

Studium Magisterskie

Kierunek: Analiza Danych – Big Data

Marek Kuś

Nr albumu 82394

**Zastosowanie konwolucyjnych sieci neuronowych w klasyfikacji znaków drogowych**

Praca magisterska

napisana w Katedrze/Instytucie

……………………………………………

pod kierunkiem naukowym

Dr Sebastiana Zająca

Warszawa 20…..

Spis treści

[Wstęp 5](#_Toc134354965)

[Rozdział I. Historia i przegląd literatury 7](#_Toc134354966)

[I.1 Znaki drogowe i rozwój pojazdów autonomicznych 7](#_Toc134354967)

[I.2 Rozwój Computer Vision i konwolucyjnych sieci neuronowych 10](#_Toc134354968)

[I.3 Nowoczesne CNN i ich implementacje w benchmarku GTSRB 15](#_Toc134354969)

[Rozdział II. Definicja pojęć 20](#_Toc134354970)

[II.1 Sieć neuronowa (ANN)- koncept 20](#_Toc134354971)

[II.2 Konwolucyjna sieć neuronowa (CNN) – charakterystyka 23](#_Toc134354972)

[Rozdział III. Charakterystyka danych 28](#_Toc134354973)

[III.1 Podstawowe informacje o zbiorze 28](#_Toc134354974)

[III.2 Pre-processing 33](#_Toc134354975)

[Rozdział IV. Metodyka 35](#_Toc134354976)

[Rozdział V. Wyniki 35](#_Toc134354977)

[Rozdział VI. Wnioski 35](#_Toc134354978)

[Zakończenie 36](#_Toc134354979)

[LITERATURA 38](#_Toc134354980)

[Spis tabel 41](#_Toc134354981)

[Spis rysunków 41](#_Toc134354982)

[Streszczenie 42](#_Toc134354983)

[Summary 42](#_Toc134354984)

**Wykaz skrótów stosowanych w pracy**

WHO World Health Organisation

ADAS Advances Driver Assistance Systems

ONZ Organizacja Narodów Zjednoczonych

CV Computer Vision

CNN Convolutional Neural Network

CPU Central Processing Unit

GPU Graphical Processing Unit

ANN Artificial Neural Network

Wstęp

W raporcie o zapobieganiu obrażeń z ruchu drogowego Światowa Organizacja Zdrowia ostrzegała, że wypadki drogowe mogą być dużym zagrożeniem w dziedzinie zdrowia publicznego i rozwoju oraz że problem ten będzie narastał, jeśli nie zostaną podjęte odpowiednie kroki[[1]](#footnote-1). Szacuje się, że co roku w wyniku wypadków drogowych ginie około 1,3 mln osób[[2]](#footnote-2). W związku z tym, Zgromadzenie Ogólne ONZ postawiło sobie za cel zmniejszenia globalnej liczby zgonów i obrażeń wynikających z wypadków drogowych do 2030 r o 50 procent. Jest wiele narzędzi, z których pomocą świat będzie dążył do ograniczenia tego problemu. Jednym z takich środków są zaawansowane system wspomagania kierowcy, które coraz częściej możemy odnaleźć w pojazdach. Jest to każde elektroniczne urządzenie, które wspomaga kierowcę w prowadzeniu lub parkowaniu pojazdu. Do najczęściej spotykanych należą systemy wykrywania martwego pola, asystent parkowania, asystent pasa ruchu, aktywny tempomat, czy też system rozpoznawania znaków drogowych. Jako że za około 90% wszystkich wypadków drogowych odpowiedzialne są ludzkie błędy spowodowane zmęczeniem, nieuwagą lub sennością[[3]](#footnote-3), ADAS wydają się być idealnym narzędziem do ograniczanie lub w przyszłości całkowitego wyeliminowania ludzkiego błędu. Z badań wynika, że te technologie mogą zapobiec nawet 2,7 mln kolizji rocznie w samych Stanach Zjednoczonych[[4]](#footnote-4). Systemy ADAS czynią więc samochody bezpieczniejszymi i inteligentniejszymi, przyczyniając się do realizacji wizji bezkolizyjnej przyszłości. Praca ta skupia się nad jednym z tych systemów, a mianowicie systemem rozpoznawania znaków drogowych. Może on działać na różne sposoby, na przykład korzystając z informacji GPS, jednak w ostatnich latach coraz więcej uwagi poświęca się w tym zakresie konwolucyjnym sieciom neuronowym, które wraz z rozwojem technologii okazały się bardzo skutecznym sposobem identyfikacji i klasyfikacji obrazów w tym między innymi znaków drogowych.

Celem tej pracy jest więc pokazanie jak za pomocą deep learningu, a dokładnie konwolucyjnym sieci neuronowych, można stworzyć system rozpoznający i odpowiednio klasyfikujący znaki drogowe, tym samym przyczyniając się do poprawy bezpieczeństwa na drogach, zmniejszając liczbę wypadków i tym samym ich skutki ekonomicznie i społeczne.

Kończąc wprowadzenie, warto byłoby również zaprezentować strukturę pracy. W rozdziale 1 dokonano przeglądu literatury i historii omawianego zagadnienia, ponieważ z całą pewnością takie zastosowanie deep learningu było już wcześniej omawianie. Istnieją jednak różne podejścia w kontekście używanych modeli uczenia maszynowego, których aspekty mogą przydać się przy tworzeniu tej pracy i pomóc zrozumieć wybór konwolucyjnych sieci neuronowych. Rozdział 2 skupia się zdefiniowaniu pojęć, wykorzystywanych w celu rozwiązania przedstawionego problemu. W kolejnym rozdziale 3 opisane zostały dane, z których korzystać będzie stworzony model, aby następnie móc wykorzystać go do klasyfikacji znaków na drodze. Rozdział 4 przedstawia natomiast uzasadnienie wybranej metody, która rozwiązywałaby badany problem. W ostatnim rozdziale znajdują się wnioski z pracy, implementacji i zastosowania modelu na danych spoza zbioru treningowego i możliwe dalsze kierunki rozwoju.

# Historia i przegląd literatury

Rozdział ten prezentuje jak przez lata rozwijany był problem wpierw detekcji, a potem rozpoznawania i klasyfikacji znaków drogowych, oraz jak użyte zostały sieci konwolucyjne do rozwiązania tego zagadnienia. Przedstawia także kluczowe prace w tematyce computer vision na przestrzeni lat.

## Znaki drogowe i rozwój pojazdów autonomicznych

Znaki drogowe nabrały ogromnego znaczenia wraz z rozwojem motoryzacji. Jeden z pierwszych współczesnych systemów znaków drogowych został opracowany przez włoski Touring Club w 1895 roku[[5]](#footnote-5). Jednak podstawowe wzory większości znaków drogowych zostały ustalone przez Międzynarodowy Kongres Drogowy, który odbył się w Rzymie w 1908 roku. Rok później, dziewięć rządów państw europejskich uzgodniło stosowanie czterech symboli obrazkowych, oznaczających "wyboje", "zakręty", "skrzyżowania" i "przejazdy kolejowe". Intensywne prace nad międzynarodowymi znakami drogowymi prowadzone w latach 1926-1949 doprowadziły ostatecznie do opracowania europejskiego systemu znaków drogowych. Do 1949 roku Europa przyjęła wspólny system, podczas gdy rząd USA odrzucił go i zaprojektował własny system. Z czasem zmienił on jednak swoje podejście i latach 60-tych XX wieku amerykanie zaczęli implementować do swojego systemu symbole międzynarodowe.

Kolejnym krokiem ku standaryzacji była konwencja wiedeńska o znakach i sygnałach drogowych. Jest to traktat podpisany w 1968 roku, dzięki któremu udało się ujednolicić znaki drogowe w różnych krajach. Około 52 krajów podpisało ten traktat, w tym 31 krajów z Europy. Konwencja szeroko sklasyfikowała znaki drogowe w siedmiu kategoriach oznaczonych literami od A do H. Taka standaryzacja bardzo mocno wspomogła rozwój systemów rozpoznawania znaków drogowych.

Pomimo to rozpoznawanie znaków drogowych nadal jest trudnym problemem. Angażuje on uwagę społeczności zajmującej się zagadnieniami Computer Vision od ponad 30 lat. Według niektórych źródeł, por. Escalera et al. (2011)[[6]](#footnote-6), pierwsze badanie zautomatyzowanego rozpoznawania znaków drogowych zostało przeprowadzone w Japonii w 1984 roku, choć pierwsze prace skupione na detekcji i rozpoznawaniu znaków drogowych, które można znaleźć w bazie Scopus, sięgają lat 1983-1984. I to właśnie w latach 80. i 90. XX wieku nastąpił istotny przełom, kiedy to po raz pierwszy zaczęto poważnie traktować problem wspomagania kierowców za pomocą wizji komputerowej.

W 1987 roku, Laboratorium Rozpoznawania Obrazów na Uniwersytecie w Koblencji we współpracy z Daimler-Benz opracowało klasyfikator przeznaczony do wykrywania i lokalizowania znaków drogowych. Nazwano go „Traffic Sign Classificator”, w skrócie TSC. Rozpoznawał on głównie znaki ograniczenia prędkości i wykorzystywał klasyczne algorytmy oparte na segmentacji obrazu oraz dopasowaniu szablonów. Proces rozpoznawania trwał średnio około 0,50 s. Ze względu na opracowywany wówczas sprzęt i związane z tym ograniczone możliwości przetwarzania danych, system nie do końca działał w czasie rzeczywistym. Mimo to zintegrowano go z pojazdem testowym VITA II, który według zespołu potrafił poruszać się autonomicznie na autostradach[[7]](#footnote-7).

W latach 90-tych, rozwój pojazdów autonomicznych przyspieszył wraz z wprowadzeniu mikrokomputerów, które umożliwiały przetwarzanie danych w czasie rzeczywistym w systemach mobilnych. Od tego czasu opracowano wiele metod wykrywania i identyfikacji znaków drogowych, takich jak ekstrakcja krawędzi, segmentacja oparta na kolorze, ekstrakcja wektorów cech, sztuczna sieć neuronowa itp.

