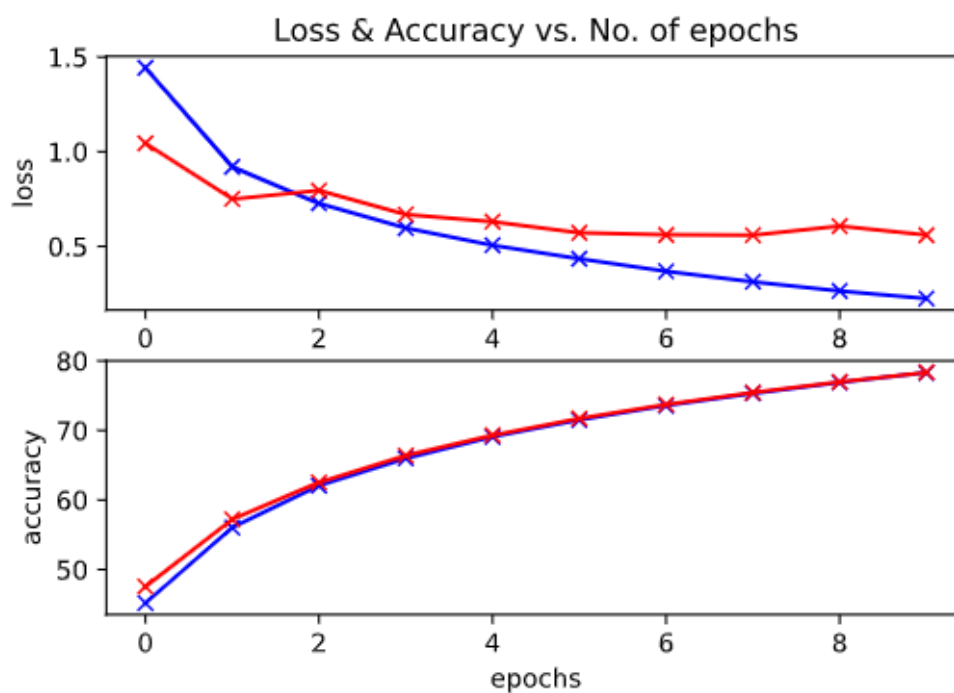
### • Model 3

Funkcja przejścia Tanh zastąpiona została bardziej uniwersalną funkcją przejścia ReLu (rectified linear activation function), dodatkowo model został rozbudowany o dwie powłoki liniowe. Dodatkowe powłoki liniowe dodadzą projektowanemu modelowi głębi, co powinno pozwolić na znalezienie lepszej reprezentacji klasyfikowanych obiektów.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 6, 252, 252]	456
ReLU-2	[-1, 6, 252, 252]	0
MaxPool2d-3	[-1, 6, 126, 126]	0
Conv2d-4	[-1, 16, 122, 122]	2,416
ReLU-5	[-1, 16, 122, 122]	0
MaxPool2d-6	[-1, 16, 61, 61]	0
Conv2d-7	[-1, 120, 57, 57]	48,120
ReLU-8	[-1, 120, 57, 57]	0
MaxPool2d-9	[-1, 120, 28, 28]	0
Flatten-10	[-1, 94080]	0
Linear-11	[-1, 1024]	96,338,944
ReLU-12	[-1, 1024]	0
Linear-13	[-1, 512]	524,800
ReLU-14	[-1, 512]	0
Linear-15	[-1, 256]	131,328
ReLU-16	[-1, 256]	0
Linear-17	[-1, 8]	2,056

Total params: 97,048,120  
 Trainable params: 97,048,120  
 Non-trainable params: 0

Zdjęcie 18 - architektura modelu 3



Wykres 3 - wykres funkcji straty oraz skuteczności sieci

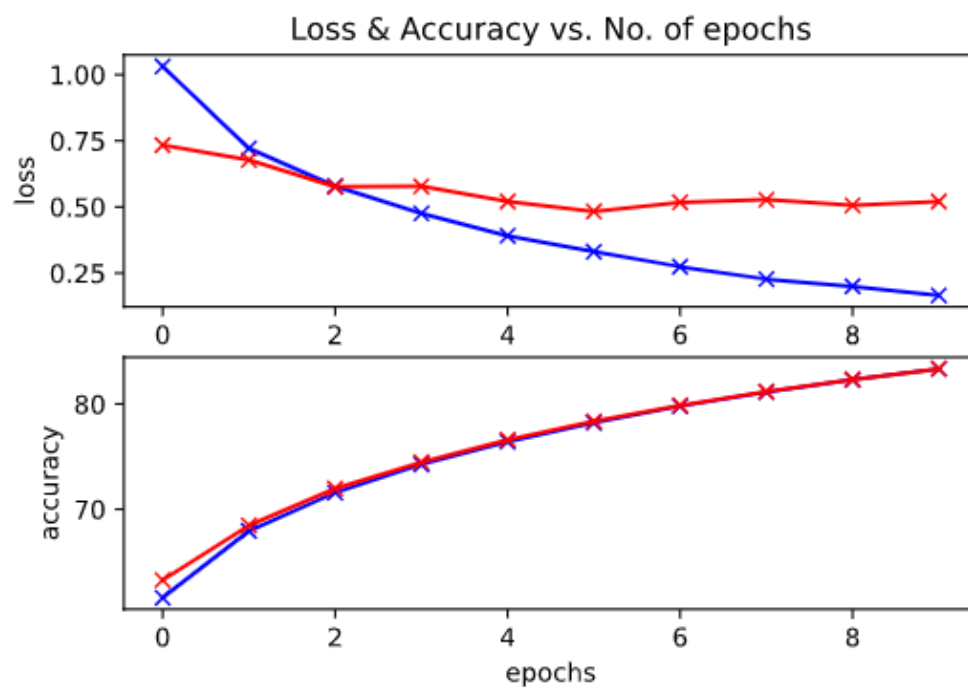
Zgodnie z przewidywaniami zmiany w architekturze poprawiły skuteczność modelu o kilka procent (wykres nr 3).

- **Model 4**

Pomiędzy warstwami konwolucyjnymi, dodana została funkcja normalizująca wartości wyjściowe.

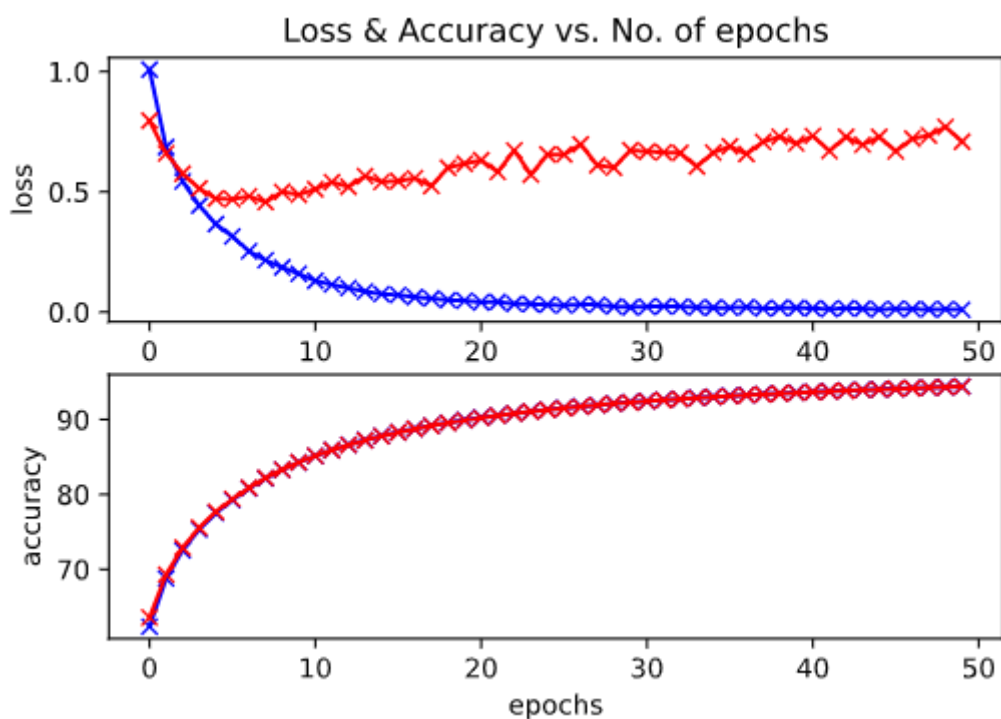
Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 6, 252, 252]	456
ReLU-2	[-1, 6, 252, 252]	0
MaxPool2d-3	[-1, 6, 126, 126]	0
BatchNorm2d-4	[-1, 6, 126, 126]	12
Conv2d-5	[-1, 16, 122, 122]	2,416
ReLU-6	[-1, 16, 122, 122]	0
MaxPool2d-7	[-1, 16, 61, 61]	0
BatchNorm2d-8	[-1, 16, 61, 61]	32
Conv2d-9	[-1, 120, 57, 57]	48,120
ReLU-10	[-1, 120, 57, 57]	0
MaxPool2d-11	[-1, 120, 28, 28]	0
BatchNorm2d-12	[-1, 120, 28, 28]	240
Flatten-13	[-1, 94080]	0
Linear-14	[-1, 1024]	96,338,944
ReLU-15	[-1, 1024]	0
Linear-16	[-1, 512]	524,800
ReLU-17	[-1, 512]	0
Linear-18	[-1, 256]	131,328
ReLU-19	[-1, 256]	0
Linear-20	[-1, 8]	2,056
=====		
Total params: 97,048,404		
Trainable params: 97,048,404		
Non-trainable params: 0		

*Zdjęcie 19 - architektura modelu 4*



*Wykres 4 - wykres funkcji straty oraz skuteczności sieci*

Ponieważ sieć uzyskała skuteczność na poziomie ponad 80% jest to dobry moment, aby sprawdzić jak sieć zachowuje się w przypadku nauki na większej liczbie epok. Proces uczenia został powtórzony z poprzednimi parametrami ale z liczbą epok zwiększoną do 50.



Wykres 5 - wykres funkcji straty oraz skuteczności sieci

Ponad 95% skuteczności (wykres nr 5) to świetny wynik! Wykres straty wygląda jednak niepokojąco. Niestety mamy do czynienia z przetrenowaniem sieci. Wysoka skuteczność wynika bezpośrednio z faktu, iż sieć uczy się “na pamięć”. Najlepszym sposobem na wykrycie zjawiska przetrenowania sieci jest porównanie wykresu straty (wykres nr 5) dla zdjęć pochodzących z zestawu uczącego (kolor niebieski) z wykresem straty dla zdjęć pochodzących z zestawu walidacyjnego (kolor czerwony). Gdy proces uczenia przebiega prawidłowo, wartość strat dla obu zestawów stopniowo zmniejsza się z każdą iteracją algorytmu. W przypadku trenowanego modelu zauważyć można, że strata zestawu walidacyjnego w pewnym momencie “oddziela się” od straty zestawu uczącego i z każdą kolejną iteracją rośnie zamiast spadać. Okazuje się, iż zjawisko to wystąpiło już w przypadku wcześniejszego procesu uczenia (wykres nr 4), jednak ze względu na małą ilość iteracji jego identyfikacja nie była tak oczywista.

Wykres nr 5 nie pozostawia jednak żadnych wątpliwości, sieć jest “przeuczona”. Oznacza to, że użycie sieci do rozpoznania jakiegokolwiek obrazu pochodzącego spoza zestawu uczącego jest niemożliwe, zatem sama sieć jest bezużyteczna. Nie oznacza to jednak konieczności projektowania sieci od zera. Istnieje wiele sposobów, aby zmniejszyć lub nawet całkowicie

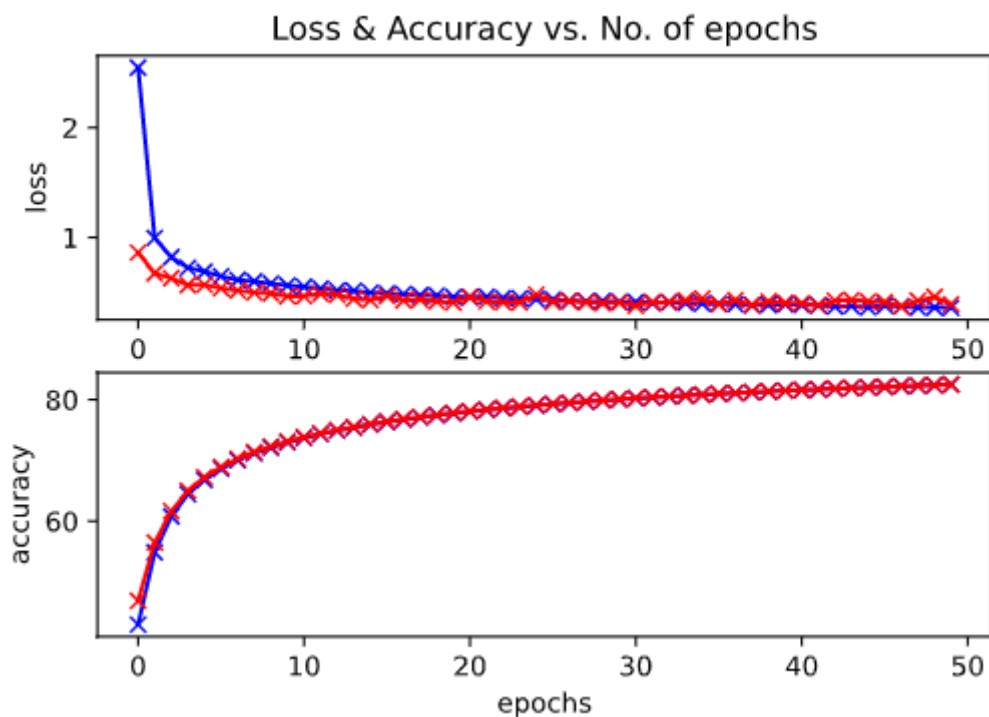
uniknąć zjawiska przetrenowania sieci. Jedno z nich polega na wprowadzeniu małej zmiany w architekturze modelu i zastosowanie w następnej wersji modelu architektury.

