XXIX Seminarium

ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE 2019
Oddział Gdański PTETiS

doi: 10.32016/1.68.07

ANALIZA SCENY PRZY UŻYCIU GŁĘBOKICH SIECI NEURONOWYCH TYPU YOLO

Mateusz MIKOŁAJCZYK[1], Arkadiusz KWASIGROCH[2], Michał GROCHOWSKI[3]

1. Politechnika Gdańska, Wydział Elektrotechniki i Automatyki
   tel.: 694 902 620 e-mail: m.u.mikolajczyk@gmail.com
2. Politechnika Gdańska, Wydział Elektrotechniki i Automatyki
   tel.: 58 347 17 42 e-mail: arkadiusz.kwasigroch@pg.edu.pl
3. Politechnika Gdańska, Wydział Elektrotechniki i Automatyki
   tel.: 58 347 29 04 e-mail: michal.grochowski@pg.edu.pl

Streszczenie: W artykule opisany został problem analizy sceny na obrazach oraz sekwencjach video. Zadanie analizy sceny polega na detekcji, lokalizacji i klasyfikacji obiektów znajdujących się na obrazach. Zaimplementowany system wykorzystuje głęboką sieć neuronową, której struktura oparta została na architekturze YOLO (You Only Look Once). Niskie zapotrzebowania obliczeniowe wybranej architektury pozwala na wykonywanie detekcji w czasie rzeczywistym z zadowalającą dokładnością. W pracy przeprowadzono również badania nad doborem odpowiedniego optymalizatora wykorzystywanego w procesie uczenia. Jako przykładową aplikację wybrano analizę ruchu ulicznego w której skład wchodzi wykrywanie i lokalizacja obiektów takich jak m.in. samochody, motocykle czy sygnalizacja świetlna. Systemy tego typu mogą być wykorzystywane w wszelkiego typu systemach analizy wizyjnej otoczenia np. w pojazdach autonomicznych, systemach automatycznej analizy video z kamer przemysłowych, systemach dozoru czy analizy zdjęć satelitarnych.

Słowa kluczowe: sztuczne sieci neuronowe, detekcja obiektów, przetwarzanie obrazu, głębokie uczenie.

1. WSTĘP

Rozwój głębokich sieci neuronowych umożliwił efektywne wykorzystanie tych algorytmów w wielu dziedzinach takich jak wspomaganie decyzji, analiza obrazów i dźwięków, klasyfikacja, prognozowanie czy wykrywanie uszkodzeń i anomalii. Szczególnie dynamicznie tego typu systemy rozwijają się w obszarze analizy obrazów. Od lat, znacznie przewyższają dokładnością działania klasyczne algorytmy oparte o ręczne przygotowanie ekstraktora cech i prostego klasyfikatora.

W pracy zajęto się nieco trudniejszym zadaniem, mianowicie problemem detekcji obiektów na obrazie. Zadanie detekcji polega na wykryciu obiektu na fotografii czyli określeniu jego rozmiaru oraz położenia (poprzez otoczenie ramką) a następnie jego klasyfikacji (określeniu klasy danego obiektu). Jest to problem znacznie bardziej złożony niż klasyfikacja, ponieważ oprócz określenie jaki obiekt znajduje się na obrazie należy wskazać jego dokładną lokalizację.

Zadanie to jest wielocelowym problemem optymalizacyjnym. Jednym z problemów spotykanych podczas projektowania systemów jest uwzględnienie kompromisu między szybkością obliczeń a dokładnością detekcji. Zwiększenie dokładności działania sieci neuronowej często prowadzi do znacznego wydłużenia czasu obliczeń, uniemożliwiając przetwarzanie danych w czasie rzeczywistym.

Systemy detekcji zbudowane w oparciu o głębokie sieci neuronowe znalazły zastosowanie m.in. w zaawansowanych systemach monitoringu, pojazdach autonomicznych, automatyzacji żmudnej pracy ludzkiej w oznaczaniu wykonanych fotografii.

