Studium Magisterskie

Kierunek: Analiza Danych – Big Data

Marek Kuś
Nr albumu 82394

Zastosowanie konwolucyjnych sieci neuronowych w klasyfikacji znaków drogowych

Praca magisterska
napisana w Instytucie Ekonometrii
pod kierunkiem naukowym
Dr Sebastiana Zająca

Warszawa, 2023

SPIS TREŚCI

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
GPS	Global Positioning System

WSTĘP

W raporcie o zapobieganiu obrażeń wynikających 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]. Szacuje się, że co roku w wyniku wypadków drogowych ginie około 1,3 mln osób[2]. W związku z tym, Zgromadzenie Ogólne Organizacji Narodów Zjednoczonych (ONZ) postawiło sobie za cel zmniejszenia globalnej liczby zgonów i obrażeń wynikających z wypadków drogowych o 50 procent do 2030 r. Jest wiele narzędzi, z których pomocą świat będzie dążył do ograniczenia tego problemu. Jednym z takich środków są zaawansowane systemy 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 systemy rozpoznawania znaków drogowych. Jako że za około 90% wszystkich wypadków drogowych odpowiedzialne są ludzkie błędy spowodowane zmęczeniem, nieuwagą lub sennością[3], zaawansowane systemy wspomagania kierowcy (ADAS) wydają się być idealnym narzędziem do ograniczanie lub w przyszłości całkowitego wyeliminowania ludzkiego błędu. Z badań wynika, że technologie te mogą zapobiec nawet 2,7 mln kolizji rocznie w samych Stanach Zjednoczonych[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 systemu nawigacji satelitarnej (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.

[1] Peden i World Health Organization, *World Report on Road Traffic Injury Prevention*.
[2] World Health Organization, *Global Status Report on Road Safety 2018*.
[3] Brookhuis, Waard, i Janssen, „Behavioural Impacts of Advanced Driver Assistance Systems–an Overview".
[4] Benson i in., „Potential Reduction in Crashes, Injuries and Deaths from Large-Scale Deployment of Advanced Driver Assistance Systems".

Celem tej pracy jest więc pokazanie jak za pomocą deep learningu, a dokładnie konwolucyjnych sieci neuronowych, można stworzyć lekki obliczeniowo, a jednocześnie bardzo skuteczny model klasyfikujący znaki drogowe, tym samym przyczyniając się do poprawy bezpieczeństwa na drogach, zmniejszając liczbę wypadków, a także skutki ekonomicznie i społeczne.

Kończąc wprowadzenie, warto również zaprezentować strukturę pracy. W rozdziale 1 dokonano przeglądu literatury i historii omawianego zagadnienia. Dotychczas powstały 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 na zdefiniowaniu pojęć, wykorzystywanych w celu rozwiązania przedstawionego problemu. W rozdziale 3 opisane zostały dane, z których korzystać będzie stworzony model, przeznaczony do klasyfikacji znaków drogowych. 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 oraz możliwe dalsze kierunki rozwoju.

ROZDZIAŁ 1. Historia i przegląd literatury

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

1.1 Znaki drogowe i rozwój pojazdów autonomicznych

Znaczenie znaków drogowych znacznie wzrosło wraz ze stopniowym rozwojem motoryzacji. Jeden z pierwszych współczesnych systemów znaków drogowych został opracowany przez włoski Touring Club w 1895 roku[5]. Jednak podstawowe wzory większości znaków drogowych zostały ustalone dopiero przez Międzynarodowy Kongres Drogowy, który miał miejsce w Rzymie w 1908 roku. Rok później, dziewięć rządów państw europejskich uzgodniło stosowanie czterech symboli, oznaczających, "zakręty", "wyboje", "skrzyżowania" i "przejazdy kolejowe". W latach 1926-1949 prowadzono intensywne prace nad międzynarodowymi znakami drogowymi, co ostatecznie doprowadziło do opracowania europejskiego systemu znaków drogowych. Do 1949 roku Europa przyjęła wspólny system, który w latach 60-tych XX wieku zaczęły implementować również Stany Zjednoczone.

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 zagadnienie to nadal jest wymagającym problemem. Angażuje on uwagę społeczności zajmującej się zagadnieniami computer vision od ponad 30 lat. Według Escalera et al. (2011)[6], pierwsze badanie zautomatyzowanego rozpoznawania

[5] Lay, „HISTORY OF TRAFFIC SIGNS. IN".
[6] Escalera i in., *Traffic-Sign Recognition Systems*.

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 roku wcześniej. 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 wykorzystując klasyczne algorytmy oparte na segmentacji obrazu oraz dopasowaniu szablonów. Proces rozpoznawania trwał średnio około 0,50s. 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].

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 czy sztuczna sieć neuronowa.

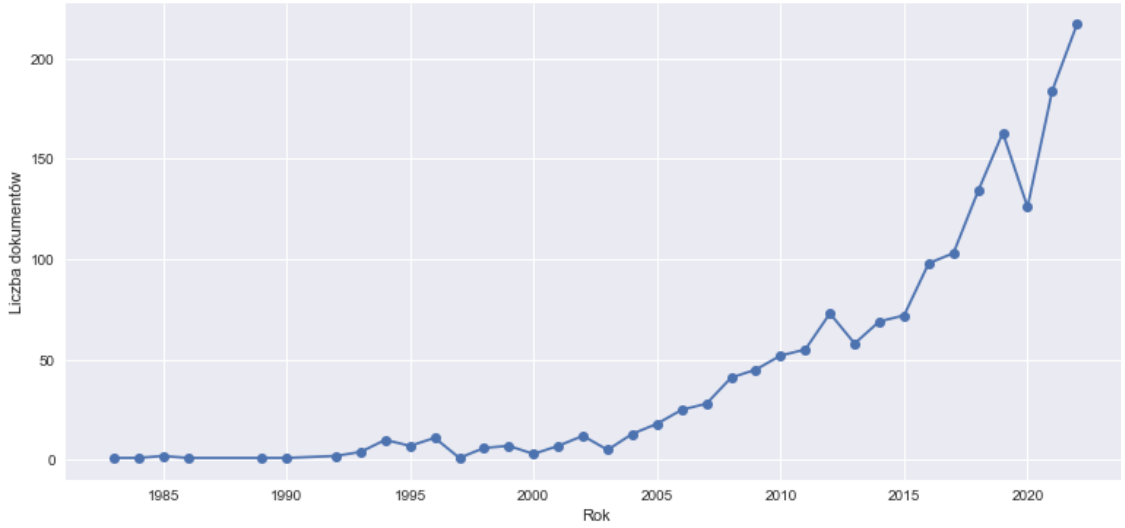
Szybki rozwój doprowadził do przeprowadzenia pierwszych poważnych zawodów dla pojazdów autonomicznych: 'DARPA Grand Challenge' w 2004 roku. Celem było przejechanie 150 mil do celu po wyznaczonej trasie. Choć żaden z pojazdów nie ukończył trasy, był to ważny krok w rozwoju pojazdów autonomicznych. W kolejnych latach odbyło się więcej podobnych konkursów, w których to różnym zespołom udało się 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 związanych z tym zagadnieniem, co można ujrzeć na rysunku numer 1. Od 2005 roku liczba prac zaczęła rosnąć lawinowo, co zbiega się w czasie z pierwszymi zawodami

[7] Priese i in., „New results on traffic sign recognition".

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 XXI wieku, wraz z udanymi zastosowaniami głębokiego uczenia w rozpoznawaniu mowy czy segmentacji semantycznej, zaczęto wprowadzać je do problemu klasyfikacji obrazów. Jedną z takich metod były sieci konwolucyjne, rozwijane przez lata w ramach dziedziny computer vision.

1.2 Rozwój Computer Vision i konwolucyjnych sieci neuronowych

Obszar Computer Vision stał się mocno popularny stosunkowo niedawno, wraz z takim wydarzeniem jak wygrana AlexNet w konkursie ImageNet Jednak dziedzina ta sama w sobie nie jest nowym obszarem nauki. Od 60 lat specjaliści starają się znaleźć różne sposoby, aby komputery mogły wydobyć istotne znaczenie z obrazów i innych danych wizualnych. Aby łatwiej zrozumieć czym są 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], opisywała główne sposoby odpowiedzi neuronów kory wzrokowej, a także 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. Dzięki swoim eksperymentom ustalili, że w pierwotnej korze wzrokowej znajdują się proste i złożone neurony. Co więcej 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 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 o tym, że proste detektory mogą być sumowane, aby stworzyć bardziej złożone detektory[9], które występują w ludzkim systemie wzrokowym, 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 zrozumiałe dla maszyn. 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 budowano z tych linii reprezentację 3D, dzięki czemu wyświetlano trójwymiarowe struktury obiektów. Autor twierdził, że procesy budowy

[8] Hubel i Wiesel, „Receptive fields of single neurones in the cat's striate cortex".
[9] Hubel i Wiesel, „Receptive fields, binocular interaction and functional architecture in the cat's visual cortex".

obrazów 2D do 3D, a następnie wyświetlania 3D do 2D były dobrym punktem wyjścia dla przyszłych badań nad komputerowo wspomaganymi systemami 3D[10].

W latach 60 sztuczna inteligencja stała się dyscypliną akademicką. Jeden z profesorów Massachusetts Institute of Technology (MIT), Seymour Papert, zrealizował projekt pod nazwą „Summer Vision Project", którego finalnym celem było stworzenie systemu będącego w stanie nazwać obiekt poprzez dopasowanie go do odpowiedniej etykiety[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". Był on samoorganizującą się sztuczną siecią prostych i złożonych komórek, potrafiącą rozpoznawać wzory i będącą niewrażliwą na zmiany ich położenia[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). Po wykonaniu pewnych obliczeń, wytwarzano zdarzenia aktywacji (tablice dwuwymiarowe), będące wejściem dla kolejnych warstw sieci. Natomiast warstwy downsamplingu zawierały jednostki, których pola recepcyjne pokrywały fragmenty poprzednich warstw konwolucyjnych. Downsampling pomagał poprawnie klasyfikować obiekty, nawet gdy te 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].

[10] Roberts, „Machine Perception of Three-Dimensional Solids".
[11] Papert, „The Summer Vision Project".
[12] Fukushima, „Neocognitron".
[13] Waibel i in., „Phoneme recognition using time-delay neural networks".

2 lata później propagacja wsteczna została zaimplementowana do algorytmu Fukushimy przez francuskiego naukowca Yanna LeCun[14]. Po kilku latach pracy nad projektem, LeCun wydał LeNet-5 - pierwszą nowoczesną, konwolucyjną sieć neuronową (CNN). Wprowadził on kilka istotnych składników, które do dziś używa się w tych sieciach. 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 zbioru danych w uczeniu maszynowym.

W międzyczasie inna grupa naukowców pod przewodnictwem Yamaguchi wprowadziła koncepcję max pooling'u[15]. Co prawda ich praca skupiła się na rozpoznawaniu izolowanych słów, jednakże max pooling jest wykorzystywany do dziś 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 są ze sobą związane i odróżnić obiekty od ich otoczenia za pomocą algorytmu teorii grafów[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, wykorzystywał nową klasę lokalnych cech obrazu. Były one niepodatne na skalowanie, translację i rotację obrazu oraz częściowo niepodatne na zmiany oświetlenia[17]. Według autora cechy te są podobne do neuronów w dolnej korze skroniowej, wykorzystywanych przez naczelne do rozpoznawania rzeczy. Eksperymentalne wyniki pokazały, że czas obliczeń potrzebnych do rozpoznania obiektu na nieidealnym obrazie wyniósł 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

[14] LeCun i in., „Backpropagation Applied to Handwritten Zip Code Recognition".
[15] Yamaguchi i in., „A Neural Network for Speaker-Independent Isolated Word Recognition".
[16] Debevec i Malik, „Recovering High Dynamic Range Radiance Maps from Photographs".
[17] Lowe, „Object recognition from local scale-invariant features".

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. Trzecim wkładem była metoda łączenia coraz bardziej złożonych klasyfikatorów w tak zwaną "kaskadę". Pozwalała ona na szybkie odrzucanie regionów tła obrazu przy jednoczesnym przeznaczeniu większej ilości obliczeń na obiecujące regiony podobne do szukanych przedmiotów[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 procesorów graficznych (GPU). Ich implementacja była 20 razy szybsza od analogicznej implementacji na procesorach CPU[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].

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 trafność 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. Był on podobny w swojej architekturze do LeNet-5 Yanna LeCuna i osiągał poziom błędu 16,4%[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.

[18] Viola i Jones, „Rapid Object Detection Using a Boosted Cascade of Simple Features".
[19] Oh i Jung, „GPU Implementation of Neural Networks".
[20] Chellapilla, Puri, i Simard, „High Performance Convolutional Neural Networks for Document Processing".
[21] Krizhevsky, Sutskever, i Hinton, „ImageNet Classification with Deep Convolutional Neural Networks".

