



Uniwersytet im. Adama Mickiewicza w Poznaniu

Wydział Matematyki i Informatyki

Kierunek Informatyka, specjalność sztuczna inteligencja

Praca magisterska

Metody wykrywania obiektów na obrazie wideo i ich zastosowanie w analizie nagrani sportowych

Methods for detecting objects in video image and their
application in the analysis of sports recordings

Jan Nowak

Numer albumu: 426206

Promotor:

prof. UAM dr hab. Krzysztof Dyczkowski

Poznań, 2022

Streszczenie

Celem niniejszej pracy dyplomowej jest przedstawienie i analiza algorytmów wygrywania obiektów z dziedziny widzenia komputerowego, z zastosowaniem w aspekcie piłki nożnej we współpracy z klubem sportowym KKS Lech Poznań. Omówiona teoria dotycząca uczenia maszynowego, widzenia komputerowego oraz wykorzystywanych architektur wykrywania obiektów, pozwoli lepiej zrozumieć sposób rozwiązania problemu oraz analizę wyników. Przedstawienie planów i etapów badań oraz narzędzia badawczego pozwoli zrozumieć sposób podejścia do problemu. Omówienie wykorzystywanych zbiorów danych, określi środowisko badanego problemu. Zaprezentowane rezultaty z przeprowadzonych eksperymentów algorytmów oraz zbiorów danych, przedstawią spójność otrzymanych wyników. Omówienie wniosków eksperymentu potwierdziło słuszność celu pracy. Przeprowadzone badania pozwoliły wybrać najskuteczniejszą architekturę wykrywania zawodników oraz piłki w materiałach wideo dostarczonych przez klub sportowy KKS Lech Poznań. Specjalnie stworzone, odpowiednio przygotowane i udokumentowane narzędzie badawcze oraz dostarczone w ramach niniejszej pracy zasoby, umożliwiają kontynuację badań oraz usprawnią rozwój technologii wykrywania obiektów szczególnie w aspekcie piłki nożnej.

Słowa kluczowe: uczenie maszynowe, widzenie komputerowe, wykrywanie obiektów, piłka nożna

Abstract

The purpose of this thesis is to present and analyze object detection algorithms of the computer vision domain. The thesis was created in cooperation with the KKS Lech Poznań sports club, therefore the main aspect was football. For a better understanding of the method of solving the problem and the architectures used, the machine learning theory was described. The plans and stages of research as well as the research tool were presented, which allowed for a better understanding of the approach to the problem. To define the research environment, the datasets were presented. The results of the experiments confirmed the reliability of the obtained results. Based on the research, the most effective architecture for detecting players and the ball in video materials provided by the KKS Lech Poznań sports club was selected. A specially developed, properly prepared and documented research tool and resources provided as part of this thesis will facilitate the continuation of research and the development of object detection technology, especially in the aspect of football.

Keywords: machine learning, computer vision, object detection, football

Spis treści

Wstęp	7
1. Zagadnienia teoretyczne	10
1.1. Sztuczna inteligencja	10
1.2. Widzenie komputerowe (ang. Computer vision)	11
1.3. Sieć neuronowa (ang. Neural network)	12
1.4. Splotowa sieć neuronowa	14
1.5. Piramidowa sieć wykrywania cech	15
1.6. Architektury detekcji obiektów	16
1.6.1. Fast R-CNN	18
1.6.2. Faster R-CNN	18
1.6.3. Mask R-CNN	20
1.6.4. Panoptic FPN	21
1.6.5. YOLO	22
1.7. Model sieci neuronowej	23
1.8. Wstępnie wytrenowane modele sieci neuronowych	24
1.9. Biblioteki sieci neuronowych	25
1.10. Metody pomiaru błędu obserwacji	26
1.10.1. Indeks Jaccarda	28
1.10.2. Średnia precyzja (AP i mAP)	29
2. Problem i metoda badań własnych	32

2.1. Problemy badawcze i hipotezy	32
2.2. Plan i etap badań	33
2.2.1. Architektury i platformy wykrywania obiektów	34
2.2.2. Narzędzie badawcze	35
2.2.3. Gromadzenie i generowanie danych	39
2.2.4. Douczenie wstępnie wytrenowanych modeli sieci neuronowych	42
2.2.5. Zapisywanie danych i prezentacja wyników	43
2.3. Metodyka badań	44
2.3.1. Przygotowanie zbiorów danych	44
2.3.2. Uczenie modeli detekcji obiektów	49
2.3.3. Przeprowadzenie badań	50
2.4. Podsumowanie	53
3. Prezentacja i analiza wyników badań własnych	54
3.1. Analiza wyników badań	54
3.1.1. Dokładność wykrywania obiektów danej architektury .	55
3.1.2. Dokładność rozpoznawania typu obiektu	56
3.1.3. Wpływ jakości materiału wideo na dokładność wykrywania obiektów	59
3.1.4. Dokładność wykrywania obiektów przy użyciu modeli douczonych	65
3.2. Wnioski z badań	67
3.2.1. Podsumowanie	70
Wnioski końcowe	72
Spis ilustracji	76

SPIS TREŚCI**6****Spis tabel** **78****Bibliografia** **83**

Wstęp

Dynamiczny rozwój technologii komputerowej w tym kart graficznych oraz dostęp i możliwość gromadzenia ogromnej ilości danych, znaczaco wpłynęło na rozwój uczenia maszynowego. Szeroko pojęta sztuczna inteligencja zaczęła być wykorzystywana w wielu dziedzinach życia. Powstało wiele ośrodków badawczych rozwijających wyspecjalizowane algorytmy. Sztuczna inteligencja znalazła zastosowanie między innymi w wyszukiwarkach internetowych, tłumaczeniu językowym, cyberbezpieczeństwie oraz w dziedzinach wykorzystujących widzenie komputerowe takich jak medycyna, autonomiczne samochody, sport.

Widzenie komputerowe w znacznym stopniu poszerzyło i ułatwiło możliwości analizowania przebiegu spotkania meczu piłki nożnej. Odpowiednie narzędzia bazujące na uczeniu maszynowym mogłyby w znacznym stopniu wspomóc kadrę szkoleniową w podejmowaniu decyzji, co mogłyby przyczynić się do zwiększenia zwycięstw klubu piłkarskiego. Narzędzie mogłyby obserwować i analizować posturę oraz sposób przemieszczania się zawodników. Dostarczać cennych informacji, jak zwiększyć skuteczność oraz zmniejszyć prawdopodobieństwo wystąpienia kontuzji u zawodnika. Znając strategię przeciwnej drużyny oraz słabości własnej drużyny kadra mogłyby polepszyć taktykę, zwiększając szansę na wygraną zespołu. Większa liczba zwycięstw prowadziłaby do większego zainteresowania ze strony kibiców. Silny zespół utrzymy-

wały się na wyższej pozycji w tabeli ligowej, osiągając przy tym większe zyski finansowe. Bogaty klub mógłby się szybciej i skuteczniej rozwijać. W przypadku kiedy rozwijający się klub nie miałby wystarczającej ilości środków finansowych lub nie miałby dostępu do urządzeń rejestrujących sytuację na boisku takich jak rejestratory GPS, czujniki zbliżeniowe, spora ilość kamery itp. lub przeprowadzane treningi odbywały się w różnych miejscach co znacznie utrudniałoby rozstawianie takiego sprzętu. Zastosowanie widzenia komputerowego byłoby trafnym rozwiązaniem tego problemu, ponieważ często wystarczyłaby tylko jedna kamera i jej operator. Zapisany materiał można byłoby wielokrotnie analizować oraz odpowiednio przetwarzać z wykorzystaniem coraz to nowszych technologii sztucznej inteligencji otrzymując lepsze wyniki i szczegółowe analizy.

Celem niniejszej pracy jest analiza algorytmów wykrywania obiektów stosowanych w widzeniu komputerowym w aspekcie piłki nożnej. Zbadana została dokładność poszczególnych architektur w wykrywaniu zawodników oraz piłki, z fragmentów nagrani spotkań meczowych. Zbadany został także wpływ jakości przetwarzanego materiału wideo na skuteczność algorytmów oraz udzielona została odpowiedź na pytanie czy douczenie modeli wstępnie wytrenowanych, obrazami związanymi typowo z piłką nożną miało wpływ na zwiększenie dokładności omawianych architektur.

Głównym źródłem wiedzy były artykuły z badań naukowych oraz książki z dziedziny sztucznej inteligencji.

W pierwszym rozdziale omówione są zagadnienia teoretyczne pozwalające na lepsze zrozumienie poruszonego problemu. Wyjaśniona jest teoria dotycząca uczenia maszynowego. Zaprezentowane są narzędzia i metryki wykorzystywane do analizy otrzymanych wyników. Przedstawione są zasady działania poszczególnych architektur.

Drugi rozdział opisuje sposób podejścia do problemu, prezentuje plan działania oraz sposoby przeprowadzania badań. Opisywane są specyfikacje wykorzystywanych modeli architektur, prezentowane jest narzędzie badawcze, opisywane są źródła danych oraz narzędzia generowania danych. Opisywany jest proces uczenia modeli. Prezentowany jest sposób zapisywania danych.

W trzecim rozdziale przedstawiane są wyniki i analizy dokładności poszczególnych metod wykrywania obiektów. Analizowany jest wpływ typu obiektu na wyniki architektury. Prezentowany jest wpływ jakości przetwarzanego materiału wideo na efektywność algorytmów. Omawiane są wyniki dokładności douczonych modeli wstępnie wytrenowanych. Podsumowywane są wyniki badań. Wypisywane są wnioski z realizowanych badań.

Wnioski płynące z przeprowadzanych badań pozwoliły stwierdzić jaki wstępnie wytrenowany model oraz jaka architektura jest najskuteczniejsza. Jak na wyniki wpłynęły typy wykrywanych obiektów, jaki wpływ na dokładność architektur miała jakość materiału wideo oraz czy douczenie modeli poprawiło rezultaty. Wyniki badań pozwoliły podjąć decyzję dotyczącą wyboru modelu oraz architektury wykorzystywanej w narzędziu, tworzonym w ramach projektu badawczo-rozwojowego realizowanego we współpracy z klubem sportowym KKS Lech Poznań.

Rozdział 1

Zagadnienia teoretyczne

1.1. Sztuczna inteligencja

Zdefiniowanie pojęcia ”Sztuczna Inteligencja” jest dość kłopotliwe, ponieważ nie istnieje jej jedna wspólna definicja. Jednym z pierwszych autorów definicji był Jhon McCarthy emerytowany profesor Stanfordu. Określił ją jako ”The science and engineering of making intelligent machines.” [24] co można przetłumaczyć jako ”nauka i inżynieria tworzenia inteligentnych maszyn.”.

W książce ”Artifical intelligence: Structures and Strategiesfor Complex Problem Solving” autorstwa George F. Luger'a pojęcie ”Sztuczna Inteligencja” zdefiniowane zostało jako ”The branch of computer science that is concerned with the automation of intelligent behavior.” [7] co można przetłumaczyć na język polski jako ”Dziedzina informatyki zajmująca się automatyzacją inteligentnych zachowań.”

Przytoczone definicje nie są dość szczegółowe, ponieważ cała dziedzina sztucznej inteligencji jest szeroko pojęta i odnosi się do wielu domen, między innymi takich jak: uczenie maszynowe, sieci neuronowe, przetwarzanie

języka naturalnego, syntezatory mowy, robotyka, planowanie i optymalizacja, widzenie komputerowe, systemy eksperckie. Na poniższym Rysunku 1.1 przedstawiony został diagram umożliwiający lepsze zobrazowanie jak szeroka jest dziedzina sztucznej inteligencji.



Rys. 1.1. Diagram Sztuczna Inteligencja. Źródło: opracowanie własne

1.2. Widzenie komputerowe (ang. Computer vision)

Algorytmy opisywane w niniejszej pracy w głównej mierze należą do dziedziny wiedzenia komputerowego. Interdyscyplinarna dziedzina widzenia komputerowego łączy ze sobą sposoby gromadzenia obrazu, odpowiedniego przetwarzania a następnie analizowania zdobytego materiału w postaci zdjęć lub filmów. Zadaniem inżynierii widzenia komputerowego jest zrozumienie i zautomatyzowanie procesów jakie wykonuje ludzki mechanizm wzrokowy, przy użyciu maszyn [4][20][29].

Zbierane dane mogą zawierać utrwalony obraz rzeczywisty, zgromadzony przy użyciu odpowiednich przetworników obrazu wykorzystywanych między innymi w aparatach i kamerach. Alternatywnym sposobem gromadzenia danych mogą być materiały wygenerowane za pomocą symulacji komputerowych, co może znacznie ułatwić i przyśpieszyć pozyskiwanie zbiorów danych niezbędnych do uczenia maszynowego [19].

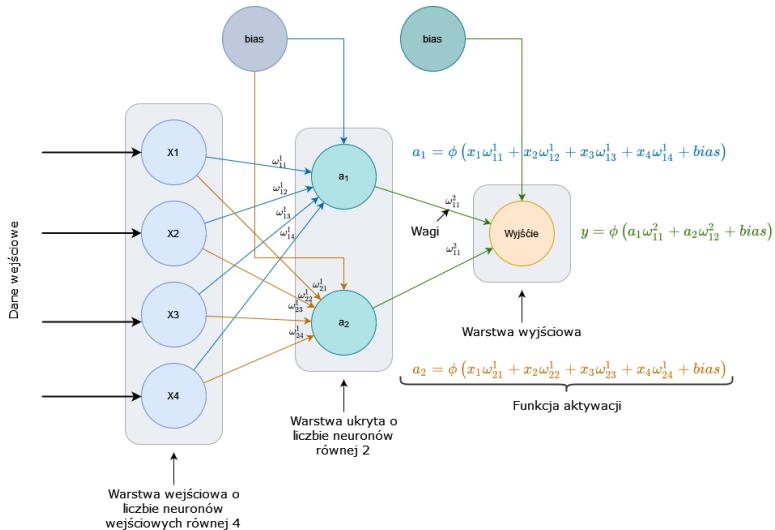
Zgromadzony materiał może zostać odpowiednio przetworzony w celu poprawy jakości oraz w celu usunięcia różnego rodzaju zakłóceń zaburzających dokładność danych. W tym celu wykorzystywane są odpowiednie algorytmy usuwania szumu, poprawiania ostrości, wyrównywania histogramu, rozcięcia histogramu, zamiany przestrzeni kolorów, przycinania i wiele innych. Dokładny opis przytoczonych metod przetwarzania obrazu można znaleźć w książce "Computer Vision: Algorithms and Applications, 2nd ed." (patrz [29]).

Zgromadzony i odpowiednio przetworzony materiał poddawany jest analizie. W zależności od oczekiwanych rezultatów może być to rozpoznawanie liter (ang. Optical character recognition, w skrócie OCR), wzorców i kształtów np. w medycynie do wykrywania guzów, rozpoznawanie obiektów takich jak samochody, zwierzęta i ludzie. W przypadku piłki nożnej uwaga została skupiona na wykrywaniu ludzi czyli zawodników oraz piłki.

1.3. Sieć neuronowa (ang. Neural network)

Wykrywanie obiektów takich jak zawodnicy czy piłka nie jest trywialne i wymaga zastosowania odpowiednich metod. Ludzki mózg potrafi dokonywać rozpoznawania percepcyjnego (np. rozpoznanie znajomej twarzy osadzonej w nieznanej scenie) w zaskakująco szybkim tempie, ponieważ zajmuje mu

to zaledwie około 100-200ms [11]. Dlatego naukowcy postanowili stworzyć sieć neuronową bazującą na budowie mózgu. „Sieć neuronowa jest bardzo uproszczonym modelem mózgu. Składa się ona z dużej liczby (od kilkuset do kilkudziesięciu tysięcy) elementów przetwarzających informację. Elementy te nazywane są neuronami, chociaż w stosunku do rzeczywistych komórek nerwowych ich funkcje są bardzo uproszczone, by nie powiedzieć sprymitywizowane. Neurony są powiązane w sieć za pomocą połączeń o parametrach (tak zwanych wagach) modyfikowanych w trakcie tak zwanego procesu uczenia. Topologia połączeń oraz ich parametry stanowią program działania sieci, zaś sygnały pojawiające się na jej wyjściach w odpowiedzi na określne sygnały wejściowe są rozwiązaniami stawianych jej zadań.” [30]. Na poniższym Rysunku 1.2 zaprezentowana została przykładowa sieć neuronowa wykorzystująca pośrednie warstwy ukryte.