- **Model 5**

Aby zapobiec przetrenowaniu, funkcje aktywacyjne pomiędzy każdą warstwą liniową zastąpione zostały funkcją Dropout, funkcja przy każdym aktywowaniu ma określoną szansę na zastąpienie wektora danych wektorem złożonym z samych zer, w efekcie sieć co jakiś czas otrzymuje wektor zakłócający proces uczenia. Skuteczność tej metody opisana została w pracy **“Improving neural networks by preventing co-adaptation of feature detectors”** autorstwa **Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, Ruslan R. Salakhutdinov.**[1]

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 6, 252, 252]	456
ReLU-2	[-1, 6, 252, 252]	0
MaxPool2d-3	[-1, 6, 126, 126]	0
BatchNorm2d-4	[-1, 6, 126, 126]	12
Conv2d-5	[-1, 16, 122, 122]	2,416
ReLU-6	[-1, 16, 122, 122]	0
MaxPool2d-7	[-1, 16, 61, 61]	0
BatchNorm2d-8	[-1, 16, 61, 61]	32
Conv2d-9	[-1, 120, 57, 57]	48,120
ReLU-10	[-1, 120, 57, 57]	0
MaxPool2d-11	[-1, 120, 28, 28]	0
BatchNorm2d-12	[-1, 120, 28, 28]	240
Flatten-13	[-1, 94080]	0
Dropout-14	[-1, 94080]	0
Linear-15	[-1, 1024]	96,338,944
Dropout-16	[-1, 1024]	0
Linear-17	[-1, 512]	524,800
Dropout-18	[-1, 512]	0
Linear-19	[-1, 256]	131,328
Dropout-20	[-1, 256]	0
Linear-21	[-1, 8]	2,056
Total params: 97,048,404		
Trainable params: 97,048,404		
Non-trainable params: 0		

*Zdjęcie 20 - architektura modelu 5*



Wykres 6 - wykres funkcji straty oraz skuteczności sieci

Skuteczność przygotowanego modelu zaczyna wyglądać obiecująco.

Zastosowanie funkcji Dropout pozwoliło na całkowite wyeliminowanie zjawiska przetrenowania sieci (dla przyjętych na początku etapu projektowania parametrów uczących).

Efektem ubocznym jest nieco niższa dokładność,

mamy natomiast pewność, że wynik jest prawidłowy a sieć nie uczy się “na pamięć”. Spadek dokładności nie stanowi problemu, ponieważ nadal nie zostały zoptymalizowane hiperparametry poszczególnych powłok oraz parametry uczące. Odpowiednie dobranie brakujących parametrów być może umożliwi uzyskanie skuteczności na poziomie ok. 90%.

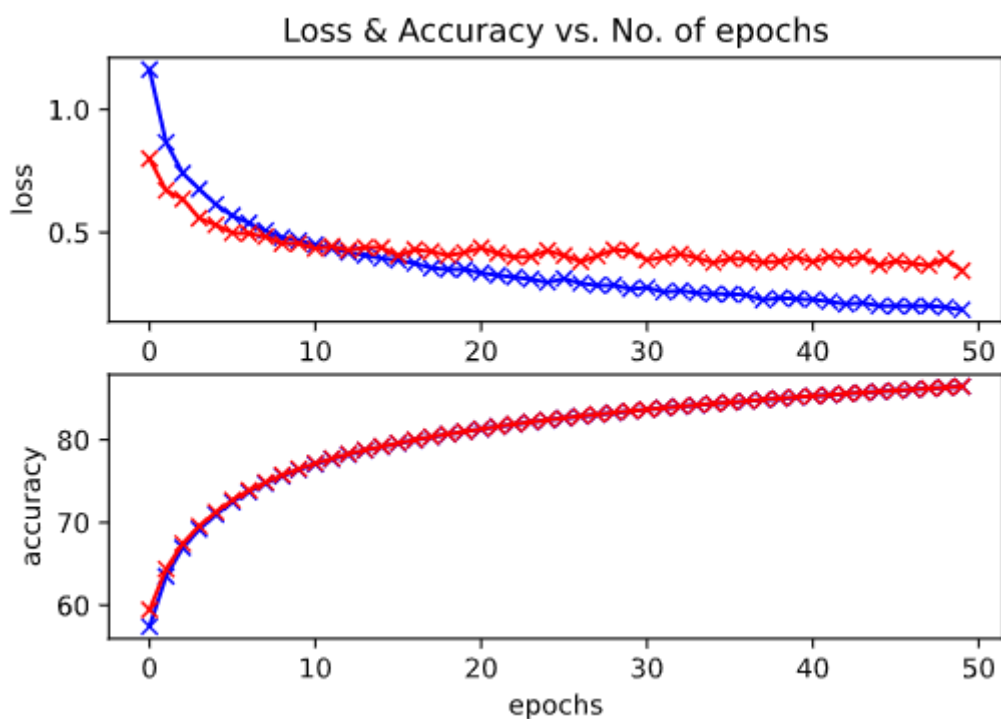
Proces projektowania sieci neuronowej zakończony zostanie przez dobór hiperparametrów poszczególnych powłok. Jest to proces, który zajmuje dużo czasu. Wynika to z faktu, iż nie ma jednej recepty na zaprojektowanie odpowiedniego modelu sieci. Bardzo duży wpływ na działanie sieci ma sam zestaw danych, dlatego dobieranie parametrów powinno być podyktowane potrzebami przygotowanych zdjęć. Samo wyznaczenie odpowiednich parametrów ma charakter eksperymentalny, a zadanie to można potraktować jak dopasowanie parametrów sieci bezpośrednio do zestawu uczącego. Ponieważ przebieg doboru parametrów



wymagał przetestowania wielu wersji modelu oraz zmiany parametrów uczących, pokazana zostanie tylko końcowa wersja sieci.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 256, 256]	896
ReLU-2	[-1, 32, 256, 256]	0
MaxPool2d-3	[-1, 32, 128, 128]	0
BatchNorm2d-4	[-1, 32, 128, 128]	64
Conv2d-5	[-1, 64, 128, 128]	18,496
ReLU-6	[-1, 64, 128, 128]	0
MaxPool2d-7	[-1, 64, 64, 64]	0
BatchNorm2d-8	[-1, 64, 64, 64]	128
Conv2d-9	[-1, 128, 64, 64]	73,856
ReLU-10	[-1, 128, 64, 64]	0
MaxPool2d-11	[-1, 128, 32, 32]	0
BatchNorm2d-12	[-1, 128, 32, 32]	256
Flatten-13	[-1, 131072]	0
Dropout-14	[-1, 131072]	0
Linear-15	[-1, 1024]	134,218,752
Dropout-16	[-1, 1024]	0
Linear-17	[-1, 512]	524,800
Dropout-18	[-1, 512]	0
Linear-19	[-1, 256]	131,328
Dropout-20	[-1, 256]	0
Linear-21	[-1, 8]	2,056
Total params: 134,970,632		
Trainable params: 134,970,632		
Non-trainable params: 0		

Zdjęcie 21 - architektura modelu 5 z dobranymi parametrami powłok



Wykres 7 - wykres funkcji straty oraz skuteczności sieci

Pomimo zniwelowania efektu przetrenowania sieci, spadek wartości straty zestawu walidacyjnego od pewnego momentu jest dość mały. Jest to związane z wartościami parametrów uczących, nie zostały one bowiem jeszcze zoptymalizowane, mało tego, część z nich nie została nawet zadeklarowana co powoduje pominięcie kilku użytecznych w procesie uczenia narzędzi. W celu jeszcze lepszej poprawy skuteczności modelu, w następnym paragrafie przedstawiony zostanie proces doboru parametrów uczących.

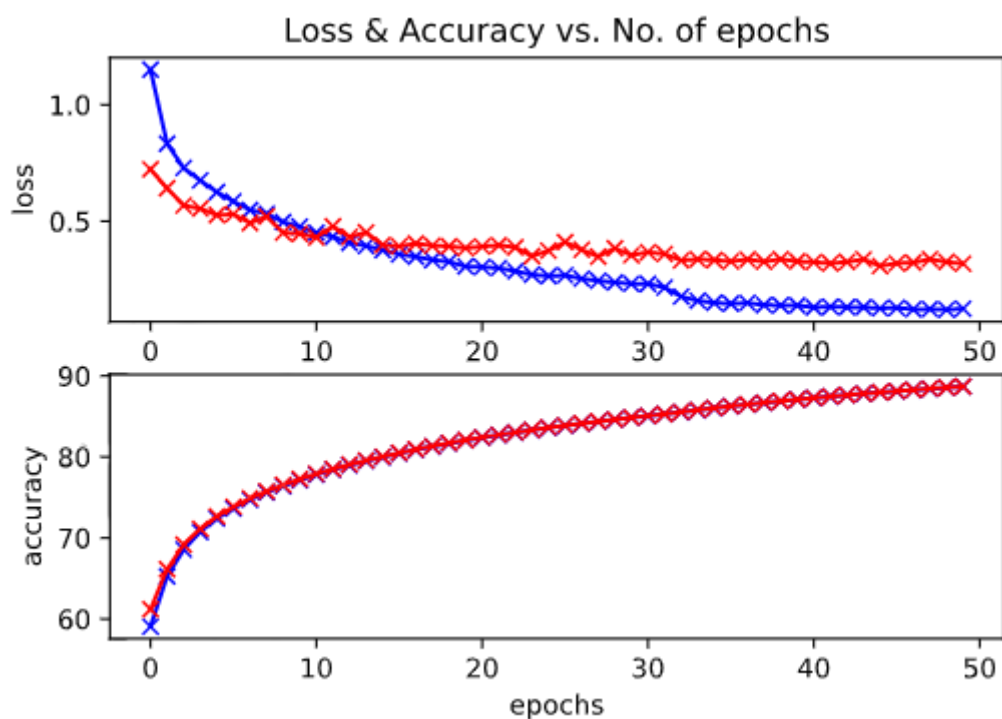
## 4.2. OPTYMALIZACJA ORAZ UCZENIE MODELU

Poza hiperparametrami poszczególnych powłok sieci, na proces uczenia bardzo duży wpływ mają parametry uczące. Odpowiedni dobór parametrów uczących może poprawić skuteczność sieci nawet o kilka procent. Przed przystąpieniem do optymalizacji sieci dobrze jest zrozumieć jak działają i na co wpływ mają poszczególne parametry.

- Krok uczenia - jest to skok funkcji optymalizacyjnej, kiedy jest za mały sieć nie jest w stanie eksplorować całego zestawu zdjęć. Kiedy krok jest za duży, funkcja nie jest w stanie dotrzeć do szczegółów zestawu.
- Liczba epok - liczba osobnych procesów uczenia, kiedy liczba epok jest za mała, sieć jest niedouczona, z kolei zbyt duża liczba epok bardzo często prowadzi do przetrenowania sieci.
- Momentum - opcjonalna wartość gradientowej funkcji optymalizacyjnej, określa zapożyczone z fizyki pojęcie pędu. Pomaga funkcji optymalizacyjnej zmierzać w istotnym dla problemu optymalizacji kierunku.
- Spadek wagi (weight decay) - spadek wag podczas procesu nauki.
- Planista (scheduler) - metoda sprawdzająca postęp procesu nauki, odpowiednie ustawienie scheduler'a pozwala na zmniejszenie szansy przetrenowania sieci.
- Liczność zestawu weryfikującego - liczba zdjęć służących tylko do weryfikacji postępów nauki, sieć nie uczy się na tym zestawie. Zestaw weryfikujący jest niezbędny do pracy scheduler'a.

Podobnie jak w przypadku parametrów poszczególnych powłok, optymalizacja parametrów uczących jest długotrwałym procesem, który wymaga wiele prób. Z tego względu w niniejszym paragrafie przedstawiona zostanie tylko ostatnia z kilkunastu przeprowadzonych prób optymalizacji parametrów.

Przeprowadzenie odpowiednich testów pozwoliło na zoptymalizowanie parametrów uczących sieci.



Wykres 8 - wykres funkcji straty oraz skuteczności sieci

Epoch: 20	Training Loss: 0.305290	Validation Loss: 0.384485
Epoch: 21	Training Loss: 0.303153	Validation Loss: 0.391150
Epoch: 22	Training Loss: 0.298664	Validation Loss: 0.396959
Epoch: 23	Training Loss: 0.285441	Validation Loss: 0.387886
Epoch: 24	Training Loss: 0.270739	Validation Loss: 0.350153
Epoch: 25	Training Loss: 0.264105	Validation Loss: 0.373035
Epoch: 26	Training Loss: 0.267424	Validation Loss: 0.410455
Epoch: 27	Training Loss: 0.252860	Validation Loss: 0.377367
Epoch: 28	Training Loss: 0.244468	Validation Loss: 0.347893
Epoch: 29	Training Loss: 0.236894	Validation Loss: 0.383495
Epoch: 30	Training Loss: 0.231493	Validation Loss: 0.354081
Epoch: 31	Training Loss: 0.230795	Validation Loss: 0.366916
Epoch 32: reducing learning rate of group 0 to 3.0000e-05.		
Epoch: 32	Training Loss: 0.215353	Validation Loss: 0.359064
Epoch: 33	Training Loss: 0.176271	Validation Loss: 0.331471
Epoch: 34	Training Loss: 0.158358	Validation Loss: 0.336751
Epoch: 35	Training Loss: 0.149648	Validation Loss: 0.333116

Zdjęcie 22 - działanie scheduler'a

Skuteczność sieci poprawiła się o ok. 5% i w najlepszej z przygotowanych sieci wynosi ok 89% (wykres nr 8). Uzyskanie takiego wyniku było możliwe dzięki zastosowaniu planisty,

którego działanie zostało zobrazowane na zdjęciu nr 22 oraz odpowiednio dobranym parametrom:

- Krok uczenia - **0.0003**
- Liczba epok - **50**
- Momentum - **0.9**
- Spadek wagi (weight decay) - **0**
- Planista (scheduler) - **10-cio krotne zmniejszenie kroku uczenia po 3 epokach bez poprawy skuteczności**
- Liczność zestawu weryfikującego - **15% głównego zestawu**

#### **4.3. SIECI WSTĘPNIE PRZETRENOWANE**

Przygotowanie architektury oraz optymalizacja parametrów pozwoliły na uzyskanie skuteczności na poziomie ok 89%. Wyższy wynik nie jest osiągalny ze względu na ograniczony rozmiar przygotowanej sieci. Chcąc osiągnąć wyższą skuteczność konieczne jest rozbudowanie jej architektury. Ponieważ skuteczność sieci zależy od wielu czynników, jej rozbudowa a następnie optymalizacja jest procesem bardzo skomplikowanym i powolnym.