W ostatnich latach wprowadzano nowe warstwy takie jak residual block (blok resztkowy), 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 on 1 miejsce w konkursie o tej samej nazwie w 2015 roku[22].

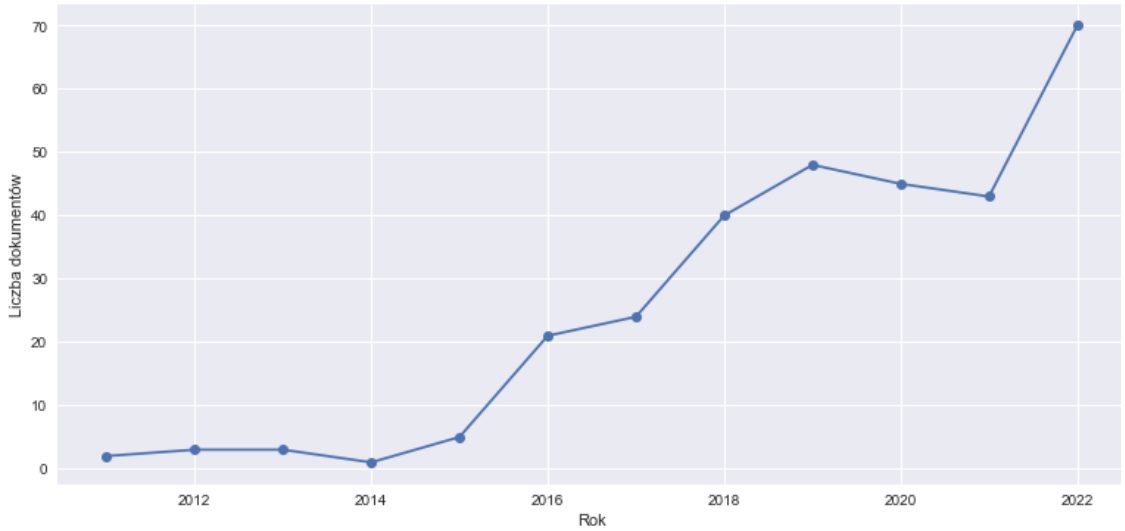
Tak więc widzenie komputerowe przeszło długą drogę w ciągu ostatnich kilku dekad. Pomimo ostatnich postępów, które są imponujące, nadal pozostaje wiele do odkrycia. Wciąż tworzone są nowe architektury sieci konwolucyjnych, potrafiące coraz lepiej rozpoznawać i klasyfikować obiekty. Jednak już teraz istnieje 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.

1.3 Nowoczesne CNN i ich implementacje w benchmarku GTSRB

Łącząc prace z zakresu computer vision i znaków drogowych w całość, powstały publikacje znajdujące się w podrozdziale 1.3. Posiłkując się raz jeszcze bazą danych Scopus można zauważyć, że takich publikacji powstało przeszło 300 w ciągu ostatnich lat.

[22] He i in., „Deep Residual Learning for Image Recognition”.

Rysunek 2. Liczba publikacji związanych z klasyfikacją znaków drogowych za pomocą sieci konwolucyjnych



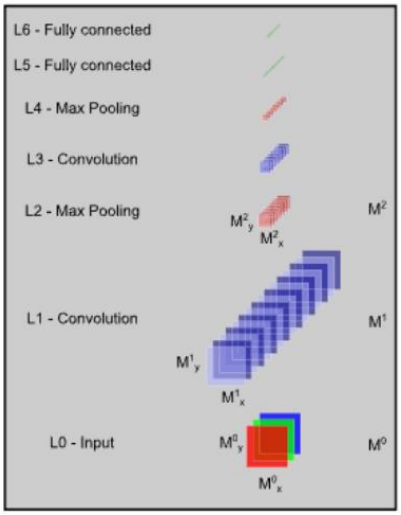
Źródło: *Opracowanie własne.*

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 1.2 stworzeniem sieci konwolucyjnej zwanej AlexNet. Warto więc przyjrzeć się niektórym z tych publikacji, 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]. 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.

[23] Cireşan i in., „Multi-Column Deep Neural Network for Traffic Sign Classification”.

Rysunek 3. Architektura sieci DNN



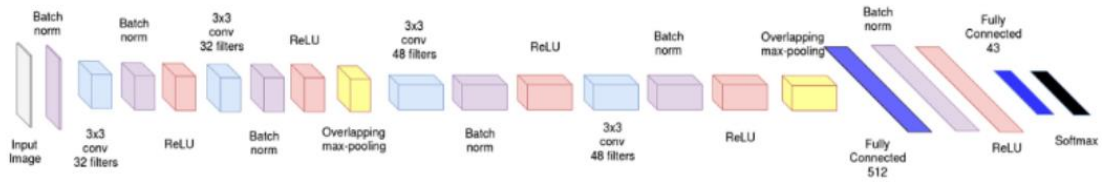
Źródło: Ciresan i in., *Multi-column deep neural network for traffic sing classification*.

Każda warstwa otrzymywała połączenia tylko od swojej poprzedniej warstwy. Opisali to jako ogólny, hierarchiczny ekstraktor cech, który mapował piksele obrazu wejściowego na wektor cech. Następnie wektor ten był klasyfikowany przez kilka, zwykle 2 lub 3, w pełni połączone warstwy. Wszystkie parametry były wspólnie optymalizowane poprzez minimalizację błędu klasyfikacji na zbiorze treningowym. Ostatecznie połączyli oni kilka sieci w jedną wielokolumnową sieć. Uśrednienie informacji wyjściowej z każdej sieci, będących odpowiednikami kolumn, 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. Pozwala to na 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, oraz w pełni połączonej warstwy ukrytej. Ostatnia z nich posiada

funkcję aktywacji softmax w celu uzyskania rozkładu prawdopodobieństwa na n etykiet klas[24]. Jej pełna architektura została przedstawiona na rysunku nr 4.

Rysunek 4. Architektura sieci CNN



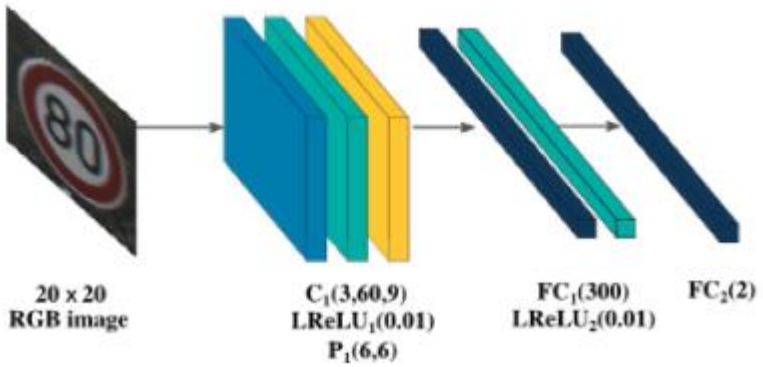
Źródło: Haque i in., *DeepThin: A novel lightweight CNN architecture for traffic sign recognition without GPU requirements*.

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%. Pokazuje to, 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. Pozwoliło to na stosowanie jej na obrazach o dowolnych rozmiarach[25]. Architektury sieci zostały przedstawione na rysunku nr 5 i 6.

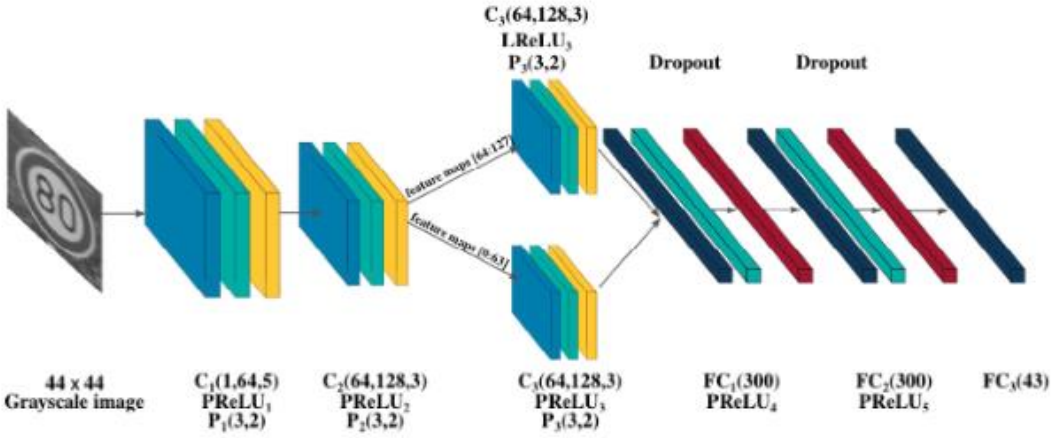
[24] Haque i in., „DeepThin: A novel lightweight CNN architecture for traffic sign recognition without GPU requirements".
[25] Habibi Aghdam, Jahani Heravi, i Puig, „A Practical Approach for Detection and Classification of Traffic Signs Using Convolutional Neural Networks".

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 zbiór danych German Traffic Sign Detection Benchmark.

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

Podsumowanie

Reasumując, w rozdziale przedstawiono rozwój problemu klasyfikacji znaków drogowych oraz zastosowania sieci konwolucyjnych. Opisano również kluczowe prace w dziedzinie Computer Vision na przestrzeni lat. Dzięki temu można było zauważyć, że przez lata konwolucyjne sieci neuronowe posiadały różne architektury i stopnie skomplikowania. Z czasem tworzone były nowe warstwy, jednak podstawowe bloki budujące, opracowane przez lata, są w większości takie same dla każdej sieci. Obecnie większość najnowszych sieci osiągnęła niemal perfekcyjny wskaźnik trafności klasyfikacji. Nadal jednak jest miejsce na poprawę w kwestii wydajności i obciążenia obliczeniowego różnych sieci. Robi się to, aby nie wymagały one mocnych procesorów GPU i mogły być implementowane w słabszych urządzeniach.

ROZDZIAŁ 2. Omówienie pojęć

Po przedstawieniu literatury i historii, warto przejść do objaśnienia pojęć. Po części pojawiły się już w rozdziale 1, a także pojawią się przy omawianiu metodyki i budowy konwolucyjnej sieci neuronowej w późniejszych rozdziałach. Zatem w rozdziale tym zostaną wyjaśnione zagadnienia, wokół których skupia się ta praca.

2.1 Sztuczna sieć neuronowa (ANN) - koncept

Konwolucyjne sieci neuronowe są wyspecjalizowanym typem sztucznych sieci neuronowych. Aby je zrozumieć warto wpierw zacząć od tych drugich. 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 (x_1, x_2, x_n), 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 (w_1, w_2, w_n), przyjmujące wartości liczbowe i kontrolujące 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 przemnożone przez wagę wartości. Taka ważona suma wejść przyjmuje postać[26]:

$$\sum_{i=1}^{n} (W_i X_i) \tag{1}$$

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 x_1, x_2, ..., x_n mają wartość 0, funkcja jest równa wartości obciążenia. Koniec

[26] Krogh, „What Are Artificial Neural Networks?”

końców funkcja progowa zwana także funkcją przekazu (transfer function) otrzymuje postać:

$$\sum_{i=1}^{n}(W_i X_i) + b \tag{2}$$

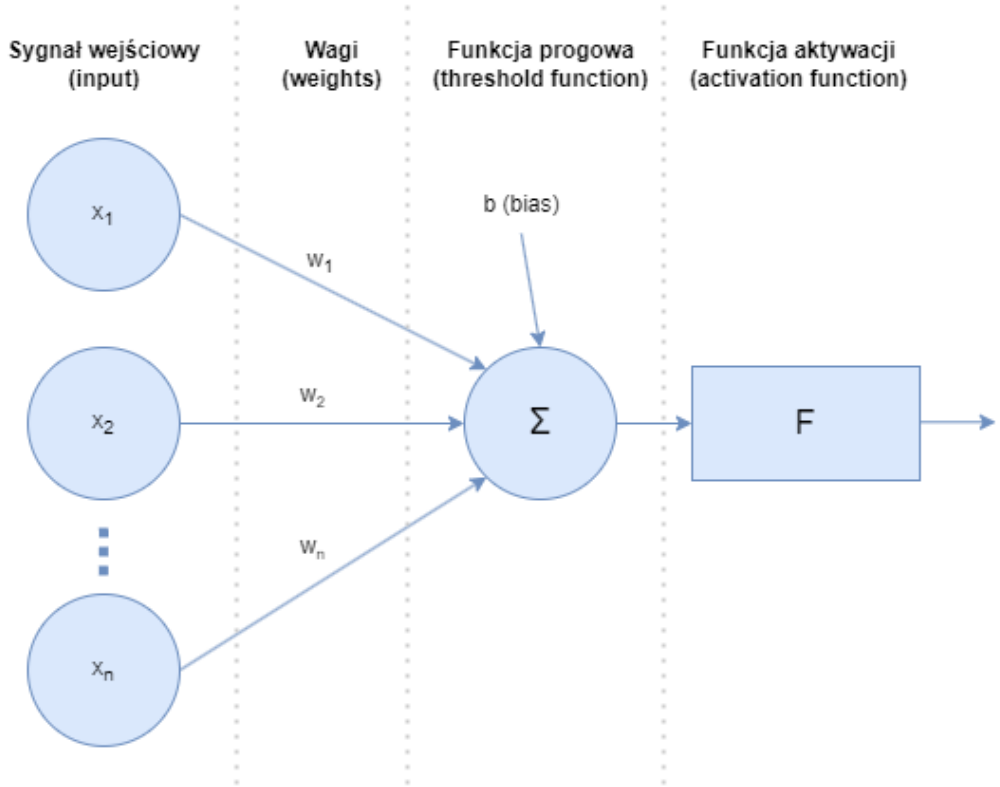
Warto w tym miejscu dodać, że wspomniane wagi oraz obciążenie są inaczej nazywane parametrami modelu sieci neuronowej. Za pomocą procesu uczenia sieć stara się znaleźć ich optymalne wartości.