Rys. 1.2. Diagram sieci neuronowej z warstwą wejściową rozmiaru czterech neuronów, jedną warstwą ukrytą oraz jedną warstwą wyjściową. Źródło: opracowanie własne wzorowane na (patrz [15])

Na przestrzeni lat powstało wiele architektur sieci neuronowych. W zależności od rozwiązywanego problemu są one specjalnie projektowane. Modyfi-

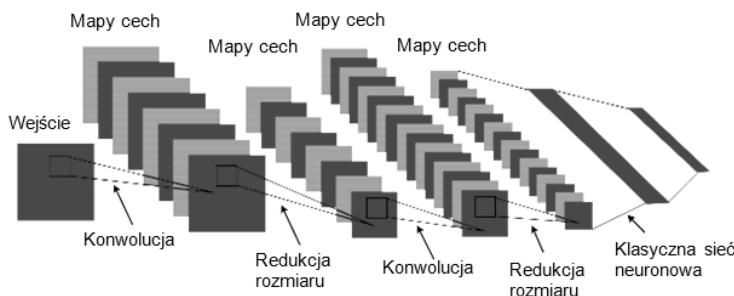
kacji mogą podlegać hiperparametry między innymi takie jak: ilość neuronów wejściowych, liczba warstw ukrytych, funkcje aktywacji, współczynnik uczenia (ang. learning rate). W przypadku dziedziny widzenia komputerowego najczęściej wykorzystywanymi architekturami są: splotowe sieci neuronowe oraz piramidowe sieci wykrywania cech [20].

1.4. Splotowa sieć neuronowa

W sieciach neuronowych parametry przekazywane są do warstwy wejściowej w postaci wektora. Podczas gdy obrazy cyfrowe zapisane są w postaci macierzy o wymiarach odpowiadających rozdzielczości obrazu, gdzie każdy piksel to jeden element macierzy. Dodatkowo dla zdjęć kolorowych w przestrzeni RGB każdy piksel będzie opisany za pomocą wektora o rozmiarze trzech elementów. Dla tradycyjnej sieci neuronowej kolorowy obraz o rozdzielczości 1920px na 1080px musiałby zostać przekształcony do wektora o rozmiarze $1920 * 1080 * 3 = 6220800$. W tym przypadku sieć otrzymałaby 6220800 parametrów. Tak ogromna liczba danych wejściowych drastycznie wpłynie na złożoność obliczeniową, wykorzystując sporą ilość pamięci, zasobów procesora oraz czasu potrzebnego do obliczenia wag. Dodatkowo poprzez zamianę obrazu cyfrowego na wektor, tracone są cenne dane np. wartości sąsiednich pikseli co w przypadku rozpoznawania kształtów ma duże znaczenie [20].

W celu rozwiązania tego problemu opracowana została splotowa sieć neuronowa (ang. Convolutional neural network, w skrócie CNN), która działa podobnie do tradycyjnej sieci neuronowej, ponieważ również składa się z neuronów, które samoptymalizują się poprzez uczenie, warstwy wejścia, ukrytych warstw oraz warstwy wyjścia. Również wykorzystywane są funkcje aktywacji oraz aktualizowane są wagi. Główną różnicą jest to, że splotowa sieć

neuronowa [...] składa się z kombinacji trzech podstawowych typów warstw: warstwa splotowa, warstwa aktywacji oraz warstwa redukująca rozmiar. [...]” [20] (patrz Rysunek 1.3) ”[...] Dodatkowo wykorzystuje się klasyczne warstwy neuronowe tzw. warstwy pełnego połączenia. [...]” [20]



Rys. 1.3. Splotowa sieć neuronowa. Źródło: [20]

W splotowej sieci neuronowej na wejściu przyjmowany jest obraz, który po kolei przetwarzany jest przez poszczególne warstwy w celu wyodrębnienia cech, filtracji oraz redukcji rozmiaru. Procedury te są powtarzane do otrzymania wektora cech, który przekazywany jest do klasycznej sieci neuronowej w celu klasyfikacji cech i zwróceniu wyniku.

1.5. Piramidowa sieć wykrywania cech

Piramidowa sieć wykrywania cech (ang. Feature Pyramid Network, w skrócie FPN) została zaprojektowana z myślą o dokładnym rozpoznawaniu obiektów przy zachowaniu odpowiedniej szybkości działania. Sama architektura nie wykrywa obiektów a jedynie służy do tworzenia mapy cech, które muszą zostać przekazane do modeli wykrywających obiekty. Pozwala ona z większą dokładnością odtworzyć pozycję wykrytego regionu na oryginalnym obrazie z dokładnością co do pikseli. Sieć FPN często używana jest równolegle z siecią CNN w celu zwiększenia dokładność modelu kosztem większych zasobów

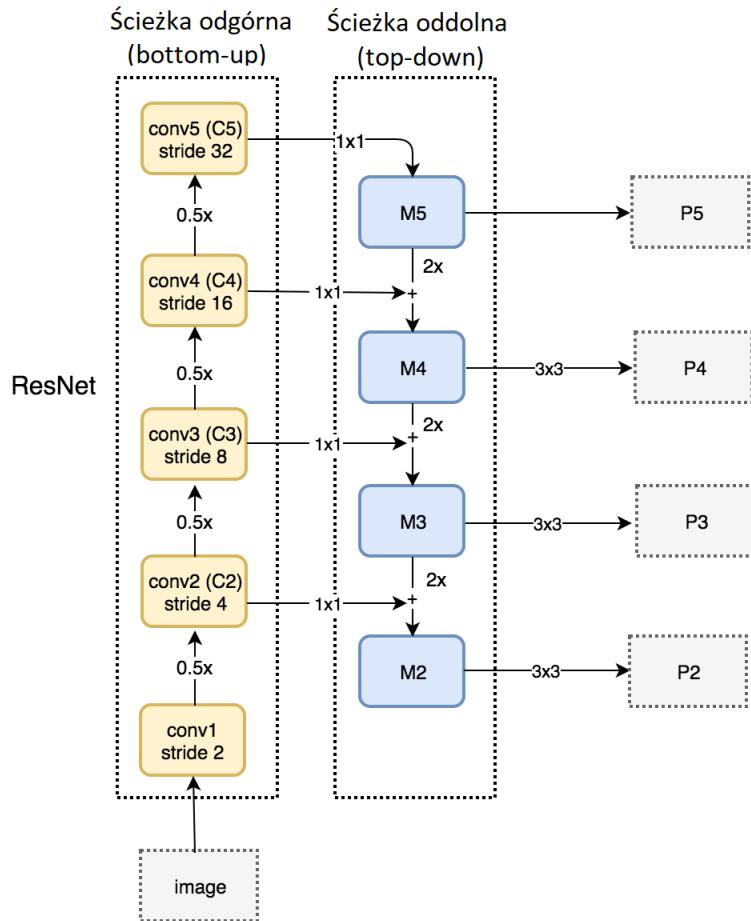
obliczeniowych [22].

Architektura składa się z ścieżki oddolnej i odgórnej. Ścieżka oddolna to zwykła sieć splotowa wykorzystywana do wyodrębniania cech z obrazu. W architekturach bazujących na samych splotowych sieciach neuronowych, z warstwy na warstwę cechy wykrywane są na obrazach o coraz niższej rozdzielczości, tracąc przy tym informację o małych obiektach [22].

W sieciach FPN wykorzystywana jest dodatkowo ścieżka odgórną, która służy do generowania warstw o wyższej rozdzielczości z warstw zawierających cechy wykryte przez sieć splotową w niższej rozdzielczości. Przed każdym powiększeniem warstwy zawierającej mapę cech, przekazywana jest ona do detektora obiektów (Rysunek 1.4P5, P4, P3, P2). Kolejna warstwa zawiera więcej cech oraz coraz lepiej odwzorowuje wykryte obiekty na oryginalnym obrazie. Warstwy cech przekazywane do detektora obiektów zawierają coraz więcej szczegółów zwiększaając dokładność szacowania ale wymagają większych zasobów obliczeniowych. Dodatkowo dzięki zastosowaniu skalowania mapy cech w górę, algorytm jest w stanie dokładniej odtworzyć pozycję pikseli wykrytego obiektu, na oryginalnym obrazie co w szczególności pozytywnie wpływa na dokładność architektur tworzących maski obiektów [22].

1.6. Architektury detekcji obiektów

W przypadku wykrywania obiektów na obrazie, architektura nie jest w stanie z góry określić ile wykryje obiektów, co w przypadku wykorzystania samych splotowych sieci neuronowych, na wyjściu generuje warstwę o zmiennej długości co powoduje problem z połączeniem jej z spłaszczoną (tradycyjną) siecią neuronową. W tym celu, architektury wykrywania obiektów posiadają algorytmy generowania propozycji regionów w których mogą znajdować się



Rys. 1.4. Architektura sieci FPN. Źródło: [22]

obiekty [27].

Propozycje wykrytych regionów są odpowiednio łączone z mapą cech i dopiero wtedy przekazywane do spłaszczonej sieci neuronowej zmniejszając przy tym sporą liczbę parametrów, które i tak najprawdopodobniej nie wpłyńłyby w żaden sposób na dokładność klasyfikacji. Zabieg ten skutecznie zmniejsza zapotrzebowanie na pamięć oraz znacznie przyśpiesza działanie architektury [33][27].

Istnieje wiele architektur detekcji obiektów bazujących na splotowych sie-

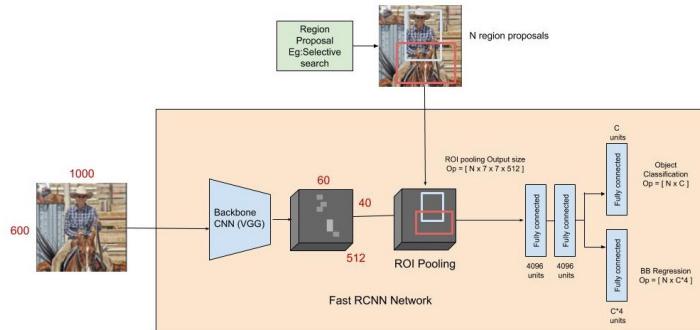
ciach neuronowych, wykrywaniu propozycji regionów oraz piramidowych sieciach wykrywania cech. W omawianej pracy do przeprowadzenia badań wykorzystane zostały architektury takie jak: Faster R-CNN [27], Mask R-CNN [12], Panoptic FPN [17] oraz YOLO [2]. Dodatkowo niektóre z wymienionych modeli zostały użyte w wersji z piramidowymi sieciami wykrywania cech.

1.6.1. Fast R-CNN

Architektura Fast R-CNN przedstawiona na Rysunku 1.5 w celu wykrycia regionów wykorzystuje algorytm selektywnego wyszukiwania "Selective Search for Object Recognition" (Region Proposal). W większości przypadków algorytm ustawiany jest tak aby wykrył 2000 proponowanych obiektów. Równolegle cały obraz przekazywany jest do sieci CNN (Backbone CNN) w celu wykrycia cech obrazu. Wykryte propozycje regionów (ang. region of interest, w skrócie RoI) nakładane są na mapę cech tak aby tworzyły wspólny wektor, który jest skalowany (RoI Pooling) na rozmiar odpowiadający wejściu sieci klasyfikacji. Sieć na wyjściu dla każdej propozycji regionu zwraca dwa wektory: poziom ufności wykrytego obiektu (prawdopodobieństwo, że dany obiekt należy do danej klasy obiektów np. jakie jest prawdopodobieństwo, że wykryty obiekt to człowiek.) (Object Classification) oraz wektor zawierający pozycję obwiedni obiektu (BB Regression) [33].

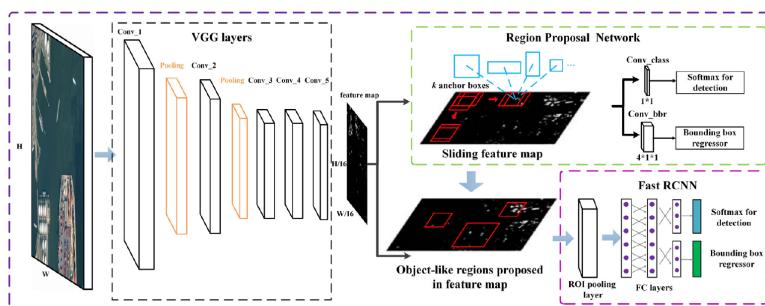
1.6.2. Faster R-CNN

Architektura Faster R-CNN przedstawiona na Rysunku 1.6 jest ulepszoną wersją architektury Fast R-CNN. Zastąpienie algorytmu selektywnego wyszukiwania na splotową siecią neuronową wykrywania regionów (ang. Region Proposal Network, w skrócie RPN) przyczyniło się do zwiększenia szybkości



Rys. 1.5. Architektura sieci Fast R-CNN. Źródło: [1]

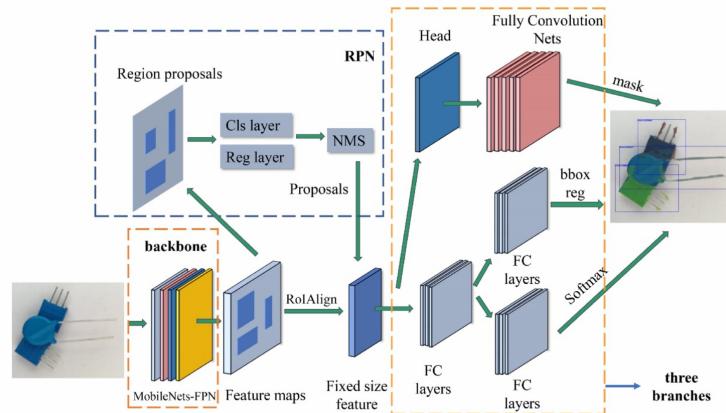
działania architektury oraz uzyskania lepszej dokładności. W architekturze Faster R-CNN w początkowym etapie cały obraz przekazywany jest do głębskiej sieci CNN (VGG layers) w celu stworzenia mapy cech obrazu (feature map). Mapa cech przekazywana jest do sieci wykrywającej propozycję regionów (Region Proposal Network). Mapa cech oraz propozycje regionów łączone są do wspólnego wektora oraz skalowane do rozmiaru wejścia sieci klasyfikującej (RoI pooling layer). Następnie sieć tak jak w przypadku architektury Fast R-CNN zwraca dwa wektory: wektor poziomu ufności oraz wektor obwiedni [27].



Rys. 1.6. Architektura sieci Faster R-CNN. Źródło: [6]

1.6.3. Mask R-CNN

Architektura Mask R-CNN przedstawiona na Rysunku 1.7 rozszerza architekturę Faster R-CNN, dodając dodatkową tradycyjną sieć neuronową do przewidywania masek segmentacji dla każdego wykrytego obiektu (ang. Instance Segmentation) oraz stosując dokładniejszą metodę skalowania mapy cech oraz propozycji regionów (RoIAlign). Sieć neuronowa wykrywająca maski działa niezależnie od sieci wykrywającej obwiednie oraz klasę obiektów. Wykrywanie maski odbywa się na pikselach należących do obiektu dostarczonego z propozycji regionów. Zastosowanie RoIAlign pozwala na dokładne odwzorowanie pikseli wykrytych w mapie cech, do pikseli z oryginalnego obrazu zwiększać przy tym dokładność nakładania maski oraz obwiedni na obiekt [12].

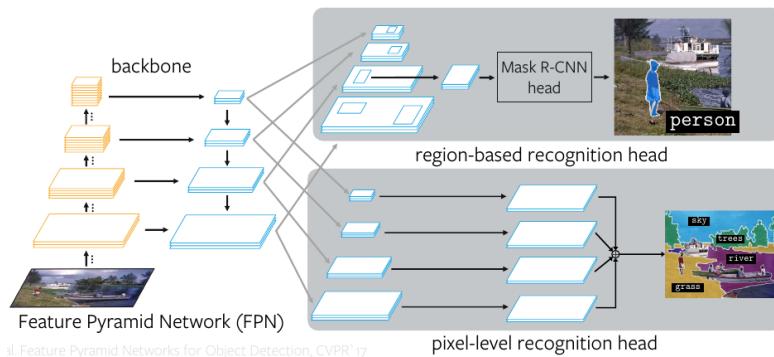


Rys. 1.7. Architektura sieci Mask R-CNN. Źródło: [37]

W celu zwiększenia dokładności wykrywania obiektów (w szczególności małych obiektów), oraz odwzorowywania maski, stosuje się architekturę Mask R-CNN FPN, w której splotowe sieci neuronowe zastąpione zostały piramidowymi sieciami wykrywania cech [12].

1.6.4. Panoptic FPN

Architektura Panoptic FPN jest rozszerzeniem architektury Mask R-CNN FPN, oprócz istniejących już gałęzi generowania obwiedni, wykrywania klasy obiektu oraz generowania maski, dodana została dodatkowa gałąź generowania segmentacji semantycznej. Jak przedstawiono na Rysunku 1.8 w początkowej fazie za pomocą piramidowych sieci wykrywania cech, tworzone są mapy cech obrazu (backbone). Mapy cech przekazywane są równolegle do sieci bazującej na architekturze Mask R-CNN w celu wykrycia, klasyfikacji, wygenerowania maski obiektu oraz do sieci generowania segmentacji semantycznej (pixel-level recognition head) [17].

