POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

KIERUNEK: Informatyka

SPECJALNOŚĆ: Grafika i Systemy Multimedialne

PRACA DYPLOMOWA MAGISTERSKA

Detekcja manipulacji zawartości zdjęć przy pomocy metod uczenia głębokiego

Image manipulation detection using deep learning techniques

AUTOR:
Jarosław Ciołek-Żelechowski

PROMOTOR:
dr inż. Paweł Ksieniewicz

OCENA PRACY:

WROCŁAW 2020

SPIS TREŚCI

1. WSTĘP

2. UCZENIE MASZYNOWE

Alan Turing w swojej pracy z 1950 roku *Computing Machinery and Inteligence*[28] zdefiniował pojęcie *obiekcji lady Lovelace*. Odnosiło się ono do krótkiej notatki[18] jaką lady Ada Lovelace poczyniła w 1843 roku podczas tłumaczenia na język angielski artykułu Luigi Menabrea[20], który to był streszczeniem wykładu Charlesa Babbage'a wygłoszonego w Turynie w 1841 roku. Wykład dotyczył projektu maszyny analitycznej której zadaniem było zautomatyzowanie niektórych obliczeń związanych z analizą matematyczną. Pojęcie to brzmi następująco:

Maszyna analityczna nie ma na celu zapoczątkowania czegokolwiek. Może wykonywać operacje, które możemy kazać jej przeprowadzać... Jej celem jest zwiększanie dostępności tego co umiemy już wykonać[28]

Turing, przywołał to pojęcie, zastanawiając się nad tym, czy komputery mogą się uczyć, tworzyć nowe rzeczy. Jak piszę on dalej w swoim artykule[28]: problem jest natury programistycznej i wymaga on stworzenia zupełnie innego, jak na tamte czasy, środowiska i sposobu pojmowania nauczania jako takiego. Wierzył jednak, że jest to możliwe.

Termin *Uczenie Maszynowe* po raz pierwszy pojawił się w 1959 roku w pracy naukowej autorstwa Arthura Samuela:

Uczenie maszynowe to dziedzina nauki dająca komputerom możliwość uczenia się bez konieczności ich jawnego programowania.[25]

Praca ta dotyczyła maszyny, która przez ok. 8 godzin *uczyła się* gry w warcaby znając jedynie jej zasady, posiadając pewnego rodzaju funkcję celu(zbijanie pionów przeciwnika), oraz macierz losowych liczb, której to zmiany i przekształcenia miały na celu reprezentować inne podejścia(nową wiedzę). Sprawdzianem sukcesu maszyny, było pokonanie jej twórcy.

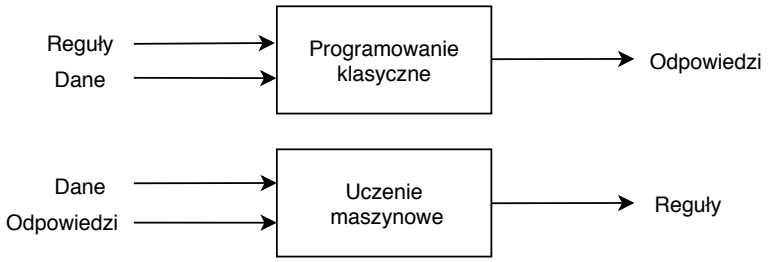
Bardziej techniczną definicję podał w 1997 roku Tom Mitchel, w rozdziale otwierającym swoją książki pt. *Machine Learning*:

Mówimy, że program komputerowy uczy się na podstawie doświadczenia E w odniesieniu do jakiegoś zadania T i pewnej miary wydajności P, jeśli jego wydajność (mierzona przez P) wobec zadania T wzrasta wraz z nabywaniem doświadczenia E.[21]

Tym samym uczenie maszynowe zostało sprowadzone do problemu, w którym to posiadamy trzy elementy T, P i E. W dalszej części Tom Mitchel podaje przykład powyżej definicji zrealizowanej dla gry w warcaby(wyraźny ukłon w stronę pracy Arthura Samuela[25]):
— Zadanie T: gra w warcaby,
— Miara Wydajności P: procent gier wygranych na oponentów,
— Doświadczenie E: granie partii przeciwko samemu sobie.

Tym samym można rozumieć uczenie maszynowe jako nowy paradygmat programowania. W programowaniu klasycznym, programista definiuje reguły według których program przetwarza dane wejściowe, generujący tym samym dane wyjściowe - odpowiedź pracy programu(patrz

rysunek 2.1). W przypadku uczenia maszynowego mamy sytuację w której programista wprowadza dane oraz odpowiedzi i oczekuje uzyskać od programu zestawu reguł, według których dane odpowiedzi zostały przypisane do konkretnych próbek. Reguły te, mają posłużyć w dalszej części do przetwarzania nowych danych.



Rys. 2.1: Uczenie maszynowe: nowy model programowania

Podsumowując, system uczenia maszynowego jest trenowany, a nie programowany w sposób jawny. Celem programisty jest przedstawienie mu odpowiedniej dużej ilości przykładów wyników, tak by sam system określił ich statystyczną strukturę, co w dalszej części pozwoli na ustalenie reguł umożliwiających automatyzację całego procesu. Ważne, by podkreślić tutaj znaczenie danych wejściowych, które to często określa się terminem *Big Data*[26].

