

Studium Magisterskie

Kierunek: Analiza Danych – Big Data

Kamil Wilnicki

Nr albumu 82361

**Wykorzystanie splotowych sieci neuronowych w procesie klasyfikacji samochodów**

Praca magisterska

napisana w Katedrze/Instytucie

……………………………………………

pod kierunkiem naukowym

Dr Sebastiana Zająca

Warszawa 2023

Table of Contents

[Wstęp 4](#_Toc133872731)

[Rozdział I. Sieci neuronowe 5](#_Toc133872732)

[I.1 Popularyzacja sieci neuronowych i ich związek ze sztuczną inteligencją 5](#_Toc133872733)

[I.2 Historia sieci neuronowych 8](#_Toc133872734)

[I.3 Ogólny zarys sieci neuronowych 10](#_Toc133872735)

[I.3.1 Neurony biologiczne 10](#_Toc133872736)

[I.3.2 Neuron 12](#_Toc133872737)

[I.3.3 Uczenie głębokie 13](#_Toc133872738)

[I.4 Funkcja aktywacji 14](#_Toc133872739)

[I.5 Propagacja wsteczna 18](#_Toc133872740)

[Rozdział II. Splotowe sieci neuronowe 21](#_Toc133872741)

[II.1 Historia splotowych sieci neuronowych 21](#_Toc133872742)

[II.2 Wartość biznesowa w poszczególnych branżach 23](#_Toc133872743)

[II.3 Architektura splotowych sieci neuronowych 24](#_Toc133872744)

[II.4 Warstwa splotowa 26](#_Toc133872745)

[II.4.1 Filtry 28](#_Toc133872746)

[II.5 Warstwa łącząca 31](#_Toc133872747)

[II.6 Warstwa w pełni połączona 33](#_Toc133872748)

[Rozdział III. Zbiór danych wykorzystany do budowy modelu 35](#_Toc133872749)

Wstęp

-hipotezy badawcze

-streszczenie rozdziałów

-

# Sieci neuronowe

Celem Rozdziału I jest ukazanie historii rozwoju dziedziny sieci neuronowych. Dany rozdział również przedstawia, dlaczego sieci neuronowe są aktualnie tematem popularnym oraz gdzie można ową dziedzinę zaklasyfikować. W danym rozdziale opisana jest zależność pomiędzy naturalnymi neuronami a sztucznymi sieciami neuronowymi oraz zostaje szczegółowo przedstawiony opis ich funkcjonowania. Ów rozdział jest kluczowy z punktu widzenia postawionych hipotez, gdyż niemożliwym jest pełne zrozumienie splotowych sieci neuronowych bez zapoznania się z założeniami opisanymi w danym rozdziale.

## Popularyzacja sieci neuronowych i ich związek ze sztuczną inteligencją

Sieci neuronowe można zdefiniować jako modele statystyczne które są podstawą uczenia głębokiego. Wraz z rozwojem technologii, sztuczne sieci neuronowe (SSN) zostały spopularyzowane w nowych branżach. Już dzisiaj SSN są jednym z kluczowych algorytmów w np. medycynie, finansach bądź branży samochodowej[[1]](#footnote-1). Wysoki wzrost istotności danych modeli, wynikł z wielu czynników, niemniej jednak najistotniejszymi były:

* ilość danych – Wraz z rozwojem technologii, znacznie zwiększył się ogólny zasób danych, które mogą być wykorzystane w tworzeniu analiz, bądź predykcji. Architektura sztucznych sieci neuronowych przyczyniła się do ich efektywności w procesie przetwarzania dużych zbiorów danych[[2]](#footnote-2).
* moc obliczeniowa – Wysokie wzrosty w mocy obliczeniowej są widoczne od ponad 30 lat[[3]](#footnote-3). Dany wzrost idealnie obrazuje prawo Moore’a, które mówi, iż liczba tranzystorów w mikroczipach zwiększa się dwukrotnie co dwa lata[[4]](#footnote-4). Dany wzrost spowodował, iż sieci neuronowe stają się wydajniejsze a czas przeznaczany na ich uczenie i stosowanie jest krótszy.
* popularność – Sieci neuronowe stały się tematem rozmów wielu inżynierów, ekonomistów, naukowców jak i inwestorów. Ci ostatni widzą w tych modelach wiele możliwości rozwoju, które mogą nieść za sobą wzrost wartości firm praktykujących używanie sieci neuronowych. Dana zależność powoduję wysoki wzrost kapitału, który jest inwestowany w rozwój uczenia głębokiego[[5]](#footnote-5).

Sztuczne sieci neuronowe posiadają istotny czynnik, który wyróżnia je od innych modeli, a mianowicie jest nim skuteczność w rozwiązywaniu obszernej grupy problemów. Dane algorytmy mogą być wykorzystane zarówno w obliczaniu prawdopodobieństwa zajścia przykładowego zdarzenia, jak i przy kategoryzowaniu tekstu, obrazu jak i dźwięku[[6]](#footnote-6).

Jak zostało to wspomniane wcześniej SSN należą do grupy algorytmów z uczenia głębokiego. Rysunek 1 przedstawia relacje pomiędzy uczeniem głębokim a dwoma terminami, które często są ze sobą mylone.

Rysunek 1 Sztuczna inteligencja, uczenie maszynowe oraz uczenie głębokie

Diagram

Description automatically generated

Źródło: Opracowanie własne na podstawie Francois Chollet, *Deep learning with Python*, Manning, Shelter Island, 2021, s.40.

Najbardziej obszerną dziedziną jest sztuczna inteligencja. Do niej możemy zaklasyfikować wszystkie czynności, których głównym celem jest automatyzacja procesów, które zazwyczaj są wykonywane przez ludzki umysł. Tak więc do danego obszaru możemy zaklasyfikować zarówno uczenie maszynowe, jak i procesy decyzyjne opierające się na warunkach utworzonych za pomocą eksperckiej bądź statystycznej wiedzy[[7]](#footnote-7). Uczenie maszynowe natomiast opiera swój proces decyzyjny na danych wejściowych. Za pomocą danych, algorytm uczy się i próbuje znaleźć odpowiednie wzorce do rozwiązania określonej analizy. Uczenie maszynowe wykorzystuje zagadnienia takie jak np. optymalizacja, prawdopodobieństwo bądź statystykę[[8]](#footnote-8). Uczenie maszynowe możemy podzielić na poniższe grupy:

* Uczenie nadzorowane – algorytmy należące do danej grupy uczą się za pomocą danych wejściowych oraz ich etykiet. Za pomocą etykiet model jest w stanie wyszukiwać wzorów w danych, aby prawidłowo wykonywać analizę[[9]](#footnote-9).
* Uczenie nienadzorowane – algorytmy uczenia nienadzorowanego, w przeciwieństwie do uczenia nadzorowanego, nie potrzebują etykiet przypisanych do danych. Modele te same odnajdują istotne wzorce, które są wykorzystane do grupowania[[10]](#footnote-10).
* Uczenie częściowo nadzorowane –To sekcja algorytmów wykonujących analizy na danych, w których tylko ich część posiada przypisane etykiety. Taka forma uczenia wykorzystywana jest w sytuacji, gdy uzyskanie danych z etykietami niesie za sobą wysokie koszty[[11]](#footnote-11).
* Uczenie wzmocnione – Jest to typ uczenia maszynowego wykorzystuje informacje pozyskane w procesie jego użytkowania. Za pomocą pozyskanych danych algorytmy uczenia wzmocnionego są w stanie dostosowywać się, aby jak najlepiej rozwiązywać określone zadanie[[12]](#footnote-12).

Uczenie głębokie natomiast jest podsekcją uczenia maszynowego. Do algorytmów uczenia głębokiego możemy zaliczyć modele, które wykorzystują wiele warstw neuronów[[13]](#footnote-13). Tak więc, do uczenia głębokiego możemy zaklasyfikować splotowe sieci neuronowe, które w swojej architekturze mogą posiadać wiele warstw.

## Historia sieci neuronowych

Za początek istnienia sztucznych sieci neuronowych podaje się rok 1943. W danym roku Warren McCulloch oraz Walter Pitts opublikowali artykuł *A Logical Calculus of the Ideas Immanen in Nervous Activity,* w którym przedstawili matematyczny model, którego celem była analiza danych oraz wnioskowanie za pomocą procesu podobnego do zwierzęcego mózgu[[14]](#footnote-14). Architektura danego modelu zakłada, iż wszystkie zmienne objaśniające muszą być zmiennymi binarnymi a funkcja aktywacji neuronów, przyjmując wyznaczoną granicę daje wartość równą 1 w przypadku, gdy suma iloczynu danych wejściowych oraz ich wag jest większa od danej granicy. W przeciwnym wypadku wyjściowa wartość danego neuronu wynosiła 0[[15]](#footnote-15).