2. PRZEGLĄD LITERATURY

Zadanie detekcji wymaga zastosowania złożonych algorytmów. Wykorzystanie głębokich sieci neuronowych pozwoliło na osiągnięcie znacznie lepszych rezultatów, w porównaniu do klasycznych metod opartych o ekstraktory cech i uczenie maszynowe.

Jedną z klas algorytmów są metody oparte na propozycjach tzw. regionów zainteresowań ROI (z ang. Regions of Interest). Regiony te to obszary na obrazie, w których prawdopodobnie znajduję się obiekt. Obiekty wyekstrahowane z tych obszarów podawane są na sieć neuronową w celu klasyfikacji. W celu detekcji obiektów na obrazie, części obrazu podawane są wielokrotnie na sieć neuronową, co znacznie wydłuża działanie algorytmu, ale zwiększa jego dokładność działania. Spośród popularnych algorytmów wykorzystujących metody ROI można wymienić algorytm OverFeat [1] czy rodzinę algorytmów Region-based Convolutional Neural Network [2]–[4].

Algorytmy bazujące na regresji oraz klasyfikacji potrzebują jedynie jednorazowego podania obrazu na sieć neuronową, w której dokonywane są wszystkie potrzebne operacje mające na celu detekcję obiektu (wskazanie lokalizacji i klasyfikacja). Powoduje to nawet kilkunastokrotne skrócenie czasu obliczeń co umożliwiając działanie systemów w czasie rzeczywistym. Odbywa się to jednak kosztem utraty dokładności detekcji. Spośród popularnych algorytmów wykorzystujących pojedynczą propagacje przez sieć neuronową można wymienić You Only Look Once w wersji pierwszej [5] i wersji drugiej [6] oraz architekturę Single Shot Detector [7].

3. BAZA DANYCH

Do uczenia głębokiej sieci neuronowej wykorzystano bazę danych COCO (ang. Common Objects in Context) [8],

która zawiera 123 tysiące zdjęć, zawierających 886 tysięcy oznaczonych obiektów spośród 80 wyróżnionych klas.

Wybrana baza danych posiada zdjęcia z dużą ilością bardzo małych obiektów. Zdjęcia te nie są łatwe do analizy ponieważ wiele obiektów jest przedstawionych tylko fragmentarycznie lub w taki sposób, że nawet człowiek ma kłopoty z ich właściwym rozpoznaniem. Ludzie czasami potrafią wywnioskować czym jest dany obiekt poprzez kontekst otoczenia, natomiast sztuczna sieć neuronowa bazuje na zdobytej podczas procesu uczenia, zdolności do rozpoznawania cech, wzorców, kształtów, charakterystycznych dla danej klasy obiektów.

Z bazy danych wybrano 7 klas reprezentujących obiekty spotykane w ruchu ulicznym, które przedstawione zostały w tablicy 1. Liczebność obiektów w poszczególnych klasach nie jest zbilansowana, co może prowadzić do mniejszej dokładności detekcji w przypadku mniejszej liczby obiektów w ramach danej klasy.

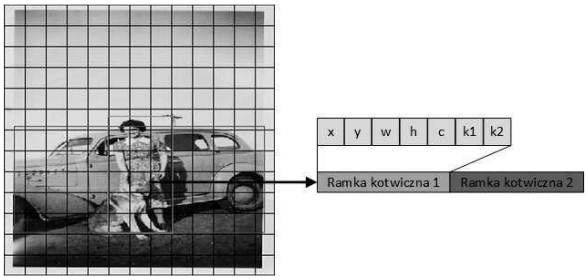
Do każdego zdjęcia dołączone są adnotacje z oznaczeniem lokalizacji oraz klasy znajdujących się na zdjęciu obiektów.