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

$$f(\sum_{i=1}^{n}(W_i X_i) + b) \tag{3}$$

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 przekazywałby 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.*

Takie 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.

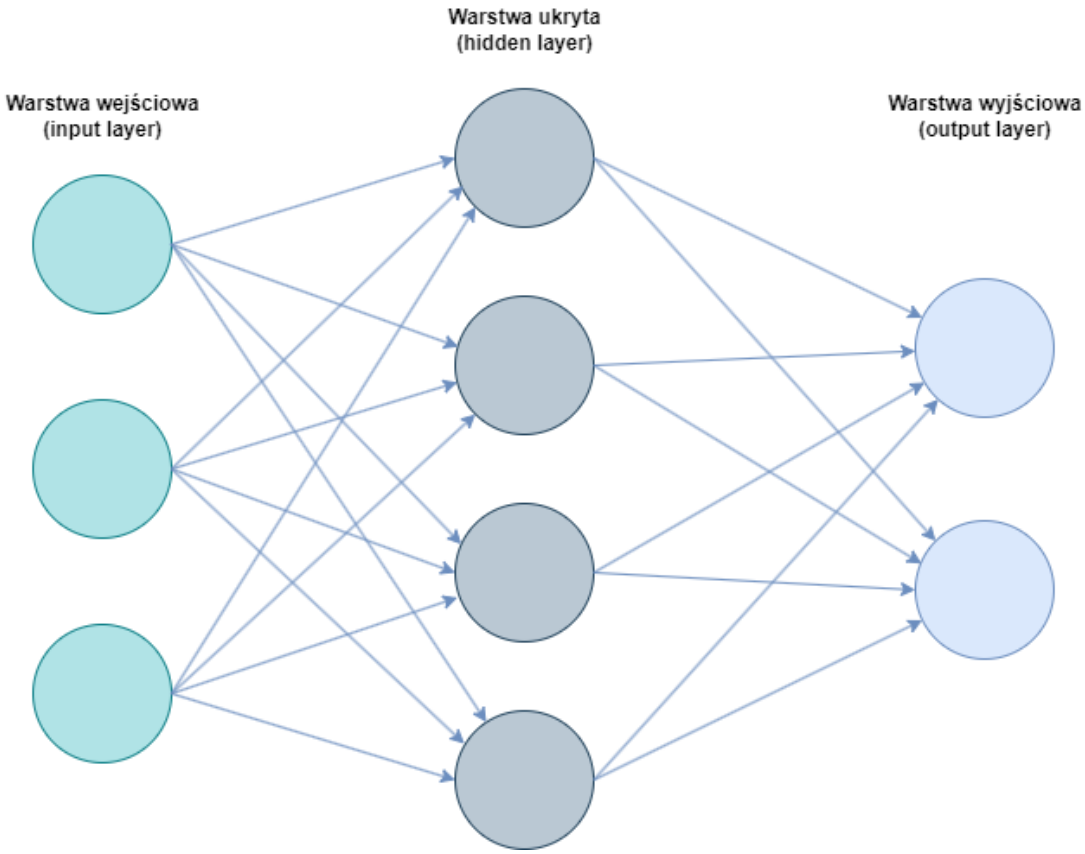
Warstwa wejściowa otrzymuje informacje i przesyła je do warstw ukrytych. Liczba neuronów w tej warstwie musi być równa liczbie atrybutów lub cech, które planuje użyć się w sieci.

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

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

Przykładowa prosta sieć neuronowa została przedstawiona na rysunku nr 8. Sieć składa się z warstwy wejściowej z 3 węzłami źródłowymi, warstwy ukrytej 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. 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.

Proces nauki w sieciach neuronowych jest więc adaptacją sieci do lepszego radzenia sobie z zadanym zbiorem treningowym. 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.

2.2 Konwolucyjna sieć neuronowa (CNN) – charakterystyka

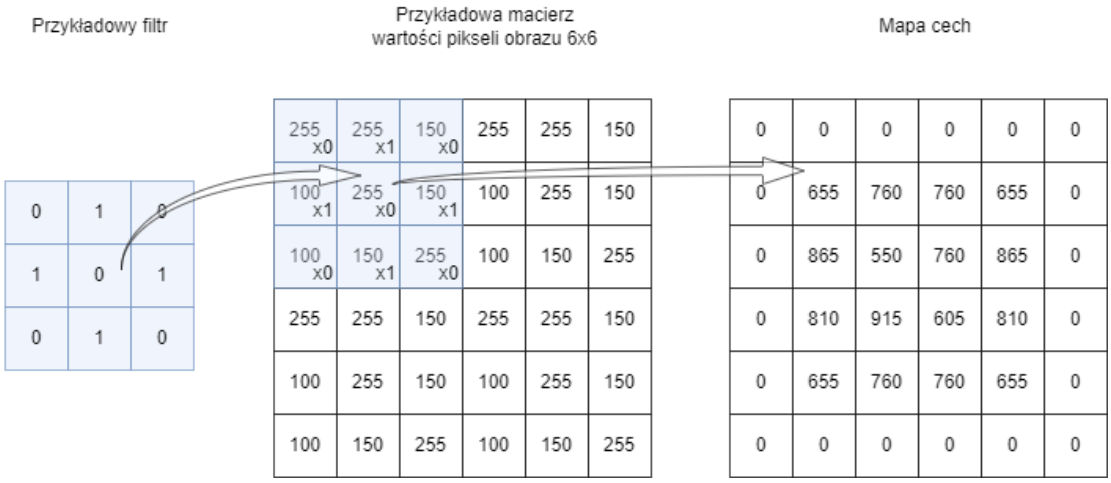
Konwolucyjne sieci neuronowe są bardzo podobne do zwykłych sieci neuronowych omawianych w podrozdziale 2.1. Różnica występuje w architekturze sieci. W sieci konwolucyjnej zakłada się, że danymi wejściowymi są obrazy, co pozwala zakodować pewne właściwości w ich architekturze. Dzięki temu są one bardziej 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 przykładowo z losowego rozkładu Gaussa. Filtr mógłby przyjmować 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:

$$f(x) = \max(0, x) \tag{4}$$

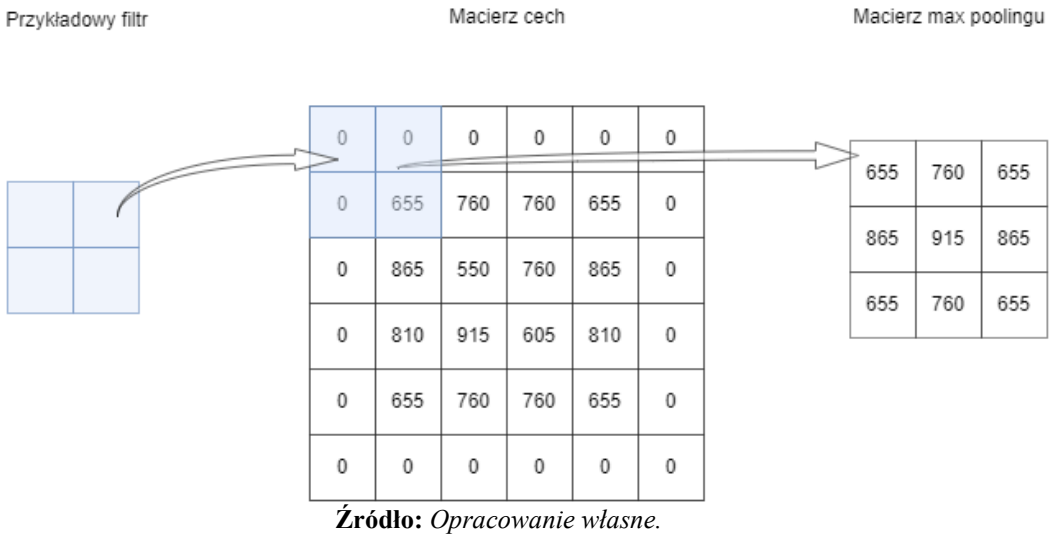
Można zauważyć, że funkcja ta zwraca 0, jeśli wartość wejściowa jest ujemna, a w przeciwnym wypadku zwraca wartość wejściową. Zwiększa to nieliniowość w danych, a jednocześnie jest mniej kosztowne obliczeniowo niż inne funkcje aktywacji takie jak tanh lub sigmoidalna.

Kolejną ważną warstwą w sieciach konwolucyjnych jest warstwa łącząca (poolingu) lub inaczej downsamplingu. Jej głównym zadaniem jest zmniejszenie rozmiarów danych, 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 operacje.

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 zwykła sieć neuronowa. 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:

$$f(x_i) = \frac{e^{xi}}{\sum_j e^{xj}} \tag{5}$$

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 odrzucania 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 dane mogą zostać przekazane do warstwy w pełni połączonej.

Podsumowanie

Podsumowując, w rozdziale omówione zostały pojęcia sztucznych sieci neuronowych (ANN) i konwolucyjnych sieci neuronowych (CNN). ANN opierają się na połączonych ze sobą jednostkach zwanych neuronami. Każde połączenie może przekazywać sygnał do innych neuronów. Natomiast każdy neuron otrzymuje sygnały wejściowe, przeprowadza odpowiednie operacje i wysyła sygnał dalej. 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.

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

ROZDZIAŁ 3. 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 rozmiary zdjęć. Zostaną omówione również statystyki dotyczące liczby obrazów w każdej klasie oraz ich rozkładu w zbiorach. Poruszone zostaną także takie kwestie jak różne warunki oświetlenia, niskiej jakości fotografie czy duża zmienność w kształcie i położeniu znaków na zdjęciu.

3.1 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]. Ujęcia 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 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. 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.*

[27] Stallkamp i in., „The German Traffic Sign Recognition Benchmark".

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

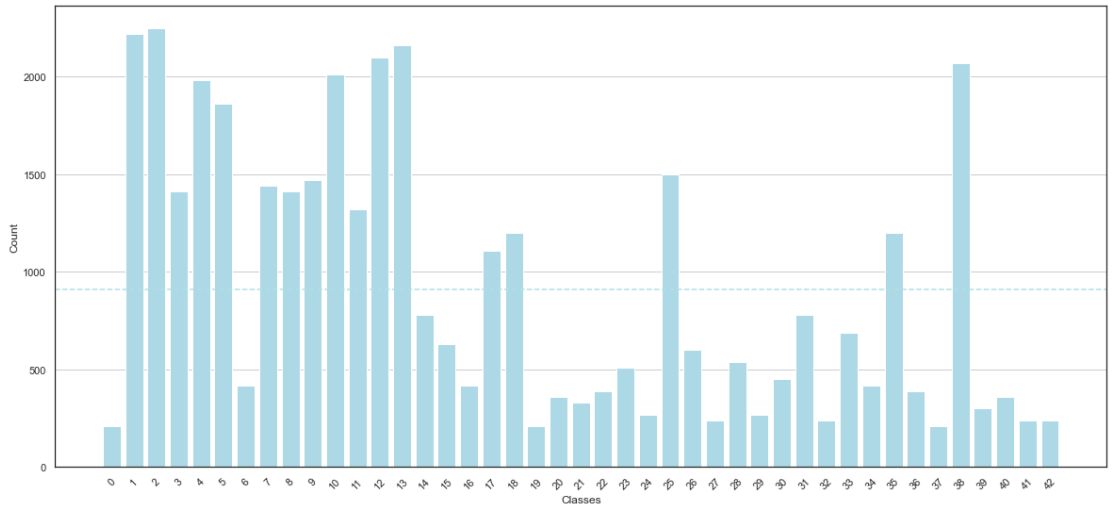
Źródło: *Opracowanie własne.*

Warto zauważyć, że obrazy są mocno zróżnicowane, jeśli chodzi o ich jakość. Część z nich jest dobrze doświetlona, jednak jest też dużo obrazów, wyglądających na zrobione 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 musiał działać na co dzień. Z drugiej strony będzie to dodatkowe wyzwanie dla sieci neuronowej, aby osiągnąć wysoki wynik poprawnej klasyfikacji.