Kolejnym istotnym przełomem w dziedzinie sztucznych sieci neuronowych jest twierdzenie Donalda Hebba, z roku 1949, o dostosowywaniu się wag poszczególnych połączeń. Cytat danego autora „*neurons that fire together wire together*” sugeruje, iż w sytuacji, gdy przykładowe neurony aktywują się w tym samym momencie, to połączenie pomiędzy nimi staje się silniejsze. Z dzieła Donalda Hebba można więc wywnioskować, że wagi pomiędzy neuronami nie powinny by stałe a wręcz powinny się zmieniać przy iteracji uczenia[[16]](#footnote-16).

W 1955 roku została zorganizowana konferencja AI w Darthmouth, której celem była wymiana wiedzy oraz pomysłów związanych z rozwojem danej dziedziny. Owa konferencja była wówczas największym zgromadzeniem ludzi specjalizującym się w zakresie AI, a także zapoczątkowała wiele nowych badań związanych z rozwojem sieci neuronowych[[17]](#footnote-17).

W okresie od 1957 do 1958 Rosenblatt oraz Wightman utworzyli pierwszy perceptron. Aby utworzyć dany model, naukowcy zbudowali komputer o łącznej wadze wynoszącej 5 ton[[18]](#footnote-18). Porównując wagę, a zarazem moc obliczeniową, owego komputera do laptopów w dzisiejszych czasach, można zauważyć diametralny rozwój technologii w danej dziedzinie. Dany perceptron bazowany był na idei modelu McCullocha oraz Pittsa oraz uczenia się modelu zaproponowanej przez Donalda Hebba. Tak więc, utworzony perceptron został zbudowany w takiej formie, aby przy każdej iteracji uczenia się modelu wagi poszczególnych połączeń dostosowywały się tak, aby predykcje perceptronu byłyby jak najbardziej przybliżone do faktycznych wartości zmiennej objaśnianej. Perceptron należy więc do grupy algorytmów uczenia nadzorowanego[[19]](#footnote-19).

Kilka lat po utworzeniu perceptronu Bernard Widrow oraz Marcian Hoff utworzyli modele: adaline oraz madaline. Model adaline różni się od perceptronu formą uczenia modelu. Perceptron uczy się porównując binarne predykcje modelu do ich prawdziwych wartości, natomiast adaline dostosowuje wagi porównując wyniki funkcji kombinacji, która przyjmuje wartość kombinacji liniowej, do faktycznych wartości zmiennej objaśnianej.[[20]](#footnote-20) Model madaline natomiast jest siecią neuronową, składającą się z wielu neuronów adaline. Owy model jest pierwszym modelem sieci neuronowych, który do dziś jest powszechnie stosowany w aplikacjach[[21]](#footnote-21).

Po danych wydarzeniach, które zostały uznane za fundamenty sztucznych sieci neuronowych nastąpiła przerwa w rozwoju danej dziedziny. Przerwa ta, która trwała od lat 60 do początku lat 80, była spowodowana przez opinie, iż utworzenie sztucznej sieci neuronowej z wieloma warstwami ukrytymi jest nieefektywne. Dane przekonanie spowodowało zanik funduszy, których celem było rozwijanie badań sztucznych sieci neuronowych. W latach 80 nastąpiła ponowna chęć rozwijania SSN, gdzie głównymi narodami, które przyczyniły się do rozwoju były Stany Zjednoczone oraz Japonia. Kluczowym odkryciem w danym okresie było stworzenie w roku 1986 techniki uczenia się modelu zwaną propagacją wsteczną[[22]](#footnote-22).

W latach 90 ponownie nastąpiła przerwa w rozwoju SSN. Dana przerwa spowodowana była między innymi przez wysoką popularność nowego algorytmu uczenia maszynowego zwanego SVM (Support Vector Machines), który na dotychczasowe możliwości mocy obliczeniowej, dawał lepsze rezultaty[[23]](#footnote-23).

W dzisiejszych czasach przebiega kolejna fala rozwoju sieci neuronowych, która w przeciwieństwie do poprzednich lat ma niskie szanse na szybki zanik. Głównymi przyczynami popularności sztucznych sieci neuronowych jest doścignięcie możliwości mocy obliczeniowej do wymagań danych algorytmów oraz ciągłe udoskonalanie danej grupy modeli.[[24]](#footnote-24)

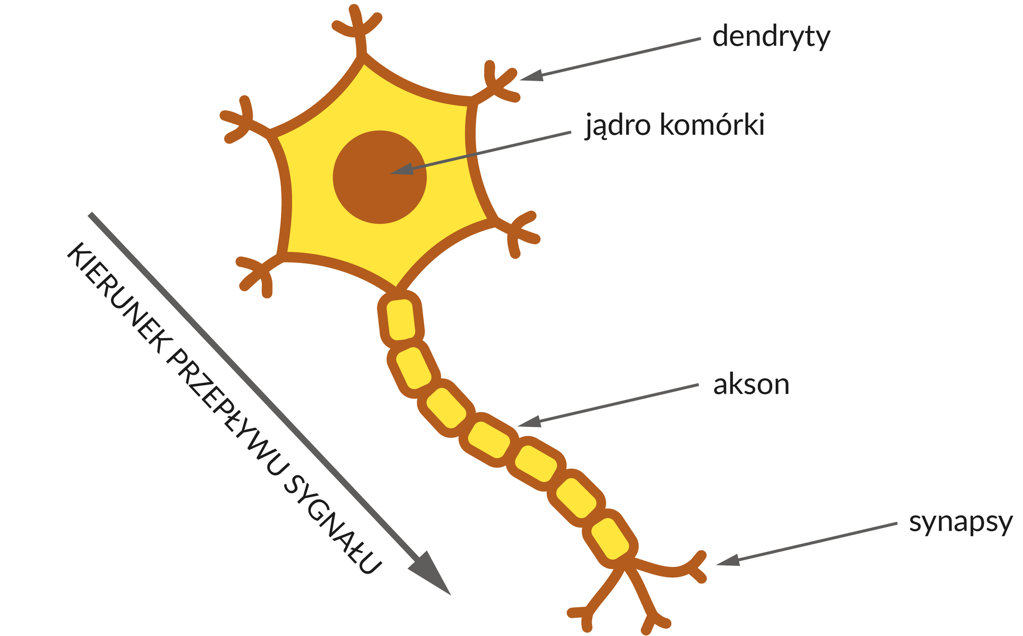
## Ogólny zarys sieci neuronowych

Ideą powstania sztucznych sieci neuronowych było utworzenie takiego algorytmu, który działałby na podobnych zasadach co biologiczny mózg. Początkowym zadaniem algorytmów SSN była predykcja zmiennych dychotomicznych, lecz wraz z rozwojem technologii oraz ulepszaniem algorytmów z danej dziedziny, sztuczne sieci neuronowe odnalazły szersze zastosowanie w wielu dziedzinach nauki.[[25]](#footnote-25)

### Neurony biologiczne

Neurony wykorzystane w budowie sztucznych sieci neuronowych przypominają neurony znajdujące się w mózgach zwierząt.[[26]](#footnote-26) Tak więc, aby zrozumieć funkcjonowanie algorytmów uczenia głębokiego, istotne jest przedstawienie budowy biologicznych neuronów.