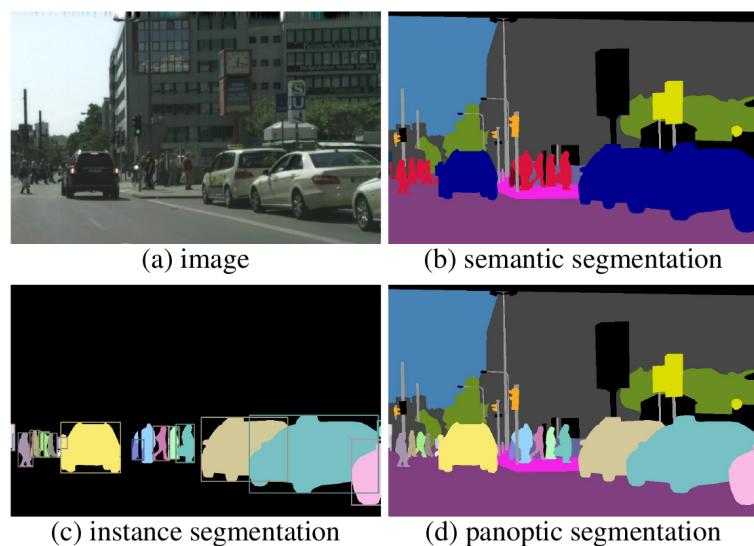


Rys. 1.8. Architektura sieci Panoptic FPN. Źródło: [16]

Segmentacja instancji (ang. Instance Segmentation) oznacza tworzenie maski dla każdego wykrytego obiektu. Jak widać na Rysunku 1.9c piksele należące do danego obiektu otrzymują ten sam kolor, natomiast każdy obiekt opisany jest innym kolorem [18].

Segmentacja semantyczna (ang. Semantic Segmentation) polega na przypisaniu etykiety klasy (samochód, osoba, niebo, ...) każdemu pikselowi na obrazie[18]. Jak przedstawiono na Rysunku 1.9b każdej klasie przypisany został odpowiedni kolor reprezentujący np. klasa samochód opisana jest kolorem niebieskim, klasa człowiek kolorem czerwonym itd.

Segmentacja panoptyczna (ang. Panoptic Segmentation) przedstawiona na Rysunku 1.9d jest połączeniem segmentacji instancji oraz segmentacji semantycznej. Do rzeczy policzalnych (człowiek, samochód, zwierzę itp.) stosowana jest segmentacja instancji. Do rzeczy niepoliczalnych lub amorficznych (niebo, woda, trawa itp.) stosowana jest segmentacja semantyczna [18].



Rys. 1.9. Typy segmentacji pikseli na obrazie. Źródło: [18]

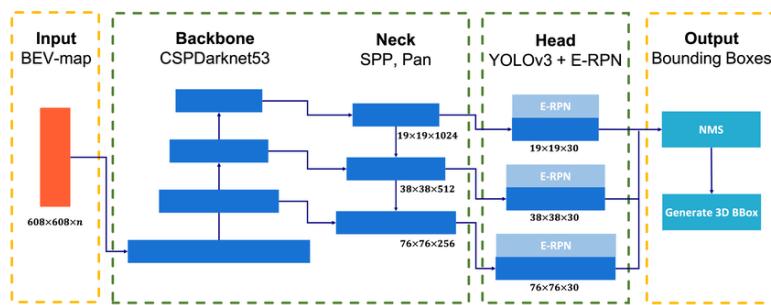
1.6.5. YOLO

Architektura YOLO w odróżnieniu od architektur dwustopniowych takich jak Fast R-CNN, Faster R-CNN, Mask R-CNN oraz Panoptic FPN jest jednostopniowa. Oznacza to, że klasyfikacja obiektów oraz generowanie obwiedni odbywa się bezpośrednio tzn. bez użycia wstępnie wygenerowanych propozycji regionów. Metoda ta znacznie przyśpiesza działanie algorytmu, kosztem mniejszej dokładności wykrywania obiektów, szczególnie obiektów małych rozmiarów [2].

Na przestrzeni lat powstało kilka wersji architektury YOLO. Dla naj-

nowszej wersji YOLOv5 nie został opublikowany jeszcze oficjalny artykuł opisujący jej budowę. W związku z tym w niniejszej pracy opisana została wersja YOLOv4.

Architektura w wersji YOLOv4 jest bardzo złożona i może się składać z wielu różnych sieci neuronowych. Architektura przedstawiona została na Rysunku 1.10. Autorzy architektury jako najlepszy kompromis pomiędzy wydajnością a dokładnością przedstawiają podstawową wersję w konfiguracji: architektura CSPDarknet53 jako rdzeń służący do wykrywania cech w obrazie, architektury SPP oraz PANet do połączenia wykrytych map cech oraz architektury YOLOv3 do wykrywania i klasyfikowania obiektów z dostarczonych map cech [2].



Rys. 1.10. Podstawowa architektura YOLOv4. Źródło: [21]

1.7. Model sieci neuronowej

Model sieci neuronowej definiuje budowę takiej sieci opisując z ilu warstw składa się sieć, jaki jest jej rozmiar wejścia oraz wyjścia, opisuje warstwy ukryte, łączenie między warstwami, funkcje aktywacji poszczególnych warstw. Modelem może być schemat, opis słowny, klasa obiektu opisana za pomocą języka programowania, której obiekt może zostać zapisany do pliku. Plik zawierający model sieci neuronowej może zostać wczytany niwelując koniecz-

ność ponownego definiowania klasy modelu.

W przypadku mocno rozbudowanych architektur możliwość eksportu oraz importu takiego modelu znacznie ułatwia i przyśpiesza pracę. Użycie wcześniej zbudowanego modelu jako rdzenia (ang. backbone) w budowanej sieci, może okazać się skutecznym podejściem rozwiązania problemu. Takie podejście daje możliwość modyfikacji sieci np. poprzez dodanie, zmodyfikowanie lub usunięcie warstw, dostosowując sieć do konkretnego problemu. Budowanie sieci na podstawie innych sieci jest jedną z technik wykorzystywanych w nauczaniu metodą transferu (ang. transfer learning) [38].

1.8. Wstępnie wytrenowane modele sieci neuronowych

Uczenie sieci neuronowych od podstaw wymaga ogromnej ilości oznaczonych danych, odpowiedniego sprzętu komputerowego oraz czasu. Powszechnym i bardzo skutecznym podejściem do uczenia głębokiego na małych zestawach danych obrazu jest użycie wstępnie wytrenowanej sieci (ang. pretrained network) [3].

Wstępnie przeszkolona sieć to zapisany wstępnie wytrenowany model sieci, która została wcześniej przeszkolona na dużym zestawie danych, zwykle w zadaniu klasyfikacji obrazów na dużą skalę. Jeśli oryginalny zbiór danych jest wystarczająco duży i wystarczająco ogólny, wówczas zbiór wyuczonych cech przez wstępnie wytrenowaną sieć może skutecznie działać jako ogólny model, a zatem jej cechy mogą okazać się przydatne w wielu różnych problemach [3].

Taką wcześniej wytrenowaną sieć (wstępnie wytrenowany model), moż-

na dodatkowo douczyć własnym niedużym zestawem danych, który dokładniej opisuje analizowane cechy obrazu dla danego środowiska, zwiększając dokładność modelu w tym konkretnym problemie. Używanie wstępnie wytrenowanych modeli należy do technik nauczania metodą transferu [38].

1.9. Biblioteki sieci neuronowych

Na rynku dostępnych jest wiele bibliotek implementujących sieci neuronowe. Należą do nich np.: Tensorflow, Microsoft CNTK, Caffe, Theano, Torch i jej odmiana zaimplementowana w języku Python - PyTorch, Shogun, Sci-Kit Learn, Apache MXNet i wiele innych. Eksperymenty omawiane w pracy, przeprowadzone zostały przy użyciu biblioteki PyTorch.

Kod źródłowy 1.9.1 zawiera przykładową implementację klasy modelu sieci neuronowej w framework'u PyTorch. Zdefiniowany model posiada trzy warstwy. Warstwę wejściową **linear1** posiadającą 100 neuronów z funkcją aktywacji ReLu. Warstwa wejściowa przekazuje dane do warstwy ukrytej **linear2** posiadającej 50 neuronów. Wagi z warstwy **linear2** podlegają normalizacji do rozkładu prawdopodobieństwa przez funkcję aktywacji softmax i przekazywane są jako warstwa wyjściowa.

Kod źródłowy 1.9.1. Definicja klasy modelu sieci neuronowej

```
1  class TinyModel(torch.nn.Module):  
2  
3      def __init__(self):  
4          super(TinyModel, self).__init__()  
5  
6          self.linear1 = torch.nn.Linear(100, 50)  
7          self.activation = torch.nn.ReLU()  
8          self.linear2 = torch.nn.Linear(50, 10)  
9          self.softmax = torch.nn.Softmax()  
10
```

```
11     def forward(self, x):
12         x = self.linear1(x)
13         x = self.activation(x)
14         x = self.linear2(x)
15         x = self.softmax(x)
16
17         return x
```

1.10. Metody pomiaru błędu obserwacji

W omawianej pracy do oceniania dokładności badanych modeli wykorzystane zostało narzędzie ”Open-Source Visual Interface for Object Detection Metrics”¹. Narzędzie pozwala na wczytanie katalogu zawierającego podstawowe pliki adnotacji (ang. Ground truth), katalogu z badanymi obrazami, pliku z formatem zapisanych adnotacji, katalogu z adnotacjami wygenerowanymi przez detektor obiektów, czyli z danymi wyjściowymi narzędzia do przeprowadzania badań, pliku z formatem adnotacji wygenerowanymi przez detektor obiektów, wybór metryk oceny detekcji.

Narzędzie bazuje na metrykach oceny dokładności modelu takich jak: średnia precyza (ang. Average Precision, w skrócie AP), średnia z średnich precyzji (ang. mean Average Precision, w skrócie mAP), cześć wspólna do całości - Indeks Jaccarda (ang. Intersection over Union, w skrócie IoU) oraz innych metryk stworzonych na potrzeby modeli COCO [26]. Do celów badawczych wykorzystane zostały metryki AP oraz mAP.

Do adnotacji położenia obiektów najczęściej wykorzystuje się obwiednie (ang. bounding box) czyli pole ograniczające w formie prostokąta. Położenie ramki ograniczającej obliczane jest na podstawie najbardziej wysuniętych

¹Repozytorium narzędzia ”Open-Source Visual Interface for Object Detection Metrics” do oceniania dokładności modelu: https://github.com/rafaelpadilla/review_object_detection_metrics

pikseli należących do wykrytego obiektu. Pozycja górnej lewej krawędzi prostokąta zależy od pozycji piksela wysuniętego najbardziej na lewo (inaczej pozycja x_1) oraz od piksela najbardziej wysuniętego w górę od obiektu (inaczej pozycja y_1). Koordynat dolnej prawej krawędzi prostokąta zależy od pozycji piksela wysuniętego najbardziej na prawo (inaczej pozycja x_2) oraz od piksela najbardziej wysuniętego w dół od obiektu (inaczej pozycja y_2).

Pozycje wykrytych obiektów mogą zostać zapisane do pliku w formie tekstu zawierającego pozycje obu krawędzi pola ograniczającego lub zawierającego pozycję górnej lewej krawędzi oraz szerokość i wysokość pola ograniczającego. W przypadku wizualizacji położenia obiektu na obrazie, rysowany jest prostokąt na podstawie obliczonych koordynat. Na Rysunku 1.11 przedstawiona została przykładowa wizualizacja pozycji obiektu na obrazie za pomocą pola ograniczającego.



Rys. 1.11. Przykład wizualizacji położenia obiektu na obrazie za pomocą pola ograniczającego w formie czerwonego prostokąta. Źródło: Opracowanie własne na podstawie kadru ze zbioru danych dostarczonego przez klub sportowy KKS Lech Poznań

1.10.1. Indeks Jaccarda

Dokładność wykrywania obiektów przez model, polega na sprawdzeniu w jakim stopniu pole obwiedni wykrytego obiektu pokrywa się z podstawowym polem obwiedni obiektu (rzeczywista pozycja obiektu) (ang. Ground truth). Szacowanie odbywa się przy użyciu współczynnika podobieństwa Jaccarda. „Jest stosunkiem części wspólnej pola obiektu wyznaczonego przez model i pola obiektu wyznaczonego przez zbiór danych testowych do sumy tych pól.” [23]. Na Rysunku 1.12 zwizualizowany został wzór równania Indeksu Jaccarda.

$$\text{IoU} = \frac{\text{Area of Overlap} \text{ (obszar wspólny)}}{\text{Area of Union} \text{ (obszar całości)}}$$


Rys. 1.12. Wizualizacja równania IoU. Źródło: [28]

Na Rysunku 1.13 przedstawione zostały: obwiednia koloru niebieskiego opisująca podstawową pozycję obiektu oraz obwiednia koloru czerwonego opisującą oszacowaną pozycję obiektu przez model detekcji obiektów.



Rys. 1.13. Przykład wizualizacji rzeczywistego położenia obiektu względem położenia obiektu oszacowanego przez model wykrywania obiektów. Źródło: Opracowanie własne na podstawie kadru ze zbioru danych dostarczonego przez klub sportowy KKS Lech Poznań

1.10.2. Średnia precyzja (AP i mAP)

W większości przypadków na analizowanym obrazie, wykrywanych jest więcej niż jeden obiekt. To czy dany obiekt został prawidłowo wykryty zależy od wartości IoU oraz od ustawionego progu akceptacji (ang. threshold). Jeżeli część wspólna podstawowego pola ograniczającego i przewidzianego pola ograniczającego jest większa bądź równa ustawionemu progowi, wtedy taki obiekt uznawany jest za prawidłowo wykryty (ang. True Positive, w skrócie TP). Jeżeli wartość IoU jest mniejsza od progu, wtedy szacowanie uznawane jest za fałszywie pozytywne (ang. False Positive, w skrócie FP). W przypadku kiedy obiekt ze zbioru testowego nie zostanie wykryty, uznawany jest za fałszywie negatywny (ang. False Negative, w skrócie FN) [25].

Precyzja (ang. precision) szacowania jest to zdolność modelu do wykrywania prawdziwych obiektów z pośród wszystkich wykrytych obiektów i jest

wyrażona wzorem:

$$\text{Precision} = \frac{TP}{TP + TN} = \frac{TP}{\text{wszystkie detekcje}}$$

Czułość (ang. recall) szacowania jest to zdolność modelu do wykrywania prawdziwych obiektów z pośród wszystkich prawdziwych obiektów i jest wyrażona wzorem:

$$\text{Precision} = \frac{TP}{TP + FN} = \frac{TP}{\text{wszystkie podstawowe obiekty}}$$

Najczęściej stosowaną metryką do badania dokładności modelu w przypadku wykrywania obiektów jest średnia precyzja (AP). W tym celu w początkowej fazie należy wyznaczyć krzywą precyzji oraz czułości (ang. Receiver operating characteristic, w skrócie ROC). Ze względu na to, że używanie metryki prawdziwie negatywne (ang. True Negative, w skrócie TN) nie ma zastosowania w wykrywaniu obiektów na obrazie, ponieważ takich obiektów może być nieskończona ilość, zamiast metryki specyficzności stosuje się metrykę precyzji do obliczania krzywej. Krzywa precyzji oraz czułości są to inaczej krzywa wartości metryk precyzji oraz czułości dla wszystkich możliwych wartości poziomu ufności wykrytych obiektów. Poziom ufności oznacza procent pewności, że dany obiekt jest np. człowiekiem. Po wyznaczeniu krzywej precyzji oraz czułości obliczane jest pole powierzchni pod tą krzywą (ang. Area Under the ROC Curve, w skrócie AUC). Obliczone pole jest średnią precyzją (AP), w praktyce AP jest średnią precyzji dla wszystkich wartości czułości [25].

W przypadku zbioru obrazów średnia precyzja liczona jest dla wszystkich obrazów wspólnie. Oznacza to, że najpierw wczytane zostają wszystkie pola ograniczające obiektów z wszystkich obrazów i następnie dla tych wszystkich

wczytanych pól liczona jest średnia precyzja.

Średnia precyzja obliczana jest dla każdej klasy obiektu osobno tj. osobno zostanie policzona metryka AP dla obiektów klasy człowiek oraz osobno dla obiektów klasy piłka. Średnia wartości z wartości AP dla obu klas jest inaczej średnią mAP.

Rozdział 2

Problem i metoda badań własnych

Przedmiotem badań przeprowadzonych w ramach niniejszej pracy, są architektury detekcji obiektów (zawodników oraz piłki) takie jak: Faster R-CNN, Mask R-CNN, Panoptic FPN oraz YOLOv5.

Celem badań jest poznanie jak architektury poszczególnych metod wykrywania obiektów wpływają na dokładność wykrywania zawodników oraz piłki. Czy typ badanego obiektu (zawodnik lub piłka) ma wpływ na rezultaty. Czy jakość danych (materiałów wideo) dostarczonych do analizy ma wpływ na wynik. Czy douczenie modelu danymi szczególnie związanymi z piłką nożną przyczynia się do uzyskania lepszych rezultatów.

2.1. Problemy badawcze i hipotezy

W celu dokładnego określenia, przedstawienia i próby odpowiedzenia na problemy badawcze omawiane w tej pracy, przedstawione zostały pytania oraz

hipotezy dotyczące przeprowadzanego eksperymentu.

Podstawowe pytanie:

1. Czy dana architektura ma wpływ na dokładność wykrywania obiektów (zawodników oraz piłki)?

Dana architektura ma wpływ na dokładność wykrywania obiektów.