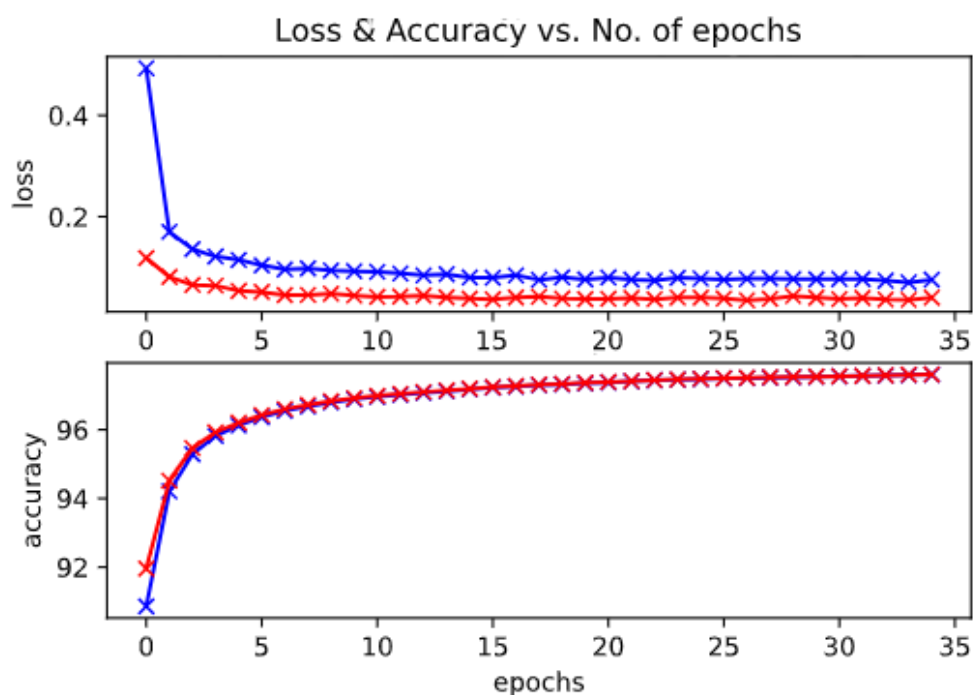
Dobłą alternatywą do rozbudowywania sieci jest skorzystanie z gotowych sieci wstępnie przetrenowanych. Są to sieci, które zostały przygotowane oraz nauczone, aby rozwiązywać konkretny problem, mogą one jednak w łatwy sposób zostać dostosowane do rozwiązania podobnego problemu.

Sieci przetrenowane nie tylko mają znacznie bardziej zaawansowaną architekturę, ale także są uczone przy użyciu dużo większych zestawów danych, co sprawia, że ich “wiedza” jest nieporównywalnie większa. Wiedzę sieci przedstawić można pod postacią **wag** reprezentujących poszczególne elementy rozpoznawalnych obiektów. Wykorzystanie zgromadzonych przez sieć wag jest możliwe ponieważ wiele obiektów dzieli wspólne cechy, dzięki czemu wiedza sieci może zostać wykorzystana do klasyfikacji innych obiektów.

Pracę z sieciami przetrenowanymi rozpocząć należy od wybrania interesującej nas sieci. Ponieważ w pracy rozpatrywany jest problem klasyfikacji obrazów, jako model dobrana została sieć ResNet34, która została stworzona właśnie do tego zadania. Następnie sieć musi zostać dostosowana do rozwiązywanego problemu, poprzez usunięcie i zastąpienie ostatniej

powłoki liniowej. Winna ona być zastąpiona powłoką, która odpowiada potrzebom projektu - w przypadku budowanego portalu jest to powłoka liniowa o ośmiu wyjściach (sieć rozpoznaje osiem obiektów).

Po dostosowaniu architektury sieci możemy przystąpić do jej uczenia. Ponieważ chcemy wykorzystać zebraną przez sieć wiedzę, proces uczenia będzie nieco różnił się od sposobu poznanego przy uczeniu poprzedniej sieci. Zachowanie nauczonych wag i parametrów jest możliwe dzięki “zamrożeniu” wszystkich poza ostatnią powłok sieci i wykonanie procesu uczenia tylko na niej.



Wykres 9 - wykres funkcji straty oraz skuteczności sieci

```
Test Accuracy: 97.0% (776.0/800.0)
Accuracy of baby : 98 %
Accuracy of cars : 98 %
Accuracy of cats : 96 %
Accuracy of dogs : 95 %
Accuracy of product : 98 %
Accuracy of food : 95 %
Accuracy of gun : 98 %
Accuracy of shoes : 96 %
```

Zdjęcie 23 - test skuteczności dla poszczególnych kategorii obiektów

Po zoptymalizowaniu parametrów uczących i przeprowadzeniu procesu nauki, sieć osiągnęła skuteczność na poziomie ok. 97%. Jest to rewelacyjny wynik, co potwierdzają przeprowadzone testy.

#### **4.4. PODSUMOWANIE**

Pierwsza z przygotowanych sieci pokazała, że tworzenie i optymalizacja architektury od podstaw jest skutecznym podejściem. Należy jednak zastanowić się, czy jest to konieczne. Tworzenie sieci od podstaw jest dobrym podejściem w przypadku bardzo zaawansowanego problemu, kiedy chcemy idealnie dostosować architekturę do rozwiązywanego zestawu danych. W przypadku, kiedy rozwiązywany problem nie jest skomplikowany, lepszym podejściem mogą okazać się sieci wstępnie przetrenowane. Przygotowana w paragrafie 4.3 architektura pozwoliła na osiągnięcie skuteczności na poziomie 97%, jest to prawie dziesięć procentowa poprawa względem sieci stworzonej od zera. Oczywiście skuteczność gorszej sieci podyktowana jest przede wszystkim jej bardzo małym rozmiarem, prawdopodobnie rozbudowanie i dostosowanie sieci do zestawu danych pozwoliłoby na osiągnięcie wyniku w okolicach 100% skuteczności. Projektowanie sieci jest jednak bardzo czasochłonne, dlatego alternatywa w postaci sieci wstępnie przeciwcichzonych będzie w zdecydowanej większości przypadków lepszym wyborem.

Ponieważ poza skutecznością różnice między obiema sieciami są znikome, na portalu internetowym zaimplementowana zostanie sieć o wyższej skuteczności.

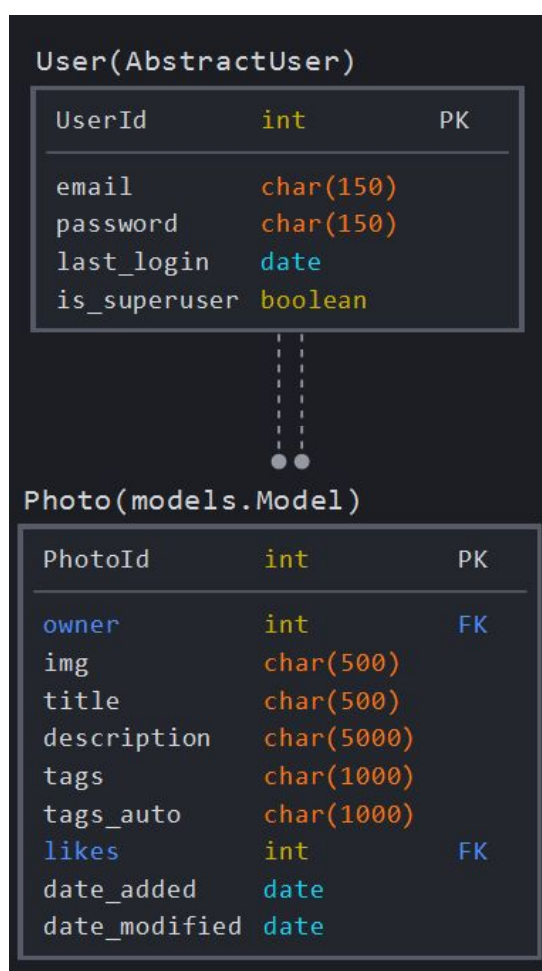
### **5. Implementacja sieci na stronie internetowej**

#### **5.1. TWORZENIE STRONY**

Przygotowywana strona internetowa w założeniu ma być prostym portalem społecznościowym. Portale tego typu zazwyczaj są jednak dużo bardziej złożone od np. sklepów internetowych, a to co widzi użytkownik takiego portalu jest tylko wierzchołkiem góry lodowej. Najważniejsze operacje i działania serwisu, dzieją się po stronie serwera i są

niewidoczne dla użytkowników. Przygotowując więc taki portal, to właśnie back-end aplikacji należy postawić na pierwszym miejscu. Z tego względu, podczas tworzenia strony, większa uwaga poświęcona zostanie back-end'owi aplikacji niż wyglądowi dostępnego dla użytkowników interfejsu. Dlatego do mniejszych postanowień projektu, dodać można fakt, iż front-end aplikacji winien być przede wszystkim użyteczny, jego wygląd dopracować można w późniejszym cyklu życia aplikacji.

Proces tworzenia strony warto zacząć od przygotowania schematu bazy danych, pozwoli to na przygotowanie najefektywniejszej architektury. W tym celu wykorzystane zostało darmowe narzędzie dostępne na stronie <https://sqldb.com>.



Zdjęcie 24 - schemat bazy danych

Przygotowanie modelu architektury, jest co prawda szczególnie użyteczne kiedy przewidywany rozmiar bazy danych jest duży. Jednakże model bazy jest przydatny w



każdych warunkach. Przygotowana architektura (zdz. nr 24) zostanie wykorzystana w następnym etapie tworzenia portalu. Dzięki niej napisanie kodu odpowiedzialnego za tabele w bazie danych będzie łatwiejsze. Ponadto, zamodelowanie architektury pozwoliło na najefektywniejsze użycie poszczególnych tabeli.

Kiedy model bazy danych jest już gotowy, możemy przejść do tworzenia back-end'u.

Pracę nad każdą aplikacją w języku Python warto zacząć od stworzenia wirtualnego środowiska, jest to osobna instancja interpretera języka, w której zainstalować możemy interesujące nas biblioteki. Jest to dobra praktyka, ponieważ pozwala na dobrą organizację rozszerzeń oraz na łatwiejsze przeniesienie programu na inną instancję interpretera (np. na innym komputerze). Stworzenie wirtualnego środowiska odbywa się poprzez wpisanie odpowiedniej komendy do konsoli PowerShell.

Następnie w aktywnym nowym środowisku, należy zainstalować niezbędne biblioteki. Jednym z najefektywniejszych narzędzi, które może być do tego wykorzystane jest instalator paczek **pip**. Przy użyciu instalatora pip dodane do wirtualnego środowiska zostały niezbędne biblioteki. Do najważniejszych z nich zaliczyć można:

- Django
- Pillow
- OpenCV
- torch
- torchvision

Po przygotowaniu odpowiedniego środowiska, należy stworzyć projekt w bibliotece Django. Służy do tego odpowiednia komenda wpisana w konsoli PowerShell.

```
$ django-admin startproject AGHDL
```

Komenda automatycznie tworzy wszystkie pliki niezbędne do działania aplikacji w folderze o nazwie **AGHDL**. Nasza aplikacja jest gotowa, aby sprawdzić czy strona działa wystarczy wpisać komendę.

```
$ python AGHDL\manage.py runserver
```

A następnie otworzyć podany przez Django adres w przeglądarce.

```
python AGHDL\manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
January 07, 2021 - 21:11:06
Django version 3.1.3, using settings 'AGHDL.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

*Zdjęcie 25 - informacja o uruchomieniu strony*

django

[View release notes](#) for Django 3.1



The install worked successfully! Congratulations!

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.



**Django Documentation**  
Topics, references, & how-to's



**Tutorial: A Polling App**  
Get started with Django



**Django Community**  
Connect, get help, or contribute

*Zdjęcie 26 - strona internetowa*

Wspaniale, stworzona strona działa, teraz wystarczy uzupełnić projekt odpowiednią zawartością.

Aby folder projektu pozostał uporządkowany, każda funkcjonalność serwisu zostanie oddzielona do nieco mniejszego folderu wewnątrz głównego folderu projektu. Każdy ze stworzonych folderów będzie tworzył aplikację, która współpracować będzie z aplikacją główną.

Chcąc stworzyć aplikację należy użyć komendy:

```
$ python manage.py startapp *nazwa aplikacji*
```

A następnie dodać stworzoną aplikację do listy aplikacji w pliku **settings.py**.

Przy użyciu wspomnianej komendy stworzone zostały następujące aplikacje oraz ich foldery:

- accounts

Aplikacja odpowiedzialna za operacje związane z kontami użytkowników.

- photos

Aplikacja zawierająca modele oraz funkcje odpowiedzialne za przechowywanie, wyświetlanie oraz dodawanie zdjęć przez użytkowników

- stream

Aplikacja zawierająca funkcje umożliwiające użytkownikom portalu przesyłanie strumieniowe (streaming) na żywo.

Poza folderami aplikacji do projektu dodane zostały foldery:

- media

Folder odpowiedzialny za przechowywanie plików udostępnionych na portalu.

- static

Folder, w którym przechowywane są pliki, do których chcemy dać portalowi dostęp globalny. Django rozwiązuje problem dostępu do plików z poziomu różnych folderów, dając do

dyspozycji tzw. **static folder**. Deklaracja folderu static odbywa się w pliku **settings.py**, dodając:

```
STATIC_URL = '/static/'  
STATICFILES_DIRS = [os.path.join(BASE_DIR, 'static')]
```

W folderze static znajdują się takie pliki jak ikony lub skrypty, które używane są na więcej niż jednej podstronie portalu. Dostęp do folderu najczęściej wykorzystywany jest w plikach HTML. Dodanie:

```
{% load static %}
```

pozwala na dostęp do folderu bez podawania jego ścieżki, np:

```
{% static 'icons/home_32.png' %}
```

Jest to szczególnie użyteczne podczas tworzenia np. paska nawigacji, czy generalnego układu strony obecnego na wszystkich podstronach portalu, ponieważ nie wymaga dostosowywania ścieżki pliku do aktualnego folderu pracy.

- templates

Folder przechowujący przygotowane szablony stron w języku HTML, które następnie używane są do budowania poszczególnych podstron witryny.

Przygotowywanie portalu warto zacząć od stworzenia modelu konta użytkownika. Dostępny w bibliotece Django model ma bowiem jedną poważną wadę - do logowania używany jest podany przez użytkownika login zamiast adresu e-mail. Jednym ze sposobów rozwiązania tego problemu jest stworzenie w bazie danych odpowiedniej tabeli, zawierającej poprawny model użytkownika a następnie uzupełnienie metod odpowiedzialnych za tworzenie i edycję konta w aplikacji.

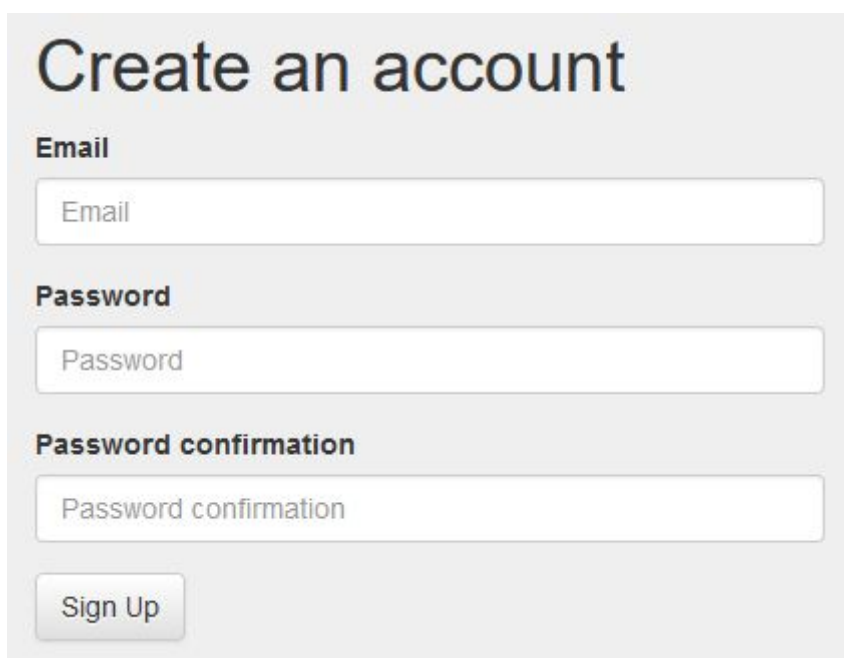
Proces ten, najlepiej rozpocząć od deklaracji modelu konta użytkownika w pliku **models.py** w folderze accounts. Na podstawie wszystkich plików models.py, automatycznie generowane są tabele oraz połączenia między nimi w bazie danych. Zaczynamy więc defacto od stworzenia odpowiedniej tabeli w bazie danych, z którą współpracuje portal. Model użytkownika jest bardzo prosty, wszystkie pola dziedziczone są z przygotowanego przez Django modelu

**AbstractUser**, a naszym zadaniem jest jedynie usunięcie pola **username** oraz wskazanie pola e-mail, jako domyślny identyfikator konta:

```
USERNAME_FIELD = 'email'
```

Następnym krokiem jest przygotowanie menedżera odpowiedzialnego za dodawanie nowych kont do bazy danych. Podobnie jak w przypadku samego modelu, wystarczy wskazać adres e-mail jako domyślny identyfikator konta.

Na koniec należy przygotować formy widoczne na stronie internetowej, umożliwiające użytkownikowi tworzenie oraz edycję konta (zdj. nr 27).



*Zdjęcie 27 - forma do tworzenia konta*

Kolejnym etapem budowy portalu jest dodanie tabeli przechowującej zdjęcia. W tym celu w folderze photos do pliku models.py dodany został model zgodny z zaprojektowaną w poprzednim paragrafie tabelą **Photo** (zdj. 24). Oznacza to, że wszystkie tabele z zaprojektowanego diagramu zostały zamodelowane.

Na razie są one jednak tylko kilkoma linijkami kodu, aby zamienić je w użyteczne tabele w bazie danych należy wykonać migrację. W tym celu wykorzystane zostały komendy:

**\$ python AGHDL\manage.py makemigrations**

```
(env) PS C:\AGH\Django> python AGHDL\manage.py makemigrations
Migrations for 'accounts':
  AGHDL\accounts\migrations\0001_initial.py
    - Create model CustomUser
Migrations for 'photos':
  AGHDL\photos\migrations\0001_initial.py
    - Create model Photo
```

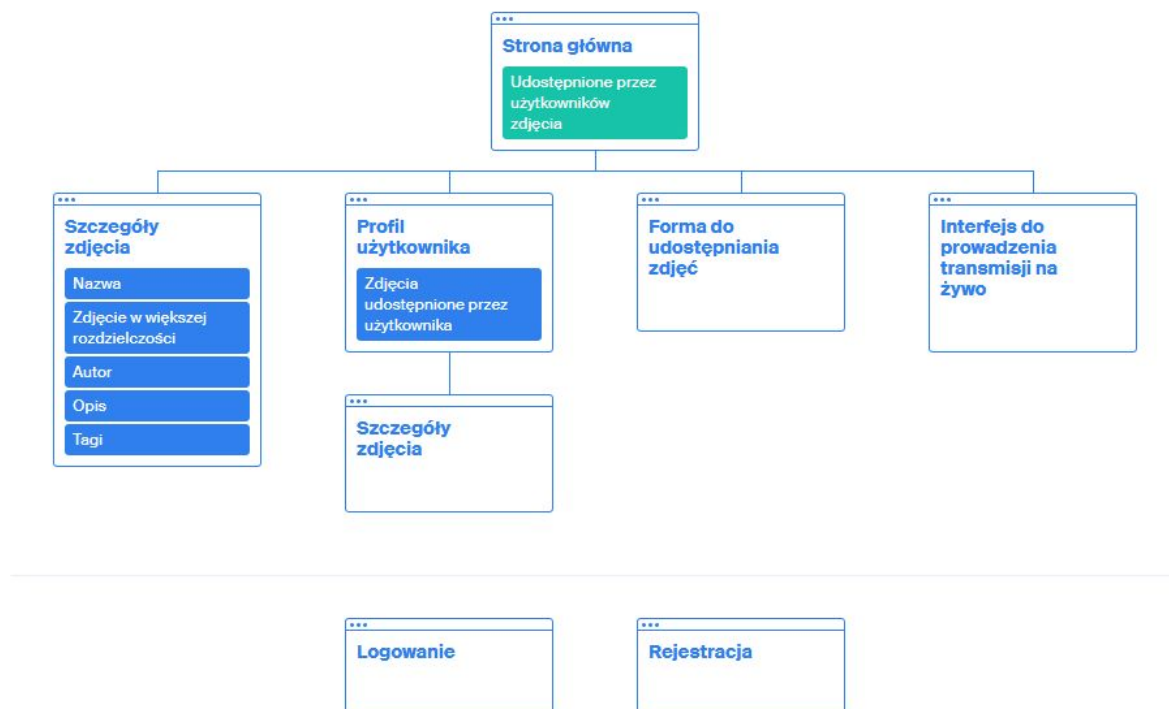
*Zdjęcie 28 - przeprowadzone migracje bazy danych*

**\$ python AGHDL\manage.py migrate**

```
(env) PS C:\AGH\Django> python AGHDL\manage.py migrate
Operations to perform:
  Apply all migrations: accounts, admin, auth, contenttypes, photos, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying accounts.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying photos.0001_initial... OK
  Applying sessions.0001_initial... OK
```

*Zdjęcie 29 - potwierdzenie przeprowadzenia migracji bazy danych*

Po wykonaniu migracji baza danych jest gotowa, najwyższy czas na przygotowanie strony internetowej. W tym celu stworzona została mapa strony (zdj. nr 30), przedstawiająca poszczególne podstrony oraz ich zawartość.



*Zdjęcie 30 - mapa strony internetowej*

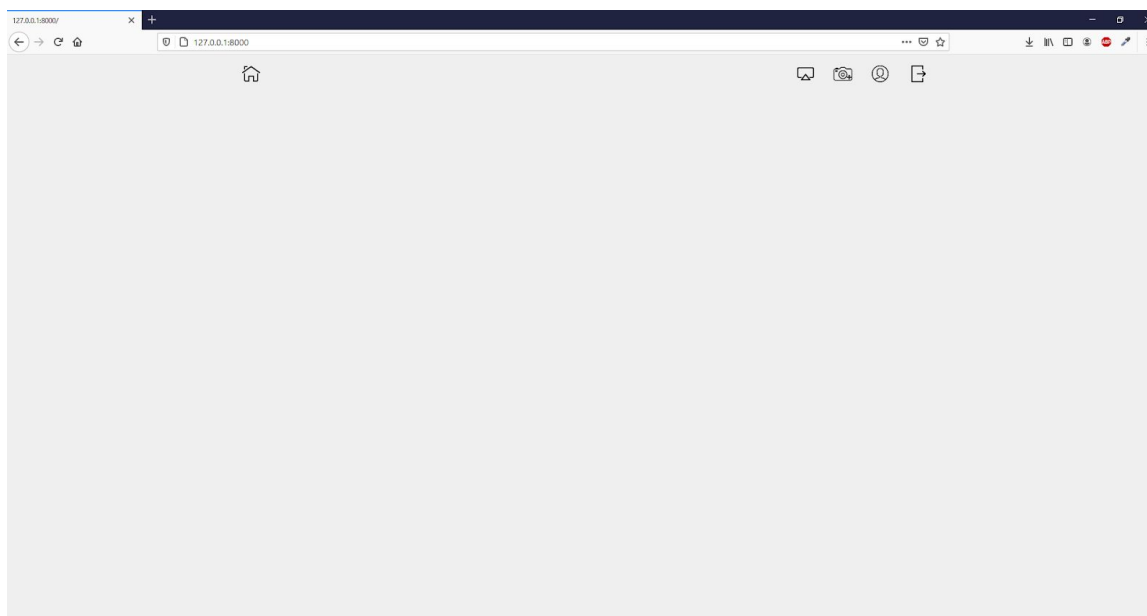
Tworzenie podstron rozpoczęte zostanie od dodania odpowiednich adresów url do pliku **urls.py**.

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path("", views.HomePage.as_view(), name='home'),
    path('accounts/', include('accounts.urls', namespace='accounts')),
    path('accounts/', include('django.contrib.auth.urls')),
    path('thanks/', views.ThanksPage.as_view(), name='thanks'),
    path('photos/', include('photos.urls', namespace='photos')),
    path('photos/', include('django.contrib.auth.urls')),
    path('stream/', include('stream.urls', namespace='stream')),
    path('stream/', include('django.contrib.auth.urls')),
]
```

Następnie w plikach **views.py** przygotowane zostały odpowiednie widoki dla każdej podstrony. Widoki odpowiedzialne są za “wyciągnięcie” informacji z bazy danych oraz przesłanie ich z serwera do front-end’u aplikacji. Dzięki połączeniu widoku z odpowiednim szablonem HTML, Django generuje całą podstronę automatycznie wraz z wyświetleniem odpowiednich rekordów pobranych z bazy danych. Przykładowa deklaracja widoku:

```
class PhotoDetailView(DetailView):  
    model = Photo  
    template_name = 'photos/photo_detail.html'
```

Po przygotowaniu widoków, strona jest praktycznie gotowa.



*Zdjęcie 31 - strona internetowa*

Co prawda jest jeszcze pusta, ale wszystkie narzędzia niezbędne do jej zapełnienia przez użytkowników są już dostępne. Na stronie można już bowiem założyć konto, dodać własne zdjęcie czy też oglądać fotografie udostępnione przez innych użytkowników. Następnym etapem jest wdrożenie inteligentnych rozwiązań wykorzystujących przygotowaną wcześniej sieć neuronową. W kolejnych paragrafach przedstawiony zostanie sposób przygotowania oraz implementacji poszczególnych rozwiązań.

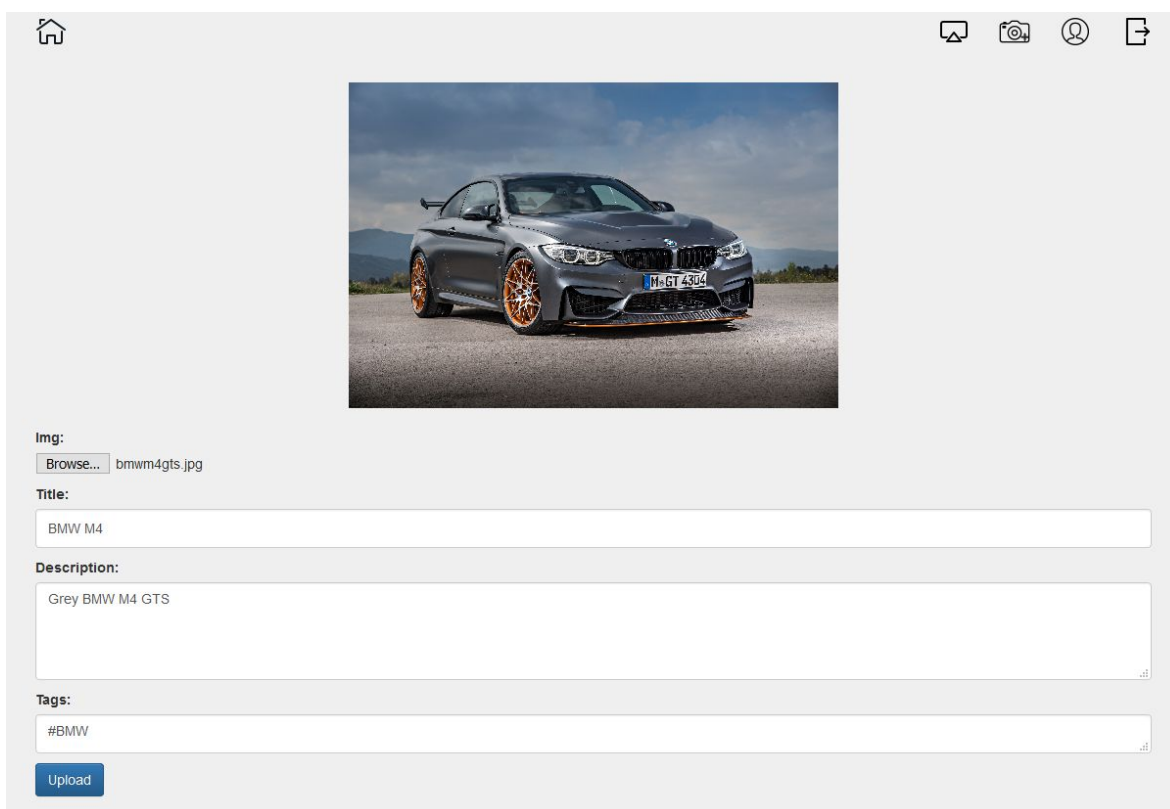


## 5.2. DODAWANIE TAGÓW DO ROZPOZNANYCH OBIEKTÓW

Pierwszym z przygotowanych rozwiązań jest automatyczne dodawanie do rozpoznanych obiektów odpowiednich etykiet ułatwiających identyfikację oraz wyszukiwanie.