Rysunek 2 Budowa neuronu



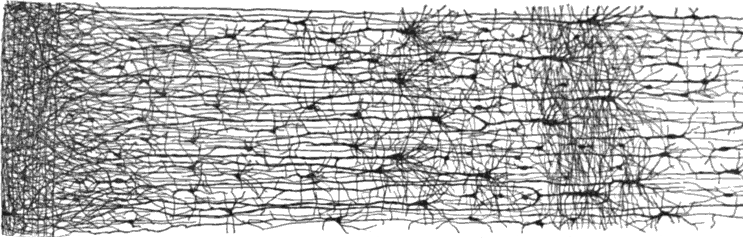
Źródło: https://zpe.gov.pl/a/przeczytaj/DqFenTtM4 (dostęp 05.04.2023)

Rysunek 2 przedstawia uproszczony wygląd neuronu z nazwami jego najistotniejszych części. Danymi częściami są:

* Dendryty- Stanowią one początek procesu zachodzącego w neuronie. Za pomocą dendrytów, neuron uzyskuje informacje przekazane przez poprzedzające go inne neurony.[[27]](#footnote-27)
* Jądro komórki- Jest to nadrzędna część, która jest kluczowa w procesie uczenia się neuronu. To właśnie w niej zachodzi proces integracji informacji otrzymanej za pomocą dendrytów oraz ich prawidłowej analizy. Wyjściowym efektem analiz w jądrze komórki jest informacja, która będzie przekazana do aksonu w przypadku, gdy przekroczy ona próg pobudzenia neuronu. W przeciwnym wypadku neuron nie jest aktywowany, co oznacza, iż nie bierze on udziału w procesie decyzyjnym.[[28]](#footnote-28)
* Akson- jest to wypustka, której celem jest przeniesienie informacji z jądra komórki do telodendronów. Długość aksona nie jest stała. Często jest on dłuższy od ciała komórki zaledwie kilkakrotnie, lecz zdarzają się neurony, których akson jest kilkaset razy dłuższy.[[29]](#footnote-29)
* Synapsy- Stanowią one zakończenie telodendronów. Ich główną funkcją jest łączenie się neuronu z dendrytami innych neuronów. Za ich pomocą dochodzi do przekazywania informacji pomiędzy neuronami.[[30]](#footnote-30)

Neuron jest tylko częścią całego procesu zachodzącego w procesie analizy informacji w zwierzęcym mózgu. Cały proces składa się z wielowarstwowej sieci neuronów, które są wspólnie od siebie zależne. Przykładowa sieć neuronowa przedstawiona jest na Rysunek 3.

Rysunek 3 Sieć neuronowa ludzkiego mózgu



Źródło: https://www.oreilly.com/library/view/neural-networks-and/9781492037354/ch01.html (dostęp 05.04.2023)

### Neuron

Jak już wcześniej zostało to przytoczone, neuron jest częścią całego procesu analizy danych w algorytmach sieci neuronowych. Przykładem neuronu, który jest omawiany w poprzednim rozdziale jest perceptron Rosenblatta oraz Wightmana. Budowa perceptronu przedstawiona jest na Rysunek 4.

Rysunek 4 Budowa perceptronu

Diagram

Description automatically generated

Źródło: *The Rise and Fall of the Perceptron* materiał zaczerpnięty ze strony: https://ai.plainenglish.io/the-rise-and-fall-of-the-perceptron-c04ae53ea465 (dostęp 05.04.2023)

Trudno jest się nie zgodzić ze stwierdzeniem, iż budowa perceptronu przypomina budowę biologicznego neuronu. Dane wejściowe *X*, które będą analizowane przez model perceptronu przechodzą przez dendryty. Dane wejściowe znajdują się w wektorze N-elementowym, co powoduję, iż w perceptronie pojawia się *N+1* dendrytów. Dodatkowy dendyt pojawia się ze względu na uwzględnienie wyrazu wolnego. W danych dendrytach znajdują się wagi poszczególnych wejść *w*.[[31]](#footnote-31) Następnie dane wejściowe oraz ich wagi przechodzą przez dwie funkcje, które znajdują się w jądrze neuronu. Do danych funkcji należy funkcja kombinacji oraz funkcja aktywacji[[32]](#footnote-32). Funkcja aktywacji szczegółowo opisana będzie w trzecim podrozdziale danego rozdziału. Funkcja kombinacji natomiast jest funkcją liniową. Tak więc wynikiem funkcji kombinacji będzie *Z*, gdzie:

[[33]](#footnote-33) (1)

Gdzie:*X* = [x1, x2, x3, …, xn, 1]; *w*= [w1, w2, w3, …, wn, wn+1]; *n* – ilość danych wejściowych w wektorze

Ostatnia wartość wektora *X*, równa 1, jest wyrazem wolnym modelu a jej przypisana waga to *wn+1*. Wynik funkcji kombinacji jest daną wejściową dla funkcji aktywacji *a(Z)*, gdzie[[34]](#footnote-34):

[[35]](#footnote-35) (2)

Wynik funkcji aktywacji jest przesyłany do warstwy wyjściowej sieci neuronowej bądź do neuronów kolejnej warstwy ukrytej.

### Uczenie głębokie

Perceptron jest siecią neuronową, która składa się z dwóch warstw. Pierwszą warstwą jest warstwa wejściowa, w której przekazywane są dane wejściowe. Ostatnia warstwa jest warstwą wyjściowa, w której wszystkie obliczenia sieci są tworzone.[[36]](#footnote-36) Wraz z rozwojem technologii oraz dziedziny sieci neuronowych, liczba algorytmów związanych z sieciami neuronowymi znacznie wzrosła. Aktualnie sieci neuronowe mogą posiadać znacznie więcej warstw w swojej architekturze. Dane warstwy, które znajdują się pomiędzy warstwą wejściowa oraz wyjściową nazywają się warstwami ukrytymi. W danych warstwach występują neurony, których celem jest zaciąganie informacji, wyłapywanie wzorców w danych i przekazywanie wyników do innych neuronów. Uczeniem głębokim można nazwać sieci neuronowe, które mają wiele warstw w swojej budowie[[37]](#footnote-37). Przykładowymi strukturami sieci neuronów, które są stosowane w dzisiejszych czasach są:

* MLP – Skrót MLP pochodzi od angielskiej nazwy Multi-Layer-Perceptron, co w polskim tłumaczeniu znaczy wielowarstwowy perceptron. Jak sama nazwa wskazuje, w danej sieci neuronowej możne znajdować się wiele warstw ukrytych. W każdej warstwie występuje określona liczba neuronów, które wykorzystują zarówno funkcje kombinacji, jak i funkcje aktywacji.[[38]](#footnote-38) W danej sieci neuronowej warstwy ukryte zaciągają sygnały z poprzedzających ich warstw, a swoje wnioski przesyłają do chronologicznie kolejnej warstwy ukrytej, bądź wyjściowej[[39]](#footnote-39). Tak więc MLP zaliczany jest do sieci jednokierunkowych. Dany typ sztucznych sieci neuronowych najczęściej wykorzystywany jest do klasyfikacji bądź regresji.[[40]](#footnote-40)
* RNN – jest to skrót od nazwy Recurrent Neural Networks. W polskiej nomenklaturze owe sieci neuronowe nazywane są rekurencyjnymi sieciami neuronowymi. Głównym celem danych sieci jest analiza szeregów czasowych. Unikatowość budowy rekurencyjnych sieci neuronowych, która występuje przez jej połączenia wsteczne, daje jej wysoką efektywność w analizowaniu szeregów czasowych. Za ich pomocą sieci te są w stanie brać pod uwagę zarówno dane wejściowe, jak i wartości, które nastąpiły w poprzednich okresach. Ze względu na to, RNN nie można zaklasyfikować do sieci jednokierunkowych. Rekurencyjne sieci neuronowe są wykorzystywane w: procesie analiz danych giełdowych, przetwarzaniu języka naturalnego jak i systemach samoczynnego prowadzenia samochodu.[[41]](#footnote-41)
* CNN – Splotowe sieci neuronowe, których angielska nazwa brzmi convolutional neural networks, są sieciami wykorzystywanymi do np. analizy obrazów. Owe sieci neuronowe używane są już od lat 80. Głównymi czynnikami, która dają wysoką efektywność w funkcjonowaniu CNN są warstwy splotowe oraz łączące. Głównym zadaniem owych warstw jest analiza oraz przekształcanie danych wejściowych a następnie zmniejszanie ich wymiarowości. Szersza analiza oraz interpretacja omawianych sieci neuronowych jest ukazana w Rozdział 2 Splotowe sieci neuronowe.[[42]](#footnote-42)

## Funkcja aktywacji

Funkcja aktywacji jest kluczową częścią efektywności sztucznych sieci neuronowych. To za jej pomocą są one w stanie rozwiązywać problemy oraz tworzyć predykcje, gdzie pomiędzy zmiennymi objaśniającymi a zmienną objaśnianą istnieje zależność nieliniowa. W sytuacji, gdyby sieć neuronowa nie posiadała jakiejkolwiek funkcji aktywacji w swoich neuronach, to wyniki danej sieci mogłyby przewidywać wyłącznie zależności liniowe.[[43]](#footnote-43) Łatwo można to potwierdzić za pomocą utworzenia sieci neuronowej, która posiada po trzy neurony w obu swoich warstwach ukrytych, gdzie nie występują wyrazy wolne, a funkcja aktywacji oddaje następujący wynik.