Tablica 1. Wykaz klas wraz z liczbą obiektów zbioru treningowego i walidacyjnego

Klasa	Liczba obiektów treningowych	Liczba obiektów walidacyjnych
Rower	7071	316
Samochód	43663	1932
Motocykl	8697	371
Autobus	6043	285
Ciężarówka	9948	415
Sygnalizacja świetlna	12820	637
Znak stop	1974	75

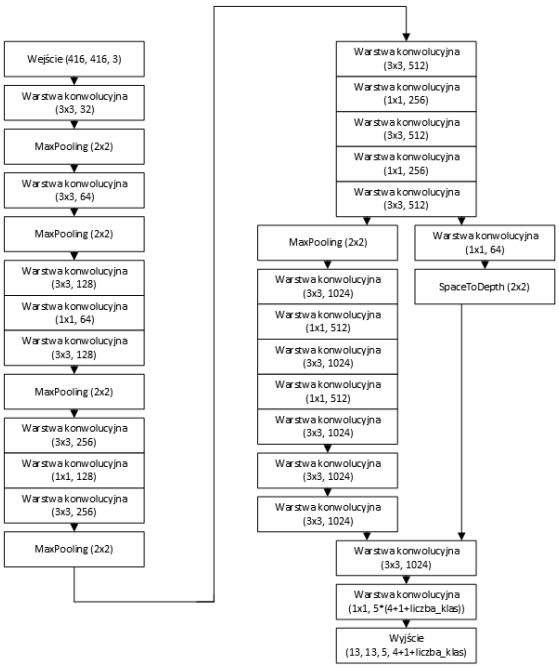
4. ARCHITEKTURA

Spośród wielu architektur postanowiliśmy wybrać architekturę YOLOv2, która umożliwia działanie w czasie rzeczywistym. YOLOv2 jest szybką, w pełni konwolucyjną siecią neuronową, co oznacza, że nie posiada klasycznych warstw w pełni połączonych. Struktura wykorzystuje jedynie jedną propagację przez sieć w celu detekcji, co znacznie przyspiesza jej działanie i umożliwia jej implementację w systemach czasu rzeczywistego. Sieć dokonuje 32-krotnej redukcji rozmiaru, co przy obrazie wejściowym o rozmiarze 416x416 pikseli daje wyjście o rozmiarze 13x13, dzieląc go na równe komórki, które odpowiadają za dane części obrazu. W każdej z tych komórek następuje detekcja 5 (hiperparametr) obiektów, co jest możliwe dzięki zastosowaniu tzw. ramek kotwicznych, czyli stałych ramek, wprowadzonych ręcznie przez programistę, których rozmiary i proporcje dobierane są w zależności od realizowanego problemu. Wykrywane obiekty przypisywane są do ramki kotwicznej, której rozmiary są najbardziej zbliżone do rozmiaru obiektu. Na Rys. 1 przedstawiono detekcję w przypadku zastosowania 2 ramek kotwicznych oraz 2 klas obiektów. Każda ramka zawiera parametry, które opisują umiejscowienie centrum obiektu względem komórki (x – odległość od lewej strony komórki, y – odległość od górnej granicy komórki), odchylenie od wartości ramki kotwicznej (w – różnica w szerokości, h – różnica w wysokości), prawdopodobieństwo obecności obiektu w danej ramce (c), oraz prawdopodobieństwo klasyfikacji dla wszystkich klas przeznaczonych do detekcji (k1, k2). Otrzymane wyniki są relatywne względem komórek oraz predefiniowanych ramek kotwicznych, w celu otrzymania parametrów ramek otaczających obiekty należy je przetworzyć. Dodatkowo, w celu eliminacji niepoprawnych wyników detekcji, należy dokonać końcowego przetwarzania wyników m.in: usunąć detekcje o niskiej pewności wykrycia obiektu oraz wielokrotne wykrycia tego samego obiektu w różnych ramkach.



Rys. 1. Przykład działania architektury YOLOv2

Wykorzystana w badaniach sieć składa się z około 51 milionów parametrów, składa się z 23 warstw konwolucyjnych, w których krok przesunięcia filtru dla wszystkich warstw to jeden piksel. Dodatkowo stosowane jest wypełnienie krawędzi obrazu zerami (tzw. padding), dzięki czemu operacje konwolucji nie powodują redukcji rozmiaru obrazu. Po każdej warstwie konwolucyjnej umiejscowiona jest warstwa normalizacji (batch normalization). Znormalizowane wartości podawane są na funkcję aktywacji typu LeakyReLU. Uproszczony schemat sieci neuronowej został przedstawiony na rys. 2.