Po przyjrzeniu się podstawowym wartościom 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 ich ponad 2000, 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.

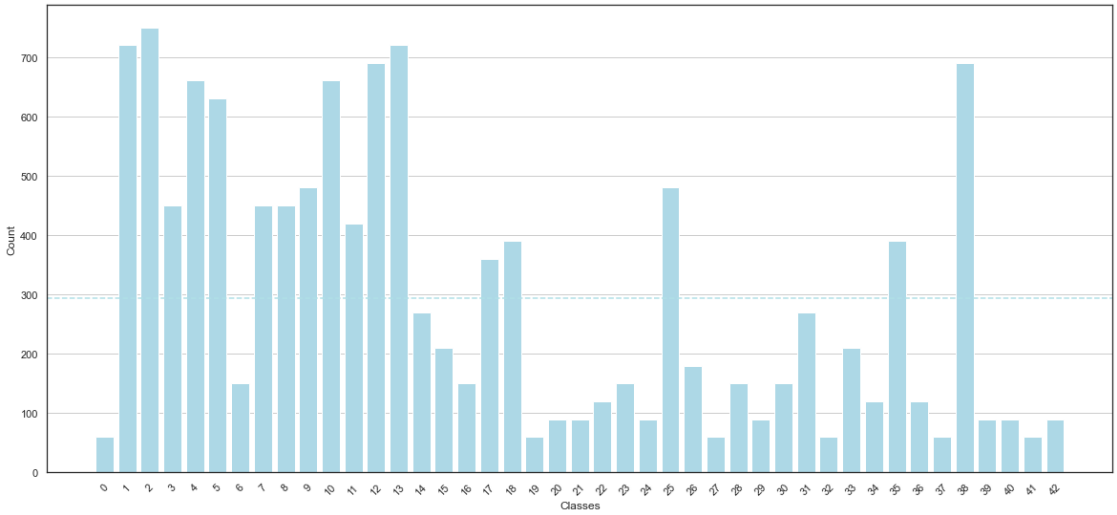
Rysunek 12. Rozkład klas w zbiorze treningowym



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. klasie 40.

Rysunek 13. Rozkład klas w zbiorze testowym

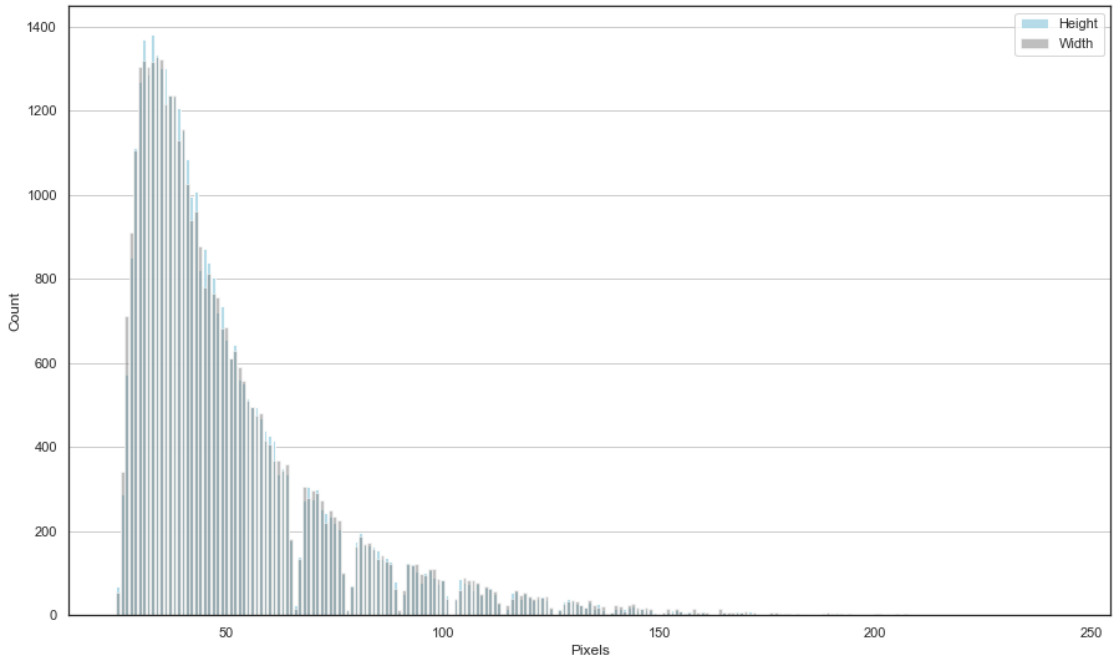


Źródło: *Opracowanie własne.*

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. Niebieskim kolorem oznaczono wysokość, a szarym szerokość obrazów. 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. Średnia wynosi około 32 pikseli dla obu wymiarów.

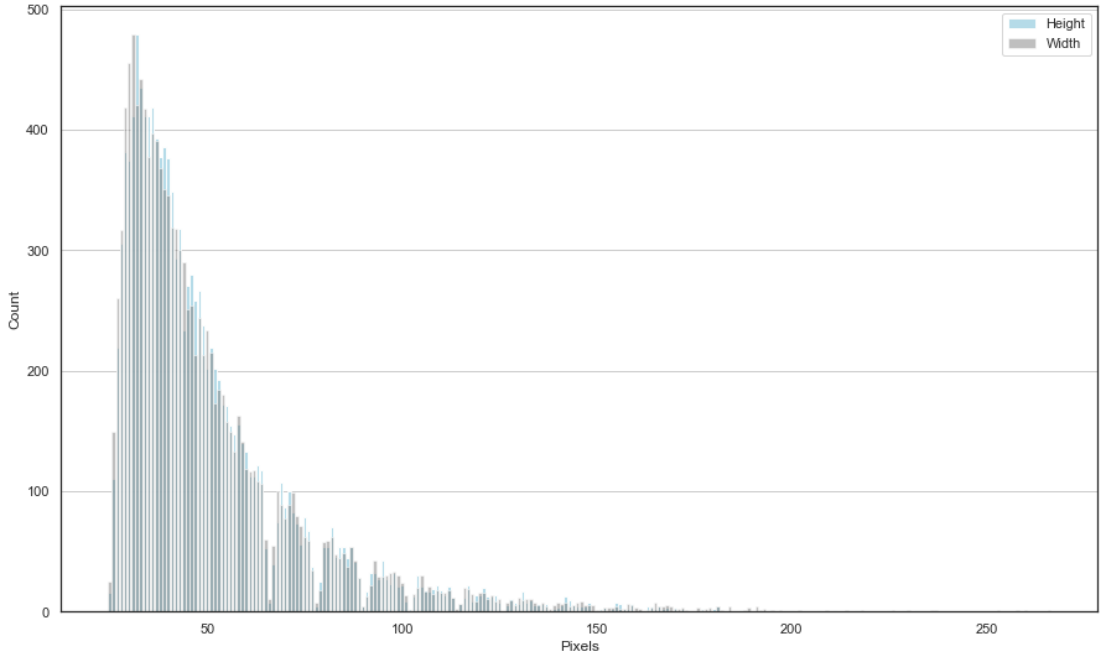
Rysunek 14. Rozkład wymiarów obrazów w zbiorze treningowym



Źródło: *Opracowanie własne.*

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

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.

3.2 Pre-processing

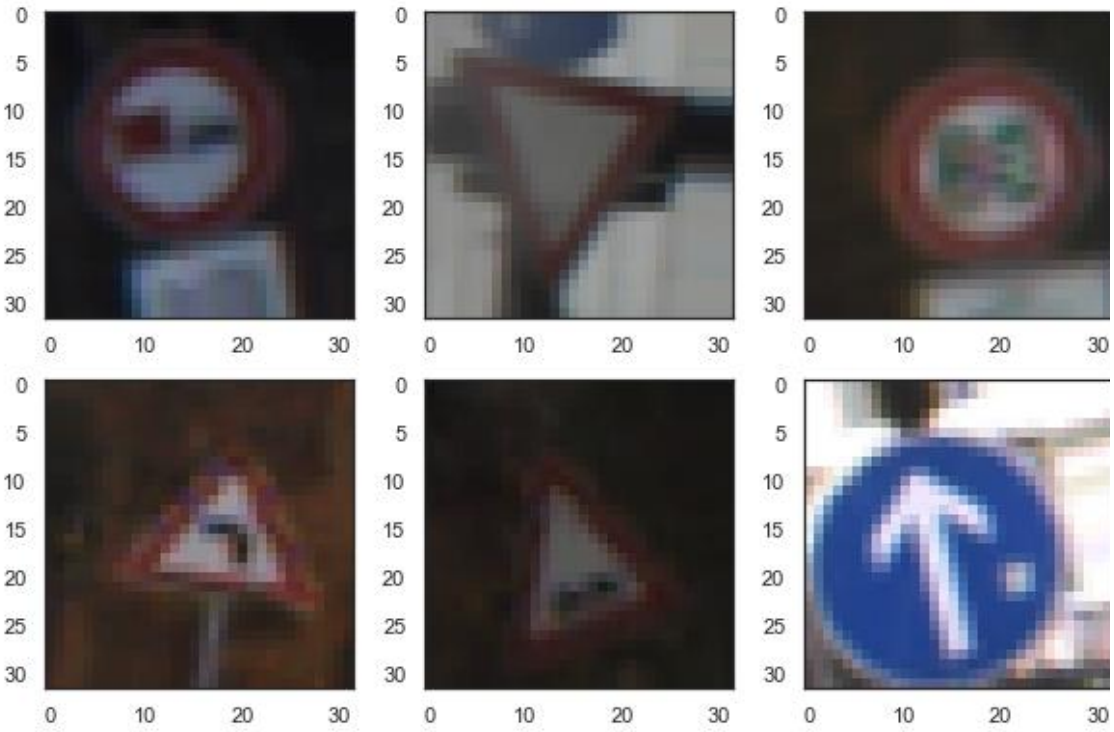
Pierwszym z kroków w pre-processingu danych było zmienienie wymiarów obrazów przy ładowaniu danych, aby ujednolicić ich wymiary. Wybrano rozmiar 32x32 pikseli, jako że wymiary większości obrazów przyjmują taką wartość. Taki zabieg ma na celu przede wszystkim przyśpieszenie trenowania CNN. Dwukrotnie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

Oprócz tego obrazy trzeba przygotować, 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 niemu mogą one być dostarczone do sieci neuronowej i poprawić predykcję. Proces ten polega na przekonwertowaniu każdej wartości kategorycznej na nową i przypisaniu wartości binarnej 1 lub 0 do kolumn. W ten sposób 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. Data augmentation jest więc procesem polegającym na sztucznym zwiększania ilości danych poprzez modyfikowanie już istniejących obserwacji. Dzięki temu powstają nowe, lekko zmienione obrazy mogące posłużyć do trenowania sieci. Dużą zaletą takiego podejścia jest to, że może to pomóc poprawić wydajność modelu. Poza tym pomaga 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 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. To przeszkadzałoby uczyć się modelowi jak faktycznie wygląda taki znak. Przykładowe zmodyfikowane obrazy można zobaczyć na rysunku nr 16.

Rysunek 16. Przykładowe zmodyfikowane obrazy



Źródło: *Opracowanie własne.*

Podsumowanie

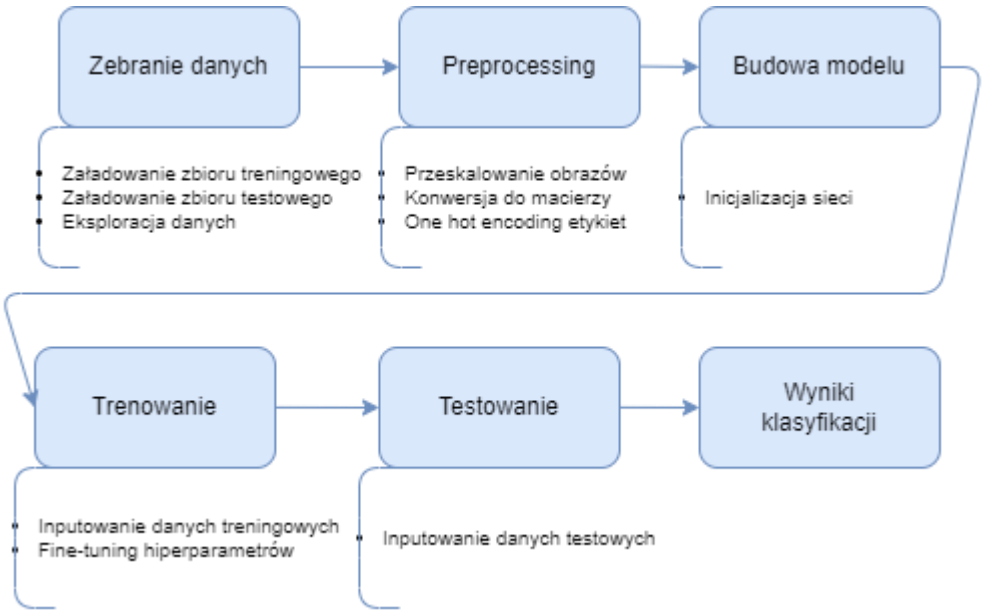
Reasumując model będzie trenowany i testowany na zbiorze "German Traffic Sign Recognition Benchmark", który jest popularnym zbiorem treningowym dla konwolucyjnych sieci neuronowych. Zbiór ten składa się z ponad 50 000 obrazów i przeszło 40 klas pochodzących z różnych znaków drogowych w Niemczech. Analiza 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.