Pytania dopełnienia:

1. Czy typ badanego obiektu ma wpływ na dokładność rozpoznawania obiektów (zawodników oraz piłki)?

Typ badanego obiektu ma wpływ na dokładność rozpoznawania obiektów.

2. Czy jakość danych dostarczonych do analizy ma wpływ na dokładność wykrywania obiektów (zawodników oraz piłki)?

Jakość danych ma wpływ na dokładność wykrywania obiektów.

3. Czy douczenie modelu danymi szczególnie związanymi z piłką nożną ma wpływ na dokładność wykrywania obiektów?

Douczenie modelu ma wpływ na dokładność wykrywania obiektów.

2.2. Plan i etap badań

Przed przystąpieniem do badań opracowana została strategia podejścia rozwiązania postawionych problemów badawczych. W pierwszym etapie określone zostało jakie architektury, będą podlegały analizie. W kolejnym etapie

uwaga została skupiona na odnalezieniu odpowiednich implementacji badanych architektur. Następnie określony został język programowania oraz środowisko w jakim będzie budowane narzędzie niezbędne do przeprowadzenia eksperymentu. Czwarty etap strategii dotyczył gromadzenia i generowania danych koniecznych do przeprowadzenia badania. W piątym etapie wybrane zostało narzędzie do analizy i prezentacji wyników. W ostatnim etapie określony został sposób realizacji badań: jak powinny zostać przygotowane dane przeznaczone do analizy, jak narzędzie badawcze powinno działać, w jaki sposób mają być prezentowane rezultaty.

2.2.1. Architektury i platformy wykrywania obiektów

Omawiany problem badawczy w głównej mierze skupia się na przetestowaniu różnego rodzaju architektur wykrywania obiektów. W tym celu przed przystąpieniem do realizacji badań, konieczne było określenie jakie są dostępne architektury oraz do jakich architektur został udostępniony kod źródłowy oraz biblioteki niezbędne do przeprowadzenia analizy.

Po zapoznaniu się listą dostępnych architektur oraz platform wykrywania obiektów wyselekcjonowane zostały architektury takie jak: Faster R-CNN, Mask R-CNN, Panoptic FPN oraz YOLOv5. Zostały one wybrane ze względu na ich większą popularność, specyfikę budowy, większą dokładność szacowania, szybkość działania oraz przede wszystkim dostępność kodu źródłowego umożliwiającą bezproblemową implementację i uruchomienie w narzędziu badawczym.

Podczas szukania implementacji wyżej wymienionych architektur, odnaleziona została platforma Detectron2 [34]. Platforma Detectron2 została stworzona przez firmę Facebook i jest następną generacją platformy Detectron.

Platforma umożliwia detekcję obiektów między innymi w postaci: obwiedni, maski oraz segmentacji. Posiada ona już wbudowane implementacje architektur: Faster R-CNN, Mask R-CNN oraz Panoptic FPN. Dodatkowo firma udostępnia zbiór ”Detectron2 Model Zoo”¹ - wstępnie wytrenowanych modeli sieci neuronowych.

W przypadku architektury YOLOv5 również dostępna jest platforma zawierająca jej implementację². Autorzy również udostępniają wstępnie wytrenowane modele dla tej architektury³.

Odnalezienie przedstawionych platform przyśpieszyło implementację narzędzia badawczego. W znacznym stopniu zmniejszyło problem gromadzenia i generowania danych potrzebnych do uczenia modeli opartych na wyżej wymienionych architekturach. Pozwoliło w większej mierze skupić się na odpowiednim i szerszym przeprowadzeniu badań.

2.2.2. Narzędzie badawcze

Do przeprowadzenia badań stworzone zostało autorskie narzędzie ODATT napisane w języku Python. Oprogramowanie zostało zaprojektowane w taki sposób aby umożliwiało uruchomienie analizy zbioru danych z materiału video lub katalogu zawierającego obrazy (w tym wypadku wyodrębnione klatki) przy użyciu platformy Detectron2 oraz platformy YOLOv5 bez konieczności tworzenia nadmiarowego kodu oraz uruchamiania osobnych skryptów. W tym celu stworzona została klasa abstrakcyjna ”Detector” narzucająca format przyjmowanych parametrów i obiektów oraz zwracająca obiekty w odpowiednim formacie wykorzystywanym do analizy wyników.

¹Detectron2 Model Zoo: https://github.com/facebookresearch/detectron2/blob/main/MODEL_ZOO.md

²YOLOv5: <https://github.com/ultralytics/yolov5>

³YOLOv5: <https://github.com/ultralytics/yolov5/releases>

W kodzie źródłowym 2.2.1 przedstawiony został kod źródłowy klasy abstrakcyjnej **Detector**, narzucającej format przyjmowanych, zwracanych danych oraz logiki działania danego detektora obiektów. W przypadku definiowania obiektu dziedziczącego tą klasę, wymagane jest ustawienie progu akceptacji wykrywania danego obiektu (**model_threshold**) oraz ścieżkę lub nazwę do modelu danej architektury wykrywania obiektów (**model_path**). W metodzie `__init__` możliwe jest zainicjalizowanie architektury wykrywania obiektów oraz dostosowanie parametrów do własnych potrzeb z zachowaniem dwóch odgórnie ustalonych parametrów. Metoda **detect_objects** musi zawierać implementacje całego mechanizmu wykrywania obiektów przez daną architekturę. Dodatkowo musi przyjmować jako parametr obraz w postaci macierzy (**image**) oraz tablicę identyfikatorów klas obiektów, które mają być wykrywane (**classes**) np. człowiek (identyfikator: 0) oraz piłka (identyfikator: 32). Funkcja musi również zwracać listę obiektów typu **DetectedObject**. Zastosowanie klasy abstrakcyjnej pozwoli określić typy danych, dzięki czemu w przypadku dodania nowej platformy detekcji obiektów, nie będzie trzeba dostosowywać całego kodu narzędzia, co czyni narzędzie ODATT bardziej uniwersalne i podatne na dalszy rozwój.

Kod źródłowy 2.2.1. Definicja abstrakcyjnej klasy detektora obiektów

```
1  from abc import ABC, abstractmethod
2
3  from detected_object import DetectedObject
4
5  class Detector(ABC):
6
7      @abstractmethod
8
9          def __init__(self, model_threshold, model_path):
10
11              """
12
13                  Args:
14
15                      model_threshold (float): Minimum score for instance prediction
16
17                      to be shown.
18
19                          model_path (str): Set model of object detector
20
21              """
22
```

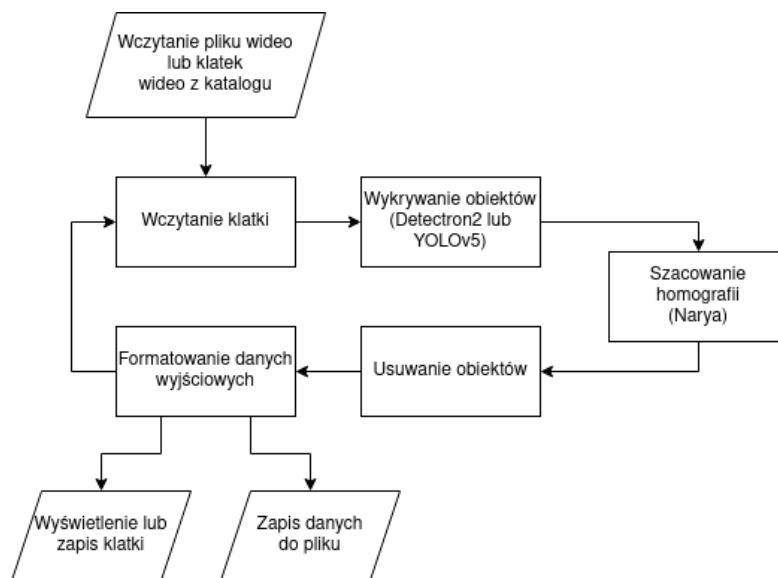
```
12     pass
13
14     @abstractmethod
15     def detect_objects(self, image, classes = [0, 32]) -> [DetectedObject]:
16         """
17         Args:
18             image (Mat): cv2.imread() or any array with image with
19             compatible format like cv2
20             classes (list): List of object class identifiers that will be
21             detected
22             Returns:
23                 list: List of DetectedObject objects
24         """
25     pass
```

Uruchomienie narzędzia wymaga odpowiedniego środowiska wykonawczego opartego na menadżerze pakietów i systemie zarządzania środowiskiem Conda⁴. Do uruchomienia narzędzia wymagany jest system operacyjny - Linux z odpowiednimi bibliotekami oraz sprzętem komputerowym składającym się między innymi z karty graficznej NVIDIA o minimalnej pojemności pamięci 6 GB. Dokładna procedura i wymagania uruchomienia narzędzia opisane zostały w ogólnie dostępnym repozytorium kodu narzędzia badawczego dostępnym pod adresem: <https://github.com/JN426206/ODATT>.

Jak pokazano na Rysunku 2.1, narzędzie w początkowej fazie działania, wczytuje materiał wideo lub klatki wideo ze ścieżki przekazanej jako parametr programu. Następnie tworzony jest obiekt klasy "Detector" zawierający implementację platformy Detectron2 lub YOLOv5 w zależności od ustalionych parametrów programu. Klatki wideo zanim zostaną poddane analizie są odpowiednio skalowane do rozmiaru wejścia sieci neuronowej oraz konwertowane do przestrzeni kolorów HSV. Detekcja obiektów oraz szacowanie obszaru boiska odbywa się na każdej klatce wideo po kolej. Platformy zostały

⁴Conda - Menadżer pakietów i system zarządzania środowiskiem: <https://docs.conda.io/en/latest/>

tak skonfigurowane aby wykrywały tylko ludzi oraz piłki. W celu ograniczenia obszaru wykrywanych obiektów do zawodników oraz piłki znajdujących się tylko na polu boiska, wykorzystana została platforma narya⁵.



Rys. 2.1. Diagram przedstawiający przepływ danych w narzędziu ODATT. Źródło: Opracowanie własne

Platforma Narya bazuje na dwóch splotowych sieciach neuronowych. Pierwsza sieć neuronowa bazuje na architekturze EfficientNET [31] i służy do wykrywania linii boiska. Druga sieć neuronowa została zbudowana na architekturze ResNet [13] i wykrywa punkty kluczowe boiska. Sieć ResNet osiąga lepszą dokładność szacowania homografii (pola boiska oraz macierzy przekształcenia perspektywicznego) ale wymaga minimum czterech punktów kluczowych boiska, które nie zawsze są widoczne ze względu na różne ujęcia kamery. Punktami kluczowymi mogą być np. rogi boiska lub punkty przecięcia się linii boiska. W przypadku kiedy liczba wykrytych punktów kluczowych jest zbyt niska, wykorzystywana jest sieć EfficientNET. Sieć EfficientNET

⁵Platforma Narya - wykrywanie obszaru boiska: <https://github.com/DonsetPG/narya>

bazuje na bezpośrednim wykrywaniu pola boiska z cech obrazu, co powoduje, że jest w stanie wykryć pole boiska w większej ilości przypadków niż sieć ResNet ale z mniejszą dokładnością [8].

Narzędzie ODATT po usunięciu obiektów nie należących do obszaru boiska (kibice, sędziowie liniowi, trenerzy, zawodnicy rezerwowi itp.) zapisuje pozycje wykrytych zawodników oraz piłki do pliku tekstowego w odpowiednim formacie, umożliwiającym analizę dokładności działania algorytmu detekcji obiektów w programie "Open-Source Visual Interface for Object Detection Metrics"⁶.

2.2.3. Gromadzenie i generowanie danych

Do przeprowadzenia badań niezbędne są odpowiednie zbiory danych. W tym przypadku są to nagrania lub wyodrębnione klatki z meczów piłki nożnej. W celu przeprowadzenia dokładnych badań, materiały te powinny być odpowiedniej jakości. Powinny zawierać nagrania różnych sytuacji taktycznych rozgrywanych podczas trwania meczu. Powinny podlegać wstępнемu przetworzeniu w celu usunięcia fragmentów nagrani nie reprezentujących sytuacji na boisku na przykład reklam lub powtórek. W przypadku uczenia modeli sieci neuronowych konieczne jest posiadanie oznaczonych danych, które są trudne do zdobycia, szczególnie za darmo lub wymagają podpisania odpowiedniej umowy. Oznaczone dane powinny zostać zweryfikowane w kwestii ich dokładności oraz szczegółowości informacji i należy stwierdzić czy mogą zostać użyte jako zbiór podstawowy. Materiały te można uzyskać z różnych źródeł. Istnieją serwisy, które udostępniają nagrania za darmo jednak bardzo rzadko serwisy te udostępniają oznaczone dane.

⁶Open-Source Visual Interface for Object Detection Metrics: https://github.com/rafaelpadilla/review_object_detection_metrics

Powszechnie dostępnym zasobem udostępniającym same nagrania, jest darmowy serwis internetowy YouTube⁷. Udostępnia on sporą liczbę nagrani z meczów piłki nożnej. Najczęściej jednak są to tylko skróty meczowe, co ogranicza możliwości analizowania różnych sytuacji na boisku. Do wad mogą należeć również: w niektórych przypadkach jakość udostępnianego materiału, która jest zbyt niska do celów analizy, brak wszystkich spotkań, wątpliwości co do legalności udostępnionego materiału, brak oznaczonych danych.

Badania przeprowadzane w ramach tej pracy odbywały się we współpracy z klubem sportowym Lech Poznań. W ramach tej współpracy otrzymany został pełen dostęp do zasobu z nagraniem meczów z udziałem klubu Lech Poznań. Jedną z udostępnionych platform była platforma wyscout. Udostępnione materiały video na rzecz współpracy były w jakości FullHD (rozdzielcość 1920x1080 px) oraz HD (1280x720 px). Materiały zawierały zapis z całego meczu, posiadały różne ujęcia kamer, dodatkowo udostępnione zostały materiały zarejestrowane przy użyciu kamery taktycznej obejmującej co najmniej połowę boiska, udostępnione zostały fragmenty nagrani z oznaczeniem typu akcji (podanie, gol, rzut karny itp.) oraz jacy zawodnicy brali w niej udział. Zasób nie zawierał oznaczonych danych, które mogłyby zostać użyte jako dane podstawowe do badań oraz uczenia modeli.

Zasobami zawierającymi oznaczone dane były: SoccerDB⁸ oraz SoccerNet⁹ [5]. Baza SoccerDB zawiera około 120 oznaczonych nagrani w jakości FullHD. Dane zawierają informację o obwiedni zawodnika oraz piłki dla co trzeciej klatki. Niestety format zapisu oznaczonych danych nie jest synchronizowany według klatki tylko według czasu, co powoduje, że oznaczone dane

⁷Serwis internetowy - YouTube: <https://www.youtube.com/>

⁸SoccerDB - baza danych oznaczonych nagrani meczów piłki nożnej: <https://github.com/newsdata/SoccerDB>

⁹SoccerNet - baza danych oznaczonych nagrani meczów piłki nożnej: <https://www.soccer-net.org/>

nie są w pełni dokładne, przez co nie mogły zostać użyte do uczenia oraz analizy modeli sieci. Baza SoccerDB dodatkowo zawiera około 45 000 oznaczonych zdjęć w rozdzielczości HD. W tym przypadku dane są dokładne i mogą zostać użyte do uczenia oraz analizowania modeli sieci neuronowej. Zasób SoccerNet zawiera około 550 oznaczonych nagrań. Dane zawierają informację o obwiedni zawodnika oraz piłki, typu przeprowadzanej akcji oraz obszaru boiska dla co 12. lub 13. klatki. Dane zostały zapisane według klatki co oznacza, że są synchronizowane z nagraniem i mogą zostać użyte w celu uczenia oraz analizowania modelu sieci neuronowej. Oba zasoby abytrzymać dostęp do materiałów, wymagają podpisania umowy NDA.

Alternatywnym sposobem na uzyskanie nagrań z meczów piłki nożnej jest wygenerowanie danych syntetycznych. W tym celu można wykorzystać otwarto źródłowe narzędzie o nazwie "Google Research Football"¹⁰ stworzone przez firmę Google. Narzędzie symuluje rozgrywkę piłki nożnej przy użyciu graficznych modeli 3D. Dostępny kod źródłowy umożliwia modyfikację programu tak aby zwracał dane zawierające pozycję i obwiednię zawodników oraz piłki. Zaletą takiego narzędzia jest łatwy sposób i szybkość generowania danych [19]. Jednak są to dane syntetyczne i mogą dobrze nie odzwierciedlać realnego obrazu, co może negatywnie wpłynąć na uczenie sieci, co za tym idzie taka sieć może osiągać słabsze wyniki w środowisku analizowania nagrań zawierającego naturalne obiekty.

Najbardziej pracochłonnym oraz czasochłonnym sposobem zdobycia oznaczonych danych jest manualne oznaczenie danych. Dokładne oznaczenie danych manualną metodą wymaga sporej ilości czasu oraz precyzji. W zamian za to, samemu decyduje się jak dokładne dane chce się osiągnąć oraz ile tych danych otrzymamy. Istnieje wiele narzędzi wspomagających oznaczanie ta-

¹⁰Google Research Football - narzędzie symulujące rozgrywkę meczu piłki nożnej:
<https://github.com/google-research/football>

kich danych. Przykładowym narzędziem może być narzędzie DarkLabel¹¹. Narzędzie to pozwala na szybkie oznaczenie dużej ilości danych dzięki wbudowanej funkcji śledzenia i wykrywania obiektów. W przypadku zaznaczenia obiektów na klatce, narzędzie automatycznie wykrywa i oznacza obiekty na następnej klatce na podstawie zaznaczonych obiektów z poprzedniej klatki. Automatycznie wykryte obiekty można łatwo i szybko korygować.