Rys. 2. Uproszczona struktura sieci neuronowej YOLO

4.1. Proces uczenia sieci neuronowej

Proces uczenia sieci neuronowej polega na minimalizacji założonej funkcji celu (kosztu) względem jej parametrów, celem znalezienia optymalnych wartości wag sieci. Funkcja kosztu w zadaniach detekcji jest znacznie

bardziej złożona niż w przypadku zadań klasyfikacji. W architekturze YOLOv2 zastosowana funkcja kosztu, która składa się z 4 części odpowiedzialnych za określenie błędów lokalizacji, wymiarów, pewności wykrycia obiektu w danym obszarze oraz klasyfikacji obiektu w danym obszarze. Dla porównania, w przypadku zagadnienia klasyfikacji, funkcja kosztu sprowadza się tylko do ostatniego członu L_{class} Zadanie optymalizacji określone jest następująco:

$$\min_{\theta} L(\theta, X, Y) \tag{1}$$

$$L = L_{x,y} + L_{w,h} + L_{obj} + L_{noobj} + L_{class} \tag{2}$$

gdzie:
θ – parametry sieci neuronowej (wagi i progi),
X – adnotacje i klasy zwracane przez sieć,
Y – adnotacje i klasy opisane w zbiorze uczącym,
$L_{x,y}$ – błąd lokalizacji współrzędnych obiektu,
$L_{w,h}$ – błąd wymiarów obiektu,
L_{obj} – błąd detekcji w przypadku obecności obiektu,
L_{noobj} – błąd detekcji w przypadku braku obiektu,
L_{class} – błąd klasyfikacji.

4.2. Algorytmy optymalizacji

Wybór odpowiedniego algorytmu optymalizacyjnego ma istotny wpływ na proces uczenia oraz ostateczny uzyskany wynik detekcji na zbiorze walidacyjnym. W pracy porównano działanie różnych popularnych algorytmów optymalizacji na proces uczenia sieci neuronowej, osiągane wyniki oraz na czas trwania treningu.

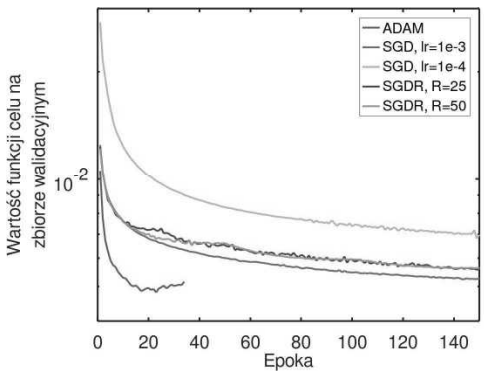
Pierwszym z wybranych algorytmów jest algorytm SGD (Stochastic Gradient Descent) [9]. Jest to jeden z klasycznych algorytmów optymalizacyjnych. Pomimo prostoty działania nadal jest szeroko używany w procesie uczenia głębokich sieci neuronowych.

Algorytm SGD-R (Stochastic Gradient Descent with warm Restarts) [10] jest modyfikacją algorytmu SGD. Wykorzystuje on zmienny współczynnik uczenia, którego wartość zmienia się w sposób kosinusoidalny względem kolejnych epok uczenia. Dodatkowo, co pewną liczbę epok następuje nagły wzrost współczynnika uczenia, którego celem jest unikanie lokalnych minimów.

Algorytm ADAM (Adaptive Moment Estimation) [11] jest wydajnym algorytmem uczącym sieci neuronowe, który dobrze sprawdza się w zadaniach charakteryzujących się dużym zbiorem uczącym oraz przy architekturach zawierających dużą liczbę parametrów. Jest to metoda dedykowana do uczenia głębokich sieci neuronowych. Algorytm określa wartości współczynnika uczenia na podstawie średniej oraz wariancji gradientu. Następnie parametry aktualizowane są wykorzystując odpowiadające im współczynniki uczenia.