ROZDZIAŁ 4. 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. Pokazany zostanie także proces, który zapewnił powstanie tej sieci.

Zaczynając od procesu, wizualną reprezentację przepływu pracy przedstawiono na rysunku nr 17. Na początku, obrazy znaków drogowych zostały zebrane i podzielone na zbiory treningowe i testowe. Następnie, zostały one wstępnie przetworzone przy użyciu metod przedstawionych w rozdziale III. W kolejnym kroku, wstępnie przetworzone dane zostały przekazane do modelu, który był trenowany na zbiorze treningowym. W międzyczasie próbowano dobrać także 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 poświęcone zbudowanemu modelowi. Natomiast wyniki klasyfikacji zostaną przedstawione w kolejnej części pracy.

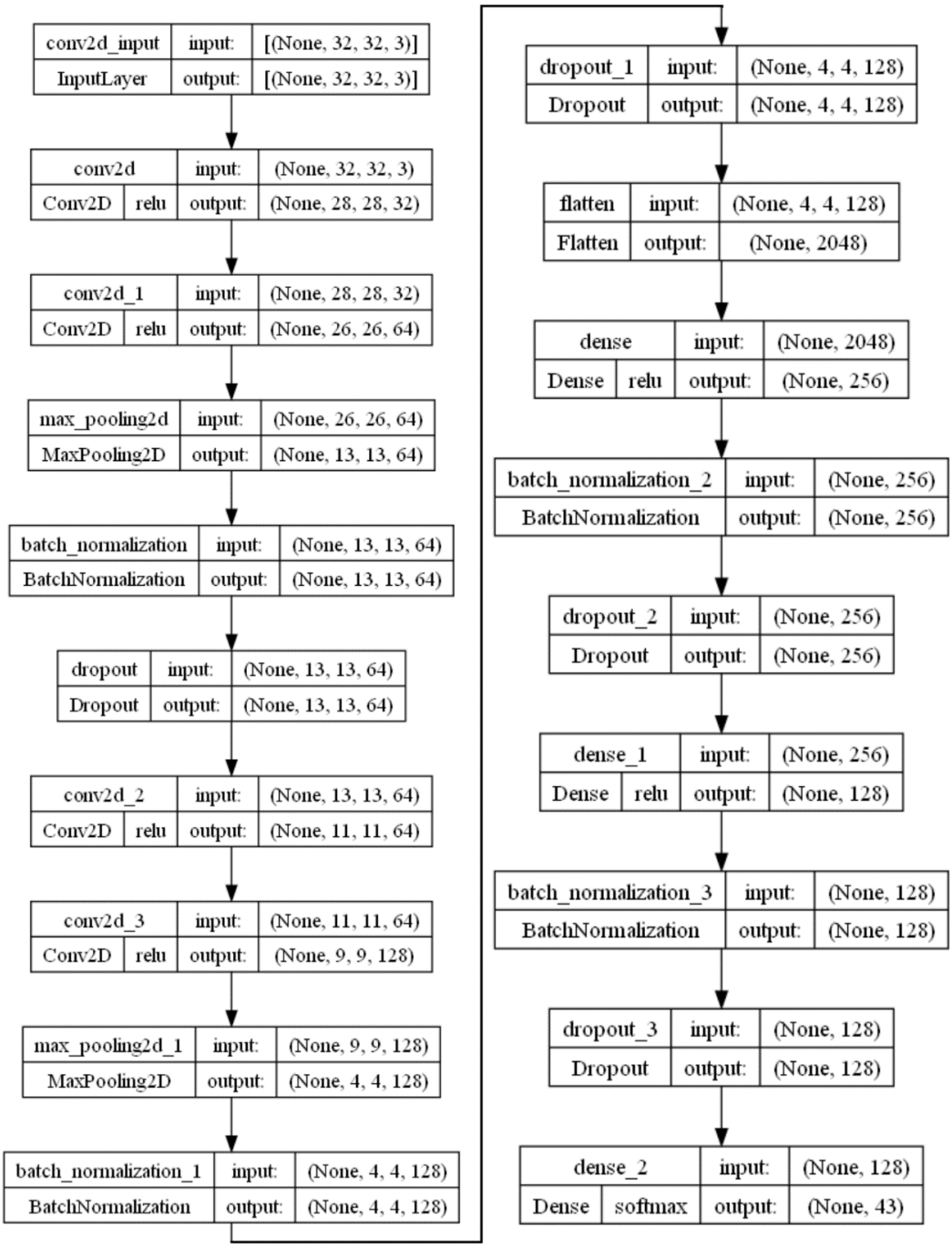
Wracając do architektury modelu. Dążenia do uzyskania lepszej dokładności w zadaniach takich jak klasyfikacja obrazów spowodowała, że powstawały coraz to głębsze sieci neuronowe z coraz większą liczbą parametrów liczoną w milionach. Jednak warto pamiętać, że samo zwiększenie głębokości niekoniecznie 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 wymiary 32x32x. Odpowiadają one wymiarom obrazów wejściowych, które uprzednio przeskalowano do takich wartości. W kolejnej warstwie konwolucyjnej użyto 32 filtry o wymiarach jądra 5x5, gdzie za funkcję 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 znajdują 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. Zwraca ona prawdopodobieństwa wystąpienia danej klasy z 43 dostępnych klas. Przy kompilacji modelu użyto

optymalizatora Adam ze zmniejszającym się tempem uczenia wraz ze wzrostem liczby epok. 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 sieci konwolucyjnej wybranej do zadania klasyfikacji, 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.

ROZDZIAŁ 5. Wyniki

W niniejszym rozdziale przedstawione zostaną wyniki klasyfikacji znaków drogowych przy użyciu modelu przedstawionego w rozdziale 4. Ponadto zostanie pokazany rozwój sieci otrzymującej kolejnej warstwy i jak doprowadziło to do wzrostu jej trafności. Dodatkowo na podstawie tych wyników wyciągnięte zostaną stosowne wnioski.

5.1 Opis wyników

Model przeszedł przez parę faz, aby przejść od stanu pierwotnego do ostatecznego, w którym osiągał on bardzo dobre wyniki na zbiorze testowym. Wyniki sieci neuronowych, które są poprzednikami modelu końcowego, zostały przedstawione w tabeli nr 2.

Tabela 2. Wyniki poprzednich modeli

Model	Acc train	Acc test	Total params
ANN 2 dense layers	5.74%	5.93%	99,755
ANN 4 dense layers	5.76%	5.93%	114,315
CNN 1 conv ,4 dense	5.75%	5.93%	938,507
CNN 1 conv, 1 dense	99.14%	86.21%	1,239,339
CNN 1 conv, 1 pool, 1 dense	99.48%	86.44%	310,539
CNN 1 conv, 1 pool, 1 dropout, 1 dense	97.32%	87.98%	310,539
CNN 2 conv, 1 pool, 1 dropout, 1 dense	98.68%	90.64%	484,523
CNN 2 conv, 2 pool, 2 dropout, 1 dense	97.35%	93.78%	118,507
CNN 2 conv, 2 pool, 3 dropout, 2 dense	95.04%	95.39%	319,979
CNN 2 conv, 2 pool, 3 dropout, 2 dense, batchnorm	99.60%	97.10%	320,875
CNN 3 conv, 2 pool, 3 dropout, 2 dense, batchnorm	99.86%	96.43%	357,931
CNN 4 conv, 2 pool, 3 dropout, 2 dense, batchnorm	99.90%	98.07%	304,747
CNN 4 conv, 2 pool, 3 dropout, 2 dense, batchnorm, changing parameters	99.89%	96.25%	546,731
CNN 4 conv, 2 pool, 4 dropout, 2 dense, batchnorm, changing parameters v2	99.64%	97.83%	391,211
CNN 4 conv, 2 pool, 4 dropout, 3 dense, batchnorm, changing parameters v3	99.82%	98.44%	697,003

CNN 4 conv, 2 pool, 4 dropout, 3 dense, batchnorm, changing learning parameters v4	99.95%	98.27%	697,003
CNN 4 conv 2 pool 4 dropout 3 dense batchnorm, changing learning parameters v4, data augmentation	99.59%	99.00%	697,003

Źródło: *Opracowanie własne.*

Pierwszy z modeli był bardzo prostą sztuczną siecią neuronową. Składał się jedynie z dwóch warstw w pełni połączonych i nie miał ani jednej warstwy konwolucyjnej. Dzięki temu łącznie posiadał niecałe 100 000 parametrów. Był to więc bardzo lekki i prosty model. Jednak prostota ta była mocno widoczna w wynikach modelu. Na zbiorze treningowym wykazywał się około 5.74% trafnością, a na zbiorze testowym wyniosła ona 5.93%. Są to bardzo słabe wyniki, jednakże można było się tego spodziewać, bo stopniu skomplikowania modelu. Sieć już po 3 epokach nie była w stanie zwiększyć swoich możliwości predykcyjnych. Kompilując model użyto kategorycznej entropii krzyżowej oraz optymalizatora Adam ze stopniem uczenia 0.001. Takie same parametry uczenia użyto w kolejnych sieciach. Dodawano jednak stopniowo warstwy. Druga sieć neuronowa miała o 2 warstwy w pełni połączone więcej, jednakże to nie pomogło w osiągnięciu zadowalających wyników. Trafność na zbiorze testowym wręcz się nie zmieniła. W związku z tym postanowiono dodać pierwszą warstwę konwolucyjną, która jest stworzona do tego typu zadań. Posiadała ona 32 filtry o wielkości jądra 3 na 3 oraz funkcję aktywacji ReLU. Prawie dziewięciokrotnie wzrosła liczba parametrów, jednak nie przyczyniło się to do wzrostu trafności modelu. Postanowiono więc zredukować liczbę warstw w pełni połączonych do jednej. Dzięki temu odnotowano skokowy wzrost trafności z około 6% do około 86% na zbiorze testowym. Jest to więc olbrzymia poprawa działania sieci, przy użyciu zaledwie dwóch warstw. Jednakże wraz ze wzrostem trafności bardzo mocno wzrosła też liczba parametrów, przekraczając tym samym próg miliona. Model także prawdopodobnie nadmiernie dopasowywał się do danych treningowych, ponieważ osiągnął tam o przeszło 10 punktów procentowych większą trafność. W obu tych sprawach pomogłaby więc warstwa poolingu, którą postanowiono dodać do modelu. Wymiary jądra filtra przyjęto na poziomie 2x2 i zastosowano max poolingu. Taki zabieg pozwolił na znaczne ograniczenie liczby parametrów z 1.2 miliona do zaledwie 300 000. Warto zauważyć, że nie spowodowało to spadku możliwości modelu. Wręcz przeciwnie, trafność wzrosła zarówno na zbiorze treningowym jak i testowym. Jednak problem przeuczenia nadał występował, tak więc dodano warstwę dropoutu, która odrzucała sygnał z 30% neuronów. Pozwoliła ona na