Cały mechanizm działania rozwiązania opiera się na funkcji umieszczonej w pliku **views.py** oraz połączonej z odpowiednim adresem url.

Funkcja współpracuje z przygotowaną wcześniej formą służącą do udostępniania zdjęć.



Home icon, Camera icon, User icon, Share icon

Img:

Title:

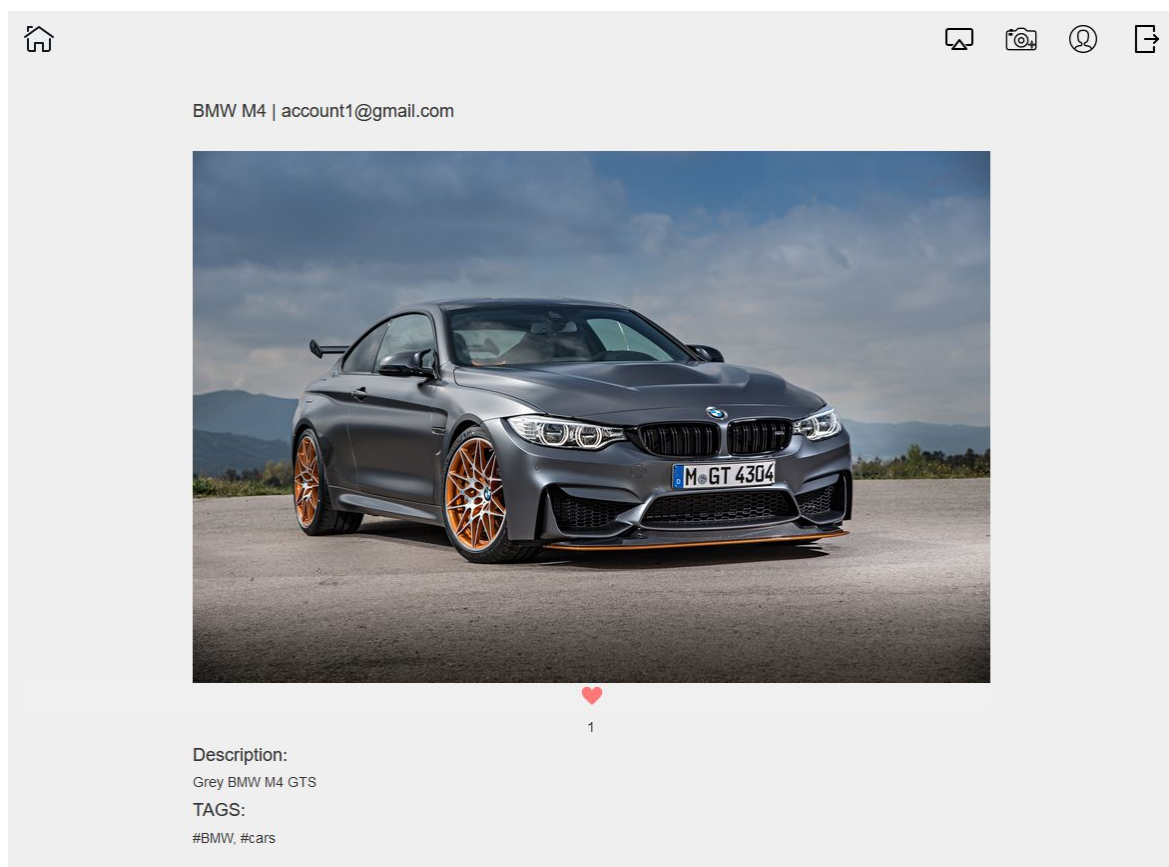
Description:

Tags:

*Zdjęcie 32 - uzupełniona forma do udostępniania zdjęć*

W momencie przesłania formy przez użytkownika, funkcja powstrzymuje aplikację przed dodaniem rekordu do bazy danych aż do momentu zakończenia operacji (jest to ważne, ponieważ chcemy aby cała operacja była wykonana podczas jednego zapisu do bazy danych). W tym czasie do pamięci serwera załadowana zostaje przygotowana wcześniej sieć neuronowa, następnie udostępnione zdjęcie zostaje przekonwertowane na odpowiednią dla przetworzenia przez sieć macierz (tensor). “Przepuszczenie” zdjęcia przez sieć generuje macierz procentowej zgodności obiektu z rozpoznawalnymi przez sieć obiektami. Jeżeli

zgodność dla któregoś obiektu wynosi ponad 80% oraz podana etykieta nie została dodana przez użytkownika to do zdjęcia dodany zostaje tag reprezentujący tę grupę obiektów. Cała operacja trwa ok. 2/3 sekundy i jest niezauważalna dla użytkownika, ponieważ dzieje się po stronie serwera.







*Zdjęcie 33 - udostępnione na portalu zdjęć*


Jak można zauważyć, w formie udostępniania zdjęć (zdj. nr 32) w polu TAGS wpisana została tylko jedna etykieta. Na zdjęciu nr 33 natomiast, w polu TAGS widnieją już dwie etykiety. Etykieta “cars” dodana została automatycznie i teraz może zostać wykorzystana do łatwiejszego wyszukania zdjęcia lub do przedstawienia w odpowiednim albumie.

### **5.3. SHADOW BANNING**

Przygotowane rozwiązanie shadow banning’u działa bardzo podobnie do przedstawionego w poprzednim paragrafie etykietowania zdjęć, a nawet opiera się o

wykorzystanie tej samej funkcji. Metoda przypisująca etykiety może bowiem dodać także etykiety, które nie są widoczne dla użytkownika. W tym celu do tabeli przechowującej zdjęcia dodane zostało specjalne pole, do którego wpisane zostają etykiety ukryte. Podobnie jak zwykle etykiety, zostają one dopisane do zdjęcia w momencie jego przesłania przez użytkownika i także są określone przez przygotowaną sieć neuronową. Ukryte etykiety dodane zostają do zdjęć naruszających regulamin portalu. Zgodnie z założeniami pracy, przygotowana sieć rozpoznaje 8 obiektów, z czego 2 z nich są niezgodne z regulaminem portalu. Jeżeli zatem użytkownik udostępni na portalu zdjęcie przedstawiające jeden z zabronionych obiektów a sieć rozpozna go z co najmniej 80% pewnością, to do ukrytych etykiet dodana zostaje informacja pozwalająca wyróżnić zdjęcie z pozostałych zdjęć znajdujących się na portalu.





**Img:**  
 240.custom\_jar\_labels.jpg

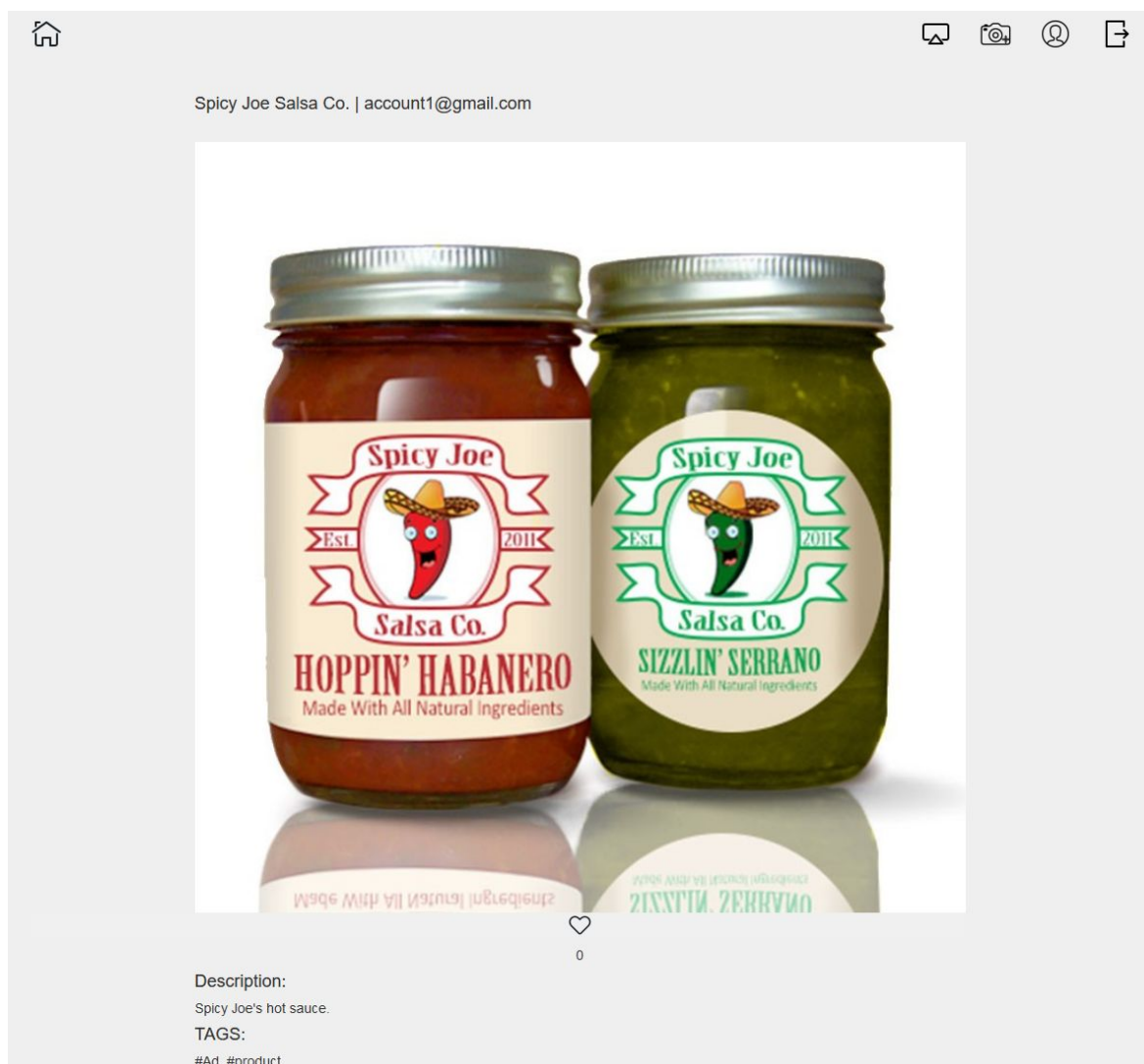
**Title:**

**Description:**

**Tags:**

*Zdjęcie 34 - uzupełniona forma do udostępniania zdjęć*

Zdjęcie 34 przedstawia wypełnioną formę wraz ze zdjęciem reklamowanego produktu, do pola **TAGS** wpisana została tylko jedna etykieta. Po przesłaniu formy, aktywowana zostaje funkcja odpowiedzialna za automatyczne dodawanie etykiet.





*Zdjęcie 35 - udostępnione na portalu zdjęcie*

Sieć neuronowa rozpoznała obiekt na zdjęciu i dodała do niego odpowiednią etykietę **product** (zdj. 35).

Ponieważ jednak udostępnianie zdjęć reklamujących produkty jest niezgodne z regulaminem portalu, do zdjęcia powinna zostać dodana także etykieta ukryta. Aby sprawdzić poprawność działania metody możemy posłużyć się panelem administratora, dostępnym pod adresem

**/admin.** Panel administratora pozwala na bezpośrednie wyświetlanie rekordów z bazy danych.

Owner:   

---

Img: Currently: [photos/uploads/240.custom\\_jar\\_labels\\_CeXKlz3.jpg](#)  
Change:  No file selected.

---

Title:

---

Description:

---

Tags:


---

Tags auto:



---

Likes: 

account1@gmail.com  
account2@gmail.com



*Zdjęcie 36 - zdjęcie łamiące regulamin widoczne w panelu administratora*

Owner:   

---

Img: Currently: [photos/uploads/bmwm4gts\\_ziNi7gg.jpg](#)  
Change:  No file selected.

---

Title:

---

Description:

---

Tags:


---

Tags auto:

---

Likes: 

account1@gmail.com  
account2@gmail.com

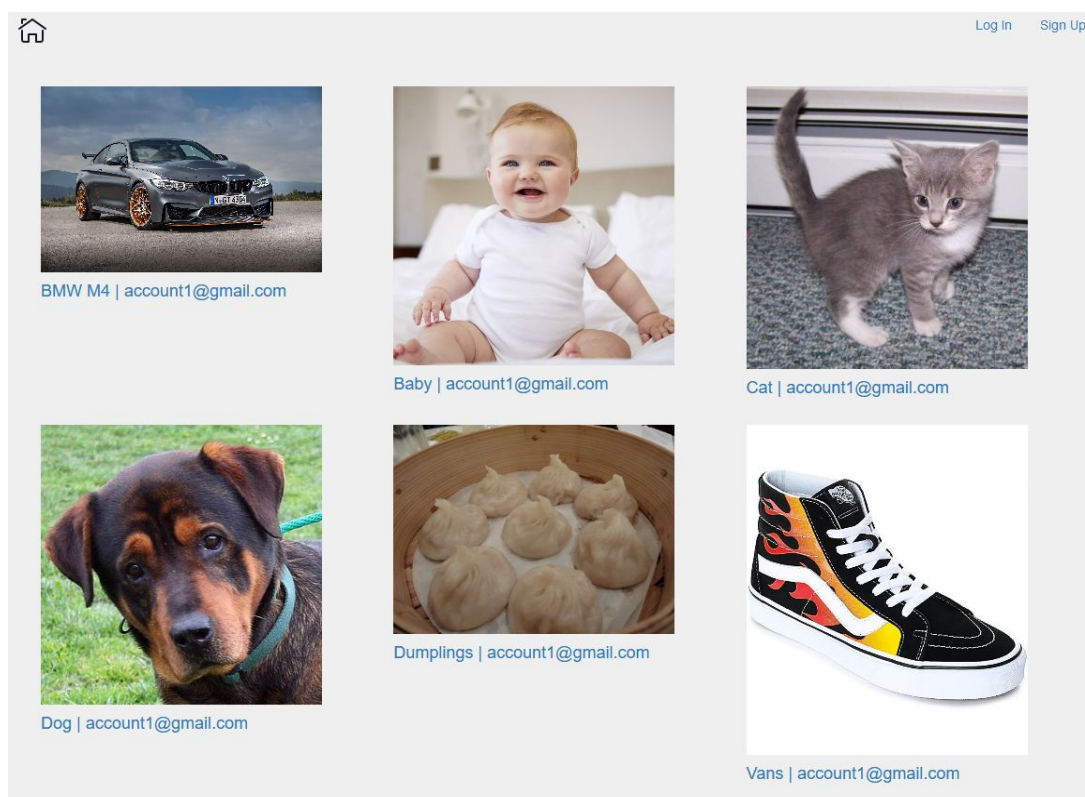


*Zdjęcie 37 - zdjęcie **nie** łamiące regulaminu widoczne w panelu administratora*

Zdjęcie nr 36 przedstawia rekord bazy danych, jest to udostępniony przed chwilą na portalu post reklamujący. Jak można zauważyć, w polu **Tags auto** wpisana została dodatkowa informacja **violation**. Jeżeli porównamy rekord (zdj. nr 36) z rekordem (zdj. nr 37) reprezentującym udostępniony wcześniej na stronie samochód (zdj. nr 33), to zauważyć możemy, że w drugim przypadku pole Tags auto jest puste - zdjęcie nie łamie regulaminu, zatem nie została dodana żadna ukryta etykieta. Oznacza to, że funkcja działa poprawnie.