5. EKSPERYMENTY

W ramach eksperymentów przetestowano opisane algorytmy uczenia. Na rysunku 3 zaprezentowano wartości funkcji celu podczas uczenia, dla różnych algorytmów optymalizacyjnych. Zaobserwować można wysoką skuteczność algorytmu ADAM – minimalna wartość błędu walidacji została osiągnięta w dużo krótszym czasie od innych algorytmów. Pomimo lepszego działania algorytmu SGD-R w zdaniach klasyfikacji, w zadaniu detekcji wyniki osiągnięte przez ten algorytm były zbliżone do wyników osiągniętych za pomocą algorytmu SGD.



Rys. 3. Wartości funkcji celu podczas uczenia

Najlepszy model, który został nauczony za pomocą algorytmu ADAM został poddany dodatkowym testom, których wyniki przedstawione są w tablicy 2.

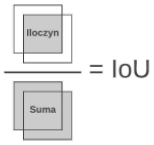
Tablica 2. Uzyskane metryki dla modelu optymalizowanego z wykorzystaniem algorytmu ADAM

Metryka	IoU	Powierzchnia	Maksymalna liczba detekcji	Wynik
Average Precision (AP)	0,50 : 0,95	Dowolna	100	0,151
	0,50	Dowolna	100	0,295
	0,75	Dowolna	100	0,142
	0,50 : 0,95	Mała	100	0,011
	0,50 : 0,95	Średnia	100	0,183
	0,50 : 0,95	Duża	100	0,359
Average Recall (AR)	0,50 : 0,95	Dowolna	1	0,148
	0,50 : 0,95	Dowolna	10	0,168
	0,50 : 0,95	Dowolna	100	0,168
	0,50 : 0,95	Mała	100	0,012
	0,50 : 0,95	Średnia	100	0,197
	0,50 : 0,95	Duża	100	0,394

Metryka AP (Average Precission) określa stosunek poprawnych detekcji do sumy poprawnych detekcji (czyli poprawnym wyznaczeniu lokalizacji i klasyfikacji) oraz fałszywych detekcji obiektów. Wysoka wartość współczynnika AP świadczy o tym, że otrzymany system nie generuje wielu fałszywych predykcji. Wartość IoU jest miarą dokładności określenia położenia oraz rozmiaru obiektu. Miara obliczana jest na podstawie rzeczywistej ramki obiektu oraz ramki zwracanej przez sieć neuronową. Zdefiniowana jest jako stosunek iloczynu (część wspólna) ramek do ich sumy, co zostało przedstawione na rysunku 4. Na podstawie wyników zawartych w tablicy 2 zaobserwować można, że wyższe wyniki osiągane są dla mniejszych wartości IoU, co oznacza że dopasowanie ramek do wymiarów obiektów nie jest idealne. System uzyskał gorsze wyniki dla małych obiektów, co było spodziewanym rezultatem. Jest to cecha sieci YOLO, której szybsze działanie uzyskane jest kosztem gorszej detekcji małych obiektów [12].

Metryka AR (Average Recall) określa stosunek poprawnych detekcji do sumy poprawnych detekcji oraz braków detekcji obiektów. Wysoka wartość współczynnika

AR świadczy o tym, że otrzymany system nie pomija obiektów znajdujących się na danym obrazie. Ilość obiektów znajdujących się na obrazie nie wpływa w znaczący sposób na wartości metryki AR. W przypadku obiektów o małej powierzchni (poniżej 32x32 px) uzyskiwane wyniki są znacznie niższe.



Rys. 4. Graficzne przedstawienie wartości IoU

Przykładowe uzyskane detekcje przedstawione zostały na rysunkach 5 i 6. Uzyskana dokładność jest zadowalająca, duże obiekty są wykrywane poprawnie.

System detekcji został stworzony w środowiskach Keras oraz Tensorflow. Uczenie trwało 11h i zostało przeprowadzono na jednostce obliczeniowej wyposażonej w kartę graficzną GeForce 980Ti. W przypadku detekcji na sekwencjach video uzyskano szybkość przetwarzania 10 klatek na sekundę przy wykorzystaniu karty graficznej Tesla K80.