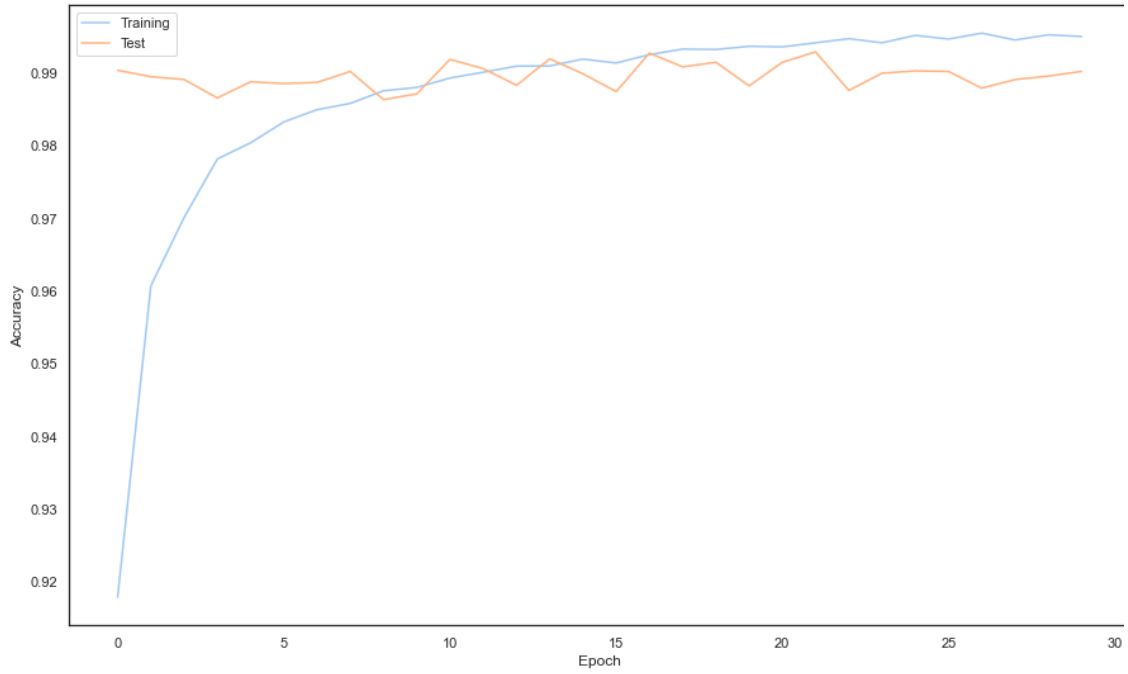
kolejne zwiększenie trafności na zbiorze treningowym do 87.98%. Z drugiej strony zabiegi te doprowadziły do obniżenia sprawności modelu na zbiorze treningowym. Dlatego też dodano drugą warstwę konwolucyjną, zmieniając w niej jedynie liczbę filtrów do 64. Następstwem tego było osiągnięcie pułapu 90% trafności na zbiorze testowym. Nadal nierozwiązany pozostawał problem przeuczenia. Z tego względu dodano jeszcze jedną warstwę poolingu oraz dropoutu. Zmniejszyło to różnicę w trafności między zbiorem treningowym i testowym do niecałych 4 punktów procentowych, a jednocześnie liczba parametrów spadła do jedynych 118 000. Wynik ten nie był jednak nadal wystarczająco dobry, więc spróbowano dodać jedną warstwę w pełni połączoną. Takie działanie jeszcze bardziej podniosło trafność na zbiorze testowym do 95.39% niemalże zrównując ją z wynikiem na zbiorze treningowym. Następnym krokiem, było dodanie warstw normalizujących. Znalazły się one po warstwach max poolingowych oraz przedostatniej warstwie w pełni połączonej. Dzięki temu trafność modelu znów znacznie wzrosła odpowiednio do 99.60% i 97.10% na zbiorach treningowym i testowym. Jednocześnie nie spowodowało to znacznego wzrostu liczby parametrów. W kolejnych krokach dodano trzecią i czwartą warstwę konwolucyjną, obie z 64 filtrami o wymiarach 3x3 i funkcjami aktywacji ReLU. W ten sposób osiągnięto niemal perfekcyjny wynik na zbiorze treningowym, czyli 99.90%. Niestety na zbiorze testowym model osiągnął jedynie 98.07% trafności. Postanowiono więc zacząć zmieniać parametry sieci, próbując znaleźć ich bardziej optymalne wartości. Tak też zmieniono liczbę filtrów w czwartej warstwie konwolucyjnej z 64 do 128. Wypróbowano różne wielkości filtrów od 5x5 do 2x2. Dodano po drodze jeszcze jedną warstwę w pełni połączoną i dropoutu. Te ostatnie także przyjmowały różne wartości od 0.2 do 0.5. Oprócz tego zwiększono wymiarowość pierwszej warstwy połączonej ze 128 do 256. Co więcej spróbowano posłużyć się malejącym tempem uczenia wraz ze wzrostem epok. Poza tym zmniejszono rozmiar batchy i zwiększono liczbę epok, jako że wraz ze wzrostem liczby parametrów potrzebowały one także więcej czasu na naukę. Zmiany parametrów pozwoliły osiągnąć trafność na poziomie 98.44% i 98.27% na zbiorach testowych. Nie był to jednak nadal satysfakcjonujący wynik, ponieważ większość modeli obecnie osiąga wartości około 99% trafności na zbiorze testowym. W związku z tym postanowiono dodać jeszcze technikę zwaną data augmentation czyli rozszerzaniem danych. Dzięki temu osiągnięto pułap 99% trafności i powstał model ostateczny, którego architektura została przedstawiona w rozdziale IV.

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

$$\text{Accuracy} = \frac{True\ Positive + True\ Negative}{True\ Positive + False\ Positive + True\ Negative + False\ Negative} \quad (6)$$

Wartości trafności jakie model osiągał przez kolejne cykle nauki zostały przedstawione na rysunku nr 19. Wzrasta ona bardzo mocno już w pierwszych epokach, aby ustabilizować się ostatecznie w okolicach 99% trafności. Wykres wskazuje, że liczba cykli była prawdopodobnie wystarczająca, jako że metryka przestała rosnąć w okolicach 25 epoki. Finalnie najwyższa trafność jaką 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 99.00%. Można więc stwierdzić, że nie doszło do problemu przeuczenia i model dobrze generalizuje, to jest równie dobrze przewiduje na nowych danych.

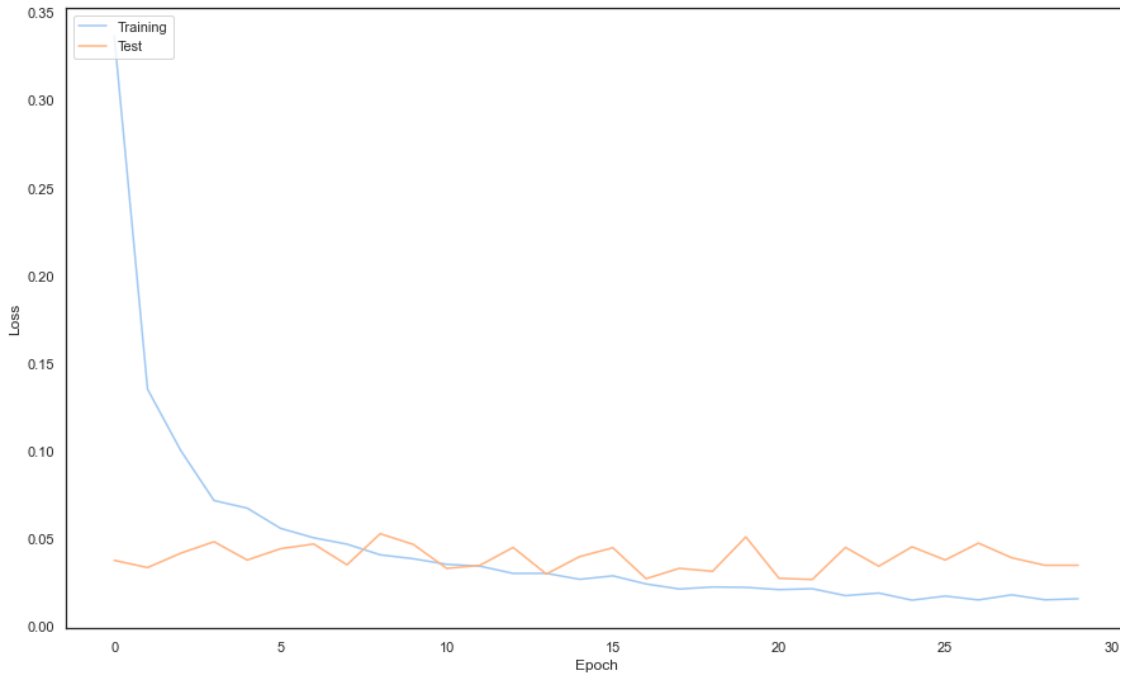
Rysunek 19. Wykres trafności modelu



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 strata dla obu zbiorów przyjmowała podobne wartości poniżej 0.05 już po około 8 epokach. Co więcej rysunek ten wskazuje również, że model prawdopodobnie nie był już zdolny do dalszego uczenia się. Dlatego zakłada się, że proces szkolenia był wystarczająco długi. Jeśli chodzi o przeuczenie można by się zastanawiać, czy nie lepiej byłoby zakończyć uczenie modelu już koło 20 epoki. Wynika to z tego, że w kolejnych cyklach wartości dla obu zbiorów zaczynają się od siebie oddalać. Jednakże są to bardzo małe różnice, 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 jego błędnego działania. 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.

Jedną z nich jest precyzja. Określa ona w jakim stopniu można zaufać pozytywnym predykcjom w danej klasie:

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{7}$$

Inaczej jest to procent pozytywnych predykcji, które faktycznie są poprawne. Tak 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 przedstawiona wzorem:

$$\text{Recall (Sensitivity)} = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{8}$$

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

$$\text{F1 score} = 2 * \frac{precision * recall}{precision + recall} \tag{9}$$

Wartości wszystkich tych miar zostały zebrane w tabeli nr 3, w której przedstawiono je w odniesieniu do każdej klasy.

Tabela 3. Raport klasyfikacyjny

class	name	precision	recall	f1-score	support
0	Speed limit (20km/h)	0.88	1	0.94	60
1	Speed limit (30km/h)	0.98	1	0.99	720
2	Speed limit (50km/h)	1	0.99	1	750
3	Speed limit (60km/h)	1	0.98	0.99	450
4	Speed limit (70km/h)	1	0.97	0.99	660
5	Speed limit (80km/h)	0.98	1	0.99	630
6	End of speed limit (80km/h)	0.99	0.99	0.99	150
7	Speed limit (100km/h)	1	1	1	450
8	Speed limit (120km/h)	1	1	1	450
9	No passing	0.99	1	1	480
10	No passing veh over 3.5 tons	1	1	1	660
11	Right-of-way at intersection	0.99	0.98	0.98	420
12	Priority road	1	0.98	0.99	690
13	Yield	1	1	1	720
14	Stop	1	1	1	270
15	No vehicles	0.98	1	0.99	210
16	Veh > 3.5 tons prohibited	1	1	1	150
17	No entry	1	1	1	360
18	General caution	1	0.97	0.98	390
19	Dangerous curve left	1	1	1	60

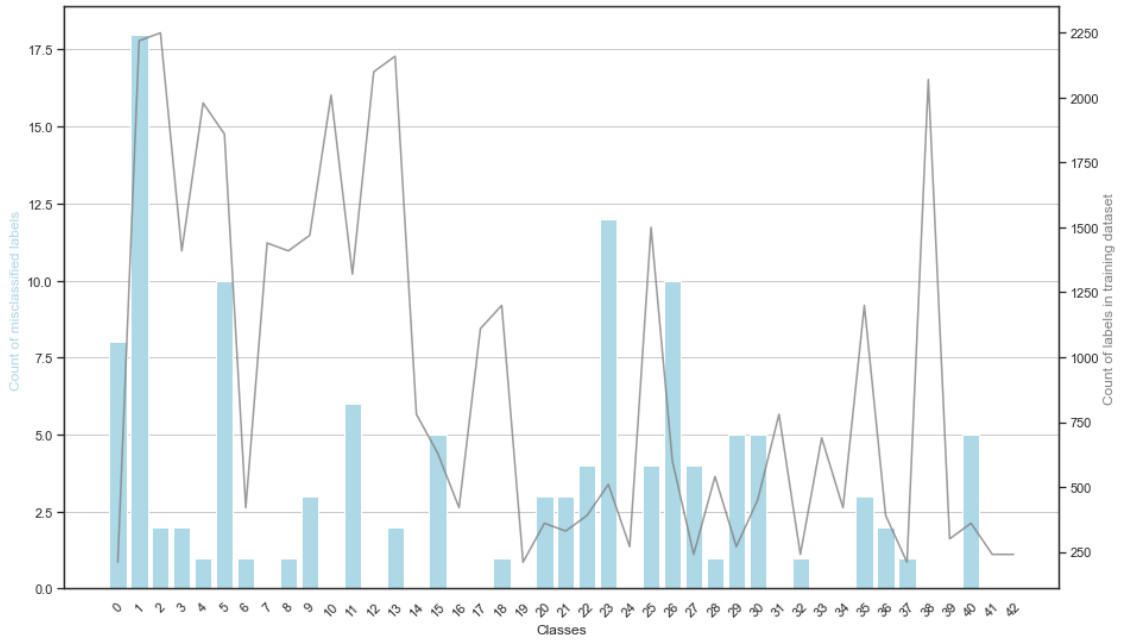
20	Dangerous curve right	0.97	1	0.98	90
21	Double curve	0.97	0.99	0.98	90
22	Bumpy road	0.97	0.93	0.94	120
23	Slippery road	0.93	1	0.96	150
24	Road narrows on the right	1	0.99	0.99	90
25	Road work	0.99	0.98	0.99	480
26	Traffic signals	0.95	0.99	0.97	180
27	Pedestrians	0.93	0.92	0.92	60
28	Children crossing	0.99	0.99	0.99	150
29	Bicycles crossing	0.95	1	0.97	90
30	Beware of ice/snow	0.96	0.9	0.93	150
31	Wild animals crossing	1	1	1	270
32	End speed + passing limits	0.98	1	0.99	60
33	Turn right ahead	1	1	1	210
34	Turn left ahead	1	1	1	120
35	Ahead only	0.99	1	1	390
36	Go straight or right	0.98	1	0.99	120
37	Go straight or left	0.98	0.98	0.98	60
38	Keep right	1	1	1	690
39	Keep left	1	0.98	0.99	90
40	Roundabout mandatory	0.95	0.97	0.96	90
41	End of no passing	1	0.97	0.98	60
42	End no passing veh > 3.5 tons	1	1	1	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.*