(3)

Dane wejściowe, które pochodzą z trzy liczbowego wektora, wraz z wagami przechodziłyby do pierwszej warstwy ukrytej. Wyjściem neuronów z danej warstwy ukrytej byłyby zatem funkcje kombinacji, które są funkcjami liniowymi. Kolejno, za pomocą funkcji kombinacji zsumowane iloczyny wag oraz danych wyjściowych z pierwszej warstwy ukrytej byłyby danymi wyjściowymi każdego neuronu z drugiej warstwy ukrytej. Zależność ta jest przedstawiona poniżej, gdzie przeanalizowane zostały dane wyjściowe z drugiej warstwy ukrytej.

Rysunek 5 Wyniki przykładowej sztucznej sieci neuronowej bez funkcji aktywacji

Text, letter

Description automatically generated

Źródło: opracowanie własne

Za pomocą wymnożenia macierzy można dojść do konkluzji, iż stosowanie wielu warstw w sztucznych sieciach neuronowych, gdzie funkcja aktywacji nie jest używana, jest bezowocne. Daną konkluzje potwierdza fakt, iż końcowym wynikiem takiego modelu będzie zawsze kombinacja liniowa danych wejściowych. Wraz ze zmianą ilości warstw w sztucznej sieci neuronowej, zmieniać się będą wyłącznie współczynniki przypisane do sygnałów wejściowych.

Funkcja aktywacji może przybrać wiele form. Poniżej przedstawione są przykładowe funkcje aktywacji, które są wykorzystywane w sztucznych sieciach neuronowych:

* Funkcja skokowa – Jest to najbardziej prymitywna forma funkcji aktywacji. Może ona być utworzona za pomocą prostej instrukcji warunku if-else w Pythonie. Jej głównym celem jest wysyłanie sygnału do neuronów w kolejnej warstwie w sytuacji, gdy wynik funkcji kombinacji przekroczy bądź jest równy określonej granicy. W przeciwnym wypadku, neuron nie zostaje aktywowany i nie wysyła żadnego sygnału. Dana funkcja najczęściej używana jest w klasyfikacji binarnej[[44]](#footnote-44). W przypadku funkcji progowej, zazwyczaj używa się funkcji skokowej Heavside’a, w której próg jest równy 0[[45]](#footnote-45). Przykład matematycznego zdefiniowania funkcji skokowej przedstawia wzór 4 a wizualizacja danej funkcji aktywacji ukazana jest na Wykres 1.

(4)

Wykres 1 Funkcja skokowa

Graphical user interface

Description automatically generated with medium confidence

Źródło: opracowanie własne

* Funkcja sigmoidalna – Jest to jedna z najczęściej stosowanych funkcji aktywacji w sztucznych sieciach neuronowych. Sigmoidalna funkcja jest funkcją nieliniową, tworzącą wyniki z przedziału od 0 do 1.[[46]](#footnote-46) Atutem danej funkcji jest fakt, iż jest ona różniczkowalna, co powoduje, iż można na niej użyć propagacji wstecznej, która jest procesem uczenia się sztucznych sieci neuronowych. Kształtem, dana funkcja przypomina literę „*S*”.[[47]](#footnote-47) Matematyczna definicja sigmoidalnej funkcji wynosi , gdzie *Z* to wynik funkcji kombinacji. Wizualizacja danej funkcji aktywacji ukazana jest na Wykresie 2.

Wykres 2 Funkcja sigmoidalna

Chart, line chart

Description automatically generated

Źródło: opracowanie własne

* Funkcja ReLU – Jest to funkcja, która jest powszechnie używana w sztucznych sieciach neuronowych. Jej spolszczona nazwa to rektyfikowana liniowa funkcja aktywacji. Przewagą funkcji ReLu, która jest funkcja aktywacji w warstwie ukrytej jest, iż nie wszystkie neurony w danej warstwie aktywują się w tym samym momencie[[48]](#footnote-48). Używanie danej funkcji aktywacji w sieciach neuronowych powoduje szybsze jej uczenie. Jest to spowodowane tym, iż budowa funkcji ReLu jest podobna do zwykłej liniowej funkcji aktywacji. Dla wartości wejściowych mniejszych od 0 funkcja relu daje wartość 0. W przeciwnym wypadku wyjściem danej funkcji jest jej wejście.[[49]](#footnote-49) Jej wizualizacja jest przedstawiona na Wykresie 3 a wzór ukazany jest we wzorze 5.

(5)

Wykres 3 funkcja ReLu

Chart, line chart

Description automatically generated

Źródło: opracowanie własne

## Propagacja wsteczna

Kluczową częścią sieci neuronowych jest proces ich uczenia. Jednym z najczęściej używanych metod uczenia jest propagacja wsteczna. W przypadku propagacji wstecznej istotne jest użycie prawidłowej funkcji kosztu. Funkcja kosztu określa jak bardzo oszacowania modelu różnią się od faktycznych wartości zmiennej celu. Dana funkcja jest podstawą w procesie optymalizacji parametrów modelu. W przypadku klasyfikacji, gdzie występują więcej niż dwie kategorie, najczęściej używana jest funkcja kosztu entropii krzyżowej.[[50]](#footnote-50) Poniższy wzór przedstawia jak obliczana jest dana funkcja kosztu.

[[51]](#footnote-51) (6)

Gdzie: K- liczba klas; tk – prawdziwa wartość dla danej klasy, wartość ta wynosi 1 bądź 0; Z- wynik funkcji kombinacji

Wyniki entropii krzyżowej są sumowane dla każdej obserwacji zbioru uczącego. Suma ta jest następnie wykorzystywana w procesie optymalizacji, gdzie głównym celem jest jej minimalizacja. Są dwa sposoby, aby minimalizować daną funkcję. Pierwszym sposobem jest zmiana wag połączeń pomiędzy neuronami. Drugim sposobem jest lepsze dopasowanie wyrazu wolnego.

Aby zoptymalizować dane parametry wykorzystuje się właśnie propagacji wstecznej, która bazuje na algorytmie gradientu prostego.[[52]](#footnote-52) Gradient przedstawia, w odniesieniu do wszystkich parametrów sieci neuronowej, kierunek, w którym funkcja błędu najbardziej się zwiększa. Wektor gradientu, składa się więc z pochodnych każdego badanego parametru funkcji kosztu.[[53]](#footnote-53) Tak więc, aby utworzyć wektor gradientu trzeba posłużyć się regułą łańcuchową.[[54]](#footnote-54) Żeby minimalizować funkcje błędu, odejmuje się od wektora wszystkich parametrów obliczony wektor gradientu. Niemniej jednak, aby propagacja wsteczna działała efektywnie stosuje się współczynnik uczenia *a*, który wymnaża wektor gradientu. Dany współczynnik nie może być za mały, ze względu na to, iż iteracyjne dążenie do minimum byłoby czasochłonne. Z drugiej strony, gdy współczynnik *a* jest za wysoki, to znalezienie minimum może okazać się niemożliwe. [[55]](#footnote-55)

Dla ukazania jak liczy się parametry w sieci neuronowej, za pomocą propagacji wstecznej, utworzona jest przykładowa sieć neuronowa, która składa się z *L* warstw, gdzie każda warstwa posiada jeden neuron. Koniec danej sieci neuronowej ukazany jest na Rysunek 6.

Rysunek 6 Sieć neuronowa

Diagram

Description automatically generated with medium confidence

Źródło: opracowanie własne

Obliczona pochodna wagi *w(L)* z funkcji błędu, która jest częścią wektora gradientu wykorzystanego w procesie optymalizacji parametrów ukazana jest następująco.

(7)

Gdzie: Z(L) – funkcja kombinacji neuronu warstwy wyjściowej, w(L)- waga przypisana do wyniku funkcji aktywacji neuronu z warstwy L-1, a(L) – funkcja aktywacji neuronu z warstwy L, E – funkcja błędu

Natomiast funkcja kombinacji neuronu z warstwy *L* ukazana jest poniżej.