2.1. BIG DATA

Popularyzację tego terminu przypisuję się do wykładu autorstwa Johna Mashey'a[19] jaki wygłosił w 1998 roku. Zauważył on, że wraz ze spadkiem cen nośników do przechowywania i gromadzenia danych, ilość zbieranych przez ludzkość informacji wzrasta z każdym rokiem. Co więcej, ilość tych danych sprawia, że ich analiza przestała być możliwa do realizacja w sposób inny niż automatyczny. Za kryterium, czy dany zbiór informacji można określić jako Big Data, często przywołuje się te podane przez Douglasa Laney'ego[15] i określane mianem *3V*, które to rozwija się jako rozmiar(ang. *volume*), różnorodność(ang. *variety*) i prędkość (ang. *velocity*). Oznaczają one kolejno:

— **rozmiar** - dane są zbyt duże by mieściły się na standardowych dyskach twardych,
— **różnorodność** - dane pochodzą z różnych źródeł, są niejednorodne i słabo ustrukturyzowane,
— **prędkość** - tempo napływu nowych danych jest znaczące, co w rezultacie utrudnia ich analizowanie.

Jednym z kluczowych zjawisk przyczyniającym się do procesu nagłego przyrostu ilości i źródeł danych jest Internet przedmiotów (ang. *Internet of Things*[12]). Według tej koncepcji, różnego typu urządzenia osadzone w urządzeniach codziennego użytku(np. odkurzacze, żarówki, instalacje grzewcze), czy też w maszynach przemysłowych, zbierają dane o otoczeniu, czy samym procesie w którym uczestniczą, a ponadto mają możliwość komunikacji pomiędzy sobą, ale również z jednostką centralną jeśli taka występuje. Wprowadza to nowy wymiar w możliwościach automatyzacji procesów i tworzy nową przestrzeń do tworzenia się złożonych, inteligentnych systemów.

2.2. RODZAJE SYSTEMÓW UCZENIA MASZYNOWEGO

Istnieje wiele metod podziału uczenia maszynowego na kategorię. Jedną z najpopularniejszych jest podział ze względu na sposób uczenia się oraz zadanie do wykonania[23]:
— Uczenie nadzorowane:
  — klasyfikacja,
  — regresja.
— Uczenie nienadzorowane:
  — klasteryzacja,
  — redukcja wymiarowości,
  — uczenie przy użyciu reguł asocjacyjnych.
— Uczenie ze wzmocnieniem.

W uczeniu z nadzorem(ang. *supervised learning*), którym zajmę się w poniżej pracy, dane trenujące przekazane algorytmowi zawierają dołączone do nich etykiety, czyli rozwiązania problemu. Celem algorytmu jest stworzenie funkcji(nazywanej też hipotezą[4]), która będzie maksymalizować swoją skuteczność względem zadanych kryteriów.

Na zbiór uczący Z_u składa się zbiór n wektorów, gdzie każdy z nich opisuje pojedynczy obiekt(patrz wzór 2.1):

$$Z_u = \{(x_1, y_1), ..., (x_n, y_n)\} \tag{2.1}$$

I tak w powyższym równaniu2.1) i-ty wektor oznaczony byłby jako x_i i odpowiadałaby mu etykieta oznaczona jako y_i. W zależności od typu danych przechowywanych pod etykietą będziemy mieć do czynienia z zadaniem klasyfikacji(etykieta należy do skończonego i przeliczalnego zbioru) lub regresji(wartości przyjmowane przez etykietę należą do przestrzeni ciągłej). Ważne żeby dodać że na wektor x_i może składać się m-liczb, gdzie każdą z tych liczb będziemy nazywać atrybutem lub cechą danego wektora(patrz wzór 2.2):

$$x_i = \{x_{1,m}, ..., x_{i,m}\} \tag{2.2}$$

Etykietą y_i, w tym rozumowaniu, oznaczamy prawdziwą wartość funkcji, którą to chcemy by nasz algorytm odwzorował. Algorytm ten, nazywamy również modelem i opisujemy go jako następującą funkcję f(patrz wzór 2.3):

$$f(x) : x \in X \mapsto y \in Y \tag{2.3}$$

Jak widać zadaniem powyższej funkcji jest przyporządkowanie wektorom wejściowym etykiet. Jej skuteczność jest mierzona przy użyciu wybranej przez nas metryki na zbiorze testowym. Zbiór testowy musi posiadać tą samą strukturę co zbiór trenujący. Można traktować metrykę M jako funkcję wyższego rzędu, której pierwszym argumentem jest sam model f, a drugim zbiór testowy Z_t. Dziedziną metryki jest zazwyczaj podzbiór liczb rzeczywistych z zakresu od 0 do 1(patrz wzór 2.4)

$$M(f, Z_t) : (f, Z_t) \mapsto m \in [0, 1] \tag{2.4}$$

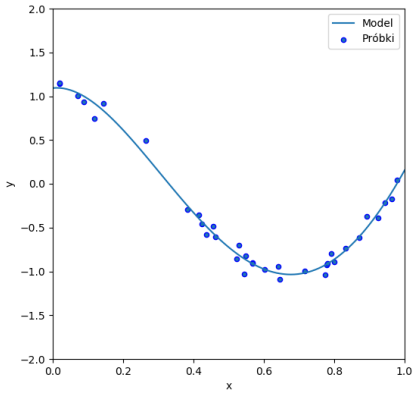
Przykładem metryk stosowanych w uczeniu maszynowym są między innymi dokładność, czułość, precyzja, a także miara *F* (ang. *F-score*), które omówione zostały szczegółowo w dalszej części rozdziału.