W tabeli możemy zauważyć, że w przypadku wszystkich 3 metryk wartości dla poszczególnych klas wahają się między 0.88 a 1.00. Im bliżej wartości jeden tym lepszy wynik. Porównując wartości metryk pomiędzy klasami warto spostrzec, że takie klasy jak limit prędkości do 100 km/h lub znak ustąp pierwszeństwa osiągają perfekcyjne wyniki we wszystkich metrykach. Można zatem założyć, że model bardzo dobrze radzi sobie z rozpoznawaniem obrazów należących do takich klas. Są jednak także klasy, gdzie sieć popełnia błędy. Jest tak chociażby w przypadku limitu prędkości do 20 km/h, nierównej drogi, uwaga piesi lub uwaga śnieg/lód. Są to klasy, dla których metryka f1 osiągnęła najniższe wartości. W tym wypadku w tabeli widać również, że we wszystkich

tych przypadkach support, czyli liczba obrazów reprezentujących daną klasę, była dość niska. Warto byłoby się w takim przypadku przyjrzeć, jak rozkłada się liczba źle sklasyfikowanych obrazów w podziale na klasy. Dodatkowo porównując to z całkowitą liczbą obrazów w każdej klasie na których obraz był uczony. Taki rozkład przedstawia rysunek nr 21.

Rysunek 21. Rozkład źle sklasyfikowanych obrazów

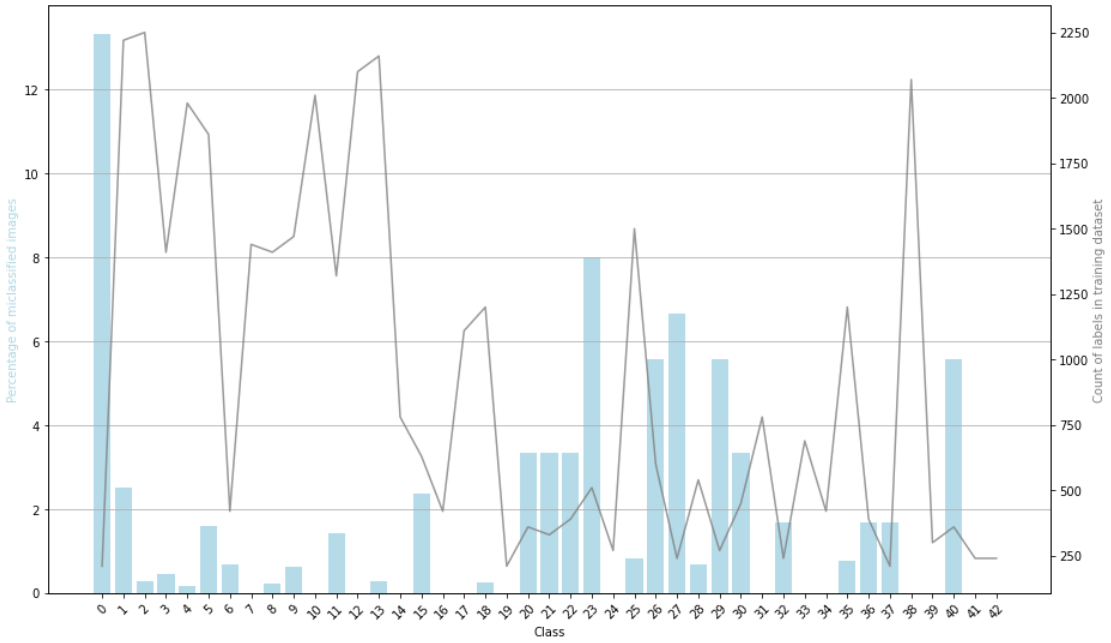


Źródło: *Opracowanie własne.*

Po lewej stronie wykresu znajduję się skala odnosząca się do błękitnych słupków, pokazujących liczbę źle sklasyfikowanych obrazów. Natomiast po prawej stronie umiejscowiona jest druga skala dotycząca szarej linii, odpowiadającej za liczbę obrazów, na których model był uczony. W większości przypadków liczba źle sklasyfikowanych obrazów nie była większa niż 5. Ciekawym przypadkiem jest klasa pierwsza (ograniczenie prędkości do 30 km/h), która posiadała bardzo dużo treningowych przykładów. Mimo to posiada najwięcej źle sklasyfikowanych obrazów, bo aż 18. Spodziewać się można było, że obrazy słabiej reprezentowane będą miały więcej błędnych klasyfikacji. Jednakże warto także pamiętać, że w skali 720 obrazów testowych daje to około 2,5% przypadków złych predykcji. Druga pod względem błędnie sklasyfikowanych obrazów jest klasa 23, czyli śliska droga. W jej przypadku liczebności obrazów w zbiorze treningowym jest jedną z najniższych, więc taki wynik nie powinien dziwić. Jednak porównując obie wymienione klasy, można spostrzec, że w procentowym

ujęciu klasa 23 wypada gorzej. Model pomylił się na zbiorze testowym w aż 8% przypadków. Procentowy rozkład poszczególnych błędów można zobaczyć na rysunku nr 22.

Rysunek 22. Procentowy rozkład źle sklasyfikowanych obrazów

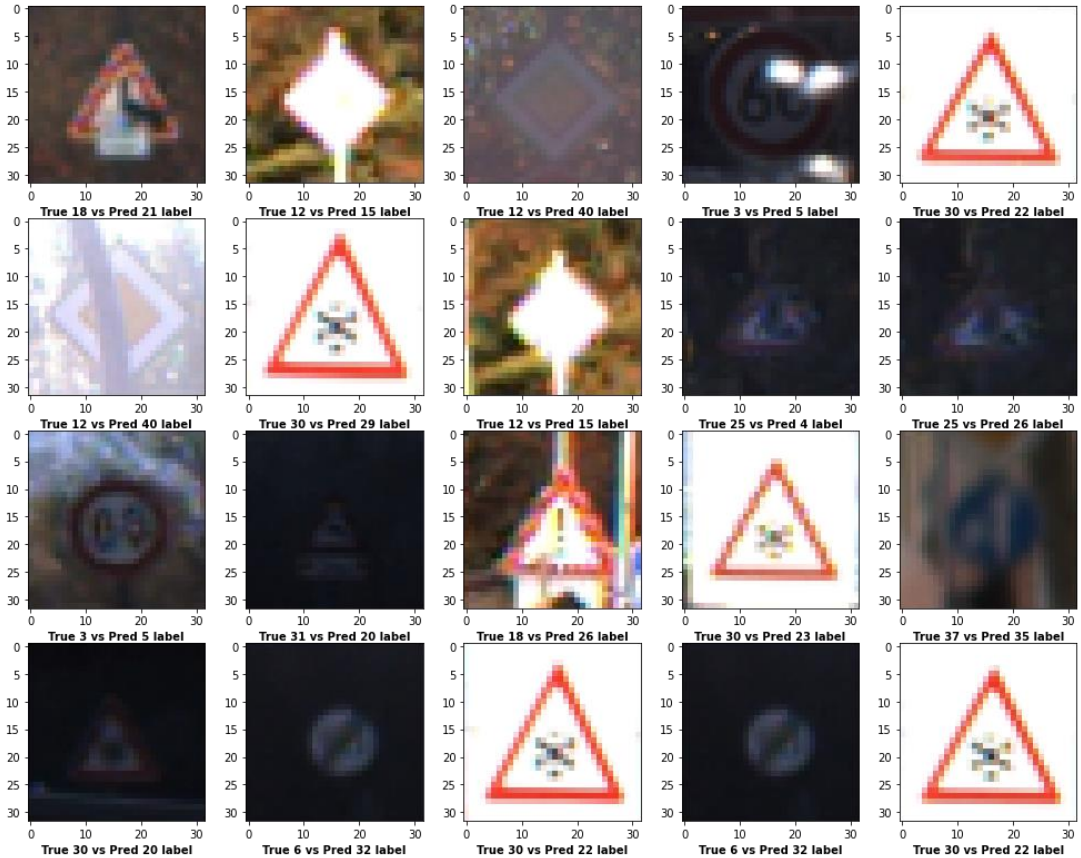


Źródło: *Opracowanie własne.*

Rysunek nr 22 zdaje się potwierdzać przypuszczenia, że to właśnie klasy z najmniejszymi zbiorami obrazów osiągają relatywnie największy odsetek źle sklasyfikowanych obserwacji. To przypomina o problemie niezbalansowanego zbioru, co było wspominano w rozdziale nr 3, przy okazji opisywania danych. Niestety nie wszystkie znaki posiadały tyle samo obserwacji, dlatego też występuje takie zróżnicowanie pomiędzy klasami. Zwiększenie liczby obrazów poprzez ich rozszerzenie na pewno pomogło sieci lepiej nauczyć się małolicznych klas, jednakże nie wyeliminowało problemu niezbalansowania. Dobrym sposobem na zobaczenie, wobec jakich znaków model może być stronniczy, jest obejrzeniem pojedynczych przypadków błędnych klasyfikacji. Na rysunku nr 23 można zobaczyć takie przypadki znaków, dla których model źle przypisał etykiety. Dla części tych przypadków można by zaryzykować stwierdzenie, że nawet człowiek miałby problem z rozpoznaniem klasy danego obiektu. Można też zauważyć czemu dane obrazy zostały źle rozpoznane. Większość z nich jest słabej rozdzielczości. Na przykład na obrazkach bardzo często powtarza się znak uwaga na lód/śnieg. Model często myli go ze znakami śliska lub nierówna droga. Jest to pewnie

spowodowane bardzo małą rozdzielczością obrazów, przez co przy ich powiększeniu stają się one dla modelu nieczytelne. Co więcej niektóre znaki są bardzo niewyraźne, zamazane lub przysłonięte jakimś refleksem świetlnym lub inną rzeczą. Przykładem jest tutaj znak ograniczenia prędkości do 60km/h lub przysłonięty znak pierwszeństwo przejazdu. Na dodatek z jednej strony część obrazów jest prześwietlona, a z drugiej strony część z nich nieodpowiednio doświetlona. Tutaj znów znak pierwszeństwa przejazdu może posłużyć jako przykład. Przez jego prześwietlenie, nie widać jego typowej żółtej barwy w środku, przez co model pomylił go z zakazem ruchu w obu kierunkach.

Rysunek 23. Przykłady błędnie sklasyfikowanych obrazów



Źródło: *Opracowanie własne.*

5.2 Wnioski

Po opisaniu wyników modelu, czas na wyciągnięcie z nich wniosków. Przedstawiony w tej pracy model osiągnął bardzo wysokie wartość trafności. Wyniki warto jednak odnieść do istniejących już badań przeprowadzonych na ten temat, aby mieć

odniesienie czy są one faktycznie satysfakcjonujące. Rezultaty przykładowych modeli zostały zestawione w tabeli nr 4.