Następnym etapem jest wykorzystanie przypisanych etykiet do zarządzania treściami na stronie. Sposób na filtrację zdjęć niezgodnych z regulaminem to tzw. shadow banning, który polega na usunięciu ze strony treści w taki sposób, aby ich autor nie zdawał sobie z tego sprawy. Sposób ten bardzo skutecznie powstrzymuje użytkowników przed wielokrotnym udostępnianiem tych samych niepożądanych treści, ponieważ nie wiedzą oni, że zdjęcie zostało usunięte.

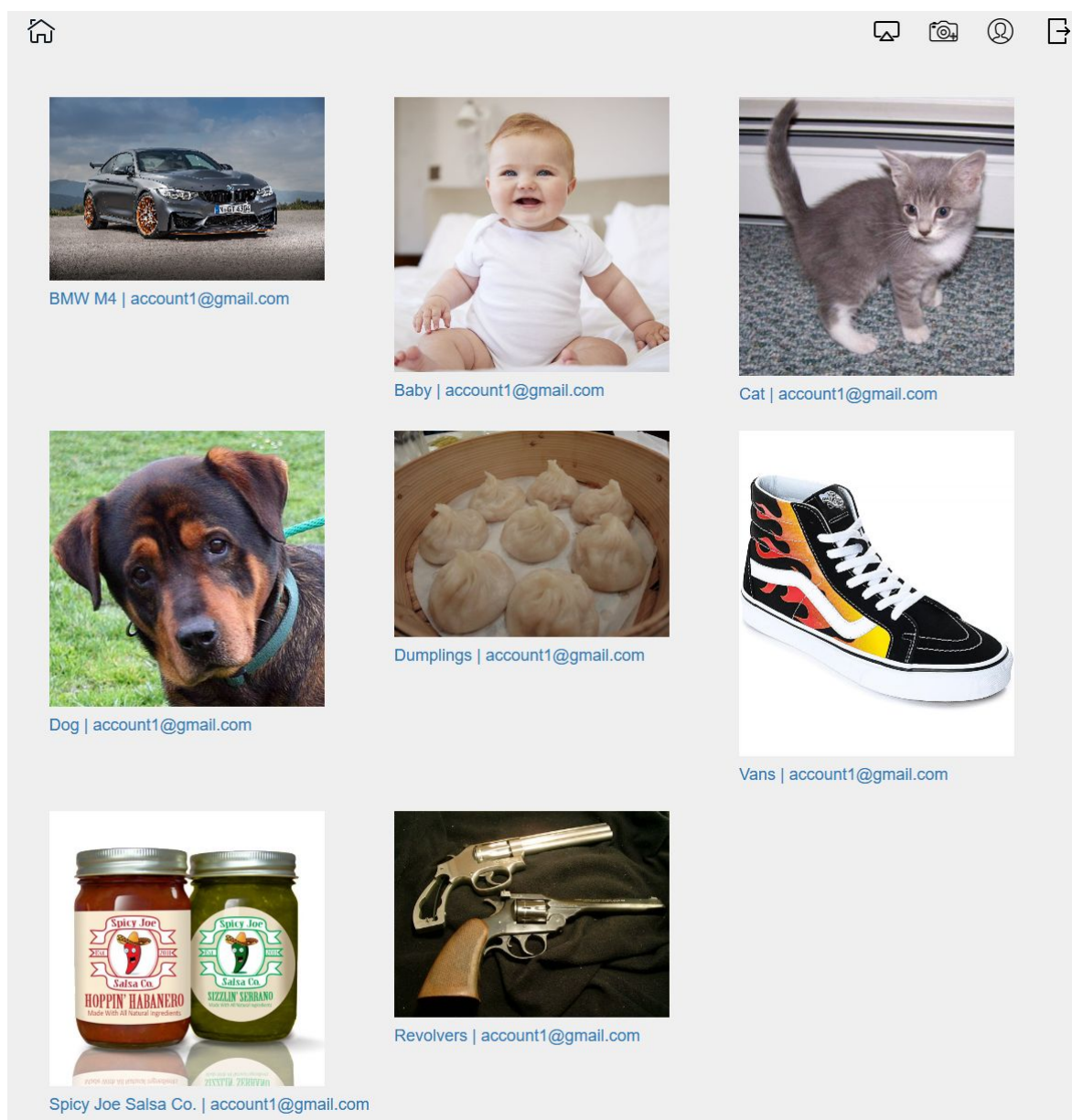
Implementacja rozwiązania jest bardzo prosta. Wspomniane wcześniej ukryte tagi, zostają wykorzystane do identyfikacji obrazów, jeżeli jakieś z nich posiada etykietę **violation**, to podczas ładowania treści z bazy danych na stronę jest ono pomijane i nigdy nie zostaje wyświetlone.



*Zdjęcie 38 - widok strony głównej dla wszystkich użytkowników*

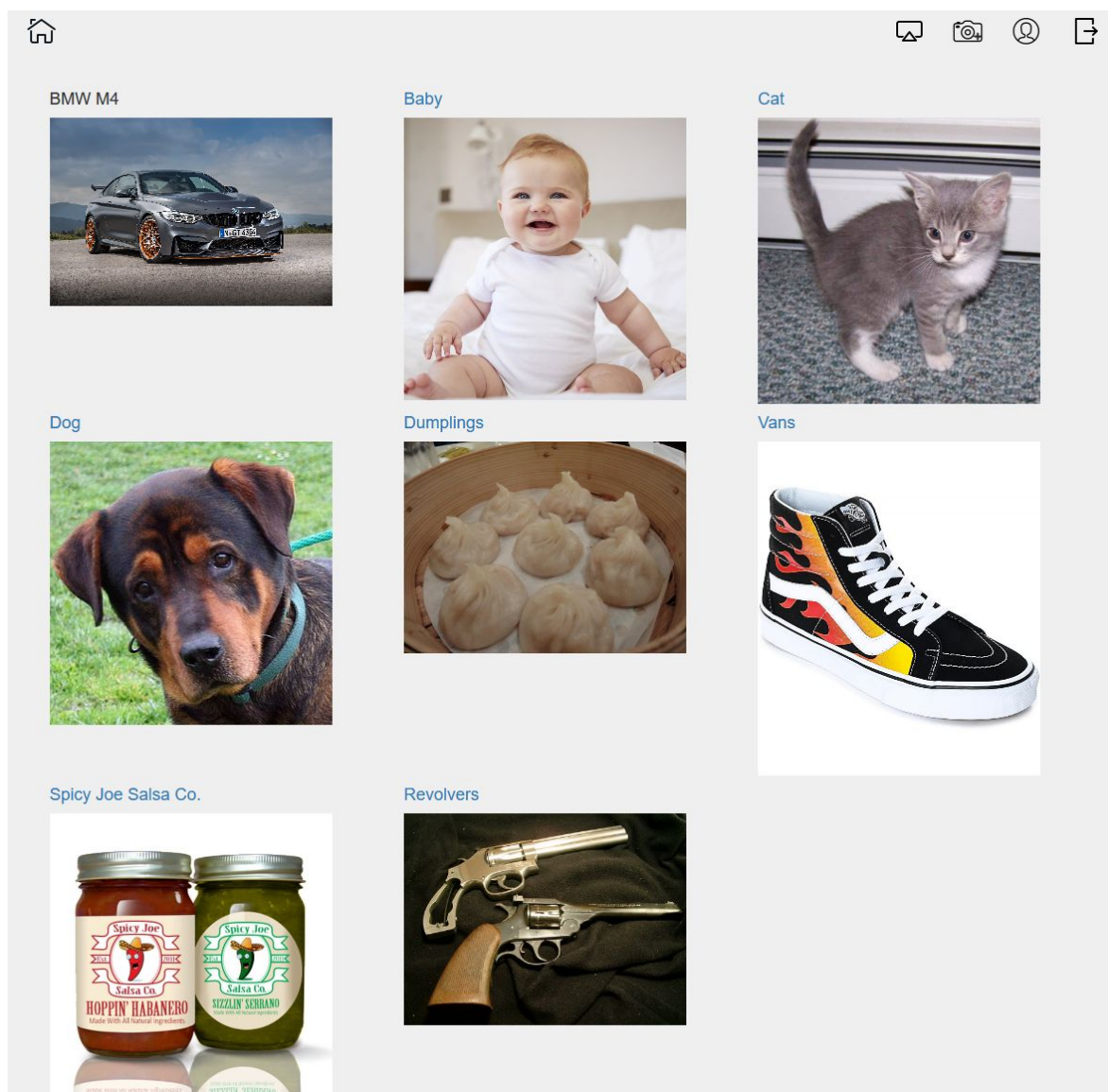


Jak widać na zdjęciu nr 38, wspomniany post reklamujący nie jest wyświetlany na stronie głównej. W ten sposób strona oraz użytkownicy chronieni są od niepożądanych treści, natomiast brakuje jeszcze zabezpieczenia przed ponownymi naruszeniami regulaminu. W tym miejscu z pomocą przychodzi wspomniany wcześniej shadow banning. Jego implementacja jest równie prosta, do funkcji sprawdzającej obecność ukrytej etykiety dołączyć należy jeden warunek - nie ukrywaj zdjęcia jeżeli prośba jego załadowania na stronie pochodzi od właściciela.



Zdjęcie 39 - widok strony głównej dla użytkownika, który udostępnił zdjęcia łamiące regulamin

Zdjęcie nr 39 przedstawia widok strony głównej dla autora zdjęć łamiących regulamin. Jak możemy zauważyć zdjęcia niezgodne z regulaminem nie zostały ukryte.



*Zdjęcie 40 - widok profilu użytkownika dla autora zdjęć łamiących regulamin*

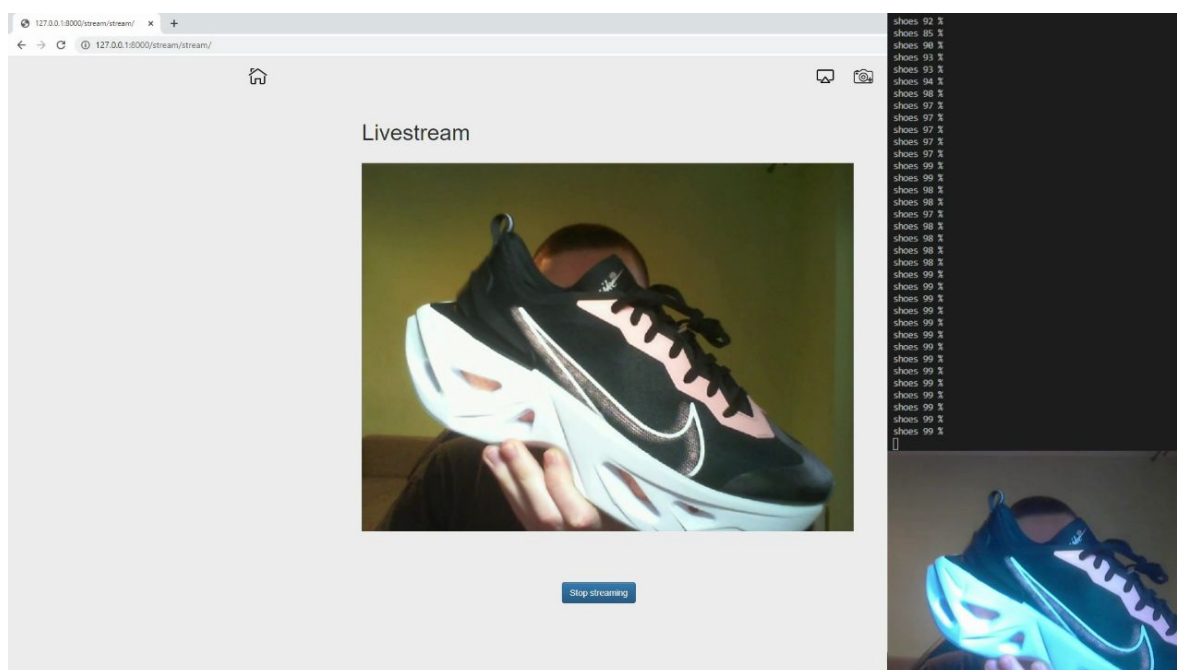
Autor może także znaleźć swoje zdjęcia na swoim profilu (zdj. 40) co dodatkowo upewnia go, że fotografie nie zostały poddane żadnej cenzurze.



## 5.4. ROZPOZNAWANIE OBIEKTÓW PODCZAS TRANSMISJI NA ŻYWO

Rozpoznawanie obiektów na żywo jest kolejnym elementem zawartym na portalu. Wykonywanie obliczeń w czasie rzeczywistym możliwe jest dzięki wykorzystaniu biblioteki **cv2** oraz **Pillow**. Rozpoznawanie obiektów odbywa się poprzez przechwycenie poszczególnych klatek obrazu przesyłanego przez kamerkę internetową a następnie “przepuszczenie” ich przez przygotowaną sieć neuronową.

Cały proces zaczyna się w momencie rozpoczęcia transmisji na żywo, to wtedy inicjalizowana jest sieć neuronowa, która pozostaje w pamięci aż do zakończenia transmisji. Następnie, przesyłany przez kamerkę obraz, dzielony jest na poszczególne klatki, które po poddaniu odpowiedniej obróbce przekazywane są do sieci neuronowej. Sieć następnie klasyfikuje obiekty na każdej z podanych klatek i zwraca ich procentową zgodność z rozpoznawalnymi obiektami.