Szybki rozwój doprowadził do przeprowadzenia pierwszych poważnych zawodów dla pojazdów autonomicznych: DARPA Grand Challenge w 2004 roku. Celem było przejechanie 150 mil do celu po wyznaczonej trasie. Choć żaden z pojazdów nie ukończył trasy, był to ważny krok w rozwoju pojazdów autonomicznych. W kolejnych latach odbyło się więcej podobnych konkursów, w których to różnym zespołom udało się już stworzyć pojazdy, które pomyślnie ukończyły trasę.

W połowie 2010 r. największe firmy samochodowe, takie jak Ford, Mercedes-Benz i BMW, a także firmy oferujące przejazdy na przykład Uber, zaczęły mocno inwestować w technologią samojezdnych pojazdów. Wzrost ten również widać w pracach poświęconych temu zagadnieniu, których rozkład przez lata można ujrzeć na rysunku numer 1. Od 2005 roku liczba prac zaczęła rosnąć lawinowo, co zbiega się w czasie z pierwszymi zawodami pojazdów autonomicznych i znaczącym wzrostem zainteresowania taką technologią ze strony dużych firm samochodowych.

|  |
| --- |
| Rysunek 1. Liczba prac związanych z detekcją i rozpoznawaniem znaków drogowych |
|  |
| **Źródło:** Opracowanie własne. |

W 21 wieku, wraz z udanymi zastosowaniami głębokiego uczenia, takimi jak rozpoznawanie mowy czy segmentacja semantyczna, metody głębokiego uczenia zostały stopniowo wprowadzone do TSR. Jedną z takich metod były sieci konwolucyjne, rozwijane przez lata w ramach dziedziny computer vision.

## Rozwój Computer Vision i konwolucyjnych sieci neuronowych

Obszar Computer Vision (wizji komputerowej) stał się mocno popularny stosunkowo nie dawno, wraz z takimi wydarzeniami jak wygrana AlexNet w konkursie ImageNet, jednak dziedzina ta sama w sobie nie jest nowym obszarem nauki. Od 60 lat ludzie związani z informatyką starają się znaleźć różne sposoby, aby komputery mogłby wydobyć jakieś istotne znaczenie z obrazów i innych danych wizualnych. Aby łatwiej zrozumieć sieci konwolucyjne, warto poznać ich historię i to jak rozwijały się przez lata.

Historię tą warto zacząć od eksperymentu biologicznego, który miał miejsce w latach 50-tych, dwudziestego wieku. W 1959 roku dwóch neurofizjologów opublikowało jedną z najbardziej wpływowych prac w dziedzinie widzenia komputerowego. Ich publikacja, zatytułowana "Receptive fields of single neurons in the cat's striate cortex" (Pola recepcyjne pojedynczych neuronów w korze wzrokowej kota)[[8]](#footnote-8), opisywała główne właściwości odpowiedzi neuronów kory wzrokowej, a także to, jak doświadczenia wizualne kota kształtowały jego architekturę korową. Naukowcy umieścili elektrody w obszarze pierwotnej kory wzrokowej w mózgu znieczulonego kota i obserwowali, aktywność neuronów w tym regionie podczas pokazywania zwierzęciu różnych obrazów. Ustalili oni dzięki swoim eksperymentom, że w pierwotnej korze wzrokowej znajdują się proste i złożone neurony oraz że przetwarzanie wzrokowe zawsze zaczyna się od prostych struktur, takich jak zorientowane krawędzie. Proste neurony reagują na krawędzie i paski o określonych orientacjach. Natomiast złożone neurony również reagują na krawędzie i paski o określonej orientacji, ale różnią się tym, że te krawędzie i paski mogą być przesunięte w obrębie sceny, a komórka nadal będzie na nie reagować. Na tej podstawie stworzyli oni koncepcję, mówiącą, że proste detektory mogą być sumowane, aby stworzyć bardziej złożone detektory[[9]](#footnote-9), które występują w całym ludzkim systemie wizualnym, a także są fundamentalną podstawą modeli konwolucyjnych.

Kolejnym ważnym punktem w historii rozwoju computer vision było wynalezienie pierwszego cyfrowego skanera obrazów. W 1957 roku Russell Kirsch i jego współpracownicy opracowali urządzenie, które pozwalało przekształcać obrazy w siatki liczb - język binarny, który mogły zrozumieć maszyny. I to właśnie dzięki ich pracy możemy teraz przetwarzać obrazy cyfrowe na różne sposoby.

6 lat później powstała kolejna praca, uważana za jedną z kluczowych w rozwoju nowoczesnych technik rozpoznawania obrazów. Publikacja "Machine perception of three-dimensional solids" (Maszynowe postrzeganie brył trójwymiarowych) opisywała proces wyciągania informacji 3D o obiektach bryłowych z fotografii 2D, czyli redukowania świata wizualnego do prostych kształtów geometrycznych. Celem opracowanego i opisanego przez Roberts’a programu było przetworzenie fotografii 2D na rysunki liniowe, następnie zbudowanie z tych linii reprezentacji 3D i wreszcie wyświetlenie trójwymiarowych struktur obiektów. Autor twierdził, że procesy budowy obrazów 2D do 3D, a następnie wyświetlania 3D do 2D były dobrym punktem wyjścia dla przyszłych badań nad komputerowo wspomaganymi systemami 3D[[10]](#footnote-10).

W latach 60 AI stało się dyscypliną akademicką i wielu naukowców bardzo optymistycznie oceniało przyszłość tej dziedziny. Jeden z profesorów MIT, Seymour Papert, stworzył projekt pod nazwą „Summer Vision Project”, którego ostatecznym celem była identyfikacja obiektów, gdzie faktycznie system będzie w stanie nazwać obiekty poprzez dopasowanie ich do słownika znanych obiektów[[11]](#footnote-11). Projekt nie powiódł się, jednak był według wielu oficjalnym narodzeniem Computer Vision jako dziedziny naukowej.

W latach 80. XX wieku dr Kunihiko Fukushima zainspirowany pracą Hubela i Wiesela na temat komórek prostych i złożonych zaproponował model "neokognitronu", który był samoorganizującą się sztuczną siecią prostych i złożonych komórek, potrafiącą rozpoznawać wzory i będącą niewrażliwą na zmiany ich położenia[[12]](#footnote-12). W modelu tym wprowadzono dwa podstawowe typy warstw: konwolucyjne oraz downsamplingu. Warstwa konwolucyjna zawierała jednostki, których pola receptywne pokrywały fragment warstwy poprzedniej. Dodatkowo pola receptywne miały wektory wag (zwane filtrami). Zadaniem tych filtrów było przesuwanie się po dwuwymiarowych tablicach wartości wejściowych (takich jak piksele obrazu) i, po wykonaniu pewnych obliczeń, wytwarzanie zdarzeń aktywacji (tablic dwuwymiarowych), które miały być wykorzystane jako wejścia dla kolejnych warstw sieci. Natomiast warstwy downsamplingu zawierały jednostki, których pola recepcyjne pokrywały fragmenty poprzednich warstw konwolucyjnych. Taka jednostka zazwyczaj obliczała średnią aktywacji jednostek w swoim fragmencie. Downsampling pomagał poprawnie klasyfikować obiekty, nawet gdy obiekty zostawały przesunięte. Neokognitron Fukushimy jest prawdopodobnie pierwszą w historii siecią neuronową, która zasługuje by być zaliczoną do modelu deep learningowego.

Jedną z pierwszych sieci konwolucyjnych była także stworzona przez Alexa Waibela w 1987 roku sieć neuronowa z opóźnieniem czasowym (TDNN). Osiągnęła ona tak zwany „shift invariance” (niezmienność na przesunięcia wejściowego obrazu) poprzez wykorzystanie współdzielenia wag w połączeniu z treningiem metodą wstecznej propagacji. W ten sposób, wykorzystując również strukturę piramidową jak w neokognitronie, była w stanie wykonać globalną optymalizację wag zamiast lokalnej[[13]](#footnote-13).

2 lata później propagacja wsteczna została zaimplementowana do algorytmu Fukushimy przez francuskiego naukowca Yanna LeCun[[14]](#footnote-14). Po kilku latach pracy nad projektem, LeCun wydał LeNet-5 - pierwszą nowoczesną, konwolucyjną sieć neuronową, wprowadzając kilka istotnych składników, które do dziś używa się w CNN-ach, a sama sieć została zastosowana przez kilka banków do rozpoznawania odręcznych numerów na zdigitalizowanych czekach. Poza tym jego praca zaowocowała stworzeniem zbioru danych MNIST dotyczących pisma ręcznego - prawdopodobnie najbardziej znanego wzorcowego zbioru danych w uczeniu maszynowym.

W międzyczasie inna grupa naukowców pod przewodnictwem Yamaguchi wprowadziła koncepcję max pooling’u[[15]](#footnote-15). Co prawda ich praca skupiła się na rozpoznawaniu izolowanych słów, jednakże max pooling jest wykorzystywany do dzisiaj w sieciach CNN do próbkowania reprezentacji wejściowej obrazu.

W latach 90 zaczęto również podejmować próby rozwiązania problemu grupowania percepcyjnego. Naukowcy chcieli skłonić algorytmy do podzielenia obrazów na sensowne części, aby automatycznie określić, które piksele na obrazie należą do siebie i odróżnić obiekty od ich otoczenia za pomocą algorytmu teorii grafów[[16]](#footnote-16).