(8)

**Podsumowanie:** Historia sztucznych sieci neuronowych rozpoczęła się w pierwszej połowie XX wieku. Od tamtego czasu dane algorytmy znacznie się rozwinęły i znalazły zastosowanie w wielu branżach. Sieci neuronowe składają się z neuronów, które analizują dane wejściowe za pomocą dwóch funkcji: kombinacji oraz aktywacji. Przykładami sieci neuronowych są splotowe sieci neuronowe, rekurencyjne sieci neuronowe oraz wielowarstwowy perceptron. Algorytmy z dziedziny uczenia głębokiego osiągają trafne wyniki predykcji ze względu na formę ich uczenia. Przykładem danej formy jest propagacja wsteczna, która bazuje na gradiencie prostym.

# Splotowe sieci neuronowe

W Rozdziale II została przedstawiona historia rozwoju dziedziny splotowych sieci neuronowych oraz ich codzienne zastosowanie w problemach biznesowych. Co więcej, w danym rozdziale opisane zostały budowy trzech najważniejszych warstw w splotowych sieciach neuronowych: warstwa splotowa, warstwa łącząca oraz warstwa w pełni połączona.

## Historia splotowych sieci neuronowych

Splotowe sieci neuronowe to najpopularniejszy algorytm związanym z kategoryzacją obrazów. Jego ideą jest odwzorowywanie kory wzrokowej, której celem jest analiza danych wizualnych. Istotnym czynnikiem w rozwoju splotowych sieci neuronowych był eksperyment przeprowadzony przez Hubela oraz Wiesela w 1962 roku. Dany eksperyment ukazał, że poszczególne neurony są aktywowane w sytuacji wykrycia bodźców wzrokowych w orientacji poziomej, bądź pionowej.[[56]](#footnote-56) Co więcej, udowodnili oni, iż poszczególne neurony wychwytują wyłącznie linie poziome, a inne neurony w tej samej sieci mogą wychwytywać pozostałe cechy obrazu. Wnioskiem danych badaczy był również fakt, iż neurony analizujące znacznie bardziej skomplikowane wzorce zazwyczaj otrzymywały dane od neuronów, których celem była interpretacja prostszych kształtów. Badania przeprowadzone przez Hubela oraz Wiesela doprowadziły do powstania pierwszego modelu splotowych sieci neuronowych, zwanego LeNet-5. Celem danego modelu, który został utworzony przez Yanna LeCuna, była analiza oraz odczytywanie cyfr wpisanych ręcznie na papierach wartościowych.[[57]](#footnote-57) Ze względu na niskie możliwości mocy obliczeniowej komputerów w latach 1990-2000, splotowe sieci neuronowe nie były rozwijane oraz implementowane w procesie kategoryzacji obrazów.

Rok 2012 przyniósł ze sobą nowy model splotowych sieci neuronowych o nazwie AlexNet.[[58]](#footnote-58) Dany model, który wyróżnił się wysoką efektywnością w procesie kategoryzacji obrazów, zainicjował znaczny rozwój dziedziny splotowych sieci neuronowych[[59]](#footnote-59). Model osiągnął sukces ze względu na jego architekturę. Jest to jeden z pierwszych modeli, których celem była kategoryzacja obrazów, gdzie zastosowano funkcję aktywacji relu, zamiast popularnych w tych czasach funkcji sigmoidalnych. W 2014 powstał algorytm VGG Net, który osiągał lepsze rezultaty od splotowych sieci neuronowych AlexNet. W danym algorytmie założono, iż pole recepcyjne neuronu w warstwie splotowej nie powinno mieć wymiarów większych od 3x3. Ze względu na to, dany algorytm nie tylko otrzymał lepsze wyniki, ale również był w stanie je otrzymać w krótszym czasie.[[60]](#footnote-60)

Tego samego roku powstał algorytm Inception, który wygrał konkurs klasyfikacji obrazów ILSVRC. Dany algorytm jest również nazywany GoogleNet. Ideą wyróżniającą dany model od innych modeli splotowych sieci neuronowych w danym okresie był fakt, iż stosował on wiele filtrów o różnych wymiarach w tym samym bloku warstw. W przykładowym bloku znajdywały się warstwy splotowe z filtrami 3x3, 5x5 oraz 1x1 oraz warstwa łącząca. Aby zminimalizować ilość parametrów w modelu zastosowano warstwy łączące z filtrem 1x1 przed każda warstwą splotową z filtrem większym od wymiarów 1x1 oraz każdą warstwą łączącą.[[61]](#footnote-61)

W roku 2015 zbudowano model ResNet, który wygrał konkurs ILSVRC. Celem twórców danego algorytmu było utworzenie takiej architektury modelu splotowych sieci neuronowych, aby przy zwiększaniu się jej ilości warstw jej efektywność nie malała. Narzędziami do osiągnięcia założeń zostały tzw. Połączenia pomijające. Przykładowo za ich pomocą wejściem warstwy splotowej *L* może być wyjście warstwy poprzedniej *L-1*, jak i warstwy *L-2*. Za pomocą danej budowy model ten nie jest narażony na problem wystąpienia zanikania gradientu[[62]](#footnote-62). Wyniki każdego przedstawionego modelu zostały ujęte w Tabeli 1.

Tabela 1 Wyniki przykładowych splotowych sieci neuronowych.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Cecha wyróżniająca | Top5 Accuracy | Ilość parametrów (miliony) | FLOP (miliardy) |
| AlexNet | Warstwa łącząca, funkcja aktywacji relu | 84.70% | 62 | 1.5 |
| VGGNet | Stałe wymiary filtrów | 92.30% | 138 | 19.6 |
| Inception | Unikatowa budowa bloków warstwowych | 93.30% | 6.4 | 2 |
| ResNet-152 | Pomijane połączenia | 95.51% | 60.3 | 11 |

Źródło: *Difference between AlexNet, VGGNet, ResNet, and Inception* materiał zaczerpnięty ze strony https://towardsdatascience.com/the-w3h-of-alexnet-vggnet-resnet-and-inception-7baaaecccc96 (dostęp 01.05.2023)

Z owej tabeli wynika, iż model ResNet osiągnął najwyższą dokładność w rozwiązaniu problemu klasyfikacji, z wynikiem 95.51%. Niemniej jednak, analizując kolumnę FLOP, można wywnioskować, iż ResNet potrzebuje relatywnie dużo czasu na jego wytrenowanie oraz użytek. Wyłącznie model VGGNet, którego wyniki predykcji są lepsze tylko od modelu AlexNet, posiada wyższy wskaźnik FLOP. Model Inception posiada najmniejszą ilość parametrów w swojej architekturze. Co więcej, czas, który jest potrzebny na jego wytrenowanie jest najkrótszy ze wszystkich analizowanych modeli. Dany model posiada drugi najlepszy wynik, jeśli chodzi o dokładność predykcji.

## Wartość biznesowa w poszczególnych branżach

Aktualnie można znaleźć wiele zastosowań splotowych sieci neuronowych w różnych branżach. W danym podrozdziale przedstawione zostały przykładowe cztery branże, które wykorzystują dane algorytmy.

* Medycyna – Jednym z głównych celów branży medycznej jest poprawna analiza wyników pacjenta oraz postawienie prawidłowej diagnozy. Już teraz używane są algorytmy konwolucyjnych sieci neuronowych do poprawnej klasyfikacji, czy na przykładowym zdjęciu pojawia się niepożądana cecha. Owe modele wykorzystywane są do diagnozowania np. gruźlicy, retinopatii cukrzycowej bądź raka skóry. Niemniej jednak uzyskanie dużej liczby oznaczonych danych jest trudne do osiągnięcia, więc w danej branży często stosuje się metodę data augmentation, której celem jest zwiększenie ilości danych za pomocą już zebranych zdjęć[[63]](#footnote-63).
* Transport – Zadaniem splotowych sieci neuronowych w branży transportowej jest przede wszystkim detekcja oraz klasyfikacja obiektów znajdujących się na jezdniach. Dane algorytmy tworzy się, aby monitorować przepływ ruchu drogowego na poszczególnych trasach. Za pomocą wyników uzyskanych przez owe algorytmy, osoby upoważnione są w stanie dostosowywać system oświetlenia drogowego, zarządzać zwiększaniem, bądź zmniejszaniem ilości pasów na drodze, jak również kontrolować bezpieczeństwo[[64]](#footnote-64).
* Portale społecznościowe – algorytmy CNN odnajdują zastosowanie w portalach społecznościowych, gdzie ich głównym celem jest klasyfikacja twarzy na zdjęciu. Za ich pomocą konsumenci są w stanie oznaczyć swojego znajomego w znacznie szybszym czasie[[65]](#footnote-65).
* Technologia – algorytmy konwolucyjnych sieci neuronowych są również wykorzystywane do znajdowania oraz odczytywania tekstu, który aktualnie znajduję się w polu recepcyjnym kamery. Za ich pomocą użytkownik jest w stanie w bardzo szybkim tempie skopiować cały tekst, który znajduje się na zdjęciu, dokumencie bądź jakimkolwiek innym przedmiocie zawierającym informację w formie pisemnej[[66]](#footnote-66).