2.3. PROBLEM PRZEUCZANIA, NIEDOUCZANIA

Wykorzystanie osobnego zbioru do badania jakości modelu w uczeniu nadzorowanym związane jest bezpośrednio z takimi problemami jak nadmierne dopasowanie, przeuczenia(ang. *overfitting*) oraz niedouczenie (ang. *underfitting*)[9]. Zadaniem, które dobrze nadaje się do graficznej ilustracji powyższych problemów jest regresja liniowa. Jak już zostało opisane powyżej w problemie tym, etykiety przykładów są liczbami należącymi do przestrzeni ciągłej, a zadaniem algorytmu jest wyznaczenie funkcji f(patrz wzór 2.5):

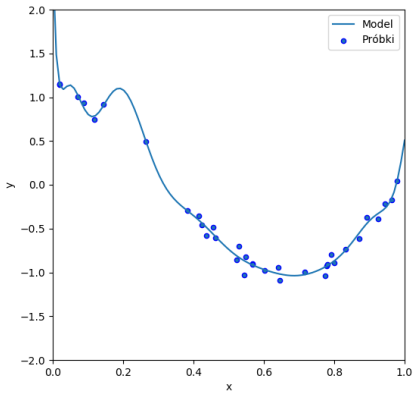
$$f(x) : x \mapsto y \in \mathbb{R} \tag{2.5}$$

Dodatkowo dla lepszej interpretacji graficznej każdy wektor w przestrzeni x będzie posiadał tylko jedną cechę. Przykład prawidłowego dopasowania wygląda następująco(patrz rysunek 2.2):



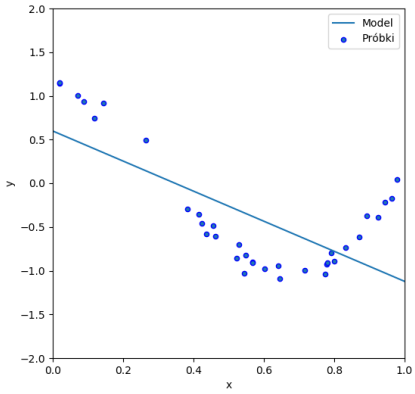
Rys. 2.2: Przykład prawidłowego dopasowania

Problem nadmiernego dopasowania cechuje się tym, że taki model zbyt mocno generuje charakterystykę dla danych trenujących, zawierając w niej również występujące tam szumy(patrz rysunek 2.3):



Rys. 2.3: Przykład nadmiernego dopasowania

Odwrotnym problem jest problem niedouczenia. Występuję on na przykład wtedy, kiedy zastosujemy zbyt prosty model w stosunku do poziomu skomplikowania zbioru danych. Wynik takiej operacji widoczny jest na rysunku 2.4



Rys. 2.4: Przykład niedostatecznego dopasowania

2.4. ZADANIE KLASYFIKACJI BINARNEJ

W niniejszej pracy będę zajmował się zadaniem należącym do problemów klasyfikacji binarnej, czyli takiej w której przestrzeń etykiet ogranicza się do dwóch elementów(patrz wzór 2.6):

$$y = \{0, 1\} \tag{2.6}$$

Klasy 0 i 1 w powyższym wzorze2.6 są przykładowe i bardziej niż ich wartość interesuje Nas liczność zbioru y. Tym samym mając zdefiniowaną przestrzeń możliwych wartości modelu można zdefiniować szereg miar, które są wykorzystywane do oceny jego wyników. Podstawowym elementem zadania ewaluacji modelu w zadaniu klasyfikacji binarnej jest macierz, tablica błędów widoczna w tabeli 2.1. Do jej skonstruowania potrzebujemy oczywiście przetestować działanie naszego modelu na zbiorze testowym Z_t. Poszczególnym reprezentantom tego zbioru, przypisywane są pozytywne lub negatywne etykiety, teraz w zależności od tego czy dany element zbioru x był faktycznie pozytywny czy negatywny można go wpisać w macierz błędów 2.1.

	Klasyfikacja pozytywna	**Klasyfikacja negatywna**
Stan pozytywny	Prawdziwie dodatnia (ang. *true positive*, TP)	Fałszywie ujemna (ang. *false negative*, FN)
Stan negatywny	Fałszywie dodatnia (ang. *false positive*, FP)	Prawdziwie ujemna (ang. *true negative*, TN)

Tabela 2.1: Tablica pomyłek, możliwe wyniki klasyfikacji binarnej

2.5. EWALUACJA MODELU - MIARY JAKOŚCI

Korzystając z opisanej powyżej macierzy błędów 2.1 w łatwy sposób można przedstawić definicję szeregu miar do oceny modelu, z których to skorzystałem w niniejszej pracy:
— **Dokładność** - procent poprawnych klasyfikacji, opisanych wzorem 2.7:

$$\mathbf{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.7}$$

— **Czułość** - stosunek prawidłowych wyników pozytywnych do sumy prawidłowych wyników pozytywnych oraz błędnych wyników negatywnych, który można rozumieć jako zdolność modelu do poprawnego etykietowania(patrz wzór 2.8),

$$\mathbf{recall} = \frac{TP}{TP + FN} \tag{2.8}$$

— **Precyzja** - stosunek prawidłowych wyników pozytywnych do sumy prawidłowych wyników pozytywnych oraz błędnych wyników pozytywnych, który można rozumieć jako zdolność modelu do niepoprawnej klasyfikacji próbek negatywnych jako pozytywne(patrz wzór 2.9),

$$\mathbf{precision} = \frac{TP}{TP + FP} \tag{2.9}$$

— **miara *F*** - będąca średnią ważoną z czułości i precyzji(patrz wzór 2.10).