Co więcej, w tym czasie wielu badaczy zaprzestało prób rekonstrukcji obiektów poprzez tworzenie ich modeli 3D i zamiast tego skierowało swoje wysiłki w stronę rozpoznawania obiektów na podstawie cech. Szczególnym tego przykładem była praca Davida Lowe'a "Object Recognition from Local Scale-Invariant Features" (Rozpoznawanie obiektów na podstawie lokalnych cech niezależnych od skali). Opisano w niej system rozpoznawania obiektów, który wykorzystuje nową klasę lokalnych cech obrazu, które są niepodatne na skalowanie, translację i rotację obrazu oraz częściowo niepodatne na zmiany oświetlenia[[17]](#footnote-17). Według autora cechy te są podobne do neuronów w dolnej korze skroniowej, wykorzystywanych przez naczelne do rozpoznawania obiektów. Eksperymentalne wyniki pokazały, że rozpoznawanie obiektów mogło być osiągnięte w zatłoczonych obrazach przy czasie obliczeń poniżej 2 sekund.

Niedługo potem, bo w 2001 roku, stworzony zostały pierwszy system wykrywania twarzy, działający w czasie rzeczywistym. Algorytm stworzony przez Paula Viola i Michaela Jones wprowadził nową reprezentację obrazu zwaną "obrazem integralnym", która pozwalała na bardzo szybkie obliczanie kluczowych cech. Drugim jego atrybutem było wybieranie niewielkiej liczby krytycznych cech wizualnych z większego zbioru dzięki czemu osiągano niezwykle wydajny klasyfikator oraz trzecim wkładem była metoda łączenia coraz bardziej złożonych klasyfikatorów w tak zwaną "kaskadę", która pozwalała na szybkie odrzucenie regionów tła obrazu przy jednoczesnym przeznaczeniu większej ilości obliczeń na obiecujące regiony podobne do obiektów[[18]](#footnote-18).

Wraz z ciągłym rozwojem, konwolucyjne sieci neuronowe wymagały coraz to większych mocy obliczeniowych. Jednym ze sposobów pokonania tego problemu, okazało się wykorzystanie procesorów graficznych. W 2004 roku koreańscy badacze wykazali, że sieci neuronowe mogą działać o wiele szybciej na układach GPU. Ich implementacja była 20 razy szybsza od analogicznej implementacji na CPU[[19]](#footnote-19). Natomiast pierwsza implementacja konwolucyjnej sieci neuronowej na GPU została opisana 2 lata później i okazała się 4 razy szybsza od analogicznej implementacji na CPU[[20]](#footnote-20).

Około 2012 roku konwolucyjne sieci neuronowe odnotowały ogromny wzrost popularności, który trwa do dziś. Stało się to za sprawą sieci konwolucyjnej zwanej AlexNet, która osiągnęła najwyższą wydajność w etykietowaniu obrazów w wyzwaniu ImageNet. Twórca sieci, Alex Krizhevsky wraz z innymi badaczami opublikowali pracę "ImageNet Classification with Deep Convolutional Neural Networks" (Klasyfikacja ImageNet za pomocą głębokich neuronowych sieci konwolucyjnych) opisującą zwycięski model AlexNet. Model był podobny w swojej architekturze do LeNet-5 Yanna LeCuna i osiągał poziom błędu 16,4%[[21]](#footnote-21). Na dzisiejsze standardy wynik może nie wydawać się przełomowy jednak jeszcze w 2010-11 roku wskaźnik błędnie zaklasyfikowanych obrazów w tym samym konkursie wynosił mniej więcej 26%. Był to więc moment przełomowy dla CNN-ów. Model ten położył podwaliny pod kolejne sieci CNN, gdzie po warstwie konwolucyjnej następowała funkcja aktywacji, a następnie max pooling’u. Dzięki temu w kolejnych latach wskaźniki błędów w klasyfikacji obrazów w konkursie ImageNet spadły do kilku procent, a zwycięzcami, od 2012 roku regularnie były konwolucyjne sieci neuronowe.

W ostatnich latach wprowadzano nowe warstwy jak blok resztkowy (residual block), który umożliwił uczenie jeszcze głębszych sieci neuronowych. Zespół złożony z takich sieci resztkowych osiągnął błąd klasyfikacji na poziomie 3,57% na zbiorze testowym ImageNet. Dzięki temu zdobył 1 miejsce w konkursie klasyfikacyjnym ImageNet w 2015[[22]](#footnote-22).

Widzenie komputerowe przeszło długą drogę w ciągu ostatnich kilku dekad. Pomimo ostatnich postępów, które są imponujące, nadal pozostaje wiele do odkrycia. Wciąż tworzone są nowe architektury sieci konwolucyjnych, potrafiące coraz lepiej rozpoznawać i klasyfikować obiekty. Jednak już teraz istnieje wiele gałęzi gospodarki, które znalazły sposoby na zastosowanie systemów Computer Vision oraz konwolucyjnych sieci neuronowych do rozwiązywania rzeczywistych problemów. Jedną z takich branż jest branża motoryzacyjna.

## Nowoczesne CNN i ich implementacje w benchmarku GTSRB

Przedstawione prace z zakresu konwolucyjnych sieci neuronowych i problemu klasyfikacji znaków drogowych, były po części wprowadzeniem do niniejszego podrozdziału, w którym przedstawione zostaną publikacje łączące oba te zagadnienia. Posiłkując się raz jeszcze bazą danych Scopus można zauważyć, że wyszukując dokumenty zawierające w sobie takie wyrazy jak konwolucyjne sieci neuronowe i klasyfikacja znaków drogowych otrzymujemy przeszło 300 publikacji.

|  |
| --- |
| Rysunek 2. Liczba publikacji związanych z klasyfikacją znaków drogowych za pomocą sieci konwolucyjnych |
|  |
| **Źródło:** Opracowanie własne. |

Liczby z rysunku nr 2 wskazują na stosunkowo szybkie tempo wzrostu prac na ten temat w szczególności po roku 2014. Można również zauważyć, że temat ten jest stosunkowo nowy, jako że w bazie nie znajdują się prace starsze niż z 2011 roku. Zbiega się to w czasie ze wspomnianym w podrozdziale 2 stworzeniem sieci konwolucyjnej zwanej AlexNet. Warto więc przyjrzeć się niektórym z tych prac, aby spojrzeć na różne podejścia do przedstawionego w niniejszej pracy problemu.

Rozpoczynając od pracy szwajcarskich badaczy z 2012 roku, która opisuje zwycięski model w niemieckim benchmarku rozpoznawania znaków drogowych (GTSRB). Był to wtedy jedyny model, który osiągnął lepszy od człowieka współczynnik rozpoznawalności na poziomie 99,46%[[23]](#footnote-23). Jako podstawowy blok konstrukcyjny wykorzystali oni głęboką hierarchiczną sieć neuronową, w której naprzemiennie występują warstwy konwolucyjne z warstwami max-poolingu, widoczne na rysunku nr 3.

|  |
| --- |
| Rysunek 3. Architektura sieci DNN |
|  |
| **Źródło:** Ciresan i in., *Multi-column deep neural network for traffic sing classification*. |

Każda warstwa otrzymywała połączenia tylko od swojej poprzedniej warstwy. Opisali to jako ogólny, hierarchiczny ekstraktor cech, który mapował piksele obrazu wejściowego na wektor cech, który był klasyfikowany przez kilka, zwykle 2 lub 3, w pełni połączone warstwy. Wszystkie regulowane parametry były wspólnie optymalizowane poprzez minimalizację błędu klasyfikacji na zbiorze treningowym. Ostatecznie połączyli oni kilka sieci w jedną wielokolumnową sieć poprzez uśrednienie informacji wyjściowej z każdej sieci, będących odpowiednikami kolumn, co dodatkowo zwiększyło wydajność rozpoznawania. Była to więc sieć dająca bardzo dobre rezultaty jednak dość skomplikowana w swojej architekturze i potrzebująca bardzo dużo mocy obliczeniowej, a więc też mocnych jednostek GPU.

Dziś jednak nie potrzeba dużych mocy obliczeniowych, żeby wyszkolić sieć konwolucyjną, dającą porównywalne wyniki predykcji. Przykładem tego jest praca „DeepThin: A novel lightweight CNN architecture for traffic sign recognition without GPU requirements” (DeepThin: Nowa lekka architektura CNN do rozpoznawania znaków drogowych bez wymagań dotyczących GPU). W proponowanej przez autorów architekturze, każda warstwa konwolucyjna zawiera mniej niż 50 cech, co pozwala ma szybkie wytrenowanie sieci neuronowej nawet bez pomocy procesora graficznego. Składa się ona z czterech warstw konwolucyjnych, dwóch nakładających się na siebie warstw max-pooling’u, po których następuje pojedyncza w pełni połączona warstwa ukryta, biegnącą do warstwy softmax w celu uzyskania rozkładu prawdopodobieństwa na 𝑛 etykiet klas[[24]](#footnote-24). Jej pełna architektura została przedstawiona na rysunku nr 4.

|  |
| --- |
| Rysunek 4. Architektura sieci CNN |
|  |
| **Źródło:** Haque i in., *DeepThin: A novel lightweight CNN architecture for traffic sign recognition without GPU requirements*. |

Natomiast rezultaty jakie osiągnęła sieć na niemieckim benchmarku rozpoznawania znaków drogowych (GTSRB) były bardzo zbliżone do omawianej sieci DNN. Dokładność klasyfikacji wyniosła średnio 99,41%, co pokazuje, że wykorzystanie o wiele mniejszej liczby parametrów i zasobów nie oznacza znacznej utraty dokładności.

Są też prace, w których osobno budowana jest sieć do wykrywania znaków drogowych i osobna do ich klasyfikacji. Mowa o publikacji “A practical approach for detection and classification of traffic signs using Convolutional Neural Networks” (Praktyczne podejście do wykrywania i klasyfikacji znaków drogowych z wykorzystaniem konwolucyjnych sieci neuronowych). Autorzy proponują dwie sieci CNN: do wykrywania i klasyfikacji znaków drogowych. Ich celem było stworzenie lekkiej sieci, która mogłaby być wykorzystywana w czasie rzeczywistym. Wpierw stworzyli sieć konwolucyjną do detekcji, a następnie przeprojektowali ją używając tylko warstw konwolucyjnych i max-pooling’u, co pozwoliło na stosowanie jej na obrazach o dowolnych rozmiarach[[25]](#footnote-25). Architektury sieci zostały przedstawione na rysunku nr 5 i 6.

|  |
| --- |
| Rysunek 5. Proponowana architektura sieci CNN do wykrywania znaków drogowych |
|  |
| **Źródło:** Habibi i in., *A Practical Approach for Detection and Classification of Traffic Signs Using Convolutional Neural Networks*. |

|  |
| --- |
| Rysunek 6. Proponowana architektura sieci CNN do klasyfikacji znaków drogowych |
|  |
| **Źródło:** Habibi i in., *A Practical Approach for Detection and Classification of Traffic Signs Using Convolutional Neural Networks*. |

Średnia dokładność dla przeprowadzonych prób wyniosła 99,34%. Benchmarkiem był zbór danych German Traffic Sign Detection Benchmark.