*Zdjęcie 41 - rozpoznawanie obiektów w czasie rzeczywistym*

Klasyfikacja jest naprawdę efektywna, rozpoznanie obuwia, pomimo słabego jakościowo obrazu odbywa się z pewnością na poziomie ok. 98/99 %. Praca narzędzia jest szybka oraz płynna, należy pamiętać jednak, że przetwarzanie ok. 30 zdjęć na sekundę bardzo obciąża

serwer i prawdopodobnie do obsługi większej ilości transmisji niezbędne będą duże zasoby sprzętowe.

## **5.5. AUTOMATYCZNE WYŁĄCZANIE TRANSMISJI**

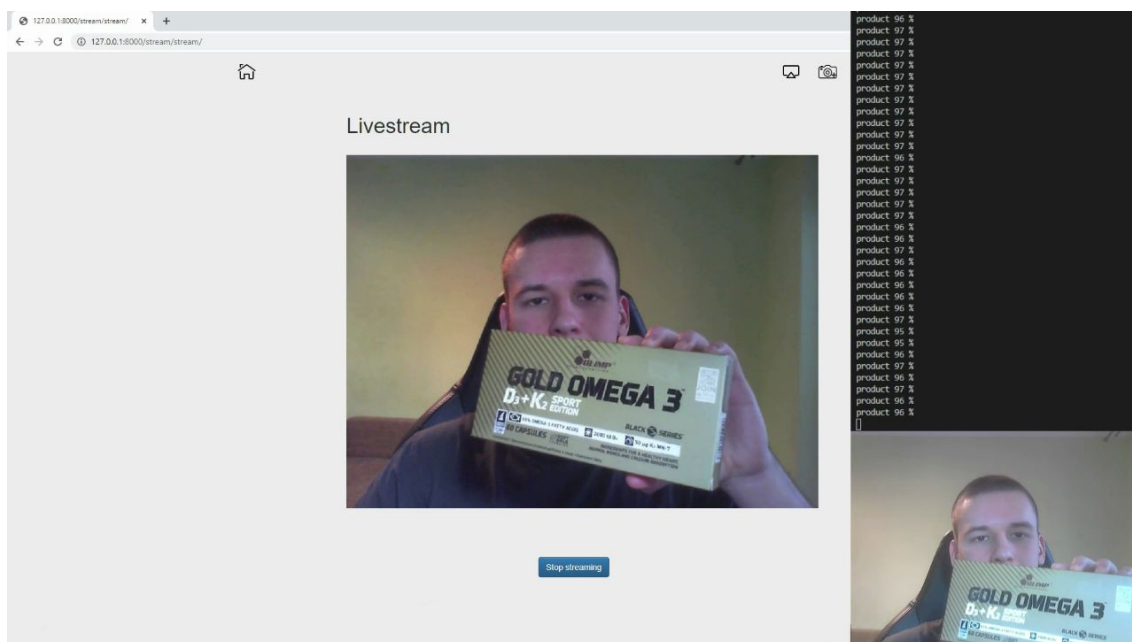
Rozpoznawanie obiektów w czasie rzeczywistym jest efektywne, na razie jednak nie ma żadnego użytecznego zastosowania.

Portal jest już chroniony przed niechcianymi zdjęciami, pozostało więc przygotować rozwiązanie zabezpieczające transmisje na żywo. Wykorzystując rozpoznawanie obiektów przygotowane zostało rozwiązanie monitorujące zgodność zawartości transmisji z regulaminem.

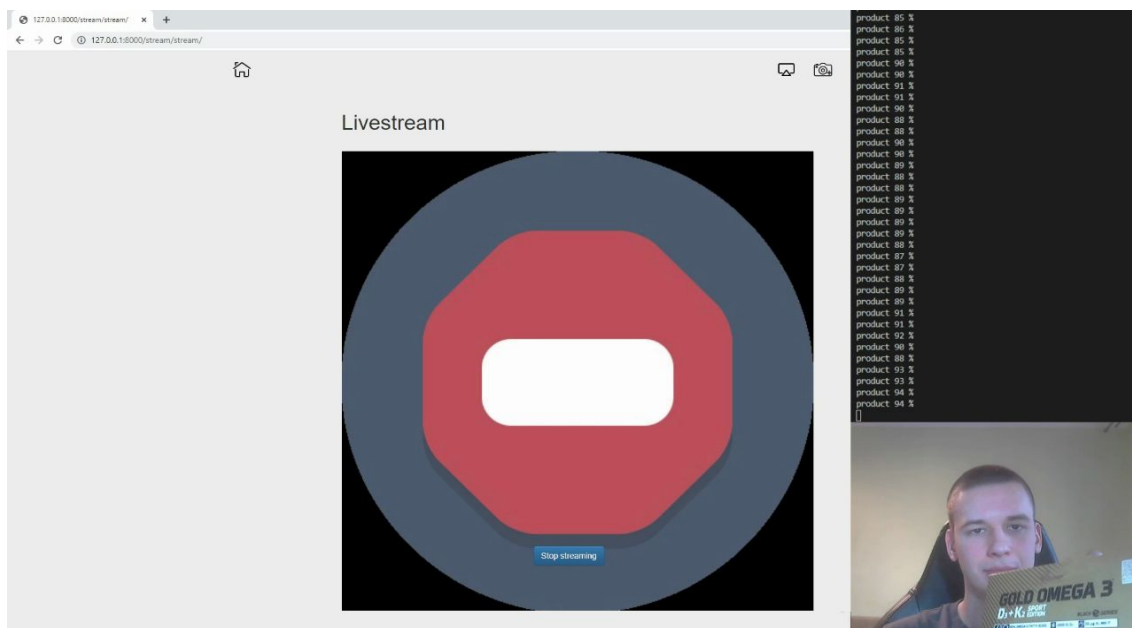
Funkcja odpowiedzialna za klasyfikowanie obiektów podczas transmisji została rozbudowana o nowe funkcjonalności. Najważniejszą z nich jest element monitorujący rozpoznane przez sieć obiekty. Ponieważ sieć poddaje klasyfikacji ok. 30 obrazów na sekundę, dość prawdopodobne staje się ryzyko chociaż jednej błędnej klasyfikacji w czasie trwania transmisji. Powoduje to, że rozwiązanie polegające na wyłączeniu transmisji w momencie pojawienia się obiektu łamiącego regulamin jest skrajnie nieskuteczne. Dobrym rozwiązaniem wydaje się zatem dodanie pewnego rodzaju licznika, który liczył będzie obrazy sklasyfikowane jako łamiące regulamin. Problemem jest ustalenie odpowiedniej liczby, przy której transmisja zostanie przerwana. Kiedy liczba będzie zbyt duża, zabezpieczenie nie zadziała wystarczająco szybko. Z drugiej strony, gdy liczba będzie zbyt mała, autorzy dłuższych transmisji mogą zostać niesłusznie ukarani, kiedy licznik osiągnie graniczną wartość, licząc tylko błędnie sklasyfikowane obrazy. Dlatego, aby zapewnić możliwie skuteczną pracę algorytmu, do licznika dodany został warunek resetujący licznik. Teraz transmisja zostaje wyłączona, gdy licznik osiągnie graniczną wartość 200 obrazów niezgodnych z regulaminem podczas, gdy licznik resetowany jest co 1000 klatek. Wartości te nie są przypadkowe, przeprowadzone testy jednoznacznie wykazały ich wysoką skuteczność. Połączenie tych dwóch wartości pozwala na bardzo szybkie reagowanie w przypadku łamania przez użytkownika postanowień regulaminu oraz pozwala ograniczyć niesłuszne przerywanie transmisji.

Reset licznika następuje co 1000 klatek, dlatego w przypadku kamerek nagrywających obraz w 30 klatkach na sekundę, do przerywania transmisji wystarczy, że dany obiekt był widoczny

przez ok. 7 sekund w trwającym przez ok. 33 sekundy pomiarze. Resetowanie licznika prawie całkowicie eliminuje problem przerywania transmisji na podstawie losowych błędów klasyfikacji obiektów, co sprawia, że rozwiązanie może być z powodzeniem stosowane na portalu.



*Zdjęcie 42 - rozpoznawanie obiektów w czasie rzeczywistym*



*Zdjęcie 43 - przerywanie transmisji*

## 5.6. AUTOMATYCZNE UKRYWANIE ELEMENTÓW ODWRACAJĄCYCH UWAGĘ

Ukrywanie elementów rozpraszających słuchaczy jest rozwiązaniem odpowiadającym na najbardziej bieżące potrzeby. Globalna pandemia zmusiła większość przedsiębiorstw do przeniesienia swojej działalności do sieci.

Konieczność pozostania w domu sprawiła, że zdalne rozwiązywanie problemów i spotkania online stały się koniecznością. Podczas, gdy firmy dostosowały swoją pracę do nowej codzienności, mieszkania pracowników nie stały się nagle specjalistycznymi biurkami. Szybko okazało się, że wraz z pracownikami na spotkaniach firmowych pojawiają się także ich współlokatorzy czy zwierzęcy pupile. Z pewnością każdy jest w stanie wyobrazić sobie sytuację, w której na biurko uczestnika telekonferencji wskakuje kot i kradnie całą uwagę zgromadzonych przed komputerami osób. Z pozoru niewinna sytuacja powoduje rozkojarzenie uczestników spotkania, a w efekcie prowadzi do niepotrzebnego wydłużania spotkań czy pominięcia ważnych kwestii.

Pomysł na rozwiązanie tego problemu jest bardzo prosty. Podobnie jak w przypadku automatycznego wyłączania transmisji, podstawą tego rozwiązania jest przedstawione wcześniej rozpoznawanie obiektów w czasie rzeczywistym. W tym przypadku, obraz z kamery również przetwarzany jest przez sieć neuronową, która następnie określa jaki obiekt znajduje się na zdjęciu. Aby ukrycie obiektu było skuteczne, spełnione muszą zostać dwa założenia. Przede wszystkim odwracający uwagę obiekt musi zostać ukryty możliwie jak najszybciej. Niemalże równie ważne jest ukrycie obiektu w odpowiedni sposób. Wyświetlenie czerwonego sportowego samochodu w celu ukrycia małego psa znajdującego się na końcu pokoju da całkowicie odwrotny do zamierzonego efekt.

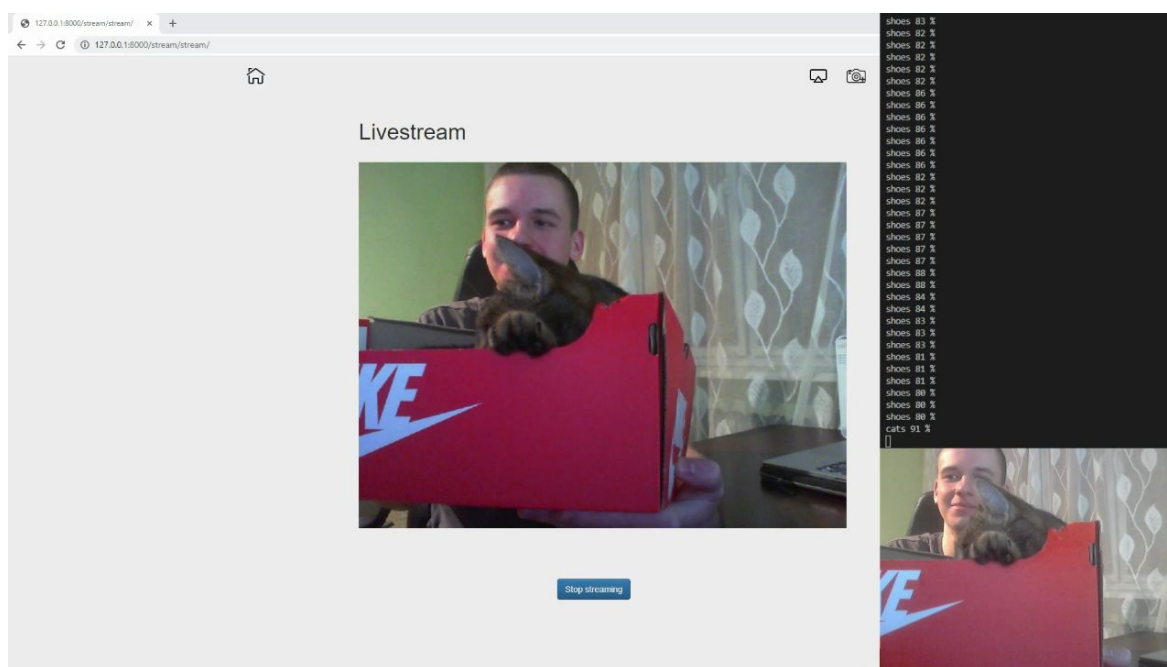
W celu zaadresowania powyższych założeń, do metody klasyfikującej obiekty dodane zostały odpowiednie modyfikacje. Przede wszystkim, algorytm podejmuje akcje natychmiast po wykryciu obiektu, nie czeka aby mieć pewność poprawnej klasyfikacji. Uzyskanie płynnego dla ludzkiego oka obrazu jest możliwe przy częstotliwości ok 30 klatek na sekundę, algorytm może ukryć zdjęcie na podstawie tylko jednej klatki pobranej z kamery. Oznacza to, że bardzo często obiekt zostanie ukryty zanim jeszcze będzie możliwe jego zauważenie na obrazie z kamery internetowej.