Tabela 4. Przykładowe rezultaty modeli

Publikacja	Model	Parametry (milion)	Trafność (%)
Szegedy et al. (2015)	GoogleNet	3.7	98.31
Krizhevsky et al. (2012)	AlexNet	15.78	97.08
Cireşan et al. (2012)	MCDNN	90	99.46
Garcia et al. (2018)	CNN with 3 Spatial Transformers	14.6	99.71
Jin et. al. (2014)	Enesemble CNNs	23	99.65
Stallkamp et al. (2012)	Human average	-	98.84
	Stworzony model	0.7	99.00

Źródło: *Opracowanie własne.*

Pod względem trafności stworzony model plasuje się mniej więcej pośrodku tabeli. Istnieją sieci mające nieznacznie gorszą trafność takie jak omawiany w pierwszym rozdziale AlexNet. Są też takie, które są trochę lepsze pod względem tej metryki, na przykład MCDNN, czyli wielokolumnowa głęboka sieć neuronowa, która również została omówiona w pierwszym rozdziale. Jednak różnice te nie są znaczne. Do najlepszego modelu, który znajduje się na oficjalnej stronie GTSRB, stworzony model traci jedynie 0.71% trafności. Warto w tym momencie przyjrzeć się jak wielka przepaść dzieli te modele, jeśli chodzi o liczbę parametrów, potrzebną do osiągniecia takich wyników. Wszystkie przedstawione modele posiadają miliony parametrów, które posłużyły do ich wytrenowania. Najlepszy z nich pod względem trafności potrzebował ich aż 14.6 miliona. Do obliczenia takich sieci wymagane jest więc zdecydowanie więcej czasu, a także mocniejszy sprzęt. Zatem stworzony model jest poniekąd kompromisem między trafnością, która i tak jest na bardzo wysokim poziomie, a skomplikowaniem modelu, który w porównaniu z innymi jest bardzo prosty. Warto również zauważyć, że model przedstawiony w tej pracy radzi sobie lepiej niż średni wynik otrzymany przez grupę 32 osób testowanych na takim samym zbiorze. Wynik ten jest oznaczony w tabeli nr 3 jako „human average”.

Podsumowanie

Ogółem ujmując, w rozdziale opisano wyniki klasyfikacji znaków drogowych przy użyciu modelu przedstawionego w rozdziale IV. Dodatkowo wyciągnięto na ich podstawie wnioski. W celu oceny skuteczności modelu, użyto popularnych metryk w uczeniu maszynowym, takich jak trafność, precyzja, czułość i f1 score. Trafność modelu była bardzo dobra i wyniosła około 99%. Inne metryki również wskazały na bardzo dobre wyniki sieci. Co ważne, niedoszło do jej przeuczenia. Porównano także otrzymane rezultaty z innymi modelami. Na ich tle wypadają one dobrze, jeśli weźmie się pod uwagę liczbę parametrów potrzebnych do wytrenowania modelu. Najlepszy model potrzebował ich przeszło 14 milionów, w porównaniu z 0.7 miliona w stworzonym w tej pracy modelu.

ROZDZIAŁ 6. Podsumowanie

Zadaniem pracy było pokazanie zastosowania konwolucyjnej sieci neuronowej w celu klasyfikacji znaków drogowych. Dlatego też stworzono lekki obliczeniowo, prosty a jednocześnie skuteczny model. Trenowanie i testowanie sieci odbyło się na zbiorze niemieckich znaków drogowych GTSRB. Pierwotne modele nie spełniły wszystkich oczekiwań, jednak po rozbudowaniu architektury sieci i fine tuningu hiperparametrów udało się osiągnąć bardzo dobre wyniki. Ostatecznie model składał 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. Liczba jej parametrów wyniosła około 0.7 miliona, co w porównaniu z innymi modelami, jest bardzo niskim wynikiem. Natomiast jeśli chodzi o trafność stworzona sieć osiągnęła wynik 99% skutecznych predykcji. Tak więc model skutecznie wykrywa i klasyfikuje znaki drogowe o różnych kształtach, rozmiarach i kolorach. Utrzymuje jednocześnie bardzo wysoką trafność w klasyfikowaniu znaków drogowych na przestrzeni wszystkich klas. Wskazuje to na jego odporność i możliwość adaptacji w rzeczywistych scenariuszach. Dodatkowo jest on prosty w budowie i jego architektura jest lekka, co świadczy o dobrej wydajności proponowanego modelu.

Pracą, którą można by wykonać w przyszłości jest skupienie się na klasach, z którymi model radził sobie najsłabiej. Co więcej warto by spojrzeć na problemy takie jak niedoświetlenie lub prześwietlenie obrazów. Możliwe, że bardziej zaawansowany preprocessing byłby w stanie zniwelować niektóre z tych problemów. Pozwoliłoby to zwiększyć jeszcze bardziej trafność modelu bez konieczności zwiększania liczby jego parametrów i skomplikowania.

Co więcej ciekawym eksperymentem mogłaby być próba wytrenowania i przetestowania modelu na innych zbiorach znaków drogowych takich jak zbiór belgijski (BTSC).

Innym eksperymentem mogłoby być wypróbowanie możliwości modelu w rzeczywistych warunkach. Wymagałoby to obudowania go w bardziej ogólny system, który wpierw lokalizowałby znaki drogowe na drodze, a następnie klasyfikował je.

BIBLIOGRAFIA

Benson, A, B.C. Tefft, A.M. Svancara, i W.J. Horrey. „Potential Reduction in Crashes, Injuries and Deaths from Large-Scale Deployment of Advanced Driver Assistance Systems". AAA Foundation for Traffic Safety, 26 wrzesień 2018. https://aaafoundation.org/potential-reduction-in-crashes-injuries-and-deaths-from-large-scale-deployment-of-advanced-driver-assistance-systems/.

Brookhuis, Karel A., Dick de Waard, i Wiel H. Janssen. „Behavioural Impacts of Advanced Driver Assistance Systems–an Overview". *European Journal of Transport and Infrastructure Research* 1, nr 3 (1 czerwiec 2001). https://doi.org/10.18757/ejtir.2001.1.3.3667.

Chellapilla, Kumar, Sidd Puri, i Patrice Simard. „High Performance Convolutional Neural Networks for Document Processing". Suvisoft, 2006. https://hal.inria.fr/inria-00112631.

Cireşan, Dan, Ueli Meier, Jonathan Masci, i Jürgen Schmidhuber. „Multi-Column Deep Neural Network for Traffic Sign Classification". *Neural Networks*, Selected Papers from IJCNN 2011, 32 (1 sierpień 2012): 333–38. https://doi.org/10.1016/j.neunet.2012.02.023.

Debevec, Paul E, i Jitendra Malik. „Recovering High Dynamic Range Radiance Maps from Photographs", b.d.

Escalera, Sergio, Xavier Baró, Oriol Pujol, Jordi Vitrià, i Petia Radeva. *Traffic-Sign Recognition Systems*. SpringerBriefs in Computer Science. London: Springer, 2011. https://doi.org/10.1007/978-1-4471-2245-6.

Fukushima, Kunihiko. „Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". *Biological Cybernetics* 36, nr 4 (kwiecień 1980): 193–202. https://doi.org/10.1007/BF00344251 .

Habibi Aghdam, Hamed, Elnaz Jahani Heravi, i Domenec Puig. „A Practical Approach for Detection and Classification of Traffic Signs Using Convolutional Neural Networks". *Robotics and Autonomous Systems* 84 (1 październik 2016): 97–112. https://doi.org/10.1016/j.robot.2016.07.003.

Haque, Wasif Arman, Samin Arefin, A. S. M. Shihavuddin, i Muhammad Abul Hasan. „DeepThin: A Novel Lightweight CNN Architecture for Traffic Sign Recognition without GPU Requirements". *Expert Systems with Applications* 168 (15 kwiecień 2021): 114481. https://doi.org/10.1016/j.eswa.2020.114481.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, i Jian Sun. „Deep Residual Learning for Image Recognition". arXiv, 10 grudzień 2015. https://doi.org/10.48550/arXiv.1512.03385.

Hubel, D. H., i T. N. Wiesel. „Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". *The Journal of Physiology* 160, nr 1 (styczeń 1962): 106-154.2.
———. „Receptive fields of single neurones in the cat's striate cortex". *The Journal of Physiology* 148, nr 3 (październik 1959): 574–91.

Krizhevsky, Alex, Ilya Sutskever, i Geoffrey E Hinton. „ImageNet Classification with Deep Convolutional Neural Networks". W *Advances in Neural Information Processing Systems*, T. 25. Curran Associates, Inc., 2012. https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html.

Krogh, Anders. „What Are Artificial Neural Networks?" *Nature Biotechnology* 26, nr 2 (luty 2008): 195–97. https://doi.org/10.1038/nbt1386.

Lay, M. G. „HISTORY OF TRAFFIC SIGNS. IN: THE HUMAN FACTORS OF TRANSPORT SIGNS". *Publication of: CRC Press LLC*, 2004. https://trid.trb.org/view/702336.

LeCun, Y., B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, i L. D. Jackel. „Backpropagation Applied to Handwritten Zip Code Recognition". *Neural Computation* 1, nr 4 (grudzień 1989): 541–51. https://doi.org/10.1162/neco.1989.1.4.541.

Lowe, D.G. „Object recognition from local scale-invariant features". W *Proceedings of the Seventh IEEE International Conference on Computer Vision*, 2:1150–57 t.2, 1999. https://doi.org/10.1109/ICCV.1999.790410.

Oh, Kyoung-Su, i Keechul Jung. „GPU Implementation of Neural Networks". *Pattern Recognition* 37, nr 6 (czerwiec 2004): 1311–14. https://doi.org/10.1016/j.patcog.2004.01.013.

Papert, Seymour A. „The Summer Vision Project", 1 lipiec 1966. https://dspace.mit.edu/handle/1721.1/6125.

Peden, M. M., i World Health Organization, red. *World Report on Road Traffic Injury Prevention*. Geneva: World Health Organization, 2004.

Priese, L., J. Klieber, R. Lakmann, V. Rehrmann, i R. Schian. „New results on traffic sign recognition". W *Proceedings of the Intelligent Vehicles '94 Symposium*, 249–54. Paris, France: IEEE, 1994. https://doi.org/10.1109/IVS.1994.639514.

Roberts, Lawrence G. „Machine Perception of Three-Dimensional Solids". Thesis, Massachusetts Institute of Technology, 1963. https://dspace.mit.edu/handle/1721.1/11589.

Stallkamp, Johannes, Marc Schlipsing, Jan Salmen, i Christian Igel. „The German Traffic Sign Recognition Benchmark: A multi-class classification competition". W *The 2011 International Joint Conference on Neural Networks*, 1453–60, 2011. https://doi.org/10.1109/IJCNN.2011.6033395.

Viola, P., i M. Jones. „Rapid Object Detection Using a Boosted Cascade of Simple Features”. W *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1:I-511-I–518. Kauai, HI, USA: IEEE Comput. Soc, 2001. https://doi.org/10.1109/CVPR.2001.990517.

Waibel, A., T. Hanazawa, G. Hinton, K. Shikano, i K.J. Lang. „Phoneme recognition using time-delay neural networks”. *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37, nr 3 (marzec 1989): 328–39. https://doi.org/10.1109/29.21701.

World Health Organization. *Global Status Report on Road Safety 2018: Supporting a Decade of Action*. Geneva, Switzerland: WHO, 2018.

Yamaguchi, Kouichi, Kenji Sakamoto, Toshio Akabane, i Yoshiji Fujimoto. „A Neural Network for Speaker-Independent Isolated Word Recognition”. W *First International Conference on Spoken Language Processing (ICSLP 1990)*, 1077–80. ISCA, 1990. https://doi.org/10.21437/ICSLP.1990-282.

SPIS TABEL

SPIS RYSUNKÓW

STRESZCZENIE

Niniejsza praca przedstawia zastosowanie konwolucyjnych sieci neuronowych w klasyfikacji znaków drogowych. W tym celu podjęto się stworzenia lekkiego obliczeniowo i skutecznego modelu klasyfikującego znaki drogowe. Mógłby on potencjalnie przysłużyć się poprawie bezpieczeństwa na drogach, gdyby zastosować go w systemie ADAS.

Praca rozpoczyna się od wprowadzenia do problemu wypadków drogowych i czynnika ludzkiego. Następnie przechodzi przez historię rozwoju pojazdów autonomicznych, w których takie systemy są niezbędne. Co więcej przedstawia historię rozwoju wizji komputerowej oraz modeli z nią związanych. Kolejno opisane zostają pojęcia związane z sieciami neuronowymi i ich działanie. W następnym rozdziale scharakteryzowany zostaje zbiór GTSRB służący do treningu i testowania. Opisana została także obróbka obrazów przed wysłaniem ich do sieci neuronowej, której architektura składa się tylko z 18 warstw i 0.7 miliona parametrów. Przedstawione zostają wyniki, w których sieć osiągnęła trafność na poziomie 99%. Dokonuje się także porównania rezultatów sieci z innymi modelami z wybranych publikacji. Koniec końców przedstawione są możliwe modyfikacje mogące poprawić trafność oraz potencjalne połączenie klasyfikatora z systemem detekcji znaków.

Zaproponowana w pracy sieć co prawda nie osiąga najlepszej trafności spośród przytoczonych metod, jednakże oferuje ona najlepszy kompromis między skomplikowaniem i obciążeniem obliczeniowym, a skutecznością predykcji.