Istnieje jeszcze wiele innych prac, które przedstawiają inne podejścia do detekcji i klasyfikacji znaków drogowych, oparte na innych architekturach sieci CNN, korzystające z nowych warstw, jednak nie sposób jest je wszystkie przytoczyć.

**Podsumowanie**

Jak można zauważyć, obecnie konwolucyjne sieci neuronowe posiadają różne architektury i stopnie skomplikowania. Czasem dokładane są nowe warstwy, jednak podstawowe bloki budujące są w większości takie same dla każdej sieci. Także większość najnowszych sieci osiągnęła niemal perfekcyjny wskaźnik dokładności klasyfikacji równy blisko 100%. Jednak nadal jest miejsce na poprawę w kwestii wydajności i obciążenia obliczeniowego różnych sieci, tak aby nie wymagały one mocnych procesorów GPU i mogły być implementowane w słabych urządzeniach.

# Definicja pojęć

Po przedstawieniu literatury i historii, warto przejść do objaśnienia pojęć, które po części już pojawiły się w rozdziale I, a także pojawią się przy omawianiu metodyki i budowy konwolucyjnej sieci neuronowej w późniejszych rozdziałach. Zatem w rozdziale tym zostaną wyjaśnione zagadnienia, wokół których skupią się ta praca.

## Sieć neuronowa (ANN)- koncept

Konwolucyjne sieci neuronowe są wyspecjalizowanym typem sztucznych sieci neuronowych stąd, aby je zrozumieć warto zacząć od zwykłych sieci neuronowych. ANN opiera się na zbiorze połączonych jednostek lub węzłów zwanych neuronami, które nawiązują do neuronów w mózgu. Schematyczna ilustracja sztucznego neuronu została przedstawiona na rysunku nr 7. Każde połączenie, podobnie jak synapsy w mózgu, może przekazywać sygnał do innych neuronów, jednak zaczynając od początku schematu. Sztuczny neuron wpierw odbiera sygnały wejściowe (x1, x2, xn), będące liczbami, w których zakodowano jakąś informację. Mogą to być surowe dane wejściowe lub sygnały od innych perceptronów. Następnie sygnały te są mnożone przez wagi (w1, w2, wn), które przyjmują wartości liczbowe i kontrolują poziom ważności każdego wejścia. Tak więc waga zwiększa lub zmniejsza siłę sygnału na połączeniu. Im wyższa wartość wagi, tym ważniejsza informacja wejściowa. Waga jest dostosowywana w trakcie uczenia. W kolejnym kroku neuron gromadzi wszystkie te informacje sumując wszystkie przemnożone przez wagę wartości. Taka ważona suma wejść przyjmuje postać[[26]](#footnote-26):

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

a do tego dodaje się również obciążenie przyjmujące literę *b*. Jej celem jest zapewnienie, by funkcja progowa nie wyniosła zero. A więc w przypadku, gdy wszystkie wejścia x1, x2, ..., xn mają wartość 0, to funkcja jest równa wartości obciążenia. Koniec końców funkcja progowa zwana także funkcją przekazu (transfer function) otrzymuje postać:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

Warto w tym miejscu dodać, że wspomniane wagi oraz obciążenie są inaczej nazywane parametrami modelu sieci neuronowej. Za pomocą procesu uczenia sieć stara się znaleźć ich optymalne wartości.

Następnie ważona suma wejść przekazywana jest do funkcji aktywacji *f*:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

Jej zadaniem jest wprowadzenie nieliniowości do modelu oraz porównanie sumy z zadanym jej progiem. Jeśli próg zostanie przekroczony, funkcja aktywuje neuron i przekaże jego sygnał do następnego sztucznego neuronu. Funkcja ta może przyjmować różne formy o różnych właściwościach. Jest ona rdzeniem sztucznego neuronu. Bez niej neuron przekazywałyby swoje ważone wejścia bez wcześniejszego filtrowania informacji. Dlatego ważny jest odpowiedni dobór funkcji aktywacji do danych wejściowych, aby uzyskać sensowne wyniki.

|  |
| --- |
| Rysunek 7. Sztuczny neuron i jego elementy |
|  |
| **Źródło:** Opracowanie własne. |

Pojedyncze neurony są zwykle łączone w różne warstwy, tworząc sieć połączonych ze sobą warstw. Sygnały w takiej strukturze podróżują od warstwy wejściowej poprzez jedną lub więcej warstw ukrytych do warstwy wyjściowej. Różne warstwy mogą wykonywać różne transformacje na swoich wejściach.

Warstwa wejściowa otrzymuje informacje i przesyła je do warstw ukrytych. Liczba neuronów w tej warstwie musi być równa liczbie atrybutów lub cech, które planuje się użyć w sieci.

Warstwa ukryta znajduje się pomiędzy warstwami wejściową i wyjściową. Zachodzi w niej między innymi proces uczenia, czyli szukania zależności między zmiennymi zarówno tych liniowych jak i nieliniowych. Warstw tych może być wiele w zależności od modelu i liczby danych.

Warstwa wyjściowa dostarcza predykcji. To w niej zwracany jest wyniki.

Taka przykładowa prosta sieć neuronowa została przedstawiona na rysunku nr 8. Sieć składa się z warstwy wejściowej z 3 węzłami źródłowymi, warstwą ukrytą z 4 neuronami i warstwy wyjściowej z 2 neuronami.

|  |
| --- |
| Rysunek 8. Prosta wielowarstwowa sieć neuronowa typu feedforward |
|  |
| **Źródło:** Opracowanie własne. |

Pomiędzy warstwami możliwe są różne sposoby połączeń. Jeśli każdy neuron w warstwie jest połączony ze wszystkimi neuronami w poprzedniej warstwie to jest to w pełni połączona sieć neuronowa. Mogą być także połączenia, w których grupa neuronów w jednej warstwie łączy się z pojedynczym neuronem w kolejnej warstwie, zmniejszając tym samym liczbę neuronów w tej warstwie. Co więcej połączenia w warstwach niekoniecznie są równe. Każde z nich może mieć inną siłę lub wagę. Łącząc wielokrotnie dane neurony między sobą doprowadza się do wzmocnienia połączenia i w ten sposób sieć uczy się, które z nich są najważniejsze.

Proces nauki w sieciach neuronowych jest więc adaptacją sieci do lepszego radzenia sobie z zadaniem poprzez dane treningowe. Dzieje się to dzięki dostosowywaniu wag i obciążeń przy użyciu wybranego algorytmu optymalizacji. Zazwyczaj taki trening rozpoczyna się od ustawienia wszystkich wag w sieci na małe liczby losowe. Następnie dla każdego przykładu wejściowego sieć zwraca jakąś predykcję. Za pomocą wybranej funkcji straty lub kosztu mierzy się różnicę pomiędzy danymi wyjściowymi a danymi pożądanymi. Celem jest zminimalizowanie tego błędu. Proces takiego szkolenia składa się z wielu iteracji, zwanych epokami. Uczenie może zostać zakończone, gdy badanie dodatkowych obserwacji nie zmniejsza w istotny sposób błędu.

Znając już budowę podstawowej sieci neuronowej i sposób jej działania można przejść do specyficznego przypadku takiej sieci, którym jest konwolucyjna sieć neuronowa.

## Konwolucyjna sieć neuronowa (CNN) – charakterystyka

Konwolucyjne sieci neuronowe są bardzo podobne do zwykłych sieci neuronowych omawianych w poprzednim podrozdziale. Różnica występuje w architekturze sieci. W sieci konwolucyjnej zakłada się, że danymi wejściowymi są obrazy, co pozwala zakodować pewne właściwości w ich architekturze. Dzięki temu są one bardziej wydajne w implementacji i znacznie zmniejsza się liczba niezbędnych parametrów w sieci.

Sieci te również składają się z wielu warstw, z czego 3 główne typy to warstwy: konwolucyjne, łączące (pooling) i w pełni połączone (fully connected).

Warstwa konwolucyjna jest pierwszą warstwą, która przyjmuje dane wejściowe. Jej zadaniem jest wydobywanie ważnych cech z obrazów. Dodając każdą kolejną warstwę zwiększa się złożoność sieci, co pozwala identyfikować coraz większe fragmenty obrazu. Wcześniejsze warstwy skupiają się na prostych cechach, takich jak kolory i krawędzie. Wraz z przechodzeniem danych przez kolejne warstwy CNN, zaczyna ona rozpoznawać większe elementy lub kształty obiektu, aż w końcu identyfikuje zamierzony obiekt. Warstwa ta działa więc jak filtr do obrazów. Filtr jest definiowany przez jądro, które składa się z macierzy wartości wag, początkowo wybieranych losowo z np. losowego rozkładu Gaussa. Przykładowy filtr mógłby mieć wymiary 3x3x1 jak na rysunku nr 9.

|  |
| --- |
| Rysunek 9. Działanie warstwy konwolucyjnej |
|  |
| **Źródło:** Opracowanie własne. |

Co więcej obraz również można przedstawić jako macierz wartości danych pikseli jak na rysunku nr 9. Dzięki temu można nałożyć na niego filtr i obliczyć ważoną sumę wartości odpowiednich pikseli obrazu. Następnie filtr jest zwykle przesuwany wzdłuż o 1 piksel i ponownie obliczana jest ważona suma wartości. Ze względu na rozmiar jądra filtra nie można obliczyć wartości brzegowych, dlatego zwykle stosuje się tam wartość wypełnienia, która zazwyczaj wynosi 0. Dzięki temu generowana jest mapa cech takiej samej wielkości jak obraz, która stanie się danymi wejściowymi do kolejnej warstwy.