## Architektura splotowych sieci neuronowych

W przypadku analizowania obrazów istotne jest uświadomienie sobie jak dużo informacji niesie za sobą jeden obraz. W przypadku gdy przykładowe zdjęcie ma rozmiar 80x80 i składa się wyłącznie z odcieni szarości to warstwa wejściowa sieci neuronowej analizującej obrazy składać się będzie z 6400 neuronów. Warto podkreślić, iż splotowe sieci neuronowe są sieciami jednokierunkowymi, co oznacza, iż informacje, które są analizowane przez dane algorytmy przechodzą chronologicznie przez wszystkie warstwy[[67]](#footnote-67). Ze względu na to, w przypadku, gdyby pierwsza warstwa ukryta splotowej sieci neuronowej miała 1500 neuronów, gdzie każdy neuron jest połączony z każdym neuronem poprzedniej warstwy sieci, to łącznie można byłoby odnotować 9 600 000 połączeń już w pierwszej warstwie ukrytej. Aby rozwiązać problem ogromnej ilości parametrów modelu, stosuje się warstwy częściowo połączone. Warstwy częściowo połączone posiadają połączenia z określoną ilością neuronów poprzedniej warstwy neuronowej, które znajdują się w ich polu recepcyjnym. Dana forma architektury sieci neuronowej powoduje, iż pierwsze warstwy sieci skupiają się na poszczególnych, ogólniejszych wzorach obrazu, a neurony znajdujące się w kolejnych warstwach analizują bardziej skomplikowane i złożone cechy obrazu[[68]](#footnote-68). Przykładowa sieć neuronowa, która posiada warstwy częściowo połączone znajduje się w Rysunek 7.

Rysunek 7 Sieć neuronowa z warstwami częściowo połączonymi

Bubble chart

Description automatically generated

Źródło: opracowanie własne

Splotowe sieci neuronowe składają się z warstw splotowych, łączących oraz z warstw na końcu sieci, które są w pełni połączone. Kolejne podrozdziały danego rozdziału dedykowane są interpretacji wszystkich przedstawionych warstw. Rysunek 8 natomiast przedstawia przykładową budowę splotowej sieci neuronowej wykorzystanej do kategoryzacji obrazów.

Rysunek 8 Przykład budowy splotowej sieci neuronowej

Diagram, engineering drawing

Description automatically generated

Źródło: A comprehensive Guide to Convolutional Neural Networks – the ELI5 way materiał zaczerpnięty ze strony: https://saturncloud.io/blog/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way/ (dostęp 16.04.2023)

## Warstwa splotowa

Dane wejściowe, które przedstawiają obraz oraz analizowane są w splotowych sieciach neuronowych, są ukazane w trzech wymiarach w przypadku, gdy zdjęcie jest kolorowe. Pierwszy wymiar przedstawia szerokość zdjęcia, drugi wymiar ujawnia jego wysokość a trzeci wymiar ukazuje nasilenie koloru czerwonego, zielonego oraz niebieskiego. W przypadku analizowania obrazu z wyłącznie odcieniami szarości, dane wejściowe w sieci neuronowej posiadają dwa pierwsze wymiary. Intensywność barwy ukazana jest za pomocą liczby od 0 do 256. Piksele posiadające wartość 0 są całkowicie czarne, natomiast wzrost wartości powoduję rozjaśnianie się określonej części obrazu.[[69]](#footnote-69)

Warstwa splotowa, która po angielsku jest nazywana convolutional layer, jest kluczowa w funkcjonowaniu splotowych sieci neuronowych. Jak już zostało to wspomniane w poprzednim podrozdziale, neurony w warstwie splotowej połączone są z wybranymi neuronami poprzedniej warstwy. Obszar neuronów, z których neurony warstwy splotowej pobierają informację nazywany jest polem recepcyjnym. Tak więc, neuron w warstwie splotowej, który znajduje się w wierszu *w* oraz w kolumnie *k*pobiera informację z neuronów znajdujących się w wierszach od *w* do *w-1+fw* i kolumnach od *k* do *k-1+fk* poprzedniej warstwy. W przedstawionych notacjach matematycznych założono, iż pierwszy wiersz oraz pierwsza kolumna macierzy mają przypisane indeksy o wartości 0. Parametry *fw* oraz *fk* określają wysokość oraz szerokość pola widzenia neuronu[[70]](#footnote-70). Rysunek 9 ukazuje jakie neurony z warstwy *L-1* są analizowane przez neuron z warstwy splotowej *L* z pierwszego wiersza oraz pierwszej kolumny, gdy parametry *fw* oraz *fk* osiągają wartość równą 3.

Rysunek 9 Relacja pomiędzy warstwami splotowymi

Line chart

Description automatically generated

Źródło: opracowanie własne

Z Rysunek 9 można wywnioskować, iż w sytuacji, gdy *fw* oraz *fk* osiągają wartości większe od 1 to wynikiem warstwy splotowej będzie obraz o mniejszym wymiarze. W sytuacji jednak, gdy pożądane jest utrzymanie wysokiego wymiaru wyjścia warstwy splotowej to używa się sposobu uzupełniania zerami. Za pomocą danego procesu do danych wejściowych dokłada się zerowe wartości przy krawędziach obrazu.[[71]](#footnote-71)

W sytuacji jednak, gdy obraz, który jest analizowany przez warstwę splotową, osiąga wysoką rozdzielczość to można zastosować metodę krokową, której celem jest minimalizowanie zapotrzebowania na moc obliczeniową. Przy zastosowaniu danej metody, efektem wyjściowym przykładowej warstwy splotowej będą obrazy ze znacznie mniejszą rozdzielczością niż dane wejściowe. Dana metoda zmienia pole recepcyjne neuronów, gdzie w przypadku, gdy *w* oraz *k* określają położenie neuronu w warstwie splotowej, to danymi wejściowymi danego neuronu będą neurony z poprzedniej warstwy usytuowane w wierszach od *w\*sw* do *w\*sw+fw-1* oraz kolumnach od *k\*sk* do *k\*sk+fk-1*, gdzie parametry *sw* oraz *sk* ukazują krok w wierszach oraz kolumnach.[[72]](#footnote-72)

Aby określić wymiary wyjścia splotowej warstwy, używa się danej formuły matematycznej: , gdzie *Z* określa wymiary danych wejściowych, *R* przedstawia wymiary pola recepcyjnego, *U* ukazuje ilość warstw obrazu utworzonych za pomocą metody uzupełniania zerami a *S* określa wielkość kroku. Wyniki danej formuły, które nie są liczbami naturalnymi oznaczają, iż metoda kroku została użyta w sposób nieprawidłowy.[[73]](#footnote-73) W sytuacji, gdy pożądane jest zatrzymanie wielkości kroku a wymiary wyjścia splotowej warstwy nie pochodzą z liczb całkowitych to można zastosować funkcje uzupełniania zerami w taki sposób, aby liczby określające wymiary nie były wartościami ułamkowymi.

### Filtry

Każdy neuron warstwy splotowej posiada swoje pole recepcyjne. Dane, które znajdują się w danym polu połączone są z przykładowym neuronem za pomocą wag. Przykładowo patrząc na Rysunek 9 można zauważyć, iż do neuronu warstwy splotowej przypisane będą 9 neuronów warstwy poprzedniej w wymiarach 3x3x1. Tak więc, wagi pomiędzy warstwą poprzednią a warstwą splotową, również mogą być przedstawione w wymiarach 3x3x1. W zależności od określonego celu wyłapywania wzorców z obrazu stosuje się różny schemat rozłożenia wag. Zbiór wag określany jest jako filtr.[[74]](#footnote-74) Jedna warstwa splotowa może posiadać wiele filtrów, których celem jest uzyskanie istotnych informacji w procesie analizowania obrazu. Dla każdego filtru tworzona będzie nowa warstwa neuronów w pojedynczej warstwie splotowej. Wynikiem warstwy, która stosuję określony oraz jednakowy filtr, jest mapa cech[[75]](#footnote-75). Budowa warstwy splotowej powinna powodować, iż jest ona w stanie wychwycić określone wzorce na obrazie, niezależnie, gdzie się one znajdują. Aby dany cel osiągnąć wykorzystuje się współdzielenie wag. Co więcej, współdzielenie wag znacznie zmniejsza ilość parametrów, które są optymalizowane w procesie uczenia się sieci neuronowej, oraz zmniejsza zapotrzebowanie na moc obliczeniową.[[76]](#footnote-76)

Wygląd przykładowego filtra (tabela zielona), który jest wykorzystany w warstwie splotowej przy analizowaniu danych z warstwy poprzedniej (tabela czerwono-czarna) widoczna jest na Rysunek 10.