$$\mathbf{F1} = \frac{2 * precision * recall}{precision + recall} \tag{2.10}$$

2.6. SPLOTOWE SIECI NEURONOWE

Splotowe, inaczej konwolucyjne, sieci neuronowe (ang. *CNN - convolutional neural networks*) stanowią wynik badań nad korą wzrokową i od 1980 roku są używane w zadaniach rozpoznawania obrazów[9]. Podstawową różnicą w stosunku do sieci neuronowych jest stosowanie wielowymiarowej operacji splotu, realizowanej za pomocą szeregu filtrów, których to parametry dobierane są podczas trenowania sieci. Pozwala to siecią konwolucyjnym na nie przetwarzanie obrazów piksel po pikselu, a raczej poprzez zauważanie ogólnych cech obrazu i budowaniu z nich nowych struktur.

W analizie matematycznej operacją splotu(reprezentowana jako $*$) dwóch funkcji f i g, jest trzecia funkcja s, patrz wzór 2.11:

$$\begin{aligned} s(t) = (f * g)(t) &= \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau = \\ &= \int_{-\infty}^{\infty} f(t - \tau) g(\tau) d\tau \end{aligned} \tag{2.11}$$

, a w przypadku operacji dyskretnych, patrz wzór 2.12:

$$s(t) = (f * g)(t) = \sum_{\tau=-\infty}^{\infty} f(\tau)(g - \tau) \tag{2.12}$$

W literaturze opisującej sieci neuronowych przyjęło się określać $f(\tau)$ jako wejście, $g(t-\tau)$ jako jądro lub filtr(z ang. *kernel*, *filter*), a wynik samej operacji jako mapę atrybutów lub aktywacji (z ang. *feature map*, *activation map*)[22]. Zadanie dla typowego problemu klasyfikacji obrazów, operującego na dwuwymiarowym obrazie wejściowym M przy pomocy filtra K z założeniem wykorzystania wielowymiarowego splotu $S(i, j)$, można sformułować następująco 2.13:

$$\begin{aligned} S(i, j) = (M * K)(i, j) &= \sum_k \sum_l M(k, l) K(i - k, j - l) \\ = (K * M)(i, j) &= \sum_k \sum_l M(i - k, j - l) K(k, l) \end{aligned} \tag{2.13}$$

Budowa sieci splotowych, a dokładnie fakt, że nie wymagały one połączeń pomiędzy wszystkimi neuronami w każdej z warstw oraz fakt, że zastosowanie operacji splotu, pozwala na zmniejszenie wymiarów zdjęcia pozwoliło budować sieci o większej ilości warstw ukrytych [23]. Za pierwszą pracę z stosującą uczenie głębokie(funkcje splotu, wsteczną propagację) uważa się tą z 1989 roku, której autorzy stworzyli model rozpoznający cyfry zawarte w kodach pocztowych[16]. Sam model okazał się sukcesem, problemem był jednak czas uczenia - $\sim$3 dni. Dodatkowo w tamtym okresie na popularności zyskiwały metody jądrowe(maszyny wektorów nośnych) oraz wszelkiego rodzaju algorytmy oparte o drzewa decyzyjne [9]. Warto tutaj również dodać, że zbiory danych konstruowane w latach 80 i 90 ubiegłego wieku, były zazwyczaj małe w porównaniu do dzisiejszych standardów, co premiowało algorytmy oparte o manualną ekstrakcję cech [3].

Krokiem w kierunku popularyzacji podejścia splotowego i samej idei uczenia głębokiego była praca Hintona i Salakhutdinova z 2006 roku, w której to autorzy zaproponowali sposób budujący coraz głębszy model składający się z ograniczonych maszyn Boltzmanna [11]. Pokazali oni że czas od pracy LeCuna z 1989 roku [16] zapewnił potrzebny postęp sprzętowy co przedstawiło dotychczasowy problem długiego czasu trenowania sieci jako problem optymalizacyjny. Zgodnie z danymi podawanymi przez F. Cholleta [3] szybkość dostępnych dla standardowych użytkowników procesorów wzrosła pomiędzy latami 1990 a 2010 o około 5000 razy. Dodatkowo stworzenie i udostępnienie takich zbiorów danych jak ImageNet, zawierających powyżej 14 milionów tagowanych zdjęć wysokiej rozdzielczości i zbudowanie wokół niego konkursu ImageNet Large Scale Visual Recognition Challenge [24] sprawiło że praktycznie każdy jest w stanie przeprowadzić proste eksperymenty w oparciu o uczenie głębokie.

Dalsza demokratyzacja uczenia głębokiego w postaci serwisu *kaggle.com*, który to hostuje nie tylko zbiory danych, ale również konkursy na najlepsze modele, w połączeniu z łatwością w korzystaniu z technologii CUDA(z ang. *Compute Unified Device Architecture*) już nie tylko w języku C++, ale również Python - sprawia że odpalanie modelów sieci głębokich jest proste i równe szybkie co modeli opartych o klasyczne uczenie maszynowe[3].