Warto również dodać, że w warstwie konwolucyjnej często stosuje się wiele różnych jąder filtrów. Każdy filtr wytwarza inną mapę cech, po czym wszystkie są przekazywane do następnej warstwy sieci.

Ponadto zanim dane zostaną przekazane do kolejnej warstwy, zazwyczaj są przekazywane do funkcji aktywacji. Jedną z najczęściej używanych jest funkcja ReLU (Rectified Linear Activation), która posiada następujący wzór:

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Można zauważyć, że funkcja ta zwraca 0, jeśli wartość wejściowa jest ujemna, a w przeciwnym wypadku zwraca wartość wejściową. Zwiększa to nieliniowość w danych, a jednocześnie jest mniej kosztowne obliczeniowo niż inne funkcje aktywacji takie jak tanh lub sigmoidalna.

Kolejną ważną warstwą w sieciach konwolucyjnych jest warstwa poolingu lub inaczej downsamplingu. Jej głównym zadaniem jest zmniejszenie rozmiarów danych, w tym wypadku obrazu, jednocześnie zachowując kluczowe cechy, które zostały wcześniej wyodrębnione. Takie działanie pozwala zmniejszyć liczbę parametrów do wytrenowania przez co potrzebna jest także mniejsza moc obliczeniowej do przetworzenia danych. Dodatkowo pomaga zapobiegać przeuczeniu sieci.

Warstwa łącząca działa podobnie do warstwy konwolucyjnej, czyli na macierz nakładany jest filtr, który przeprowadza odpowiednie kalkulacje a następnie tworzona jest nowa macierz wartości. Różnica polega na tym, że filtr nie posiada wag. Za to przeprowadza się inne obliczenia.

Dwa najczęściej używane sposoby poolingu to max pooling i average pooling. Pierwszy z nich zwraca maksymalną wartość z części obrazu objętej przez filtr i został przedstawiony na rysunku nr 10. Natomiast drugi rodzaj poolingu zwraca średnią wszystkich wartości z części obrazu objętej filtrem.

|  |
| --- |
| Rysunek 10. Działanie warstwy poolingu |
|  |
| **Źródło:** Opracowanie własne. |

Konwolucyjna sieć neuronowa kończy się często warstwą w pełni połączoną (fully connected layer). Łączy ona każdy neuron w jednej warstwie z każdym neuronem w kolejnej warstwie. Jest to więc to samo, co w zwykłej sieci neuronowej. Warstwa ta dokonuje klasyfikacji na podstawie cech wyodrębnionych przez poprzednie warstwy. Aby zapewnić, że wyniki będą sumować się do zera, na końcu sieci stosuje się funkcję SoftMax:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Aby zawrzeć wartości między 0 a 1, funkcja normalizuje dane wyjściowe sieci neuronowej. Normalizacja jest obliczana poprzez podzielenie wartości eksponencjalnej danej klasy przez sumę wartości eksponencjalnej każdej możliwej klasy. Dzięki temu każdej klasie odpowiada dane prawdopodobieństwo.

Oprócz przedstawionych wcześniej warstw stosowane są także warstwy odrzucania (dropout layer) i warstwy spłaszczające (flatten layer). Pierwsza z nich stosowana jest po to, aby zapobiec problemowi overfittingu czyli nadmiernego dopasowania do danych. Poprzez przeuczenie model nie będzie w stanie generalizować, tego co nauczył się na zbiorze treningowym. W związku z tym warstwa dropoutu losowo eliminuje utworzone mapy cech, aby model nie polegał zbyt mocno na żadnej z nich w procesie uczenia. Natomiast warstwa spłaszczająca służy do przekształcenia map cech w wektor wartości. W takiej formie może ono zostać przekazane jako dane wejściowe do warstwy w pełni połączonej.

**Podsumowanie**

Podsumowując, omówione zostały pojęcia sztucznych sieci neuronowych (ANN) i konwolucyjnych sieci neuronowych (CNN). ANN opierają się na połączonych ze sobą jednostkach zwanych neuronami, które są inspirowane neuronami w mózgu. Każde połączenie może przekazywać sygnał do innych neuronów, a każdy neuron otrzymuje sygnały wejściowe, przeprowadza odpowiednie operacje i wysyła sygnał dalej. Proces ten jest powtarzany przez wiele warstw, aby uzyskać ostateczny wynik.

Natomiast CNN są rodzajem ANN, które są specjalnie stworzone do przetwarzania obrazów. Wykorzystują one warstwy konwolucyjne, warstwy łączące oraz warstwy w pełni połączone do uczenia się i klasyfikowania cech w obrazach. W warstwach konwolucyjnych filtr nakładany jest na obraz w celu uzyskania mapy cech. Warstwa łącząca zmniejsza rozmiar map cech, a warstwa w pełni połączona pozwala dostarczyć ostateczną predykcję.

Oczywiście nie sposób przekazać wszystkich informacji na temat CNN w tak krótkim rozdziale. Przedstawione zostały jedynie najważniejsze rzeczy, potrzebne do zrozumienia działania budowanej w dalszych rozdziałach sieci konwolucyjnej. Zanim jednak to nastąpi, omówiony zostanie zbiór danych, na którym będzie szkolona i testowana sieć neuronowa.

# Charakterystyka danych

W rozdziale III zawarta została analiza zbioru danych „German Traffic Sign Recognition Benchmark”, który posłuży do trenowania konwolucyjnej sieci neuronowej. Na początku rozdziału przedstawione zostaną podstawowe informacje o zbiorze GTSRB, takie jak liczba klas znaków drogowych czy rozmiar zdjęć. Zostaną omówione również statystyki dotyczące liczby obrazów w każdej klasie oraz ich rozkładu w zbiorach oraz poruszone zostaną także takie kwestie jak różne warunki oświetlenia, niskiej jakości zdjęcia czy duża zmienność w kształcie i położeniu znaków na zdjęciu.

## Podstawowe informacje o zbiorze

Zbiór niemieckich znaków drogowych od dawna jest jednym z najpopularniejszym zbiorów treningowych dla sieci neuronowych, których zadaniem jest rozpoznawanie znaków drogowych. Powstał on w 2010 roku z około 10 godzinnego materiału wideo, nagranego w czasie jazdy po różnego typu drogach w Niemczech[[27]](#footnote-27). Obrazy znaków drogowych zostały wyciągnięte z obrazów 1360×1024 pikseli uchwyconych przez zamontowaną w pojeździe kamerę.

Zbiór zawiera ponad 40 klas i ponad 50 000 obrazów. Dokładne liczby zostały przedstawione w tabeli nr 1. Co więcej zbiór danych zawiera tylko jeden ścieżkę dla każdego fizycznego znaku drogowego. Ścieżka to sekwencja obrazów pochodzących z jednego, fizycznego znaku drogowego w świecie rzeczywistym. Maksymalna długość takiej ścieżki została ograniczona do 30 obrazów, ponieważ zbadano, że kolejne obrazy powoli mijanego znaku drogowego były do siebie bardzo podobne przez co nie przyczyniały się do różnorodności zbioru danych.

|  |
| --- |
| Tabela 1. Podsumowanie liczebności danych |
| |  |  | | --- | --- | | Liczebność danych | Wartości | | Zbiór treningowy | 39209 | | Zbiór testowy | 12630 | | Liczba klas | 43 |   **Źródło:** Opracowanie własne. |

Na każdym z obrazów zawartych w zbiorze znajduje się tylko jeden znak. Wielkości obrazów różnią się zaczynając od obrazów 15x15 pikseli, a kończąc na obrazach 250x250 pikseli. Przykładowe obrazy ze zbioru treningowego można zobaczyć na rysunku nr 11.

|  |
| --- |
| Rysunek 11. Przykładowe obrazy z treningowego zbioru danych |
|  |
| **Źródło:** Opracowanie własne. |

Warto zauważyć, że obrazy są mocno zróżnicowanie, jeśli chodzi o ich jakość. Część z nich jest dobrze doświetlona, jednak jest też dużo obrazów wygląda jakby była uchwycona późnym wieczorem, przy słabych warunkach oświetleniowych. Oprócz tego w związku z robieniem zdjęć w ruchu, niektóre obserwacje są mocno niewyraźne. Z jednej strony jest to bardzo dobry znak, jako że są to realne warunki w jakich klasyfikator będzie sobie musiał radzić na co dzień. Z drugiej strony będzie to dodatkowe wyzwanie dla sieci neuronowej, aby osiągnąć wysoki wynik poprawnej klasyfikacji.

Po przyjrzeniu się podstawowym wartością warto dokładniej przyjrzeć się zbalansowaniu zbioru. Zaczynając od klas w zbiorze treningowym na rysunku nr 12 można zauważyć, że są one rozłożone mocno nierównomiernie. Na klasę w zbiorze przypada średnio około 912 obrazów. Są jednak klasy, które posiadają ponad 2000 obserwacji, ale są też takie które mają ich poniżej 300. Tak duże zróżnicowanie wynika z częstości poszczególnych znaków na drogach. Przyglądając się bliżej etykietą, okazuje się, że najwięcej obrazów posiadają takie znaki jak ograniczenie prędkości do 30km/h, ograniczenie prędkości do 50 km/h lub znak ustąp pierwszeństwa. Natomiast na drugim końcu spektrum znajdują się takie znaki jak niebezpieczny zakręt w lewo, jedź prosto czy też koniec zakazu wyprzedzania pojazdów powyżej 3.5 tony, który nie często spotyka się na drogach.

|  |
| --- |
| Rysunek 12. Rozkład klas w zbiorze treningowym |
|  |
| **Źródło:** Opracowanie własne. |

Taki sam wykres stworzono dla klas obrazów znajdujących się w zbiorze testowym, co przedstawia rysunek nr 13. Jest on bardzo podobny do rysunku nr 12, z tym, że najbardziej liczne klasy mają powyżej 700 obserwacji, a najmniej liczne nawet poniżej 200. Średnia liczba obrazów na klasę wynosi 294. Pod względem rozkładu wykresy te są bliźniaczo podobne z niewielkimi odchyleniami względem proporcji ze zbioru treningowego w niektórych klasach np. 40.

|  |
| --- |
| Rysunek 13. Rozkład klas w zbiorze testowym |
|  |
| **Źródło:** Opracowanie własne. |

Po przeanalizowaniu rozkładów klas w obu zbiorach można dojść do wniosku, że zbiór ten jest przykładem zbioru niezbalansowanego. Warto mieć to na uwadze przy trenowaniu modelu, ponieważ algorytm, który otrzymuje znacznie więcej przykładów z jednej klasy, może być wobec niej stronniczy.