Rys. 5. Przykładowa detekcja

Rys. 6. Przykładowa detekcja

6. PODSUMOWANIE

Zaproponowany system umożliwia uzyskanie zadowalających wyników detekcji przy czasie obliczeń umożliwiającym przetwarzanie materiałów video w czasie rzeczywistym. Program poprawnie dokonuje detekcji obiektów o dużym polu powierzchni, umożliwiając zastosowanie systemu w wielu systemach wizyjnych. Obiekty duże, nieprzysłonięte, znajdujące się na pierwszym planie obrazu są poprawnie lokalizowane oraz klasyfikowane. W przypadku detekcji obiektów o małej powierzchni detekcje nie są zadowalające, co wynika z kompromisu między jakością a szybkością działania. Możliwe jest zwiększenie dokładności predykcji kosztem czasu propagacji poprzez zastosowanie innych architektur takich jak Faster R-CNN czy Mask R-CNN.

Duży wpływ na jakość działania sieci oraz czas treningu ma dobór odpowiedniego algorytmu optymalizującego. Zastosowanie algorytmu ADAM pozwoliło na znaczne skrócenie czasu treningu sieci neuronowej.

7. BIBLIOGRAFIA

1. P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, i Y. LeCun, „OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks”, *ArXiv13126229 Cs*, 2013.
2. R. Girshick, J. Donahue, T. Darrell, i J. Malik, „Rich feature hierarchies for accurate object detection and semantic segmentation”, *ArXiv13112524 Cs*, 2013.
3. R. Girshick, „Fast R-CNN”, *ArXiv150408083 Cs*, kwi. 2015.
4. S. Ren, K. He, R. Girshick, i J. Sun, „Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, *ArXiv150601497 Cs*, 2015.
5. [„YOLO: Real-Time Object Detection”. Dostępne na: https://pjreddie.com/darknet/yolov2/. [Udostępniono: 12.2018].
6. J. Redmon i A. Farhadi, „YOLO9000: Better, Faster, Stronger”, *ArXiv161208242 Cs*. 2016.
7. W. Liu *i in.*, „SSD: Single Shot MultiBox Detector”, *ArXiv151202325 Cs*, t. 9905, s. 21–37, 2016.
8. T.-Y. Lin *i in.*, „Microsoft COCO: Common Objects in Context”, *ArXiv14050312 Cs*, 2014.
9. S. Ruder, „An overview of gradient descent optimization algorithms”, *ArXiv160904747 Cs*, 2016.
10. Loshchilov i F. Hutter, „SGDR: Stochastic Gradient Descent with Warm Restarts”, *ArXiv160803983 Cs Math*, 2016.
11. D. P. Kingma i J. Ba, „Adam: A Method for Stochastic Optimization”, *ArXiv14126980 Cs*, grudz. 20
12. J. Huang *i in.*, „Speed/accuracy trade-offs for modern convolutional object detectors”, w *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017

SCENE ANALYSIS USING YOLO NEURAL NETWORK

The paper describes the problem of scene analysis in images and video sequences. The task of scene analysis is to detect, locate and classify objects in images. As an example of an application, traffic analysis was chosen, which includes the detection and location of objects such as cars, motorcycles or traffic lights. The implemented system uses a deep neural network, whose structure is based on the YOLO (You Only Look Once) architecture. Low computing requirements of the chosen architecture allows performing real-time detection with satisfactory accuracy. The work also included a study on the selection of an appropriate optimizer used in the learning process. The program correctly detects objects with a large surface area, allowing the system to be used in traffic analysis. The work also showed that using the ADAM algorithm allowed significantly shorten the training time of the neural network. Systems of this type can be used in many types of video analysis systems such as autonomous vehicles, automatic video analysis systems with CCTV cameras, surveillance systems or satellite image analysis.

Keywords: artificial neural networks, object detection, image processing, deep learning.