3. ANALIZA ISTNIEJĄCYCH METOD W OBRĘBIE DZIEDZINY

Badanie manipulacji zdjęć jest dość obszernym dziedziną w której to, co więcej, można wyróżnić szereg podejść i prób zrozumienia problemu. Stanem wiedzy o podejściach statystycznych, lub takich opartych o metadane zdjęcia jest książka *Photo Forensics*, autorstwa Hany Farida [7]. Podejścia oparte o analizę poszczególnych zdjęć bez używania uczenia maszynowego w większości opierają się o metaznaczniki zawarte w strukturze plików *.JPEG. Metaznaczniki te są zarówno wspierane po stronie aparatów cyfrowych jak i oprogramowania do edycji zdjęć. Dla zdjęć cyfrowych generowane są dane dotyczące modelu aparatu, jego ustawień(takich jak czas naświetlania, wartość przesłony czy czułość matrycy w ISO), daty wykonania zdjęcia, czy nawet współrzędne GPS miejsca w którym zdjęcie zostało zrobione. Dla programów do obróbki zdjęć generowane są informację o nazwie użytego programu i w zależności od rodzaju oprogramowania - część danych utworzonych przez aparat zostaje *wymazana*. Takie podejście sprawia, że do sprawdzenia czy dane zdjęcie zostało przerobione, wystarczy przeczytać jego metadane [7].

Innym podejściem, choć ciągle bazującym na metaznacznikach jest zbudowanie *odcisku palca* aparatu o zadanych ustawieniach i porównywaniu spreparowanych zdjęć do obrazów rozpatrywanych jako potencjalnie przerobione [27]. Problemów z zaproponowanym przez A. Swaminathan, M. Wu oraz K. J. Ray Liu jest kilka i sami wspominają o nich w swojej pracy [27]: całość wymaga zbudowania modelu poszczególnych aparatów, co znacząco zimniejsza możliwości skalowania aplikacji oraz dodatkowo, co stwierdzają sami autorzy, im więcej rozpatrywanych aparatów, tym mniejsza dokładność predykcji - aparaty tego samego producenta, będące nie daleko od siebie pod względem rodziny modelu produkują bardzo podobne *odciski palców*. Co więcej, do działania i samego treningu, opisywany model wymaga zdjęć z wypełnionymi znacznikami.

Inną rodziną podejść są te zorientowane na pomijanie metadanych, podział zdjęcia na mniejsze części, a następnie sprawdzanie ich właściwości względem siebie. Przykładem może być praca z 2015 roku autorstwa M. Goljan and J. Fridrich, w której opisują stworzony przez siebie wariant metody CFA(z ang. *color filter array*), nazwany przez siebie CRM(z ang. *color rich model*) [8]. Jego zadaniem jest stworzenie szeregu statystyk opisujących relacje pomiędzy kolorami pikseli w określonym obszarze. Następnie mając te informacje sprawdzane jest czy występują w obrazie nagłe przejścia w których zbierane statystyki wskazują na obiekt spoza oryginalnej przestrzeni. Takie podejście świetnie sprawdza się w sytuacji w której manipulacja polega na wstawieniu obcego elementu w zdjęcie. Problemem są oczywiście takie manipulacje, które ingerują w cały obszar zdjęcia, a mówiąc bardziej szczegółowo, w przestrzeń kolorów zdjęcia.

Podobny do opisanego powyżej jest pomysł pracy autorstwa D. Cozzolino, G. Poggi, L. Verdoliva z 2015 roku, z tą różnicą że zamiast przestrzeni barw autorzy sprawdzają szum występujący pomiędzy pikselami poszczególnych elementów zdjęcia [5]. Nagłe zmiany, lub pewne nieciągłości pozwalają autorom nie tylko stwierdzić czy zdjęcie zostało przerobione czy nie -

pozwala też zlokalizować miejsce w którym do takiej manipulacji potencjalnie doszło.

Innym podejściem bazującym na poprzednich ale jednak bardziej zwróconym w stronę uczenia głębokiego, była praca naukowców z USA z 2017 roku [1]. Podobnie jak poprzednicy, wykorzystali podział obrazu na mniejsze elementy z tą jednak różnicą że tym razem nie definiowali funkcji ekstrahującej z pomniejszych elementów cech - zamiast tego stanowiły one wejście klasyfikatora LSTM, którego pamięć została wykorzystana do przetworzenia pojedynczego zdjęcia. Z racji jednak że badali oni zmiany lokale natrafili na ograniczenia takie same jak inni uczeni [8] [5].

Kolejnym typem prób rozpoznawania czy dane zdjęcie zostało zmanipulowane czy nie jest wykorzystanie podwójnej kompresji JPEG - skorzystanie z algorytmu ELA(z ang. *Error Level Analysis*), co zostało opisane dość dokładnie w pracy N. Krawetza z 2007 roku[13]. Podczas zapisywania plików JPEG dochodzi do kompresji danych, domyślnie wartość kompresji jest ustawiona na 90%. Oznacza to że algorytm kompresji będzie przeglądał obszary wielkości 8x8 pikseli na zdjęciu w celu zaoszczędzenia $\sim 10\%$ danych. Ideą algorytmu ELA jest zapisanie zdjęcia podwójnie z kompresją pomiędzy $\sim 80\%$ - $\sim 90\%$, a następnie *odjęcie* od zdjęcia oryginalnego. Taki zabieg sprawi że dojdzie do wytworzeniu artefaktów kompresji - miejsc gdzie *poziom* uproszczenia jest różny dla elementów na zdjęciu. Tym samym będzie to oznaczało, że elementy te przed zastosowanie algorytmu ELA były różnej jakości, czyli nie należą do tego samego obrazka [17].