2.2.4. Douczenie wstępnie wytrenowanych modeli sieci neuronowych

Modele wstępnie wytrenowane umożliwiają wykrywanie obiektów różnego typu (ludzi, zwierzęta, pojazdy, przedmioty). Ze względu na to, że modele te uczone są na bardzo urozmaiconych zbiorach danych, mogą osiągać mało satysfakcjonujące wyniki wykrywania obiektów w specyficznych warunkach, takich jak na przykład murawa boiska. W celu poprawienia wyników detekcji obiektów możliwe jest dotrenowanie takich modeli dostosowując je do środowiska i zwiększając ich precyzję detekcji.

Model sieci neuronowej wstępnie wytrenowanej za pomocą dużego zbioru danych posiada dużą liczbę użytecznych cech wykrywania obiektów. Technika nauczania metodą transferu pozwala zainicjalizować sieć neuronową wagami z modelu wstępnie wytrenowanego, zamrozić większą część warstw i douszyć niewielkim zbiorem danych jedną lub kilka ostatnich warstw sieci zwiększając jej dokładność wykrywania obiektów w danym środowisku.

¹¹DarkLabel - narzędzie do oznaczania materiałów wideo: <https://github.com/darkpgmr/DarkLabel>

2.2.5. Zapisywanie danych i prezentacja wyników

Narzędzie opracowane do przeprowadzania badań, generuje wynikowe pliki tekstowe zawierające niezbędne dane umożliwiające pomiar dokładności badanych architektur oraz materiałów wideo. W przypadku analizowania katalogu z obrazami jest to osobny plik dla każdego obrazu. W przypadku pliku wideo generowany jest osobny plik dla każdej klatki. Wynikowe plik w każdej linii zawierają dane takie jak: typ wykrytego obiektu, dokładność szacowania typu obiektu, obwiednia zawodnika lub piłki zapisana w postaci koordynatów górnego lewego punktu oraz szerokości i wysokości obwiedni. Dane te oddzielone są znakiem spacji tworząc kolumny. Poniżej przedstawiony został nagłówek opisujący każdą kolumnę:

```
object_class score bb_x bb_y bb_width bb_height
```

Znaczenie poszczególnych kolumn:

object_class - typ obiektu przyjmujący wartość 0 w przypadku zawodnika lub 1 w przypadku piłki

score - dokładność szacowania typu obiektu

bb_x - pozycja x górnego lewego punktu obwiedni

bb_y - pozycja y górnego lewego punktu obwiedni

bb_width - szerokość obwiedni

bb_height - wysokość obwiedni

Dane opisujące obwiednie nie są normalizowane a więc są zależne od rozmiaru przetwarzanego materiału. W celu prawidłowego odtworzenia danych wynikowych, konieczne jest posiadanie informacji jakiej rozdzielczości był przetwarzany materiał.

Plik wynikowy został zapisany z nazwą odpowiadającą nazwie przeanalizowanego obrazu.

zowanemu obrazowi lub zawierającej numer klatki wideo przeanalizowanego materiału wideo. Poniżej przedstawiona została przykładowa zawartość pliku:

```
player 1.0 267 735 90 96
player 1.0 945 273 30 76
player 1.0 282 671 36 105
player 1.0 1617 551 36 99
ball 0.99 1609 443 13 14
player 0.99 1758 384 33 85
...
...
```

2.3. Metodyka badań

2.3.1. Przygotowanie zbiorów danych

Do przeprowadzenia badań wykorzystane zostały cztery zbiory danych. Każdy zbiór posiada wyodrębnione klatki z nagrani wideo meczu piłkarskiego:

1. Zbiór danych zawierający 240 klatek wideo w rozdzielczości Full HD (1920 px na 1080 px) pochodzący z materiału wideo o przepływności 7799 kb/s [9]. Materiał zawierał zapis wideo jednego spotkania piłkarskiego pochodzącego z bazy zbiorów SoccerNet.

W pierwszym etapie, materiał wideo został przekonwertowany za pomocą programu FFmpeg¹² w celu ustawienia prawidłowej synchronizacji klatek i usunięcia poziomych linii mogących mieć wpływ na dokładność wykrywania obiektów. Zachowana została oryginalna rozdzielcość

¹²FFmpeg - narzędzie do nagrywania, konwertowania i przesyłania strumieniowego treści audiowizualnych: <https://ffmpeg.org/>

oraz przepływność materiału wideo. Następnie wyodrębnione zostały klatki wideo zgodnie z dostępnymi oznaczeniami danych czyli co 13. klatkę wideo. Zbiór został zweryfikowany i usunięte zostały klatki o słabej jakości oznaczeń. Zbiór ten został nazwany jako "20161019 BM FHD".

Zbiór "20161019 BM FHD 1k" zawiera klatki z wideo o zmniejszonej przepływności do 1000 kb/s z zachowaniem oryginalnej rozdzielczości.

Zbiór "20161019 BM FHD 100" zawiera klatki z wideo o zmniejszonej przepływności do 100 kb/s z zachowaniem oryginalnej rozdzielczości.

Na Rysunku 2.2 przedstawione zostały kadry dla różnych przepływności materiału wideo. Między Rysunkiem 2.2a a Rysunkiem 2.2b nie widać dużej różnicy w jakości materiału wideo. W przypadku Rysunku 2.2c widać znaczny spadek jakości obrazu, co może mieć znaczenie dla dokładności wykrywania obiektów przez badane architektury.



Rys. 2.2. Kadry dla różnych przepływności materiału wideo. Kadr (a) prezentuje oryginalną przepływność materiału wideo. Kadr (b) przedstawia kadr z materiału wideo o przepływności 1000kb/s. Kadr (c) przedstawia kadr z materiału wideo o przepływności 100 kb/s. Źródło: Kadry pochodzą z materiału wideo ze zbioru danych SoccerNet.

Zbiór "20161019 BM HD" zawiera klatki w rozdzielczości 1280x720 przeskalowane ze zbioru "20161019 BM FHD".

Zbiór "20161019 BM SD" zawiera klatki w rozdzielczości 720x405 przeskalowane ze zbioru "20161019 BM FHD".

Zbiory wymienione powyżej zostały użyte do zbadania wpływu jakości materiału wideo na dokładność wykrywania obiektów przez algorytmy detekcji obiektów.

2. Zbiór "LP_GZ_HOM HD" zawiera ujęcia (136 klatek) z kamery taktycznej z jednego z meczów klubu sportowego KKS Lech Poznań, została wybrana co 1000. klatka wideo. Na każdej klatce zostały wstępnie wykryte obiekty za pomocą narzędzia badawczego a następnie ręcznie zweryfikowane i skorygowane przy użyciu narzędzia DarkLabel. Za pomocą specjalnie stworzonego skryptu "convert_labels.py", dane zostały przekształcone do formatu zgodnego z narzędziem do analizy i badania metryk. Materiał wideo został dostarczony przez klub sportowy KKS Lech Poznań.
3. Zbiór "SoccerDB HD" zawiera 208 klatek z jednego meczu ze zbioru SoccerDB. Klatki wideo zostały wyodrębnione z pierwszej połowy meczu w różnych odstępach czasowych. Dodatkowo zbiór został ręcznie zweryfikowany pod kątem dokładności oznaczeń. Oznaczenia niskiej dokładności zostały usunięte.
4. Zbiór "SoccerDB2 HD" zawiera 1057 klatek z trzech meczów ze zbioru SoccerDB i nie zawiera klatek ze zbioru "SoccerDB HD". Zbiór nie został zweryfikowany pod kątem jakości oznaczeń i został wykorzystany do zbadania dokładności modeli douczonych.

Oznaczenia ze zbioru "SoccerDB HD" oraz "SoccerDB2 HD" zostały przekształcone do formatu zgodnego z narzędziem do analizy i badania metryk za pomocą specjalnie stworzonego skryptu "convert_labels.py". Dane zostały przekształcone z formatu znormalizowanego i zapisanego w postaci:

```
object_class nbb_cx nbb_cy nbb_width nbb_height
```

do formatu w którym ostatecznie zapisane zostały oznaczenia wszystkich zbiorów:

```
object_class bb_x bb_y bb_width bb_height
```

Znaczenie poszczególnych kolumn dla oryginalnych oznaczeń zbiorów "SoccerDB HD" oraz "SoccerDB2 HD":

object_class - typ obiektu przyjmujący wartość 0 w przypadku zawodnika lub 1 w przypadku piłki

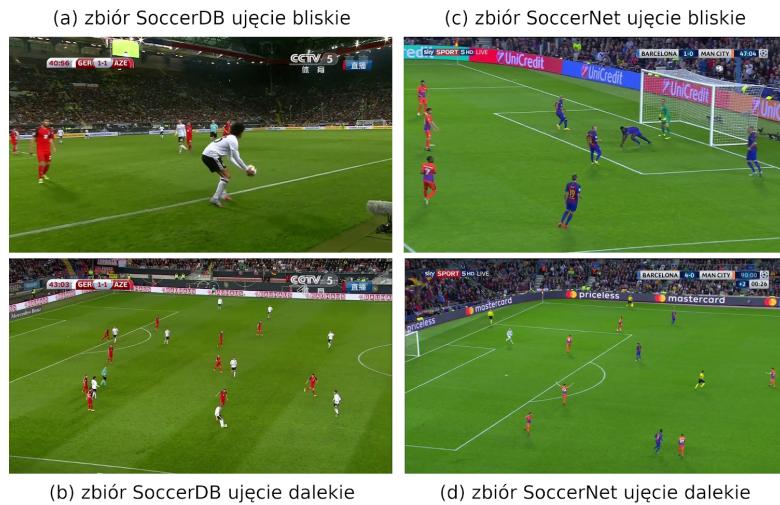
nbb_cx - znormalizowana pozycja x centralnego punktu obwiedni

nbb_cy - znormalizowana pozycja y centralnego punktu obwiedni

nbb_width - znormalizowana szerokość obwiedni

nbb_height - znormalizowana wysokość obwiedni

Zbiory pierwszy, trzeci i czwarty zawierają ujęcia z różnych pozycji kamery, oznacza to, że zawodnicy oraz piłka w niektórych przypadkach są większych lub mniejszych rozmiarów jak przedstawiono na Rysunku 2.3. W przypadku zbioru drugiego odległość pozycji kamery od zawodników jest stała i większa jak przedstawiono na Rysunku 2.4. Z tego powodu zawodnicy oraz piłka są mniejszych rozmiarów co może mieć wpływ na ich dokładność wykrywania.



Rys. 2.3. Różne pozycje kamery dla zbioru SoccerDB oraz zbioru SoccerNet. Jak przedstawiono na kadrze (a) i (c) rozmiar zawodników oraz piłki jest większy ze względu na bliską pozycję kamery. Na kadrze (b) oraz (d) zawodnicy oraz piłki są mniejszych rozmiarów ze względu na bardziej oddaloną pozycję kamery. Źródło: Kadry ze zbioru danych SoccerDB oraz SoccerNet



Rys. 2.4. Ze względu na stałą odległość pozycji kamery dla nagrania z meczu klubu sportowego Lech Poznań, zawodnicy oraz piłka są mniejszych rozmiarów. Źródło: Kadr z nagrania meczu klubu sportowego Lech Poznań

2.3.2. Uczenie modeli detekcji obiektów

Do uczenia modeli wykorzystane zostały zbiory "20161019 BM HD", "LP_GZ_HOM HD", "SoccerDB HD" z których powstał przetasowany zbiór "dataset_shuffle" zawierający 467 klatek dla zbioru trenującego i 117 dla zbioru testującego.

Na potrzeby douszczania modeli stworzony został skrypt "train_detectron2.py" bazujący na platformie Detectron2. Skrypt umożliwia wybranie dowolnego modelu dostępnego w platformie Detectron2, ustawienie parametrów uczenia między innymi takich jak: rozmiar paczki definiującej ilość równolegle przetwarzanych obrazów na dany krok uczenia (ang. batch size), współczynnik uczenia (ang. learning rate), liczbę epok uczenia oraz inne parametry uczonego modelu. Platforma Detectron2 zaleca dostarczenie danych uczących i testujących w formacie zgodnym z COCO¹³, w tym celu został stworzony skrypt "create_JSON_COCO_file.py" przekształcający dane z formatu zgodnego z narzędziem do analizy do formatu COCO.

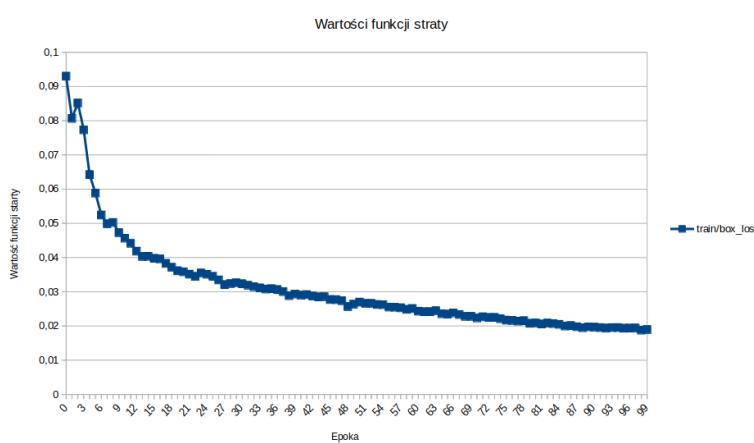
W przypadku uczenia modelu "YOLOv5x6" wykorzystany został skrypt "convert_labels.py" do przetworzenia oznaczonych danych na format zgodny z formatem YOLO. Do trenowania modelu wykorzystany został skrypt "train.py" dostarczony wraz z platformą YOLOv5.

Zgodnie z techniką uczenia metodą transferu, zarówno w przypadku modeli pochodzących z platformy Detectron2 oraz dla modelu "YOLOv5x6" zamrożone zostały pewne wagi, dla początkowych warstw. W przypadku modelu "YOLOv5x6" zamrożonych zostało dziewięć początkowych warstw, natomiast dla modeli z platformy Detectron2 zamrożone zostały dwie pierwsze warstwy, a w kolejnych wagi ulegały aktualizacji.

Na Rysunku 2.5 przedstawiony został wykres zachowania wartości funkcji

¹³COCO - format zapisu wykrytych obiektów <https://cocodataset.org/#format-data>

straty podczas trenowania modelu "YOLOv5x6" dla dokładności wykrywania obwiedni zawodników oraz piłki. Model był trenowany przez 100 epok ze współczynnikiem uczenia równym 0,01. Z obserwacji wynika, że wartości funkcji straty od 90. epoki zaczynały się stabilizować, co oznacza, że model osiągał już duże dopasowanie do zbioru uczącego. Dalsze kontynuowanie uczenia mogłoby skutkować przeuczeniem modelu.



Rys. 2.5. Wykres zachowania wartości funkcji straty dla obwiedni, podczas treningu modelu YOLOv5x6. Źródło: Opracowanie własne

2.3.3. Przeprowadzenie badań

Do badania wykorzystane zostało specjalnie stworzone narzędzie ODATT zawierające implementację algorytmów detekcji obiektów takich jak Faster R-CNN, Mask R-CNN, Panoptic FPN, Large-MASK-RCNN, YOLOv5 oraz zawierający implementację algorytmu wykrywania pola boiska Narya. Narzędzie umożliwiało wybranie odpowiedniego algorytmu detekcji obiektów, źródła danych (na potrzeby badania były to katalogi zawierające wyodrębnione klatki z materiałów wideo) oraz możliwość ustawienia parametru progu akceptacji (ang. threshold) detekcji obiektów. Narzędzie zwracało wyniki w

podanej lokalizacji zgodne z formatem narzędzia do analizy i badania metryk opisanym w rozdziale 1.10.

Po odpowiednim przygotowaniu zbiorów testujących oraz zbioru trenującego, za pomocą narzędzia ODATT badaniu poddane zostały modele:

Wstępnie wytrenowane:

Faster R-CNN R50 FPN 3x - Model bazujący na architekturze Faster R-CNN oraz sieciach neuronowych wykrywania obiektów ResNet [13] w połączeniu z siecią FPN. Sieć ma 50 warstw ukrytych i była szkolona według potrójnego harmonogramu co według twórców platformy Detectron2 oznacza około 32. epoki.

Faster R-CNN X101 32x8d FPN 3x - Model bazujący na architekturze Faster R-CNN oraz sieciach neuronowych wykrywania obiektów ResNeXt [36] w połączeniu z siecią FPN. Sieć ma 101 warstw ukrytych, kardynalność (wielkość zbioru przekształceń) [36] równą 32, szerokością wąskiego gardła równą osiem i była szkolona według potrójnego harmonogramu.