Po rozwiązaniu problemu z szybkością działania algorytmu, przyszedł czas na zaimplementowanie odpowiedniego filtra. Ponieważ najważniejszym zadaniem algorytmu jest utrzymanie uwagi uczestników spotkania, ukrycie obiektu powinno odbyć się szybko i

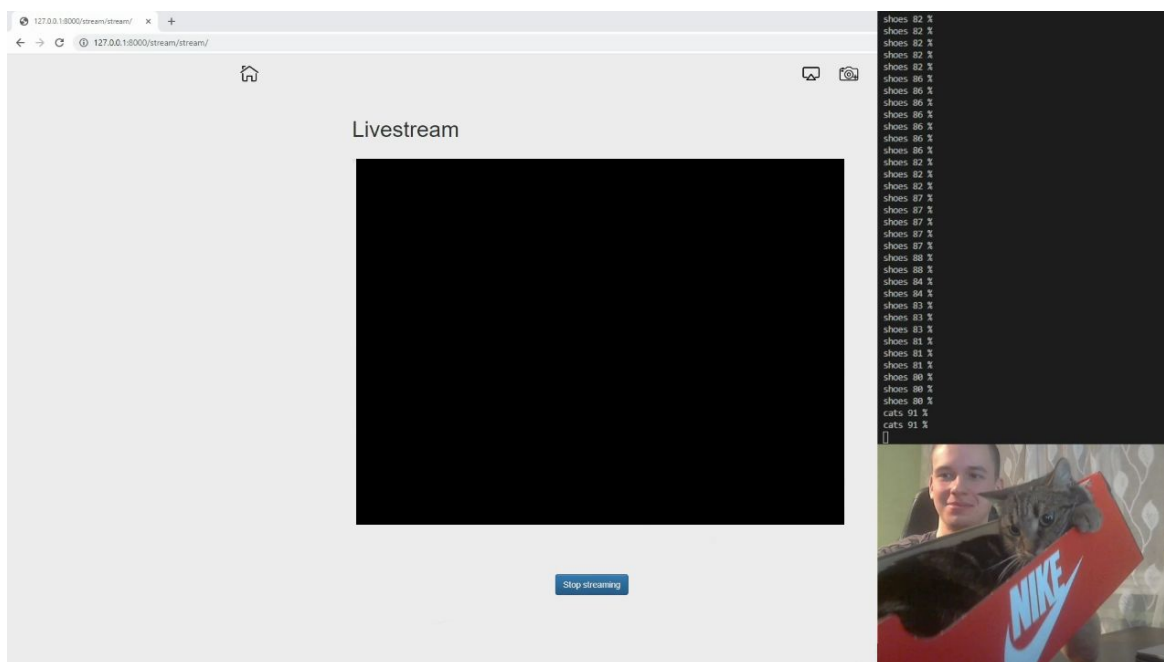
niezauważalnie. Dlatego zrezygnowano z całkowitego wyłączenia kamery czy akcji podejmowanych bezpośrednio w oknie telekonferencji - bardzo często zmieniają one układ strony, co również odwraca uwagę.

Zamiast tego, ukrycie obiektów odbywa się poprzez zakrycie obrazu z kamery. Aby przyciągać jak najmniejszą uwagę, obraz z kamery zastąpiony zostaje jednokolorowym zdjęciem. Rozwiązanie jest skuteczne, ponieważ jednokolorowy ciemny obraz nie kontrastuje z wideo pozostałych uczestników spotkania.

Ze względu na złożoność systemów do prowadzenia telekonferencji, w procesie tworzenia portalu został on pominięty. Na potrzeby prezentacji działania algorytmu wykorzystana została forma do prowadzenia transmisji na żywo.



*Zdjęcie 44 - rozpoznawanie obiektów w czasie rzeczywistym*



*Zdjęcie 45 - automatyczne ukrywanie obiektów*

Kiedy na zdjęciu nr 45 pojawia się kot, klatki przesyłane przez kamerę natychmiast zostają zastąpione czarnym obrazem. Jak widać na dodatkowej kamerze kot jest cały czas widoczny w kamerze głównej, ponieważ sieć nie przestaje działać, w momencie zniknięcia kota z widoku transmisja obrazu zostanie wznowiona.

## 6. Kierunek rozwoju

### 6.1. MOŻLIWE ROZWIĄZANIA

Zastosowane na stronie rozwiązania nie wyczerpują w pełni możliwości aplikacji. Poszczególne metody przygotowane zostały w sposób umożliwiający ich rozbudowę. Pierwszym elementem, który powinien zostać rozwinięty jest sama sieć neuronowa. Zwiększenie ilości rozpoznawalnych obiektów jest kluczowe podczas rozbudowy portalu. Rozpoznawane przez sieć osiem kategorii obiektów nie jest imponujące z perspektywy portalu, na którym znajdują się miliony zdjęć. Po rozbudowaniu sieci dobrym kierunkiem rozwoju będzie ulepszenie lub dodanie funkcjonalności do przygotowanych rozwiązań.

Jednym z nich może być np. rozbudowanie systemu etykiet zdjęć. Sklasyfikowane obiekty, mogą należeć do kilku przygotowanych grup naraz, w ten sposób np. zdjęcie zabawek możemy dodatkowo oznaczyć jako treść odpowiednią dla dzieci. Przygotowanie dodatkowych etykiet znacznie poprawi przejrzystość strony oraz przyczyni się do większego zaangażowania użytkowników. Ponadto, etykiety mogą zostać wykorzystane do odpowiedniego pozycjonowania zdjęć jako płatne reklamy, czemu pomoże także automatyczne ukrywanie reklam nieopłaconych (paragraf 5.3).

Kolejnym obszarem, który można rozbudować jest moduł ukrywający obraz podczas telekonferencji. Poprawione mogą zostać skuteczność wykrywania obiektów czy też sposób ukrywania obrazu z kamery. Uzyskanie maksymalnej skuteczności może niestety wiązać się z koniecznością przygotowania odrębnej aplikacji, która od zera zbudowana zostanie do współpracy ze wspomnianym rozwiązaniem.

Poza rozbudowaniem istniejących na portalu rozwiązań możliwe jest także dodanie nowych metod. W pracy bowiem nie został poruszony temat naruszania regulaminu poprzez komentarze użytkowników. Jeżeli portal posiadałby możliwość umieszczania komentarzy, do strony dodana powinna zostać także metoda filtrująca te z nich, które łamią regulamin. Zastosowanie shadow banning'u do komentarzy łamiących regulamin prawdopodobnie okazałoby się bardzo skuteczne a do rozpoznania niechcianych komentarzy wystarczyłoby rozbudowany słownik wyrazów zabronionych. Chcąc pójść o krok dalej, przygotowana może zostać sieć rozpoznająca mowę, mogłaby ona zostać przeznaczona do nadzorowania transmisji na żywo i działać podobnie do sieci rozpoznającej obiekty.

Jak widać możliwości wręcz się nie kończą, metody mogą być dostosowywane do indywidualnych potrzeb portali.

## **6.2. SKALOWALNOŚĆ APLIKACJI**

Niestety głównym problemem aplikacji jest jej skalowalność. Przygotowane rozwiązania wymagają dużych zasobów mocy obliczeniowej. Podczas gdy implementacja klasyfikacji udostępnionych obrazów jest całkiem możliwa dla średniego rozmiaru portalu, skalowalność metody rozpoznającej obiekty w czasie rzeczywistym jest bardzo słaba. Aby zapewnić stabilne działanie wielu instancji metody jednocześnie potrzebne byłyby ogromne zasoby sprzętowe. Dobrym rozwiązaniem tego problemu mogłoby być przeniesienie oraz

wykonanie obliczeń nie na serwerze aplikacji a korzystając z podzespołów użytkownika. Jest to jednak rozwiązanie wymagające znacznie lepszej optymalizacji, do zapewnienia płynnego działania sieci neuronowej w aktualnym stanie niezbędna jest karta graficzna NVIDIA, co uniemożliwia korzystanie z sieci na wielu komputerach. Alternatywą jest uproszczenie działania algorytmów oraz rozbudowanie mocy obliczeniowej serwera, co niestety jest opcją bardzo kosztowną.

Na szczęście wraz z rozwojem podzespołów oraz algorytmów głębokiego uczenia ograniczenia sprzętowe będą coraz mniejsze. W końcu komputer odpowiedzialny za doprowadzenie Apollo 11 na księżyc w 1969 roku, był kilkadziesiąt tysięcy razy słabszy niż współczesne smartfony, mało tego w okresie pisania pracy, najnowsze ładowarki dodawane do tych właśnie smartfonów mają większą moc obliczeniową od wspomnianego **Apollo Guidance Computer**. [23]

Biorąc pod uwagę postęp technologiczny na przestrzeni ostatnich lat, być może za parę lat porównanie najnowszej ładowarki do komputera Apollo będzie obelgą dla ówczesnych ładowarek, tak jak teraz ma to miejsce dla dostępnych na rynku smartfonów.

Ta myśl pozwala utrzymać pozytywne nastawienie co do skalowalności przygotowanych rozwiązań. Być może w niedalekiej przyszłości podobne rozwiązania będą szeroko stosowane na portalach internetowych.

## 7. Wyniki i podsumowanie

Celem pracy było przygotowanie oraz nauczanie sieci neuronowej rozpoznającej obiekty udostępniane w internecie, a następnie wykorzystanie tej sieci do stworzenia i implementacji na portalu społecznościowym inteligentnych rozwiązań. Praca przeprowadza czytelnika przez proces tworzenia aplikacji krok po kroku.

Proces uczenia sieci przebiegł bardzo dobrze, przygotowane sieci mają skuteczność kolejno 89% oraz 97%, oznacza to że obie sieci mogą być z powodzeniem stosowane w różnych aplikacjach.

Przygotowanie portalu społecznościowego również można uznać za sukces, portal umożliwia tworzenie konta, dodawanie zdjęć czy prowadzenie transmisji na żywo.



Rozwiązania zaimplementowane na portalu okazały się skutecznie wykorzystywać przygotowaną sieć neuronową. Zabezpieczenia portalu działają bardzo sprawnie, a kary nakładane na użytkowników mają wysoką skuteczność.

Niestety skalowalność i implementacja rozwiązań na większych portalach jest aktualnie bardzo ciężkim zadaniem ze względu na wysoki koszt podzespołów.

Praca nie wyczerpuje możliwych rozwiązań i ich implementacji, przedstawione metody mogą zostać modyfikowane i ulepszone wraz z postępem technologicznym. Implementacja podobnych rozwiązań już ma miejsce na największych portalach, na razie niestety wymaga to ogromnych zasobów sprzętowych.

## 8. Literatura

[1] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, Ruslan R. Salakhutdinov. "Improving neural networks by preventing co-adaptation of feature detectors" (2012)

[2] "How to Control the Stability of Training Neural Networks With the Batch Size"

Dostępny:

<https://machinelearningmastery.com/how-to-control-the-speed-and-stability-of-training-neural-networks-with-gradient-descent-batch-size/> Odwiedzona 10.10.2020

[3] "Google Images Download documentation" Dostępny:

<https://google-images-download.readthedocs.io/en/latest/index.html> Odwiedzona 10.10.2020

[4] "About Feature Scaling and Normalization"

[https://sebastianraschka.com/Articles/2014\\_about\\_feature\\_scaling.html](https://sebastianraschka.com/Articles/2014_about_feature_scaling.html) Odwiedzona 10.10.2020

[5] "Normalization vs Standardization - Quantitative analysis"

<https://towardsdatascience.com/normalization-vs-standardization-quantitative-analysis-a91e8a79cebf> Odwiedzona 11.10.2020

[6] "Gap CV" <https://pypi.org/project/gapcv/> Odwiedzona 1.11.2020

[7] "The Ultimate Guide to Convolutional Neural Networks (CNN)"

<https://www.superdatascience.com/blogs/the-ultimate-guide-to-convolutional-neural-networks-cnn> Odwiedzona 2.11.2020

- [8] “Cat and Dog dataset to train a DL model”  
<https://www.kaggle.com/tongpython/cat-and-dog> Odwiedzona 10.11.2020
- [9] “Food Images (Food-101)” <https://www.kaggle.com/kmader/food41> Odwiedzona 10.11.2020
- [10] “Learning Rate Tuning in Deep Learning”  
<https://mlexplained.com/2018/01/29/learning-rate-tuning-in-deep-learning-a-practical-guide/>  
Odwiedzona 10.11.2020
- [11] “Why normalize images by subtracting dataset’s image mean, instead of the current image mean in deep learning?”  
<https://stats.stackexchange.com/questions/211436/why-normalize-images-by-subtracting-data-sets-image-mean-instead-of-the-current> Odwiedzona 10.11.2020
- [12] “Who Owns Which Car Brands”  
<https://www.consumerreports.org/cars-who-owns-which-car-brands/> Odwiedzona 10.11.2020
- [13] “America’s favourite guns”  
<https://www.cbsnews.com/pictures/most-popular-guns-in-america/> Odwiedzona 10.11.2020
- [14] “Image normalization in PyTorch”  
<https://forums.fast.ai/t/image-normalization-in-pytorch/7534/7> Odwiedzona 10.11.2020
- [15] “ptrblck/DiscoGAN”  
[https://github.com/ptrblck/DiscoGAN/blob/master/discogan/run\\_inference.py](https://github.com/ptrblck/DiscoGAN/blob/master/discogan/run_inference.py) Odwiedzona 12.11.2020
- [16] “Deep learning basics - weight decay”  
<https://medium.com/analytics-vidhya/deep-learning-basics-weight-decay-3c68eb4344e9>  
Odwiedzona 12.11.2020
- [17] “Stochastic Gradient Descent with momentum”  
<https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d>  
Odwiedzona 15.11.2020
- [18] “Understanding regularization with PyTorch”  
<https://medium.com/analytics-vidhya/understanding-regularization-with-pytorch-26a838d94058> Odwiedzona 15.11.2020
- [19] “How to Use Weight Decay to Reduce Overfitting of Neural Network in Keras”  
<https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/> Odwiedzona 16.11.2020

[20] “Building your own Object Recognition in PyTorch - A Guide to Implement HardNet in PyTorch”

<https://analyticsindiamag.com/building-your-own-object-recognition-in-pytorch-a-guide-to-implement-hardnet-in-pytorch/> Odwiedzona 16.11.2020

[21] “Designing Your Neural Networks”

<https://towardsdatascience.com/designing-your-neural-networks-a5e4617027ed> Odwiedzona 20.11.2020

[22] “A Beginner’s Guide to Data Engineering - Part I”

<https://medium.com/@rchang/a-beginners-guide-to-data-engineering-part-i-4227c5c457d7> Odwiedzona 10.11.2020

[23] “Apollo 11 Guidance Computer (AGC) vs USB-C Chargers”

<https://forrestheller.com/Apollo-11-Computer-vs-USB-C-chargers.html> Odwiedzona 04.01.2021

[24] “An overview of gradient descent optimization algorithms”

<https://ruder.io/optimizing-gradient-descent/> Odwiedzona 01.12.2020

[25] “Frot-Coutaz: YouTube’s priority is tackling harmful content”

<https://www.ibc.org/trends/frot-coutaz-youtubes-priority-is-tackling-harmful-content/4208.article> Odwiedzona 12.01.2021

[26] “Illustrated: 10 CNN Architectures”

<https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d> Odwiedzona 20.10.2020

[27] “PYTORCH DOCUMENTATION” <https://pytorch.org/docs/stable/index.html>

Odwiedzona 15.10.2020

[28] “Image hashing with OpenCV and Python”

<https://www.pyimagesearch.com/2017/11/27/image-hashing-opencv-python/> Odwiedzona 10.11.2020

[29] “API documentation” <https://www.flickr.com/services/api/> Odwiedzona 10.10.2020