Jednym z ciekawszych i najbliższych mojej propozycji rozwiązania problemu jest praca z 2016 roku, która wykorzystuje autoencoder [29]. Ideą pracy autoencodera jest kompresja informacji wejściowej, a następnie próba odtworzenia jej w jak najlepszym stopniu. Okazuje się, że gdy zbadamy obraz oryginalny, skompresowany i odtworzony to wystąpią różnice w jakości odtwarzania w zależności od tego czy dany obszar był poddany manipulacji czy nie. Tym samym ponownie nie dość że dostaniemy wynik klasyfikacji jako nie manipulowane zdjęcie lub zmanipulowane, to jeszcze dostaniemy obszar który nasz model wytypował jako zmieniony względem oryginału.

4. ZAŁOŻENIA METODOLOGICZNE

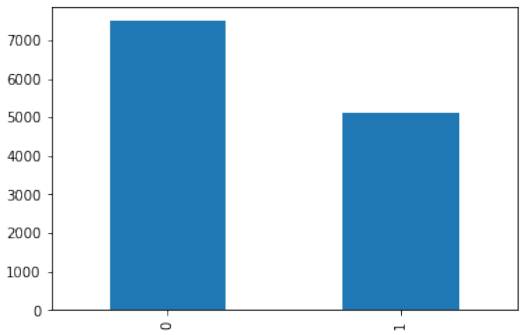
W poniższym rozdziale przedstawię elementy składowe jakie stanowiły części wykonanych eksperymentów.

4.1. WYBRANE ZBIORY DANYCH

W pracy posłużyłem się dwoma zbiorami danych:
— Casia image tampering detection evaluation database [6]
— The PS-Battles Dataset [10]

4.1.1. CASIA v.2

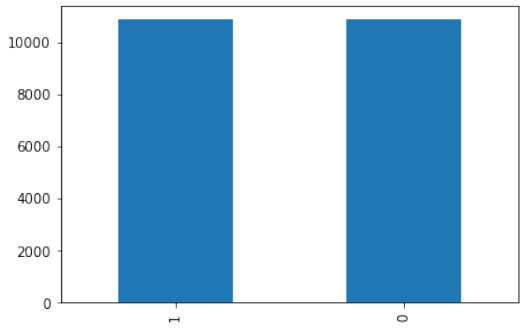
Opisywany zbiór posiada 12614 zdjęć, z czego 7941 z nich stanowi zdjęcia oryginalne, a kolejne 5123 to zdjęcia zmanipulowane(patrz rysunek 4.1). Zgodnie z artykułem [6] elementy zbioru różnią się wielkością od 320×240, aż do 800×600 pikseli. Manipulacja zdjęć jest wykonana poprzez wycinanie, kopiowanie i rozmazanie. Twórcy zbioru danych podzielili go na 9 kategorii: scena, zwierzę, architektura, postać, roślina, artykuł, natura, wnętrze i tekstura - na potrzeby pracy interesuje Nas tylko podział na dane prawdziwe i zmanipulowane. Dodatkowo zdjęcia zawarte w zbiorze zapisane są jako *.JPEG, *.BMP, *.TIFF.



Rys. 4.1: Ilość przedstawicieli danej klasy w zbiorze CASIA v.2

4.1.2. The PS-Battles Dataset

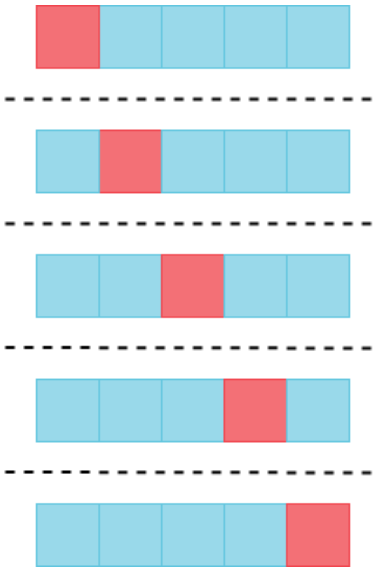
Powyższy zbiór jest liczniejszy - 21758 elementów, z czego po 10879 należy do każdej z dwóch klas(patrz rysunek 4.2). Formatem zdjęć jest *.JPEG lub *.PNG, a wielkość jest różna od 136×68, do aż 20000×12024 pikseli. Zbiór danych został stworzony przez społeczność forum internetowego *reddit.com/r/photoshopbattles* do której na moment pisania pracy należy $\sim 16,5$ mln użytkowników, którzy to codziennie dodają kolejne ~ 92 posty ze zdjęciami. Ideą tej społeczności jest wrzucanie zdjęć nieoczywistych, śmiesznych a następnie przerabianie ich na śmieszniejsze.



Rys. 4.2: Ilość przedstawicieli danej klasy w zbiorze PS-Battles

4.2. STRATYFIKOWANA K-KROTNA WALIDACJA KRZYŻOWA

Do przeprowadzenia wiarygodnych eksperymentów i wyliczeniu odpowiednich metryk pracy modeli skorzystałem z k-krotnej stratyfikowanej walidacji krzyżowej. W samej k-krotnej walidacji krzyżowej chodzi po prostu o podział zbioru Z na k równych elementów [9]. Następnie 1 z tych k zbiorów zostaje zbiorem testowym Z_t podczas gdy pozostałe $k-1$ zbiorów zostają zbiorem uczącym Z_u. Całe zadanie jest powtarzane k razy. Całość została pokazana na rysunku 4.3



Rys. 4.3: Przebieg 5-krotnej walidacji krzyżowej

Z racji jednak że w zbiorze testowym CASIA [6] ilość przedstawicieli klas nie jest taka sama - zdecydowałem się na stratyfikowaną wersję k-krotnej walidacji krzyżowej, oznacza to że podczas podziału na podzbiory operacja ta zachowuje oryginalny lub zbliżony do oryginalnego poziom niezbalansowania danych.