Przed przejściem do pre-processingu warto również przyjrzeć się lepiej wymiarom zdjęć i ich zróżnicowaniu w zbiorze. Na rysunku nr 14 został przedstawiony rozkład wysokości i szerokości wszystkich obrazów w zbiorze treningowym. Oba wymiary zdają się mieć bardzo podobny rozkład. Na wykresie widać również jak wiele unikalnych wartości przyjmują obrazy. Jednak zdecydowana większość z nich posiada wymiary poniżej 100x100 pikseli, gdzie średnia wynosi około 30 pikseli dla obu wymiarów.

|  |
| --- |
| Rysunek 14. Rozkład wymiarów obrazów w zbiorze treningowym |
|  |
| **Źródło:** Opracowanie własne. |

Podobnie do rysunku 14 wygląda rysunek 15 przedstawiający również rozkład wysokości i szerokości wszystkich obrazów, ale w zbiorze testowym. Rozkład ten podobny do tego w zbiorze treningowym, co jest pożądanym rezultatem.

|  |
| --- |
| Rysunek 15. Rozkład wymiarów obrazów w zbiorze testowym |
|  |
| **Źródło:** Opracowanie własne. |

Przy tak zróżnicowanych wymiarach obrazów warto je ujednolicić, jednak w taki sposób, aby stracić jak najmniej informacji.

## Pre-processing

Pierwszym z kroków w pre-processingu danych było zmienienie wymiarów obrazów przy ładowaniu danych, aby ujednolicić ich wymiary. Wybrano rozmiar 32x32 pikseli, jako że wymiary większości obrazów przyjmują wartości wokół wybranego rozmiaru. Taki zabieg ma na celu przede wszystkim przyśpieszenie trenowania CNN. Dwukrotnie większy obraz wejściowy wymaga, aby sieć neuronowa uczyła się z czterokrotnie większej liczby pikseli, a to zwiększa czas treningu architektury. Oprócz tego zmniejszenie rozmiaru większych obrazów, aby dopasować ich rozmiar do mniejszych obrazów, jest często lepszym pomysłem niż zwiększenie rozmiaru małych obrazów, aby były większe. Co więcej wiele modeli wymaga, aby obrazy służące jako dane do uczenia były tej samej wielkości.

Oprócz tego obrazy trzeba przygotować tak aby mogły one posłużyć jako dane wejściowe do sieci konwolucyjnej. W związku z tym przekonwertowano każdy z nich na macierze. Dodatkowo na etykietach obrazów zastosowano one hot encoding czyli proces konwersji danych kategorycznych dzięki któremu mogą one być dostarczone do sieci neuronowej i prawdopodobnie poprawić predykcję. Proces ten polegna na przekonwertowaniu każdej wartości kategorycznej na nową i przypisaniu wartości binarnej 1 lub 0 do kolumn. Dzięki temu każda wartość jest reprezentowana jako wektor binarny. Wszystkie wartości są zerowe, a indeks jest oznaczony jako jedynka.

Co więcej dla jednego z modelów zastosowano także rozszerzanie danych (data augmentation). Może wydawać się że przeszło 50 000 obrazów w zbiorze treningowym to bardzo dużo, jednak wraz z wzrastającą liczbą parametrów modelu warto posiadać więcej danych przeznaczonych do trenowania co raz to bardziej skomplikowanych sieci neuronowych. Na tym też data augmentation polega. Jest to proces polegający na sztucznym zwiększania ilości danych poprzez modyfikowanie już istniejących danych. Dzięki temu powstają nowe, lekko zmienione obrazy mogące posłużyć do trenowania sieci. Dużą zaletą takiego podejścia jest to, że może pomóc poprawić wydajność modelu, oraz uniknąć problemu przetrenowania dzięki bardziej zróżnicowanemu zbiorowi danych. W związku z tym operacje, które zostały przeprowadzone na obrazach to rotowanie ich do 15 stopni, zwiększanie i oddalanie, ścinanie i przesuwanie względem szerokości i wysokości. Są to zazwyczaj lekkie modyfikacje, nie zdecydowano się na obracanie obrazów pionowo i poziomo, jako że duża zmiana obrazu mogłaby wprowadzać model w błąd. Przykładem mogłoby być obrócenie obrazu limitu prędkości 60 km/h względem osi x. Byłby on wtedy podobny do ograniczenia 90 km/h jednak z odwróconą dziewiątką, co nie pomagałoby modeli uczyć się jak faktycznie wygląda ograniczenie prędkości do 60km/h. Przykładowe zmodyfikowane obrazy można zobaczyć na rysunku nr 16.

|  |
| --- |
| Rysunek 16. Przykładowe zmodyfikowane obrazy |
|  |
| **Źródło:** Opracowanie własne. |

**Podsumowanie**

Reasumując model będzie trenowany i testowany na zbiorze "German Traffic Sign Recognition Benchmark", który jest popularnym zbiorem treningowym dla konwolucyjnych sieci neuronowych. Zbiór ten składa się z ponad 50 000 obrazów pochodzących z różnych znaków drogowych na drogach w Niemczech oraz przeszło 40 klas. Analiza zbioru wykazała, że klasy w zbiorze treningowym są nierównomiernie rozłożone, co może powodować stronniczość modelu. Zdecydowano przeprowadzić się także preprocessing i rozszerzenie danych, aby przygotować obrazy do wejścia do sieci CNN oraz potencjalnie poprawić wydajność modelu. W następnym rozdziale zostaną omówione ogólne założenia i postać modelu.

# Metodyka

Aby klasyfikować obrazy znaków drogowych można użyć wcześniej opracowanego modelu lub stworzyć własny. Zdecydowano się spróbować opracować własny model, który pozwoli skutecznie dopasowywać etykiety do odpowiadających im obrazów. W rozdziale nr IV omówiona zostanie struktura wybranej do zadania sieci konwolucyjnej, która wykazała się najlepszymi wynikami na przedstawionym w rozdziale III zbiorze danych i proces, który powstał, aby taki model stworzyć.

Zaczynając od procesu, wizualną reprezentację przepływu pracy przedstawiono na rysunku nr 17. Na początku, obrazy znaków drogowych zostały zebrane i użyte jako zbiory treningowe i testowe. Następnie, zostały one wstępnie przetworzone przy użyciu różnych metod w celu inputowania ich do sieci oraz potencjalnie poprawienia możliwości predykcyjnych. W kolejnym kroku, wstępnie przetworzone dane zostały przekazane do sieci, gdzie proponowany model był trenowany na zbiorze treningowym oraz w międzyczasie próbowano dobrać optymalne wartości hiperparametrów. Na koniec, sprawdzona została dokładność modelu na zbiorze testowym, aby określić jego ogólną wydajność.

|  |
| --- |
| Rysunek 17. Proces tworzenia modelu |
|  |
| **Źródło:** Opracowanie własne. |

O zbiorze danych i preprocessingu więcej zostało napisane w rozdziale III, w związku z tym, więcej uwagi w tym rozdziale zostanie poświęcona zbudowanemu modelowi. Natomiast wyniki klasyfikacji zostaną przedstawione w kolejnej części pracy.

Wracając do architektury modelu. Dążenia do uzyskania lepszej dokładności w zadaniach takich jak klasyfikacja obrazów spowodowała, że powstawały coraz to głębsze sieci neuronowe z coraz większą liczbą parametrów liczoną w milionach. Jednak warto pamiętać, że samo zwiększenie głębokości nie koniecznie prowadzi do poprawy wyników. W związku z tym po kilku próbach budowy takiej sieci, zdecydowano się na architekturę przedstawioną na rysunku nr 18. Model składa się z 18 warstw i posiada niecałe 700 000 parametrów. Nie jest więc to bardzo duży i skomplikowany model. Jego szkolenie na 6 rdzeniowym CPU AMD Ryzen 5 2600 z 3400 Mhz wyniosło około 30 minut.

Analizując architekturę dokładniej, sieć składa się z 4 warstw konwolucyjnych, 2 warstw max-poolingu, 4 warstw normalizujących, 4 warstw dropoutu, 3 warstw w pełni połączonych i 1 warstwy do spłaszczania danych.