Rysunek 10 Przykładowy filtr w splotowej sieci neuronowej

Table

Description automatically generated

Źródło: opracowanie własne

Wyjściem neuronu w warstwie splotowej *L* jest suma wymnożonych wag oraz wartości z poprzedniej warstwy[[77]](#footnote-77). Analizując przypadek z Rysunku 10, neuron znajdujący się w wierszu oraz kolumnie 0 będzie posiadał wynik funkcji kombinacji równy Z= (1\*1) + (-1\*0) + (1\*4) + (-1\*1) + (1\*-2) + (-1\*2) + (1\*3) + (-1\*4) + (1\*3), natomiast wynik funkcji aktywacji relu będzie równy 1. Oczywiście jest to prosty obliczeniowo przypadek, w którym warstwa *L-1* posiada wyłącznie 2 wymiary. W przypadku 3 wymiarowych warstw poprzedzających warstwę splotową, obliczenie funkcji kombinacji będzie znacznie bardziej skomplikowane. Wzór na wyjście neuronu znajdującego się w splotowej warstwie znajduje się poniżej[[78]](#footnote-78):

(9)

Gdzie: - wyjście neuronu znajdującego się w *w* wierszu, *k* kolumnie oraz *ft* mapie cech;

– wymiary pola recepcyjnego określonego filtra, - liczba filtrów, bądź kanałów, występujących w poprzedniej warstwie; (*u,v*) – index wiersza oraz kolumny neuronu z pola recepcyjnego filtra; (*w,k*) – index wiersza oraz kolumny analizowanego neuronu z mapy cech *ft*; - wyjście neuronu z poprzedniej warstwy z wiersza w’=w\*sw+u, kolumny k’=k\*sk+ v oraz mapy cech, bądź kanału, *ft’*; - waga połączenia pomiędzy neuronem z mapy cech *ft* a neuronem poprzedniej warstwy z mapy cech, bądź kanału, *ft’* znajdującego się w wierszu *u* oraz kolumnie *v* pola recepcyjnego; - parametr obciążenia filtra z mapy cech *ft*.

Rysunek 11 Wyniki filtrów warstwy splotowej

Diagram, engineering drawing

Description automatically generated

Źródło: opracowanie własne

Rysunek 11 przedstawia jak warstwa splotowa analizuje przykładowy obraz w zależności od filtru. W danym przykładzie użyto filtru poziomego, którego wyniki są przedstawione na zdjęciu po lewej stronie, oraz filtru pionowego, którego wyniki są ukazane na obrazie znajdującym się po prawej strony rysunku. Na samej górze natomiast znajduje się oryginalne zdjęcie samochodu.

Patrząc na wynik filtru poziomego można stwierdzić, iż wzory zdjęcia których kształtem jest linia pozioma, są bardziej widoczne od pozostałych obiektów. Podobny wniosek można wyciągnąć analizując obraz utworzony za pomocą filtra pionowego, gdzie elementy obrazu charakteryzujące się pionowym kształtem, są bardziej zauważalne. Filtrem poziomym można nazwać filtr, który w przypadku, gdy przykładowo jest macierzą 3x3 to w drugim wierszu ma wartości dodatnie a w pozostałych wierszach występują wartości ujemne bądź zerowe. Filtrem pionowym natomiast może być filtr o wymiarach 3x3, gdzie druga kolumna posiada wartości dodatnie, a pozostałe ujemne bądź zerowe.

## Warstwa łącząca

Warstwa łącząca, po angielsku zwana pooling layer, jest istotną częścią splotowych sieci neuronowych. Jest ona stosowana po utworzeniu wyników przez splotową sieć neuronową. Celami stosowania danej warstwy jest minimalizowanie czasu spędzonego na trenowaniu/stosowaniu danego modelu oraz zmniejszanie ryzyka wystąpienia przeszacowanego modelu predykcyjnego. Warstwy łączące zmniejszają ilość wierszy oraz kolumn przy każdej warstwie filtra w warstwie splotowej, niemniej jednak nie niwelują one ilości map cech.[[79]](#footnote-79) Metoda ta, mimo iż zmniejsza wymiary danych, które są analizowane w procesie użytku splotowych sieci neuronowych, to nie osłabia ich użyteczności a jedynie usuwa nieistotne informacje.[[80]](#footnote-80)

Warstwa łącząca posiada, tak samo jak warstwa splotowa, swoje pole recepcyjne. Co więcej, warstwa łącząca również stosuje metodę krokową oraz metodę uzupełniania zerami. Tak więc, wielkość wymiaru danych wyjściowych warstwy łączącej jest w głównej mierze zależna od wielkości pola recepcyjnego, wielkości kroku oraz ilości uzupełnienia zerami.[[81]](#footnote-81) Do powszechnie stosowanych metod warstw łączących możemy zaklasyfikować: average pooling, max pooling oraz mixed pooling.[[82]](#footnote-82)

* Average pooling –Sposób ten minimalizuje wymiarowość wyników splotowej sieci neuronowej poprzez obliczanie średniej wartości z pola recepcyjnego warstwy łączącej.[[83]](#footnote-83) Przykład wykorzystania metody average pooling, gdzie wymiary pola recepcyjnego są równe 2x2 a wielkość kroku wynosi 2, przedstawiony jest na Rysunku 13.

Rysunek 12 Average pooling

A picture containing table

Description automatically generated

Źródło: opracowanie własne

* Max pooling – Jest to warstwa łącząca, której wyjściem przykładowego neuronu jest maksymalna, zaobserwowana wartość z jego pola recepcyjnego. Dana metoda w przeciwieństwie do average pooling, ukazuje najbardziej wyróżniającą się cechę obrazu.[[84]](#footnote-84) Przykład wykorzystania metody max pooling można zauważyć na Rysunku 13.

Rysunek 13 Max pooling

A picture containing line chart

Description automatically generated

Źródło: opracowanie własne

* Mixed pooling – Dana forma warstwy łączącej jest połączeniem metod max pooling oraz average pooling. Kluczowym parametrem w stosowaniu danej metody jest λ. Lambda jest losowana i może przyjąć wartość 0 bądź 1. W sytuacji, gdy lambda jest równa 0 to warstwa łącząca używa metody average pooling. W przeciwnym wypadku stosowana jest metoda max pooling[[85]](#footnote-85).

Rysunek 14 przedstawia, jak wyglądają wyniki zastosowania max poolingu oraz average poolingu na zdjęciu samochodu. W przypadku obu warstw łączących zastosowano pole recepcyjne o wymiarach 10x10, wielkość kroku równą 2 oraz wypełnianie zerami.

Rysunek 14 Wyniki max pooling oraz average pooling ze zdjęcia samochodu

A picture containing text, items, different

Description automatically generated

Źródło: opracowanie własne

## Warstwa w pełni połączona

Ostatnie warstwy, które znajdują się w budowie splotowych sieci neuronowych różnią się od warstw splotowych oraz łączących. Główna różnica polega na pełnym połączeniu neuronów danych warstw z warstwą poprzednią. Ów segment splotowych sieci neuronowych może posiadać jedną bądź wiele warstw w pełni połączonych. Ostatnia warstwa splotowa bądź łącząca, przekształca wymiary swojego wyjścia w jednowymiarowy wektor, aby połączyć swoje wyniki z warstwą w pełni połączoną. W warstwach tych zachodzi proces przekształcania zebranych informacji w taki sposób, aby można było utworzyć wynik stosowny do narzuconego zadania. Tak więc, w przypadku klasyfikacji wieloklasowych, ostatnia warstwa posiada tyle neuronów, ile zmienna celu posiada klas. Co więcej, warstwa ta musi posiadać taką funkcje aktywacji, aby spełniać określone zadanie. Funkcja softmax jest najczęściej stosowana w zadaniach klasyfikacji, gdzie zmienna objaśniana posiada więcej niż dwie kategorie.[[86]](#footnote-86)

(11)[[87]](#footnote-87)

Gdzie: Z(n) – wynik funkcji kombinacji w n neuronie, N – liczba neuronów w danej warstwie

**Podsumowanie:** Splotowe sieci neuronowe są powszechnie używane do kategoryzacji obrazów. Ich efektywność jest spowodowana przez warstwy splotowe, łączące oraz w pełni połączone. Celem warstw splotowych jest wyłapanie istotnych wzorów z obrazu za pomocą filtrów. Warstwy łączące natomiast zmniejszają wymiarowość wyników warstw splotowy, czego skutkiem jest mniejsze zapotrzebowanie na moc obliczeniową i minimalizacja ryzyka przeuczenia. Ostatnie warstwy splotowej sieci neuronowej są ze sobą w pełni połączone. W danych warstwach zachodzi transformacja wyników sieci, tak aby były one interpretowalne. Dzięki określonym cechom można stwierdzić, iż splotowe sieci neuronowe mogą zostać wykorzystane jako metoda badawcza do klasyfikacji zdjęć samochodów.