4.3. PAROWE TESTY STATYSTYCZNE

Do sprawdzenia czy po przeprowadzeniu eksperymentu na tych samych modelach ale np. różnych jądrach, różnice które otrzymałem w wartościach metryk są istotne statystycznie, skorzystałem z testu *T-Studenta* z poziomem ufności $\alpha = 0,05$ [2]. Obliczenia zostały przepro-

wadzone dla każdej z metryk, tj. dokładności, czułości, precyzji i miary *F*. Za hipotezę zerową przyjąłem, że pomiędzy klasyfikatorami nie ma istotnej różnicy statystycznej, co oznacza że kiedy wyliczona przez statystykę t, wartość $p - value$ będzie mniejsza lub równa α - odrzucę hipotezę zerową. W przeciwnym wypadku($\alpha > p - value$) przyjmę hipotezę zerową.

4.4. MASZYNA WEKTORÓW NOŚNYCH(*SVM*)

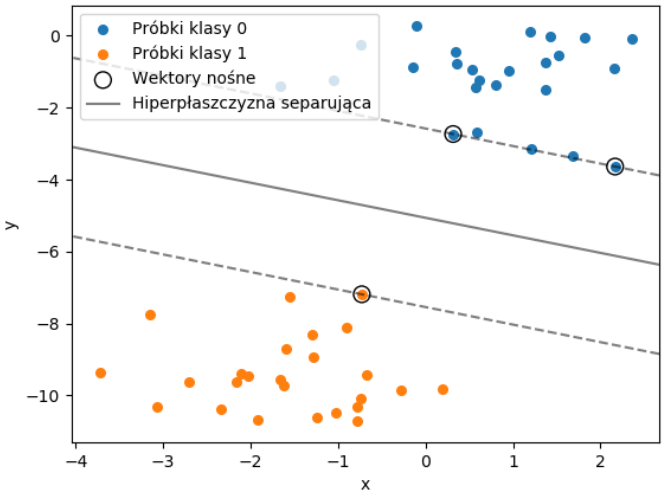
Jako bazowym klasyfikatorem binarnym posłużyłem się maszyną wektorów nośnych(ang. *Support Vector Machine - SVM*). Ideą tego algorytmu jest rzutowanie danej przestrzeni cech, na przestrzeń o większej ilości wymiarów. Taka operacja pozwala osiągnąć własność zwaną liniową separowalnością. Oznacza to nie mniej ni więcej, że istnieje możliwość rozdzielenia zbioru uczącego, tj. $(\vec{x}_1, y_1), ..., (\vec{x}_n, y_n)$ za pomocą hiperpłaszczyzny separującej, opisanej wzorem 4.1

$$\vec{w} * \vec{x} - b = 0 \tag{4.1}$$

, gdzie $\vec{w}$ jest wektorem normalnym dla tej płaszczyzny. Ponadto, wyliczane są wektory nośne, takie żeby spełniały własność opisaną wzorem 4.2

$$\begin{aligned} \vec{w} * \vec{x} - b &= 1 \\ \vec{w} * \vec{x} - b &= -1 \end{aligned} \tag{4.2}$$

Tym samym poszczególne próbki są rozdzielane ponad lub poniżej hiperpłaszczyzny separującej, co zostało zobrazowane na rysunku 4.4.

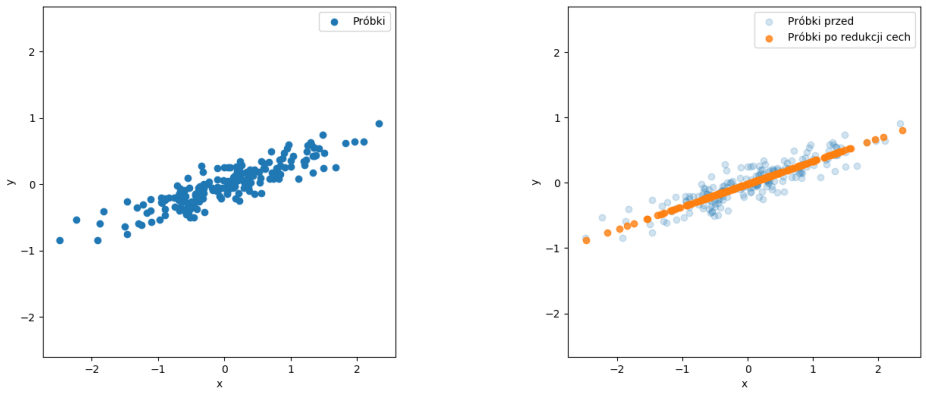


Rys. 4.4: Przykład klasyfikacji maszyny wektorów nośnych

4.5. ANALIZA GŁÓWNYCH SKŁADOWYCH

Dla uczącego zbioru danych Z_u, składającego się z n obserwacji, gdzie każda ma postać $k_1, ..., k_i$ można zdefiniować chmurę punktów, która im odpowiada. Oczywiście, mamy wtedy do czynienia z n punktów w przestrzeni i wymiarowej [14]. Na potrzeby przykładu można przyjąć że mamy do czynienia z $n = 50$ i $i = 2$. Celem analiza głównych składowych(z ang.*principal*

component analysis- PCA) jest taki obrót układu współrzędnych, aby maksymalizować wartość wariancji pierwszej współrzędnej, a następnie drugiej. Obrót ten, ma na celu skonstruowanie nowej przestrzeni, w której to początkowe współrzędne są najważniejsze. Graficzny przykład omawianej operacji widoczny jest na rysunku 4.5.