W pierwszej warstwie konwolucyjnej, będącej warstwą wejściową przyjęto wartości 32x32x3, które odpowiadają wymiarom obrazów wejściowych, które uprzednio przeskalowano do takich wartości. W kolejnej warstwie konwolucyjnej użyto 32 filtrów o wymiarach jądra 5x5, gdzie za funkcje aktywacji posłużyła funkcja ReLU. Po tej warstwie rozmiar zmniejsza się do 28x28. Następnie dane przekazywane są do kolejnej warstwy konwolucyjnej, w której to działają 64 filtry, tym razem z jądrem wielkości 3x3. Tutaj również użyto funkcji aktywacji ReLU. Po dwóch warstwach konwolucyjnych zastosowano warstwę max-poolingu, w której przyjęto rozmiar filtra 2x2. W kolejnym kroku znormalizowano paczkę danych, a następnie w warstwie dropoutu wyłączono 30% neuronów. Taką samą strukturę ma kolejne pięć warstw z tą różnicą, że w trzeciej warstwie konwolucyjnej znajduje się 64 filtry, a w czwartej 128 filtrów o rozmiarach jądra 3x3. Max-pooling i normalizacja są takie same jak poprzednio, a warstwa dropoutu tym razem wyłączą 20% neuronów. Po tych operacjach następuje spłaszczenie macierzy do wymiarów wektora. W końcowym etapie dane przechodzą przez warstwę w pełni połączoną z funkcją aktywacji ReLU, następnie są normalizowane oraz 50% wartości neuronów zostaje wyzerowanych. To samo dzieje się raz jeszcze, tyle że tylko 20% neuronów zostaje wyłączonych. W ostatniej w pełni połączonej warstwie zamiast funkcji aktywacji ReLU została użyta funkcja softmax, zwracająca prawdopodobieństwa wystąpienia jeden z 43 klas na danym obrazie. Przy kompilacji modelu użyto optymalizatora Adam ze zmniejszającym się tempem uczenia wraz ze wzrostem epoch. Jako funkcję straty zastosowano kategoryczną entropię krzyżową, a metrykami mierzącymi jakość modelu zostały dokładność (accuracy) i błąd średniokwadratowy (mse).

|  |
| --- |
| Rysunek 18. Architektura modelu CNN |
|  |
| **Źródło:** Opracowanie własne. |

**Podsumowanie**

Podsumowując omówiona została struktura wybranej do zadania klasyfikacji sieci konwolucyjnej, która wykazała się najlepszymi wynikami na przedstawionym w rozdziale III zbiorze danych. W rozdziale przedstawiono wizualną reprezentację przepływu pracy oraz przedstawiono także architekturę stworzonego modelu, składającego się z 18 warstw i niespełna 700 000 parametrów. Opisano również szczegółowo kolejne warstwy tego modelu. Natomiast w kolejnym rozdziale zostaną przedstawione wyniki klasyfikacji obrazów przy użyciu omawianego modelu.

# Wyniki

W niniejszym rozdziale przedstawione zostaną wyniki klasyfikacji znaków drogowych przy użyciu modelu przedstawionego w poprzednim rozdziale oraz na podstawie tych wyników wyciągnięte wnioski.

## Opis wyników

Wpierw warto wspomnieć model przeszedł przez parę faz, aby przejść od stanu pierwotnego do finalnego, w którym pokazywał już bardzo dobre wyniki na zbiorze testowym. Pierwszy model składał się głównie z warstw konwolucyjnych, dwóch max-poolingowych i dwóch warstw w pełni połączonych. Łącznie posiadał około 300 000 parametrów. Był to więc bardzo lekki i prosty model. Na zbiorze treningowym wykazywał się około 97% skutecznością, jednakże dokładność ta spadała do 95% na zbiorze testowym. Nie był to więc satysfakcjonujący wynik, ponieważ większość modeli obecnie osiąga wartości około 99% na zbiorze testowym. W związku z tym zaczęto stopniowo dodawać warstwy normalizujące, konwolucyjne, w pełni połączone i dropoutu. Dzięki temu powstał model ostateczny, którego architektura została przedstawiona w rozdziale IV.

Do oceny efektywności modelu zostały użyte jedne z najpopularniejszych metryk w uczeniu maszynowym – accuracy, precision, recall i f1-score. Trafność jest podstawowym wskaźnikiem jakości prognozy. Mówi on o stosunku obserwacji, które model zaklasyfikował poprawnie (true positive i true negative) względem wszystkich dokonanych klasyfikacji:

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

Wartości trafności jakie model osiągał przez kolejne cykle zostały przedstawione na rysunku nr 19. Wartość metryki dla zbioru treningowe wzrasta bardzo mocno już w pierwszych cyklach, aby ustatkować się ostatecznie w okolicach 99% trafności. Wykres wskazuje, że liczba cykli była prawdopodobnie wystarczająca, jako że trafność przestała rosnąć w okolicach cyklu 25. Finalnie najwyższe accuracy jakie osiągnął model na zbiorze treningowym wyniosło 99.3%. Natomiast jeśli chodzi o zbiór testowy można zauważyć, że osiągał on bardzo zbliżone wyniki do zbioru treningowego. Wartość trafności na zbiorze testowym wyniosła 99.02%. Można więc stwierdzić, że nie doszło do problemu przeuczenia i model dobrze generalizuje, czyli równie dobrze przewiduje na nowych danych.

|  |
| --- |
| Rysunek 19. Wykres trafności modelu |
|  |
| **Źródło:** Opracowanie własne. |

Dodatkowo stworzono również wykres straty, który można zobaczyć na rysunku nr 20. Wykres ten również daje wgląd w to, jak dobrze model uczył się przez kolejne cykle. Tutaj również można dostrzec, że dla obu zbiorów strata przyjmowała podobne wartości poniżej 0.05 już po około 8 epizodach. Co więcej rysunek ten również wskazuje, że model prawdopodobnie nie był już zdolny do dalszego uczenia się, więc proces szkolenia został zatrzymany w odpowiednim momencie. Jeśli chodzi o przeuczenie można by się zastanawiać, czy nie lepiej byłoby zakończyć uczenie modelu już koło 20 epoki, jako że w kolejnych wartości dla obu zbiorów zaczynają się od siebie oddalać. Jednakże są to wartości bardzo małe, rzędu setnych czy nawet tysięcznych miejsc. Co więcej wartości te stabilizują się i nie odbiegają coraz bardziej od siebie wraz z dalszym wzrostem epok.

|  |
| --- |
| Rysunek 20. Wykres straty dla modelu |
|  |
| **Źródło:** Opracowanie własne. |

Omawiając trafność warto nie zapomnieć, że ma ona sporą wadę. W zbiorach niezbalansowanych, takich jak ten, może ona wskazywać na wysoką skuteczność modelu, pomimo że model będzie źle działa. Sieć może dobrze przewidywać klasę najczęściej występującą w zbiorze, natomiast nie będzie umiała przewidzieć klasy rzadszej. Mimo to trafność będzie wysoka. Dlatego też warto przyjrzeć się innym metrykom. Precyzja określa w jakim stopniu można zaufać pozytywnym predykcjom w danej klasie:

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

Inaczej jest to procent pozytywnych predykcji, które faktycznie są poprawne. A więc przyglądając się precyzji wyznaczonej dla każdej klasy, można wywnioskować czy faktycznie model jest tak dobry jak wskazywałaby na to trafność. Podobnym wskaźnikiem jest czułość przedstawiony wzorem:

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

Jest to wskaźnik, który informuje, ile elementów z wybranej klasy zostało poprawnie rozpoznanych. Oprócz tego wyliczono jeszcze miarę F1-score, który jest niejako połączeniem tych miar i pozwala ułatwić porównania między jakością modeli. Jego wzór wygląda następująco:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

Wartości wszystkich tych miar zostały zebrane w tabeli nr 2, w której przedstawiono je w odniesieniu do każdej kategorii.

Tabela 2. Raport klasyfikacyjny

|  |
| --- |
|  |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | **class** | **precision** | **recall** | **f1-score** | **support** | | 0 | 0.88 | 1.00 | 0.94 | 60 | | 1 | 0.98 | 1.00 | 0.99 | 720 | | 2 | 1.00 | 0.99 | 1.00 | 750 | | 3 | 1.00 | 0.98 | 0.99 | 450 | | 4 | 1.00 | 0.97 | 0.99 | 660 | | 5 | 0.98 | 1.00 | 0.99 | 630 | | 6 | 0.99 | 0.99 | 0.99 | 150 | | 7 | 1.00 | 1.00 | 1.00 | 450 | | 8 | 1.00 | 1.00 | 1.00 | 450 | | 9 | 0.99 | 1.00 | 1.00 | 480 | | 10 | 1.00 | 1.00 | 1.00 | 660 | | 11 | 0.99 | 0.98 | 0.98 | 420 | | 12 | 1.00 | 0.98 | 0.99 | 690 | | 13 | 1.00 | 1.00 | 1.00 | 720 | | 14 | 1.00 | 1.00 | 1.00 | 270 | | 15 | 0.98 | 1.00 | 0.99 | 210 | | 16 | 1.00 | 1.00 | 1.00 | 150 | | 17 | 1.00 | 1.00 | 1.00 | 360 | | 18 | 1.00 | 0.97 | 0.98 | 390 | | 19 | 1.00 | 1.00 | 1.00 | 60 | | 20 | 0.97 | 1.00 | 0.98 | 90 | | 21 | 0.97 | 0.99 | 0.98 | 90 | | 22 | 0.97 | 0.93 | 0.94 | 120 | | 23 | 0.93 | 1.00 | 0.96 | 150 | | 24 | 1.00 | 0.99 | 0.99 | 90 | | 25 | 0.99 | 0.98 | 0.99 | 480 | | 26 | 0.95 | 0.99 | 0.97 | 180 | | 27 | 0.93 | 0.92 | 0.92 | 60 | | 28 | 0.99 | 0.99 | 0.99 | 150 | | 29 | 0.95 | 1.00 | 0.97 | 90 | | 30 | 0.96 | 0.90 | 0.93 | 150 | | 31 | 1.00 | 1.00 | 1.00 | 270 | | 32 | 0.98 | 1.00 | 0.99 | 60 | | 33 | 1.00 | 1.00 | 1.00 | 210 | | 34 | 1.00 | 1.00 | 1.00 | 120 | | 35 | 0.99 | 1.00 | 1.00 | 390 | | 36 | 0.98 | 1.00 | 0.99 | 120 | | 37 | 0.98 | 0.98 | 0.98 | 60 | | 38 | 1.00 | 1.00 | 1.00 | 690 | | 39 | 1.00 | 0.98 | 0.99 | 90 | | 40 | 0.95 | 0.97 | 0.96 | 90 | | 41 | 1.00 | 0.97 | 0.98 | 60 | | 42 | 1.00 | 1.00 | 1.00 | 90 | | accuracy |  |  | 0.99 | 0.9903 | | macro avg | 0.98 | 0.99 | 0.98 | 12630 | | weighted avg | 0.99 | 0.99 | 0.99 | 12630 | |
| **Źródło:** Opracowanie własne. |

## Wnioski

# Podsumowanie

Zakończenie

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.