Mask R-CNN X101 32x8d FPN 3x - Model bazujący na architekturze Mask R-CNN oraz sieciach neuronowych wykrywania obiektów ResNeXt w połączeniu z siecią FPN. Sieć ma 101 warstw ukrytych, kardynalność równą 32, szerokością wąskiego gardła równą osiem i była szkolona według potrójnego harmonogramu.

Cascade Mask R-CNN X152 32x8d FPN IN5k - Model bazujący na architekturze Mask R-CNN oraz sieciach neuronowych wykrywania obiektów ResNeXt w połączeniu z siecią FPN. Sieć ma 152. warstwy ukryte, kardynalność równą 32, szerokością wąskiego gardła równą osiem i była szkolona na dużym zbiorze ImageNet-5k. Specjalnie przygotowany model dla uzyskania jak najlepszej dokładności wykrywania obiektów.

Panoptic FPN R101 3x - Model bazujący na architekturze Panoptic FPN oraz sieciach neuronowych wykrywania obiektów ResNet w połączeniu z siecią FPN. Sieć ma 101 warstw ukrytych i była szkolona według potrójnego harmonogramu.

YOLOv5x6 - Model bazuje na architekturze YOLOv5 v6.0 składającej się z sieci neuronowych CSP-Darknet53, SPPF, CSP-PAN oraz YOLOv3 Head [10][35]. Rozszerzenie "x" oznacza, stopień skalowania szerokości, głębokości oraz rozdzielczości sieci. Oznaczenie "x" oznacza sieć z największą szerokością, głębokością oraz rozdzielczością sieci zapewniając największą dokładność modelu z pośród dostępnych modeli [32]. Cyfra "6" oznacza, że użyty model jest w wersji YOLOv5 v6.0.

Douczone:

Modele uczone były w różnych konfiguracjach liczby iteracji oraz liczbie równolegle przetwarzanych obrazów. Ostatecznie wybrane zostały modele osiągające najlepsze wyniki dla obu zbiorów testujących ("dataset_shuffle" oraz "SoccerDB2 HD").

Faster R-CNN R50 FPN 3x - model uczony z parametrami: współczynnik uczenia: 0.0125 batch size: 8 liczba iteracji: 1500

Faster R-CNN X101 32x8d FPN 3x - model uczony z parametrami: współczynnik uczenia: 0.0125 batch size: 3 liczba iteracji: 1500

Cascade Mask R-CNN X152 32x8d FPN IN5k - model uczony z parametrami: współczynnik uczenia: 0.0125 batch size: 1 liczba iteracji: 1500

YOLOv5x6 - model uczony z parametrami: współczynnik uczenia: 0.01 batch size: 6 liczba iteracji: 7783

2.4. Podsumowanie

Odpowiednie przygotowanie planu i strategii badawczej. Zdobycie niezbędnych materiałów - kodów źródłowych architektur, bibliotek, wstępnie wytrenowanych modeli sieci neuronowych oraz gotowych implementacji wykorzystywanych platform. Zgromadzenie materiału badawczego. Stworzenie narzędzia badawczego ODATT. Wygenerowanie danych do analizy. Wykorzystanie oprogramowania do interpretacji wyników, pozwoliło dokładnie przeprowadzić badania i uzyskać odpowiedzi na postawione problemy badawcze.

Rozdział 3

Prezentacja i analiza wyników badań własnych

3.1. Analiza wyników badań

Analiza wyników badań została podzielona na cztery sekcje. Każda sekcja dotyczy danego problemu. Pierwsza sekcja przedstawia wyniki dotyczące wpływu typu architektury na dokładność wykrywania obiektów. Druga sekcja dotyczy analizy dokładności rozpoznawania typu obiektu przez daną architekturę. W trzeciej sekcji przedstawione zostały wyniki dokładności szacowania danej architektury w zależności od jakości przetwarzanego materiału. Sekcja trzecia została podzielona na dwie podsekcje w celu osobnego opisania wpływu rozdzielczości oraz przepływności materiału wideo na wyniki. Ostatnia - czwarta sekcja odpowiada na pytanie czy doczelenie modelu architektury wpływa na dokładność wykrywania obiektów. Najlepsze wyniki w tabelach w celu uwidocznienia, zostały zapisane pogrubioną czcionką. Wyniki zostały zapisane przy użyciu metryk średniej precyzji (AP) oraz średniej z średnich

precyzji (mAP). Metryki zostały opisane w rozdziale 1.10.2.

3.1.1. Dokładność wykrywania obiektów danej architektury

Celem badania było stwierdzenie czy algorytm oraz budowa danej architektury mają wpływ na dokładność detekcji zawodników i piłki. Badaniu podległy opisane w rozdziale 2.3.3 architektury: ”Faster R-CNN R50 FPN 3x”, ”Faster R-CNN X101 32x8d FPN 3x”, ”Mask R-CNN X101 32x8d FPN 3x”, ”Cascade Mask R-CNN X152 32x8d FPN IN5k”, ”Panoptic FPN R101 3x”, ”YOLOv5x6” oraz niektóre modele douczone.

Badanie przeprowadzone zostało na zbiorach ”20161019 BM HD”, ”LP_GZ_HOM HD”, ”SoccerDB HD”, ”SoccerDB2 HD”, zbiorze połączonym składającym się ze zbiorów ”20161019 BM HD”, ”LP_GZ_HOM HD”, ”SoccerDB HD” oraz zbiorze testującym ze zbioru ”dataset_shuffle”. Spośród przeprowadzonych badań dla poszczególnych zbiorów, wybrane zostały architektury z najlepszymi wynikami.

Na podstawie wyników z Tabeli 3.1 dla modeli wstępnie wytrenowanych, architektura ”YOLOv5x6” największą dokładność uzyskuje w przypadku zbioru zawierającego większe obiekty. Architektura ”Faster R-CNN X101 32x8d FPN 3x” największą dokładność uzyskuje w przypadku zbioru zawierającego mniejsze obiekty. Architektura ”Cascade Mask R-CNN X152 32x8d FPN IN5k” największą dokładność uzyskuje w przypadku zbioru zawierającego mieszanego rozmiaru obiekty. Dla modeli douczonych architektura ”Faster R-CNN X101 32x8d FPN 3x” uzyskuje najlepszą dokładność wykrywania zawodników oraz piłki.

Tab. 3.1. Tabela wyników najlepszych architektur dla danego zbioru danych.

Zbiór danych	Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
20161019 BM FHD	YOLOv5x6	0,97	0,69	0,83
LP_GZ_HOM HD	Faster R-CNN X101...	0,90	0,33	0,61
SoccerDB HD	Cascade Mask R-CNN...	0,94	0,22	0,58
SoccerDB2 HD	Cascade Mask R-CNN...	0,88	0,21	0,54
Połączone zbiory	YOLOv5x6	0,92	0,39	0,65
Zbiór danych	Model douczony	zawodnik AP	piłka AP	mAP
dataset_shuffle	Faster R-CNN X101...	0,95	0,75	0,85
SoccerDB2 HD	Faster R-CNN X101...	0,92	0,79	0,85

3.1.2. Dokładność rozpoznawania typu obiektu

Celem badania było stwierdzenie czy detekcja zawodnika oraz piłki wpływa na dokładność rozpoznawania typu obiektu przez architekturę detekcji obiektów. Zawodnicy są większych rozmiarów powierzchniowych niż piłka co podczas skalowania obrazu przez sieć neuronową może mieć duży wpływ na jej dokładność. Badanie pozwoli stwierdzić jaka architektura powinna zostać użyta w celu wykrywania obu typów obiektów równolegle lub pojedynczo. Badanie przeprowadzone zostało na połączonych zbiorach "20161019 BM HD", "LP_GZ_HOM HD", "SoccerDB HD" w których w sumie znajdowały się 584 klatki. Zbiór ten zawierał klatki z różnymi ujęciami oraz pozycjami kamery. Dodatkowo przeprowadzone zostało osobne badanie zbioru "LP_GZ_HOM HD" ze względu na stałą oraz oddaloną pozycję kamery względem pozostałych zbiorów. Poniżej przedstawione zostały dwie tabele (Tabela 3.2 oraz Tabela 3.3) z wynikami wykrywania poszczególnych obiektów dla danej architektury.

Tabela 3.2 prezentuje wyniki badania architektur dla połączonych zbiorów danych "20161019 BM HD", "LP_GZ_HOM HD", "SoccerDB HD". Dla każdej architektury wyniki dla obiektu typu zawodnik wahają się między 0,91

Tab. 3.2. Tabela wyników dokładności rozpoznawania typu obiektu dla połączonych zbiorów "20161019 BM HD", "LP_GZ_HOM HD", "SoccerDB HD".

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,91	0,31	0,61
Faster R-CNN X101 32x8d FPN 3x	0,93	0,34	0,63
Mask R-CNN X101 32x8d FPN 3x	0,93	0,32	0,62
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,93	0,36	0,64
Panoptic FPN R101 3x	0,91	0,34	0,63
YOLOv5x6	0,92	0,39	0,65

Tab. 3.3. Tabela wyników dokładności rozpoznawania typu obiektu dla zbioru "LP_GZ_HOM HD".

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,86	0,30	0,58
Faster R-CNN X101 32x8d FPN 3x	0,90	0,33	0,61
Mask R-CNN X101 32x8d FPN 3x	0,90	0,24	0,57
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,88	0,30	0,59
Panoptic FPN R101 3x	0,87	0,29	0,58
YOLOv5x6	0,87	0,13	0,50

a 0,93 AP, natomiast dla piłki wartości są dużo niższe i wahają się między 0,31 a 0,39 AP. Widoczna jest więc znaczna różnica między dokładnością wykrywania zawodnika a piłki. W przypadku wykrywania zawodników największą dokładność uzyskały architektury "Faster R-CNN X101 32x8d FPN 3x", "Mask R-CNN X101 32x8d FPN 3x" oraz "Cascade Mask R-CNN X152 32x8d FPN IN5k". W przypadku wykrywania piłki najlepsze wyniki osiągnęła architektura "YOLOv5x6". Architektura "YOLOv5x6" również osiągnęła najlepszy średni wynik wykrywania zawodników oraz piłki.

W Tabeli 3.3 przedstawione zostały wyniki badania architektur dla zbioru "LP_GZ_HOM HD", który zawiera specyficznie trudne ujęcia ze względu na stałą oraz oddaloną pozycję kamery. Taka pozycja kamery powoduje, że zarówno zawodnicy jak i piłka są mniejszych rozmiarów w porównaniu do po-

zostałych zbiorów. Na podstawie wyników oraz w porównaniu do Tabeli 3.2 można stwierdzić, że dokładność wykrywania zawodników oraz piłki spadła. W przypadku zawodników różnica jest na poziomie od 0,02 do 0,06 punktów procentowych, natomiast w przypadku piłki jest od 0,01 do 0,26 punktów procentowych. Największy spadek punktów procentowych średniej dokładności odnotowany został dla architektury "YOLOv5x6". Najlepszy wyniki osiągnęła architektura "Faster R-CNN X101 32x8d FPN 3x". Na podstawie wyników można wywnioskować, że architektura "YOLOv5x6" osiąga gorsze wyniki w przypadku wykrywania małych obiektów. Architektura "Faster R-CNN X101 32x8d FPN 3x" najlepiej sprawdza się w przypadku wykrywania małych obiektów.

Niska dokładność wykrywania piłki może być związana z problemem skalowania obrazu podczas wyodrębniania cech i przekształcania ich do postaci wektora przez architekturę. Piłka w porównaniu do zawodnika jest znacznie mniejsza. W przypadku kiedy badany materiał video jest słabej jakości, piłka szybko się przemieszcza lub jest na tle innych przedmiotów jej rozmiar w pikselach może być bardzo mały lub w pewnej części zniekształcony, co powoduje, że podczas skalowania obrazu, piksele piłki zostają utracone co negatywnie wpływa na dokładność detekcji obiektu.

Problemem w przypadku wykrywania zawodników może być wzajemne nakładanie się zawodników co skutkuje wykryciem takich nałożonych się zawodników jako jeden duży zawodnik. Wykrycie jednego zawodnika zamiast kilku, zaburza poprawność detekcji i tym samym pogarsza wyniki.

Odpowiednie wykorzystanie danej architektury pozwoli osiągnąć lepsze wyniki. W przypadku wykrywania większych obiektów lepiej sprawdzi się architektura "YOLOv5x6". W przypadku wykrywania małych obiektów lepiej sprawdzi się architektura "Faster R-CNN X101 32x8d FPN 3x". W celu

zwiększenia dokładności detekcji obiektów, możliwe jest połączenie dwóch architektur.

3.1.3. Wpływ jakości materiału wideo na dokładność wykrywania obiektów

Celem badania było stwierdzenie czy jakość materiału wideo ma wpływ na dokładność detekcji zawodników oraz piłki. Badanie zostało podzielone na dwie pod sekcje. Pierwsza podsekcja dotyczyła wpływu rozdzielczości materiału wideo na dokładność detekcji obiektów. Druga podsekcja dotyczyła wpływu przepływności materiału wideo na dokładność detekcji obiektów. Badanie przeprowadzone zostało na zbiorach "20161019 BM FHD", "20161019 BM FHD 1k", "20161019 BM FHD 100" "20161019 BM HD", "20161019 BM SD".

Badanie wpływu rozdzielczości materiału wideo na dokładność wykrywania obiektów

Badanie ma na celu zweryfikowanie czy rozdzielcość materiału wideo wpływa na dokładność wykrywania obiektów przez daną architekturę. Badaniu podane zostały zbiory danych zawierające klatki odpowiednio w rozdzielczości 1920x1080 px (zbiór "20161019 BM FHD"), 1280x720 px (zbiór "20161019 BM HD") oraz 720x405 px (zbiór "20161019 BM SD"). W przypadku platformy Detectron2 jeżeli obrazy wejściowe są większe niż 1333 px dla dowolnej krawędzi, są wtedy skalowane do rozdzielczości 1333 px z zachowaniem proporcji. W platformie YOLOv5 obrazy większe niż 1280 px również są skalowane z zachowaniem proporcji do rozdzielczości 1280 px. Z powodu skalowania obrazów wejściowych dokładność szacowania dla rozdzielczości 1920x1080 px

Tab. 3.4. Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD" z obrazami w rozdzielczości 1920x1080 px.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,98	0,45	0,71
Faster R-CNN X101 32x8d FPN 3x	0,98	0,54	0,76
Mask R-CNN X101 32x8d FPN 3x	0,97	0,52	0,75
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,98	0,54	0,76
Panoptic FPN R101 3x	0,96	0,49	0,73
YOLOv5x6	0,97	0,69	0,83

Tab. 3.5. Tabela wyników dokładność wykrywania obiektów dla zbioru "20161019 BM HD" z obrazami w rozdzielczości 1280x720 px.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,96	0,45	0,71
Faster R-CNN X101 32x8d FPN 3x	0,97	0,54	0,75
Mask R-CNN X101 32x8d FPN 3x	0,96	0,52	0,74
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,97	0,54	0,75
Panoptic FPN R101 3x	0,96	0,49	0,72
YOLOv5x6	0,96	0,66	0,81

oraz 1280x720 px może być bardzo podobna. Poniżej przedstawiono trzy tabele (Tabela 3.4, Tabela 3.5 oraz Tabela 3.6) prezentujące wyniki dla poszczególnych rozdzielczości.

Tabela 3.4 prezentuje wyniki dokładności modeli wstępnie wytrenowanych dla zbioru "20161019 BM FHD" z obrazami w rozdzielczości 1920x1080 px. W przypadku wykrywania zawodników modele osiągały bardzo zbliżone wartości w przedziale od 0,96 do 0,98 AP. W przypadku wykrywania piłki wyniki były znacznie gorsze względem wyników dla zawodników. Największą skutecznością wykrywania piłki osiągnął model "YOLOv5x6". Model "YOLOv5x6" również osiągnął najlepszy średni wynik na poziomie 0,83 mAP.

W Tabeli 3.5 przedstawione zostały wyniki dokładności modeli detekcji

Tab. 3.6. Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM SD" z obrazami w rozdzielczości 720x405 px.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,97	0,44	0,70
Faster R-CNN X101 32x8d FPN 3x	0,96	0,51	0,74
Mask R-CNN X101 32x8d FPN 3x	0,96	0,50	0,73
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,97	0,53	0,75
Panoptic FPN R101 3x	0,96	0,48	0,72
YOLOv5x6	0,96	0,22	0,59

obiektów wstępnie wytrenowanych dla zbioru "20161019 BM HD" z obrazami w rozdzielczości 1280x720 px. Podobnie jak w Tabeli 3.4 w przypadku wykrywania zawodników modele osiągały bardzo zbliżone wartości w przedziale od 0,96 do 0,97 AP. W przypadku wykrywania piłki podobnie jak w Tabeli 3.4 wyniki były na podobnie słabym poziomie. W tym przypadku również największą skutecznością wykrywania piłki (0,66 AP) osiągnął model YOLOv5x6. Model "YOLOv5x6" także osiągnął najlepszy średni wynik na poziomie 0,81 mAP.