Rys. 4.5: Dane przed i po redukcji cech przy pomocy PCA(n_components=1)

4.6. VGG NET

5. IMPLEMENTACJA I INTERPRETACJA WYNIKÓW BADAŃ

5.1. SVM

5.2. IMPLEMENTACJA ŚRODOWISKA RAMOWEGO WYKORZYSTUJĄCEGO ISTNIEJĄCE ARCHITEKTURY SIECI

5.3. IMPLEMENTACJA AUTORSKIEJ METODY WYKORZYSTANIA UCZENIA GŁĘBOKIEGO W ZADANIU KLASYFIKACJI

6. PODSUMOWANIE

BIBLIOGRAFIA

[1] Bappy, J.H., Roy-Chowdhury, A.K., Bunk, J., Nataraj, L., Manjunath, B., *Exploiting spatial structure for localizing manipulated image regions*, IEEE International Conference on Computer Vision. 2017.

[2] Billingsley, P., *Prawdopodobieństwo i miara* (Wydawnictwo Naukowe PWN, 2009).

[3] Chollet, F., *Deep Learning with Python* (Manning Publications, 2017).

[4] Cichosz, P., *Systemy uczące się* (Wydawnictwo Naukowo-Techniczne, 2000).

[5] Cozzolino, D., Poggi, G., Verdoliva, L., *Splicebuster: a new blind image splicing detector*, IEEE International Workshop on information forensics and security. 2015.

[6] Dong, J., Wang, W., Tan, T., *Casia image tampering detection evaluation database*, IEEE China Summit and International Conference on Signal and Information Processing. 2013.

[7] Farid, H., *Photo Forensics* (MIT Press, 2016).

[8] Goljan, M., Fridrich, J., *Cfa-aware features for steganalysis of color images*, Proceedings of SPIE - The International Society for Optical Engineering. 2015.

[9] Géron, A., *Hands-On Machine Learning with Scikit-Learn and TensorFlow* (O'Reilly Media, 2017).

[10] Heller, S., Rossetto, L., Schuldt, H., *The ps-battles dataset - an image collection for image manipulation detection*, Computing Research Repository. 2018.

[11] Hinton, G., Salakhutdinov, R., *Reducing the dimensionality of data with neural networks*, Neural Computation. 2006.

[12] Holler, J., Tsiatsis, V., Mulligan, C., Avesand, S., Karnouskos, S., Boyle, D., *From machineto-machine to the internet of things: Introduction to a new age of intelligence*, Academic Press. 2014.

[13] Krawetz, N., *A picture's worth...*, Hacker Factor Solutions. 2007.

[14] Krzanowski, W., *Principles of Multivariate Analysis* (OUP Oxford, 2000).

[15] Laney, D., *3d data management: Controlling data volume, velocity, and variety*, tech. rep., META Group. 2001.

[16] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W., Jackel, L.D., *Backpropagation applied to handwritten zip code recognition*, Neural Computation. 1989.

[17] Liu, Q., *Detection of misaligned cropping and recompression with the same quantization matrix and relevant forgery*, ACM workshop on Multimedia in forensics and intelligence. 2011.

[18] Lovelace, A., Menabrea, L.F., *Sketch of the analytical engine invented by charles babbage... with notes by the translator. translated by ada lovelace*, Scientific Memoirs. 1843.

[19] Mashey, J.R., *Big data... and the next wave of infrastress*, Slides from invited talk. 1998.

[20] Menabrea, L.F., *Sketch of the analytical engine invented by charles babbage*, Bibliothèque Universelle de Genève, No. 82. 1842.

[21] Mitchell, T.M., *Machine Learning* (McGraw-Hill, Inc., 1997).

[22] Piczak, K., *Klasyfikacja dźwięku za pomocą splotowych sieci neuronowych*, Prasa akademicka. 2018.

[23] Raschka, S., *Python Machine Learning* (Packt Publishing, 2015).

[24] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., S, M., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L., *Imagenet large scale visual recognition challenge*, International Journal of Computer Vision. 2015.

[25] Samuel, A.L., *Some studies in machine learning using the game of checkers*, IBM Journal of Research and Development(Volume: 3, Issue: 3). 1959.

[26] Singh, S., Singh, N., *Big data analytics*, Communication, Information Computing Technology (ICCICT). 2012.
[27] Swaminathan, A., Wu, M., Liu, K.J.R., *Digital image forensics via intrinsic fingerprints*, IEEE transactions on information forensics and security. 2008.
[28] Turing, A.M., *Computing machinery and inteligence*, Mind(Volume: LIX, Issue: 236). 1950.
[29] Zhang, Y., Goh, J., Win, L., Thing, V., *Image region forgery detection: A deep learning approach*, Proceedings of the Singapore Cyber-Security Conference. 2016.

SPIS RYSUNKÓW

SPIS TABEL