LITERATURA

**Książki**

1. Bradley K., *Podstawy metodyki PRINCE 2*, Warszawa 2002.
2. Bukowski M., *Korzyści z wdrożenia metodyki PRINCE2 w wybranej jednostce administracji publicznej*, op. naukowy dr inż. W. Dąbrowski, Warszawa, czerwiec 20.
3. Flasiński M., *Zarządzanie projektami informatycznymi,* Wydawnictwo Naukowe PWN, Warszawa 2006.
4. Frączkowski K., *Zarządzanie projektem informatycznym. Projekty w środowisku wirtualnym. Czynniki sukcesu i niepowodzeń projektów*, Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław 2003, s.125-126.
5. Gunia G., *Implementacja zintegrowanych systemów informatycznych w małych i średnich przedsiębiorstwach*, „Zarządzanie Przedsiębiorstwem”, 2009.
6. Gunia G., *Wdrażanie zintegrowanych systemów informatycznych,* Wydawnictwo Fundacji Centrum Nowych Technologii, Bielsko Biała 2009.
7. Januszewski A., *Funkcjonalność informatycznych systemów zarządzania. Tom 1*, Wydawnictwo Naukowe PWN, Warszawa 2008.
8. Kisielnicki J., *Zintegrowane systemy informatyczne, dobre praktyki wdrożeń systemów klasy ERP*, Wydawnictwo Naukowe PWN
9. Kisielnicki J., *Systemy informacyjne biznesu,* Placet, Warszawa 1999.
10. Koszlajda A.*, Zarządzanie projektami IT*, Gliwice 2010.
11. Lech P., Zintegrowane Systemy Zarządzania ERP/ERP2. *Wykorzystanie w biznesie wdrażanie,* Difin, Warszawa 2003.
12. Miłosz M. (red.), *Wdrażanie i eksploatacja systemów informatycznych. Wybrane problemy,* Polskie Towarzystwo Informatyczne, Lublin 2002.
13. Office of Government Commerce, *Skuteczne zarządzanie projektami PRINCE2*, London: TSO, wydanie 2005.
14. Rokicka-Broniatowska A., *Wstęp do informatyki gospodarczej*, Szkoła Główna Handlowa, Warszawa 2006.
15. Stefanowicz B., *Informacyjne systemy zarządzania. Przewodnik,* Szkoła Główna Handlowa, Warszawa 2007.
16. Szyjewski Z., *Metodyki zarządzania projektami informatycznymi,* PLACET, Warszawa 2004.
17. Trocki M., *Metodyki zarządzania projektami*, Biblioteka Project Managera, Warszawa 2011.
18. Waćkowski K., Chmielewski J., *Wspomaganie zarządzania projektami informatycznymi. Poradnik dla menadżera*, Helion, Gliwice 2007.

**Artykuły i studia**

1. Dębowski L.,*ZASTOSOWANIE KOMPUTERÓW W NAUCE I TECHNICE’ 2006*, Zeszyty Naukowe Wydziału Elektrotechniki i Automatyki Politechniki Gdańskiej Nr 22
2. IE. 1991. Competition in manufacturing leads to MRP II. 23 (July) 10-13
3. WJ Hopp, ML Spearman Commissioned Paper To Pull or Not to Pull: What Is the Question, Manufacturing & Service Operations Management, 2004

**Strony internetowe**

1. Poksiński P., *PRINCE2*, materiał dostępny w wersji elektronicznej pod adresem internetowym: http://www.poksinski.com/pdf\_files/PRINCE2.pdf [18-09-2012]
2. *Słownik APICS*, http://www.apics.org/dictionary/dictionary-information?ID=2399 [4-01-2013]
3. Schmidt P., *PRINCE2 i techniki planowania*, materiał dostępny w wersji elektronicznej pod adresem internetowym: http://ww.4pm.pl/artykul/prince2\_i\_techniki\_planowania-56-1247.html [17-10-2012]
4. *Cloud Computing*, materiał zaczerpnięty ze strony internetowej: http://www.it.integro.pl/pl/aplikacje-w-chmurze-cloud-computing [4-01-2013]
5. Trąbka J., *Zarządzanie projektem wdrożeniowym systemu klasy ERP – autorska metodyka*, materiał dostępny w wersji elektronicznej pod adresem internetowym: http://kkio2012.agh.edu.pl/presentations/KKIO2012-C1\_3.pdf [10-11-2012]
6. *Skuteczne wdrożenia systemu ERP dzięki metodyce Microsoft Sure Step;* http://www.it.integro.pl/pl/metodyka-wdrozeniowa. [28-09-2012]
7. *Nasza metodyka na wdrożenie ERP w twojej firmie;* http://www.profidata.com.pl/erp/metodyka-pit-stop.html. [28-09-2012]
8. *Manifesto for Agile Software Development*; Cytat ze strony http://www.agilemanifesto.org/ [10-10-2012]
9. *Metodyka wdrożenia QlikView- Business Intelligence;* , http://www.businessintelligence.pl/pl/metodyka-wdrozenia. [28-09-2012]
10. *QlikView przełamuje bariery rynku – Business Intelligence;* http://www.businessintelligence.pl/pl/a/QlikView-przelamuje-bariery-rynku [11-10-2012]
11. *ERP- centrum wiedzy o systemach ERP na decyzje-IT.pl;* http://decyzje-it.pl/centrum-wiedzy/erp.html [10-10-2012]
12. *The history of Prince2- Project Smart*; http://www.projectsmart.co.uk/history-of-prince2.html [18-09-2012]
13. *Oracle- Hardware and Software, Enginnered to work together*; http://www.oracle.com/index.html [23-09-2012]
14. CRP for Oracle R12- Welcom to Oracle ERP http://www.oracleerp4u.com/2010/06/crp-for-oracle-r12.html [189-10-2012]
15. *Przykłady zastosowań technologii RFID w magazynach* http://rfid-lab.pl/przyklady-zastosowan-technologii-rfid-w-magazynach [07-01-2013]

Spis tabel

[Tabela 1. Lorem ipsum dolor sit amet 14](#_Toc355852181)

[Tabela 2. Lorem ipsum dolor sit amet, consectetur adipisicing elit 15](#_Toc355852182)

Spis rysunków

[Rysunek 1. Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet 17](#_Toc355852186)

[Rysunek 2. Lorem ipsum dolor sit amet, consectetur 22](#_Toc355852187)

Streszczenie

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Summary

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

1. Peden i World Health Organization, *World Report on Road Traffic Injury Prevention*. [↑](#footnote-ref-1)
2. World Health Organization, *Global Status Report on Road Safety 2018*. [↑](#footnote-ref-2)
3. Brookhuis, Waard, i Janssen, „Behavioural Impacts of Advanced Driver Assistance Systems–an Overview”. [↑](#footnote-ref-3)
4. Benson i in., „Potential Reduction in Crashes, Injuries and Deaths from Large-Scale Deployment of Advanced Driver Assistance Systems”. [↑](#footnote-ref-4)
5. Lay, „HISTORY OF TRAFFIC SIGNS. IN”. [↑](#footnote-ref-5)
6. Escalera i in., *Traffic-Sign Recognition Systems*. [↑](#footnote-ref-6)
7. Priese i in., „New results on traffic sign recognition”. [↑](#footnote-ref-7)
8. Hubel i Wiesel, „Receptive fields of single neurones in the cat’s striate cortex”. [↑](#footnote-ref-8)
9. Hubel i Wiesel, „Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex”. [↑](#footnote-ref-9)
10. Roberts, „Machine Perception of Three-Dimensional Solids”. [↑](#footnote-ref-10)
11. Papert, „The Summer Vision Project”. [↑](#footnote-ref-11)
12. Fukushima, „Neocognitron”. [↑](#footnote-ref-12)
13. Waibel i in., „Phoneme recognition using time-delay neural networks”. [↑](#footnote-ref-13)
14. LeCun i in., „Backpropagation Applied to Handwritten Zip Code Recognition”. [↑](#footnote-ref-14)
15. Yamaguchi i in., „A Neural Network for Speaker-Independent Isolated Word Recognition”. [↑](#footnote-ref-15)
16. Debevec i Malik, „Recovering High Dynamic Range Radiance Maps from Photographs”. [↑](#footnote-ref-16)
17. Lowe, „Object recognition from local scale-invariant features”. [↑](#footnote-ref-17)
18. Viola i Jones, „Rapid Object Detection Using a Boosted Cascade of Simple Features”. [↑](#footnote-ref-18)
19. Oh i Jung, „GPU Implementation of Neural Networks”. [↑](#footnote-ref-19)
20. Chellapilla, Puri, i Simard, „High Performance Convolutional Neural Networks for Document Processing”. [↑](#footnote-ref-20)
21. Krizhevsky, Sutskever, i Hinton, „ImageNet Classification with Deep Convolutional Neural Networks”. [↑](#footnote-ref-21)
22. He i in., „Deep Residual Learning for Image Recognition”. [↑](#footnote-ref-22)
23. Cireşan i in., „Multi-Column Deep Neural Network for Traffic Sign Classification”. [↑](#footnote-ref-23)
24. Haque i in., „DeepThin: A novel lightweight CNN architecture for traffic sign recognition without GPU requirements”. [↑](#footnote-ref-24)
25. Habibi Aghdam, Jahani Heravi, i Puig, „A Practical Approach for Detection and Classification of Traffic Signs Using Convolutional Neural Networks”. [↑](#footnote-ref-25)
26. Krogh, „What Are Artificial Neural Networks?” [↑](#footnote-ref-26)
27. Stallkamp i in., „The German Traffic Sign Recognition Benchmark”. [↑](#footnote-ref-27)