Tabela 3.6 przedstawia wyniki dokładności modeli wstępnie wytrenowanych dla zbioru "20161019 BM SD" z obrazami w rozdzielczości 720x405 px. Podobnie jak w Tabeli 3.4 oraz Tabeli 3.5 w przypadku wykrywania zawodników modele osiągały bardzo zbliżone wartości w przedziale od 0,96 do 0,97 AP. Natomiast w przypadku wykrywania piłki wyniki były nieznacznie gorsze niż w poprzednich tabelach za wyjątkiem modelu "YOLOv5x6" dla którego wyniki znacznie pogorszyły się względem pozostałych modeli. Największą dokładność wykrywania piłki (0,53 AP) osiągnął model "Cascade Mask R-CNN X152 32x8d FPN IN5k". Model ten również osiągnął najlepszy średni wynik na poziomie 0,75 mAP.

W przypadku modeli dostarczonych przez platformę Detectron2 rozdzielcość badanego materiału nieznacznie wpływa na dokładność wykrywania

Tab. 3.7. Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD" z obrazami, z materiału wideo o przepływności 7799 kb/s.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,98	0,45	0,71
Faster R-CNN X101 32x8d FPN 3x	0,98	0,54	0,76
Mask R-CNN X101 32x8d FPN 3x	0,97	0,52	0,75
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,98	0,54	0,76
Panoptic FPN R101 3x	0,96	0,49	0,73
YOLOv5x6	0,97	0,69	0,83

obiektów i wyniki są bardzo zbliżone, różnią się na poziomie kilku punktów procentowych. Natomiast w przypadku platformy YOLOv5 widoczny jest znaczy spadek dokładności szacowania piłki dla zbioru "20161019 BM SD". Spadek jest co najmniej trzykrotny. Ze względu na lepszą wydajność modelu "YOLOv5x6" najlepiej zastosować ten model do wykrywania obiektów dla zbiorów danych w rozdzielcości większej bądź równej 1280x720 px, natomiast w przypadku zbiorów danych o rozdzielcości mniejszej niż 1280x720 px najlepiej zastosować model "Cascade Mask R-CNN X152 32x8d FPN IN5k".

Badanie wpływu przepływności materiału wideo na dokładność wykrywania obiektów

Badanie ma celu zweryfikowanie czy przepływność materiału wideo ma wpływ na dokładność wykrywania obiektów przez daną architekturę. Badaniu podane zostały zbiory danych zawierające klatki o przepływności 7799 kb/s (zbiór "20161019 BM FHD"), 1000 kb/s (zbiór "20161019 BM FHD 1k") oraz 100 kb/s (zbiór "20161019 BM FHD 100"). Poniżej przedstawione zostały trzy tabele (Tabela 3.7, Tabela 3.8 oraz Tabela 3.9) prezentujące wyniki dla poszczególnych przepływności.

Tab. 3.8. Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD 1k" z obrazami, z materiału wideo o przepływności 1000 kb/s.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,96	0,41	0,68
Faster R-CNN X101 32x8d FPN 3x	0,96	0,48	0,72
Mask R-CNN X101 32x8d FPN 3x	0,95	0,44	0,70
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,97	0,54	0,75
Panoptic FPN R101 3x	0,94	0,45	0,69
YOLOv5x6	0,96	0,55	0,76

Tabela 3.7 prezentuje wyniki dokładności modeli wstępnie wytrenowanych dla zbioru "20161019 BM FHD" z obrazami, z materiału wideo o przepływności 7799 kb/s. Jest to oryginalna jakość materiału wideo. W przypadku wykrywania zawodników modele osiągały bardzo zbliżone wartości w przedziale od 0,96 do 0,98 AP. W przypadku wykrywania piłki wyniki były znacznie gorsze. Największą dokładność wykrywania piłki oraz najlepszy średni wynik wykrawania obu typów obiektów osiągnął model "YOLOv5x6".

Tabela 3.8 prezentuje wyniki dokładności modeli wstępnie wytrenowanych dla zbioru "20161019 BM FHD 1k" z obrazami, z materiału wideo o przepływności 1000 kb/s. W przypadku wykrywania zawodników modele osiągnęły nieznacznie mniejsze wartości niż w przypadku oryginalnej jakości materiału. W przypadku wykrywania piłki wyniki osiągneły większą różnicę w porównaniu do oryginalnego materiału wideo. Różnica ta wynosi od 0,04 do 0,14 punktów procentowych. Największą różnicę uzyskał model YOLOv5x6 względem oryginału, pomimo dużego spadku model ten nadal miał największą dokładność wykrywania zawodników oraz piłki.

W Tabeli 3.9 przedstawione zostały wyniki detekcji obiektów modeli wstępnie wytrenowanych dla zbioru "20161019 BM FHD 100" z obrazami, z materiału wideo o przepływności 100 kb/s. Dokładność szacowania znacznie się

Tab. 3.9. Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD 100" z obrazami, z materiału wideo o przepływności 100 kb/s.

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,58	0,02	0,30
Faster R-CNN X101 32x8d FPN 3x	0,49	0,02	0,26
Mask R-CNN X101 32x8d FPN 3x	0,53	0,03	0,28
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,75	0,19	0,47
Panoptic FPN R101 3x	0,56	0,05	0,30
YOLOv5x6	0,48	0,06	0,27

pogorszyła. W przypadku wykrywania zawodników modele osiągnęły prawie o połowę mniejsze wyniki niż w przypadku oryginalnego materiału wideo. W przypadku wykrywania piłki wyniki przedstawiają znaczne pogorszenie względem wyników w Tabeli 3.7. Różnica ta wynosi średnio około 40 punktów procentowych. Najwyższą dokładność szacowania osiągnął model "Cascade Mask R-CNN X152 32x8d FPN IN5k".

Na podstawie przeprowadzonych badań można stwierdzić, że przepływność materiału wideo ma wpływ na dokładność wykrawania obiektów przy użyciu omawianych architektur. W przypadku przepływności powyżej 1000 kb/s różnica nie jest duża co oznacza, że materiały wideo z taką przepływnością mogą być używane jako materiał wideo dobrej jakości. W przypadku materiałów wideo o przepływności bliskiej wartości 100 kb/s widać znaczny spadek dokładności wykrywania obiektów co oznacza, że takie materiały nie nadają się do analizy.

Podsumowanie badań wpływu jakości materiału wideo na dokładność wykrywania obiektów

Na podstawie przeprowadzonych badań dotyczących wpływu jakości materiału wideo na dokładność wykrywania obiektów przez omawiane architektury,

można stwierdzić, że jakość analizowanego materiału wpływa na dokładność wykrywania obiektów. Tym samym można stwierdzić, że aby architektura skutecznie wykrywała zawodników oraz piłkę, materiał powinien być w rozdzielczości co najmniej 1280x1080 px oraz przepływności większej niż 1000 kb/s. W przypadku materiałów wideo dobrej jakości najlepiej sprawdza się model "YOLOv5x6" natomiast w przypadku materiałów słabej jakości najlepiej sprawdza się model "Cascade Mask R-CNN X152 32x8d FPN IN5k"

3.1.4. Dokładność wykrywania obiektów przy użyciu modeli douczonych

Celem badania było stwierdzenie czy douczenie modelu poprawia dokładność wykrywania zawodników oraz piłki. W tym celu douczone zostały cztery modele: dwa bazujące na algorytmie "Faster R-CNN" z różną liczbą ukrytych warstw oraz konfiguracją architektury oraz model "Cascade Mask R-CNN X152 32x8d FPN IN5k" i model "YOLOv5x6". Badane architektury zostały opisane w rozdziale 2.3.3. Badanie przeprowadzone zostało na zbiorze testującym "dataset_shuffle" oraz zbiorze "SoccerDB2 HD".

Tabela 3.10 przedstawia wyniki detekcji obiektów przy użyciu modeli wstępnie wytrenowanych oraz douczonych. Na podstawie wyników można wywnioskować, że modele douczone mają od 0,01 do 0,05 punktów procentowych większą dokładność detekcji zawodników niż modele wstępnie wytrenowane, co na tak wysokim poziomie dokładności oznacza jeszcze lepszy wynik. W przypadku detekcji piłki widać większą poprawę. Wyniki wzrosły o od 0,14 do 0,44 punktów procentowych, co oznacza znaczną poprawę. Największy wzrost dokładności wykrywania zawodników oraz piłki uzyskały modele "Faster R-CNN R50 FPN 3x" oraz "Faster R-CNN X101 32x8d FPN 3x".

Tab. 3.10. Tabela wyników dokładności wykrywania obiektów przy użyciu modeli douczonych dla zbioru testującego ze zbioru "dataset_shuffle".

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,90	0,31	0,60
Faster R-CNN X101 32x8d FPN 3x	0,91	0,31	0,61
Mask R-CNN X101 32x8d FPN 3x	0,91	0,31	0,61
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,92	0,38	0,65
Panoptic FPN R101 3x	0,90	0,32	0,61
YOLOv5x6	0,90	0,42	0,66
Model douczony	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,95	0,75	0,85
Faster R-CNN X101 32x8d FPN 3xP	0,95	0,75	0,85
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,93	0,52	0,72
YOLOv5x6	0,94	0,69	0,82

Zbiór testowy "dataset_shuffle" zawierał tylko 117 klatek, z tego względu douczone modele zostały dodatkowo zweryfikowane na modelu "SoccerDB2 HD", który zawierał 1057 klatek. Wyniki zbioru "SoccerDB2 HD" zostały zaprezentowane w Tabeli 3.11. Podobnie jak w poprzedniej Tabeli 3.10 dokładność douczonych modeli zwiększyła się, szczególnie w przypadku wykrywania piłki. Niektóre z douczonych modeli osiągnęły jednak gorsze wyniki co może wynikać z większej różnorodności zbioru "SoccerDB2 HD", przeuczenia lub niedouczenia modelu. Najlepsze wyniki osiągnął model "Faster R-CNN X101 32x8d FPN 3x".

Na podstawie wyników przeprowadzonych badań douczenia modeli, można wywnioskować, że douczenie wstępnie wytrenowanego modelu w znacznym stopniu poprawia dokładność detekcji zawodników oraz szczególnie piłki. Nawet niewielki zbiór trenujący składający się z 467 klatek, z różnych ujęć oraz pozycji kamery, pozwala skutecznie dotrenować wstępnie wyszkolony model sieci neuronowej. Przy użyciu wstępnie wytrenowanych modeli niepotrzebny jest zatem duży zbiór uczący aby uzyskać znaczną poprawę detekcji

Tab. 3.11. Tabela wyników dokładności wykrywania obiektów przy użyciu modeli douczonych dla zbioru testującego "SoccerDB2 HD".

Model wstępnie wytrenowany	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,85	0,16	0,50
Faster R-CNN X101 32x8d FPN 3x	0,86	0,17	0,51
Mask R-CNN X101 32x8d FPN 3x	0,86	0,16	0,51
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,88	0,21	0,54
Panoptic FPN R101 3x	0,84	0,19	0,52
YOLOv5x6	0,85	0,22	0,53
Model douczony	zawodnik AP	piłka AP	mAP
Faster R-CNN R50 FPN 3x	0,91	0,75	0,83
Faster R-CNN X101 32x8d FPN 3x	0,92	0,79	0,85
Cascade Mask R-CNN X152 32x8d FPN IN5k	0,90	0,74	0,82
YOLOv5x6 1280 B6 E100 F9	0,89	0,58	0,74

obiektów.

3.2. Wnioski z badań

Na podstawie przeprowadzonych badań, postawione hipotezy potwierdziły się. Różne scenariusze różne wpływają na wyniki danej architektury. W przypadku zbiorów danych zawierających większe obiekty najlepszą architekturą bazującą na modelach wstępnie wytrenowanych, okazała się architektura "YOLOv5x6". Dla zbiorów danych zawierających obiekty różnych rozmiarów najlepsze wyniki osiągnęła architektura "Cascade Mask R-CNN X152 32x8d FPN IN5k". W przypadku zbiorów danych, zawierających większość mniejszych obiektów, ze względu na oddaloną pozycję kamery najlepsze wyniki uzyskała architektura "Faster R-CNN X101 32x8d FPN 3x". Na podstawie wyników z rozdziału 3.1.1 potwierdzona została hipoteza, że dana architektura ma wpływ na dokładność wykrywania obiektów.

Architektury wykrywania obiektów potrafią również rozpoznać typ wy-

krytego obiektu. W przypadku piłki nożnej oraz na potrzeby przeprowadzanych badań, typem wykrywanych obiektów były zawodnik oraz piłka. Ze względu na różny rozmiar zawodnika oraz piłki, pojawia się problem z dokładnością wykrywania tych obiektów w szczególności piłki. W tym celu przeprowadzone zostało badanie pozwalające stwierdzić czy typ obiektu wpływa na dokładność rozpoznawania obiektów oraz czy odpowiednio dobrana architektura, pozwoli osiągnąć lepsze wyniki. Na podstawie wyników z rozdziału 3.1.2 potwierdzona, została hipoteza, że typ badanego obiektu ma wpływ na dokładność rozpoznawania obiektów, ponieważ dokładność wykrywania piłki była znacznie gorsza od wyników wykrywania zawodników. Można także stwierdzić, że najbardziej uniwersalną architekturą do wykrywania zawodników oraz piłki bazującą na modelach wstępnie wytrenowanych jest architektura "YOLOv5x6", ponieważ osiągnęła najlepsze średnie wyniki dla obu typów obiektów.

Hipoteza stwierdzająca, że jakość danych ma wpływ na dokładność wykrywania danych, również została potwierdzona. Na podstawie wyników z rozdziału 3.1.3 można zaobserwować, że jakość materiału wideo poniżej rozdzielczości 1280x720 px powoduje zmniejszenie dokładności wykrywania obiektów. W przypadku przepływności wyniki są bardziej skrajne i dla przypadku kiedy dostarczony materiał wideo jest w przepływności bliżej lub niższej 100 kb/s wyniki drastycznie spadają osiągając średnią dokładność 0,47 mAP dla najlepszej architektury, w porównaniu do zbioru oryginalnej jakości, gdzie średnia dokładność dla najlepszej architektury wynosi 0,83 mAP.

Ostatnia hipoteza dotyczyła stwierdzenia, że douczenie modelu ma wpływ na dokładność wykrywania obiektów. Na podstawie wyników z rozdziału 3.1.4, hipoteza została potwierdzona. Wyniki z badania prezentują znaczy wzrost dokładności wykrywania obiektów, w szczególności piłki. Średnia wy-

Tab. 3.12. Tabela wyników uzyskanych przez SoccerDB Źródło: [14]

Architektura	zawodnik AP	piłka AP	mAP
RetinaNet	0,75	0,43	0,59
Faster R-CNN	0,76	0,42	0,59

niku dla najlepszej architektury ("Cascade Mask R-CNN X152 32x8d FPN IN5k") bazującej na modelu wstępnie wytrenowanym wynosi 0,54 mAP, w przypadku najlepszej architektury ("Faster R-CNN X101 32x8d FPN 3x") bazującej na modelu douczonym wynosi 0,85 mAP co daje o około 57% lepszy wynik średniej dokładności wykrywania obiektów. Na podstawie tych wniosków można stwierdzić, że modele wstępnie wytrenowane nie są dostosowane do rozwiązywania konkretnego problemu. Problem ten można rozwiązać, wykorzystując technikę nauczania metodą transferu. Douczenie modelu zbiorem zawierającym dane dotyczące danego środowiska umożliwia dostosowanie sieci neuronowej do rozwiązywania konkretnego problemu i znaczne poprawienie dokładności architektur.

W Tabeli 3.12 przedstawione zostały wyniki badań dokładności wykrywania zawodników oraz piłki uzyskane przez autorów artykułu "Comprehensive Soccer Video Understanding" [14]. Badanie przeprowadzone zostało na pełnym zbiorze SoccerDB. Badaniu poddane zostały dwie architektury: RetinaNet oraz Faster R-CNN. W przypadku wykrywania zawodników architektura RetinaNet osiągnęła o 0,01 punktu procentowego gorszy wynik niż architektura Faster R-CNN. Natomiast w przypadku wykrywania piłki architektura RetinaNet osiągnęła o 0,01 punktu procentowego lepszy wynik niż architektura Faster R-CNN. Ostatecznie średni wynik dokładności wykrywania obu typów obiektów jest taki sam dla architektury RetinaNet oraz Faster R-CNN i wynosi 0,59 mAP.

Na podstawie wyników uzyskanych przez SoccerDB, przedstawionych w

Tabeli 3.12 oraz własnych wyników uzyskanych dla modeli douczonych, opisanych w rozdziale 3.1.4, widać znaczną poprawę dokładności rozpoznawania zawodników oraz w szczególności piłki, dla modeli douczonych.

3.2.1. Podsumowanie

Przeprowadzenie badań pozwoliło stwierdzić jak różnego rodzaju problemy dotyczące danego zagadnienia wpływają na dokładność działania architektur detekcji obiektów. W przypadku rozwiązywania problemu dotyczącego piłki nożnej, uwaga skupiona została na dokładności detekcji zawodników oraz piłki. Dodatkowy wpływ na dokładność używanych architektur, miał dostarczony materiał wideo. Duży wpływ na rezultaty miała również pozycja kamery oraz jakość zbioru danych. W związku ze współpracą z klubem sportowym KKS Lech Poznań, uwaga została również skupiona na przeprowadzeniu badań przy użyciu danych dostarczonych przez klub.

Na podstawie wniosków płynących z badań, można stwierdzić, że w przypadku modeli wstępnie wytrenowanych, materiałów wideo z różnymi pozycjami kamery, w środowisku boiska piłkarskiego dla materiałów wideo dobrej jakości największą dokładność osiągnęła architektura "YOLOv5x6". W przypadku materiałów słabej jakości największą dokładność uzyskała architektura "Cascade Mask R-CNN X152 32x8d FPN IN5k". Dla materiałów wideo zawierających mniejsze obiekty (takich jak zbiór danych "LP_GZ_HOM HD" dostarczony przez klub sportowy KKS Lech Poznań) najlepsze wyniki osiągnęła architektura "Faster R-CNN X101 32x8d FPN 3x". W przypadku modeli douczonych najlepsze wyniki również osiągnęła architektura "Faster R-CNN X101 32x8d FPN 3x". Na tej podstawie wybrana została architektura oraz model wykorzystywany w narzędziu, tworzonym w ramach projektu badawczego, realizowanego we współpracy z klubem sportowym KKS Lech

Poznań. Podsumowanie wyników badań pomaga stwierdzić jaka architektura wykrywania obiektów powinna zostać użyta w celu rozwiązania danego problemu.

Wnioski końcowe

Celem pracy było przeprowadzenie badań dokładności różnego rodzaju algorytmów wykrywania obiektów, stosowanych w widzeniu komputerowym dla problemu wykrywania zawodników oraz piłki w dziedzinie sportowej - piłce nożnej. Przeprowadzone badania miały na celu stwierdzenie czy dana architektura wykrawania obiektów, typ badanych obiektów, jakość badanego zbioru danych oraz czy douczenie modelu architektury wpływa na dokładność wykrywania zawodników oraz piłki w środowisku meczu piłki nożnej. Rezultatem pomyślnego ukończenia wszystkich badanych problemów były spójne wyniki prezentujące dokładność wykorzystywanych architektur wykrywania obiektów dla każdego problemu.

W pierwszym rozdziale opisane zostały zagadnienia teoretyczne, wprowadzające i pozwalające lepiej zrozumieć technikę badań. Zaprezentowane zostały wykorzystywane narzędzia umożliwiające przeprowadzenie badań oraz teoria umożliwiająca odpowiednie zrozumienie i zinterpretowanie wyników badanych problemów.

W drugim rozdziale przedstawione zostały poruszane problemy pracy badawczej, omówiony został plan i poszczególne etapy badań. Przedstawione zostały wykorzystywane architektury oraz platformy wykrywania obiektów, omówione zostało narzędzie badawcze. Zaprezentowane zostały źródła zbiorów danych oraz sposób ich przygotowania. Opisany został sposób oraz kon-

figuracja parametrów douczania modeli. Odpowiednie przygotowanie planu realizacji badań narzędzia badawczego oraz zbioru danych umożliwiło sprawne i rzetelnie przeprowadzić badania architektur wykrywania obiektów oraz zbiorów danych.

W trzecim rozdziale zaprezentowane zostały wyniki dla postawionych hipotez. Przedstawiona została analiza uzyskanych rezultatów. Opisane zostały wnioski wynikające z przeprowadzonych badań. Na podstawie otrzymanych wyników potwierdzone zostały hipotezy. Uzyskane odpowiedzi potwierdziły, że dana architektura, typ wykrywanego obiektu, jakość zbiorów danych oraz douczenie modelu wpływają na dokładność wykrywania obiektów. Na podstawie wyników badań uzyskana została również odpowiedź jaką architekturę powinno się wykorzystać do rozwiązania danego problemu.

Z przyczyn ograniczeń czasowych, braku pełnej swobody dostępu do maszyn obliczeniowych, ograniczeniach w dostępnych oznaczonych zbiorach danych, przez co konieczne było ręczne oznaczanie oraz weryfikowanie badanych danych, przetestowane zostało tylko kilka możliwości konfiguracji parametrów uczenia wstępnie wytrenowanych modeli oraz wykorzystanych algorytmów. W przypadku większego dostępu zasobu czasowego oraz większej swobody dostępu do maszyn obliczeniowych, możliwe byłoby dokładniejsze dostrojenie konfiguracji parametrów uczenia modeli, zwiększenie zasobu oznaczonych zbiorów danych, zastosowanie odpowiedniego przetwarzania badanych danych w celu poprawienia dokładności wykrywania obiektów za pomocą narzędzia ODATT oraz wykorzystywanych architektur.

Mimo napotkanych problemów, przeprowadzone badania pozwoliły jednoznacznie określić wpływ środowiska problemu na dokładność wykrywania zawodników oraz piłki na nagraniach z meczów piłki nożnej. Z możliwych dalszych działań dotyczących problemu dokładności wykrywania zawodników

oraz piłki jest: dodatkowe przetwarzanie badanych danych poprzez zastosowanie różnego rodzaju algorytmów przetwarzania obrazu, powiększenie uczącego zbioru danych - możliwe jest zastosowanie wspomianego w rozdziale 2.2.3 symulatora rozgrywki meczu piłki nożnej, do wygenerowania syntetycznych, oznaczonych zbiorów danych. Przetestowanie dodatkowych konfiguracji parametrów uczenia modeli. Zastosowanie innych architektur wykrywania obiektów niż tych wykorzystywanych w pracy.

Na podstawie rezultatów osiągniętych w tej pracy, można stwierdzić, że dalsza praca nad poruszonymi problemami, umożliwi stworzenie pełnoprawnego narzędzia do analizowania sytuacji na boisku piłkarskim. Wyniki dostarczone przez takie narzędzie będą mogły w znacznym stopniu wspomóc kadrę szkoleniową klubu piłkarskiego w podejmowaniu kluczowych decyzji związanych z np. taktyką gry, rozstawieniem zawodników itp. zwiększając szanse na lepsze osiągnięcia klubu.

Spis rysunków

1.1.	Diagram Sztuczna Inteligencja. Źródło: opracowanie własne . . .	11
1.2.	Diagram sieci neuronowej z warstwą wejściową rozmiaru czterech neuronów, jedną warstwą ukrytą oraz jedną warstwą wyjściową. Źródło: opracowanie własne wzorowane na (patrz [15])	13
1.3.	Splotowa sieć neuronowa. Źródło: [20]	15
1.4.	Architektura sieci FPN. Źródło: [22]	17
1.5.	Architektura sieci Fast R-CNN. Źródło: [1]	19
1.6.	Architektura sieci Faster R-CNN. Źródło: [6]	19
1.7.	Architektura sieci Mask R-CNN. Źródło: [37]	20
1.8.	Architektura sieci Panoptic FPN. Źródło: [16]	21
1.9.	Typy segmentacji pikseli na obrazie. Źródło: [18]	22
1.10.	Podstawowa architektura YOLOv4. Źródło: [21]	23
1.11.	Przykład wizualizacji położenia obiektu na obrazie za pomocą pola ograniczającego w formie czerwonego prostokąta. Źródło: Opracowanie własne na podstawie kadru ze zbioru danych dostarczonego przez klub sportowy KKS Lech Poznań	27
1.12.	Wizualizacja równania IoU. Źródło: [28]	28

1.13. Przykład wizualizacji rzeczywistego położenia obiektu względem położenia obiektu oszacowanego przez model wykrywania obiektów. Źródło: Opracowanie własne na podstawie kadru ze zbioru danych dostarczonego przez klub sportowy KKS Lech Poznań	29
2.1. Diagram przedstawiający przepływ danych w narzędziu ODATT. Źródło: Opracowanie własne	38
2.2. Kadry dla różnych przepływności materiału wideo. Kadr (a) prezentuje oryginalną przepływność materiału wideo. Kadr (b) przedstawia kadr z materiału wideo o przepływności 1000kb/s. Kadr (c) przedstawia kadr z materiału wideo o przepływności 100 kb/s. Źródło: Kadry pochodzą z materiału wideo ze zbioru danych SoccerNet.	45
2.3. Różne pozycje kamery dla zbioru SoccerDB oraz zbioru SoccerNet. Jak przedstawiono na kadrze (a) i (c) rozmiar zawodników oraz piłki jest większy ze względu na bliską pozycję kamery. Na kadrze (b) oraz (d) zawodnicy oraz piłki są mniejszych rozmiarów ze względu na bardziej oddaloną pozycję kamery. Źródło: Kadry ze zbioru danych SoccerDB oraz SoccerNet . .	48
2.4. Ze względu na stałą odległość pozycji kamery dla nagrania z meczu klubu sportowego Lech Poznań, zawodnicy oraz piłka są mniejszych rozmiarów. Źródło: Kadr z nagrania meczu klubu sportowego Lech Poznań	48
2.5. Wykres zachowania wartości funkcji straty dla obwiedni, podczas trenowania modelu YOLOv5x6. Źródło: Opracowanie własne	50

Spis tabel

3.1.	Tabela wyników najlepszych architektur dla danego zbioru danych.	56
3.2.	Tabela wyników dokładności rozpoznawania typu obiektu dla połączonych zbiorów "20161019 BM HD", "LP_GZ_HOM HD", "SoccerDB HD".	57
3.3.	Tabela wyników dokładności rozpoznawania typu obiektu dla zbioru "LP_GZ_HOM HD".	57
3.4.	Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD" z obrazami w rozdzielczości 1920x1080 px.	60
3.5.	Tabela wyników dokładność wykrywania obiektów dla zbioru "20161019 BM HD" z obrazami w rozdzielczości 1280x720 px.	60
3.6.	Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM SD" z obrazami w rozdzielczości 720x405 px.	61
3.7.	Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD" z obrazami, z materiału wideo o przepływności 7799 kb/s.	62
3.8.	Tabela wyników dokładności wykrywania obiektów dla zbioru "20161019 BM FHD 1k" z obrazami, z materiału wideo o przepływności 1000 kb/s.	63

3.9. Tabela wyników dokładności wykrywania obiektów dla zbioru ”20161019 BM FHD 100” z obrazami, z materiału wideo o przepływności 100 kb/s.	64
3.10. Tabela wyników dokładności wykrywania obiektów przy uży- ciu modeli douczonych dla zbioru testującego ze zbioru ”data- set_shuffle”.	66
3.11. Tabela wyników dokładności wykrywania obiektów przy uży- ciu modeli douczonych dla zbioru testującego ”SoccerDB2 HD”. .	67
3.12. Tabela wyników uzyskanych przez SoccerDB Źródło: [14] . . .	69

Bibliografia

- [1] ANANTH, S. Fast r-cnn for object detection. <https://towardsdatascience.com/fast-r-cnn-for-object-detection-a-technical-summary-a0ff94faa022>, 2019.
- [2] BOCHKOVSKIY, A., WANG, C., AND LIAO, H. M. Yolov4: Optimal speed and accuracy of object detection. *CoRR abs/2004.10934* (2020).
- [3] CHOLLET, F. *Deep learning with Python*. Manning, 2018.
- [4] DANA H. BALLARD, C. M. B. *Computer Vision*. Prentice-Hall, Inc, 1982.
- [5] DELIÈGE, A., CIOPPA, A., GIANCOLA, S., SEIKAVANDI, M. J., DU-EHOLM, J. V., NASROLLAHI, K., GHANEM, B., MOESLUND, T. B., AND DROOGENBROECK, M. V. Soccernet-v2 : A dataset and benchmarks for holistic understanding of broadcast soccer videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops* (June 2021).
- [6] DENG, Z., SUN, H., ZHOU, S., ZHAO, J., LEI, L., AND ZOU, H. Multi-scale object detection in remote sensing imagery with convolutional neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing* 140 (2018), 1–12.

- nal neural networks. *ISPRS Journal of Photogrammetry and Remote Sensing* 145 (05 2018).
- [7] F. LUGER, G. *Artifical inteligence: Structures and Strategiesfor Complex Problem Solving*. Pearson, 2009.
- [8] GARNIER, P., AND GREGOIR, T. Evaluating soccer player: from live camera to deep reinforcement learning. *CoRR abs/2101.05388* (2021).
- [9] GUIMARAES, D. A. *Digital Transmission: A Simulation-Aided Introduction with VisSim/Comm*. Springer Berlin, Heidelberg, 2009.
- [10] HAN, S., DONG, X., HAO, X., AND MIAO, S. Extracting objects’ spatial–temporal information based on surveillance videos and the digital surface model. *ISPRS International Journal of Geo-Information* 11, 2 (2022).
- [11] HAYKIN, S. *Neural Networks and Learning Machines Third Edition*. Pearson, 2008.
- [12] HE, K., GKIOXARI, G., DOLLÁR, P., AND GIRSHICK, R. B. Mask R-CNN. *CoRR abs/1703.06870* (2017).
- [13] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. *CoRR abs/1512.03385* (2015).
- [14] JIANG, Y., CUI, K., CHEN, L., WANG, C., WANG, C., LIU, H., AND XU, C. Comprehensive soccer video understanding: Towards human-comparable video understanding system in constrained environment. *CoRR abs/1912.04465* (2019).
- [15] KHANDELWAL, R. How to solve randomness in an artificial neural network? <https://towardsdatascience.com/>

how-to-solve-randomness-in-an-artificial-neural-network-3befc4f27d45, 2020.

- [16] KIRILLOV, A. Panoptic segmentation: Task and approaches. https://www.dropbox.com/s/t6tg87t78pdq6v3/cvpr19_tutorial_alexander_kirillov.pdf?dl=0, 2019.
- [17] KIRILLOV, A., GIRSHICK, R. B., HE, K., AND DOLLÁR, P. Panoptic feature pyramid networks. *CoRR abs/1901.02446* (2019).
- [18] KIRILLOV, A., HE, K., GIRSHICK, R. B., ROTHER, C., AND DOLLÁR, P. Panoptic segmentation. *CoRR abs/1801.00868* (2018).
- [19] KURACH, K., RAICHUK, A., STANCZYK, P., ZAJAC, M., BACHEM, O., ESPEHOLT, L., RIQUELME, C., VINCENT, D., MICHALSKI, M., BOUSQUET, O., AND GELLY, S. Google research football: A novel reinforcement learning environment. *CoRR abs/1907.11180* (2019).
- [20] KWASIGROCH, A., AND GROCHOWSKI, M. Rozpoznawanie obiektÓw przez głĘbokie sieci neuronowe. *Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej* 60 (2018), 63–65.
- [21] LI, Y., WANG, H., DANG, L. M., HAN, D., MOON, H., AND NGUYEN, T. A deep learning-based hybrid framework for object detection and recognition in autonomous driving. *IEEE Access* 8 (10 2020).
- [22] LIN, T.-Y., DOLLÁR, P., GIRSHICK, R., HE, K., HARIHARAN, B., AND BELONGIE, S. Feature pyramid networks for object detection, 2016.
- [23] MAJEWSKI, M., AND PAŁKA, D. System graficznego rozpoznawania obiektów ruchomych. *Zeszyty Naukowe WWSI* 13, 21 (2019), 57–73.

- [24] MANNING, C. *Artificial Intelligence Definitions*. Stanford University, 2020.
- [25] PADILLA, R., NETTO, S. L., AND DA SILVA, E. A. B. A survey on performance metrics for object-detection algorithms. In *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)* (2020), pp. 237–242.
- [26] PADILLA, R., PASSOS, W. L., DIAS, T. L. B., NETTO, S. L., AND DA SILVA, E. A. B. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics* 10, 3 (2021).
- [27] REN, S., HE, K., GIRSHICK, R. B., AND SUN, J. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR abs/1506.01497* (2015).
- [28] ROSEBROCK, A. Intersection over union (iou) for object detection. <https://pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>, 2016.
- [29] SZELISKI, R. *Computer Vision: Algorithms and Applications*. Springer, 2021.
- [30] TADEUSIEWICZ, R. *Sieci neuronowe*. Akademicka Oficyna Wydawnicza, 1993.
- [31] TAN, M., AND LE, Q. V. Efficientnet: Rethinking model scaling for convolutional neural networks. *CoRR abs/1905.11946* (2019).
- [32] TAN, M., PANG, R., AND LE, Q. V. Efficientdet: Scalable and efficient object detection. *CoRR abs/1911.09070* (2019).

- [33] UIJLINGS, J., VAN DE SANDE, K., GEVERS, T., AND SMEULDERS, A. Selective search for object recognition. *International Journal of Computer Vision* (2013).
- [34] WU, Y., KIRILLOV, A., MASSA, F., LO, W.-Y., AND GIRSHICK, R. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019.
- [35] WUZHE. Yolov5 (6.0/6.1) brief summary. <https://github.com/ultralytics/yolov5/issues/6998#1>, 2022.
- [36] XIE, S., GIRSHICK, R. B., DOLLÁR, P., TU, Z., AND HE, K. Aggregated residual transformations for deep neural networks. *CoRR abs/1611.05431* (2016).
- [37] YANG, Z., DONG, R., XU, H., AND GU, J. Instance segmentation method based on improved mask r-cnn for the stacked electronic components. *Electronics* 9, 6 (2020).
- [38] YOU, K., LIU, Y., WANG, J., AND LONG, M. Logme: Practical assessment of pre-trained models for transfer learning. 12133–12143.