# Zbiór danych wykorzystany do budowy modelu

1. https://vitalflux.com/deep-neural-network-examples-from-real-life/ [dostęp 26.03.2023] [↑](#footnote-ref-1)
2. Aurelien Geron, Uczneie maszynowe z użyciem Scikit-lEarn i TensorFlow, Helion, 2020, s. 286. [↑](#footnote-ref-2)
3. Ibidem. [↑](#footnote-ref-3)
4. https://www.investopedia.com/terms/m/mooreslaw.asp [dostęp 26.03.2023] [↑](#footnote-ref-4)
5. Aurelien Geron, op. cit, s.287. [↑](#footnote-ref-5)
6. https://towardsdatascience.com/6-types-of-neural-networks-every-data-scientist-must-know-9c0d920e7fce [dostęp 26.03.2023] [↑](#footnote-ref-6)
7. Francois Chollet, *Deep learning with Python*, Manning, Shelter Island, 2021, s.41 [↑](#footnote-ref-7)
8. Alexey Grigorev, Machine Learning Bookcamp, Manning, Shelter Island, 2021, s.40 [↑](#footnote-ref-8)
9. Gabriel Baker, *Machine Learning For Begginers – The Comprehensive Guide to Artificial Intelligence and Data Science for Business,* Gabriel Baker, 2020, s10. [↑](#footnote-ref-9)
10. Yuxi (Hayden) Liu, *Python Machine Learning By Example*, Packt Publishing, 2017, s.41. [↑](#footnote-ref-10)
11. Ibidem. [↑](#footnote-ref-11)
12. Ibidem. [↑](#footnote-ref-12)
13. Francois Gollet, op.cit, a.50. [↑](#footnote-ref-13)
14. Aurelien Geron, op. cit, s.286. [↑](#footnote-ref-14)
15. *McCulloch Pitt’s Model of Neuron* materiał zaczerpnięty ze strony internetowej: https://www.codingninjas.com/codestudio/library/mcculloch-pitt-s-model-of-neuron (dostęp 03.04.2023) [↑](#footnote-ref-15)
16. *Hebbian Learning* materiał zaczerpnięty ze strony internetowej: https://thedecisionlab.com/reference-guide/neuroscience/hebbian-learning (dostęp 03.04.2023) [↑](#footnote-ref-16)
17. Steven Cooper, Neural Networks: A Practical Guide for Understanding and Programming Neural Networks and programming neural networks and useful insight for inspiring reinvention, Data Science, 2019, s.15-16. [↑](#footnote-ref-17)
18. *Professor’s perceptron paved the way for AI – 60 years too soon* materiał zaczerpnięty ze strony internetowej: https://news.cornell.edu/stories/2019/09/professors-perceptron-paved-way-ai-60-years-too-soon (dostęp 04.04.2023) [↑](#footnote-ref-18)
19. *Rosenblatt’s perceptron, the first modern neural* network materiał zaczerpnięty ze strony internetowej: https://towardsdatascience.com/rosenblatts-perceptron-the-very-first-neural-network-37a3ec09038a (dostęp 04.04.2023) [↑](#footnote-ref-19)
20. Sebastian Reschka, *Python Machine Learning*, Packt Publishing, 2015, s. 77-100 [↑](#footnote-ref-20)
21. Steven Cooper, op.cit, s.15-17. [↑](#footnote-ref-21)
22. Steven Cooper, op.cit, s.15-17. [↑](#footnote-ref-22)
23. Aurelien Geron, op. cit, s.287. [↑](#footnote-ref-23)
24. Ibidem. [↑](#footnote-ref-24)
25. Aurelien Geron, op. cit, s.285-289. [↑](#footnote-ref-25)
26. Ibidem [↑](#footnote-ref-26)
27. Nicoladie Tam, Neuron A Tutorial Study Guide, Nicoladie Tam, 2015, s.11-20 [↑](#footnote-ref-27)
28. Ibidem. [↑](#footnote-ref-28)
29. Aurelien Geron, op. cit, s.287. [↑](#footnote-ref-29)
30. Aurelien Geron, op. cit, s.287. [↑](#footnote-ref-30)
31. Francois Chollet, op.cit, s.78-86. [↑](#footnote-ref-31)
32. Modelowanie predykcyjne – sztuczne sieci neuronowe materiał zaczerpnięty ze strony internetowej: https://www.studocu.com/pl/document/szkola-glowna-handlowa-w-warszawie/data-mining/zajecia-8-aaadasf-fasf-asfdasgea-fawfdzg-dfhewehjrt-rksjfonseklfma/25070744 (dostęp 05.04.2023) [↑](#footnote-ref-32)
33. Francois Chollet, op.cit, s.78-86. [↑](#footnote-ref-33)
34. Francois Chollet, op.cit, s.78-86. [↑](#footnote-ref-34)
35. Ibidem. [↑](#footnote-ref-35)
36. Sebastian Reschka, op.cit, s.85-86. [↑](#footnote-ref-36)
37. Gabriel Baker, op.cit, s.12-13. [↑](#footnote-ref-37)
38. Sebastian Raschka, op.cit, s.626-628. [↑](#footnote-ref-38)
39. Stanisław Osowsji & Krzysztof Siwek & Tomasz Markiewicz, *MLP and SVM Networks – a Comparative Study*, Proceedings of the 6th Nordic Signal Processing Symposium – NORSIG, Espoo, 2004, s.37. [↑](#footnote-ref-39)
40. When to Use MLP, CNN and RNN Neural Networks materiał zaczerpnięty ze strony: https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/ (dostęp: 06.04.2023) [↑](#footnote-ref-40)
41. Aurelien Geron, op.cit, s.487-488. [↑](#footnote-ref-41)
42. Auerlien Geron, op.cit, s. 439-452. [↑](#footnote-ref-42)
43. Siddhart Sharma & Simone Sharma, *Activation Functions in Neural Networks*, International Journal of Engineering Applied Sciences and Technology, 2020, s.310 [↑](#footnote-ref-43)
44. Siddhart Sharma & Simone Sharma, op.cit, s.312. [↑](#footnote-ref-44)
45. Aurelian Geron, op.cit, s.290 [↑](#footnote-ref-45)
46. Siddhart Sharma & Simone Sharma, op.cit, s.312. [↑](#footnote-ref-46)
47. Tomasz Szandała, Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks, Wrocław University of Science and Technology, 2020, s.5. [↑](#footnote-ref-47)
48. Siddhart Sharma & Simone Sharma, op.cit, s.313. [↑](#footnote-ref-48)
49. Tomasz Szandała, op.cit, s.9-10. [↑](#footnote-ref-49)
50. Frank Millstein, *Convolutional Neural Networks in Python: Beginner’s Guide to Convolutional Neural Networks in Python*, Frank Millstein, 2019, s.60. [↑](#footnote-ref-50)
51. Cross-Entropy Loss Function materiał zaczerpnięty ze strony: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e (dostęp 10.04.2023) [↑](#footnote-ref-51)
52. Aurelien Geron, op.cit, s.294-295. [↑](#footnote-ref-52)
53. *Gradient*, materiał zaczerpnięty ze strony: https://www.medianauka.pl/gradient (dostęp 10.04.2023). [↑](#footnote-ref-53)
54. Aurelien Geron, op.cit, s.294-295. [↑](#footnote-ref-54)
55. Aurelien Geron, op.cit, s.135-136. [↑](#footnote-ref-55)
56. Frank Millstein, op.cit, s.8. [↑](#footnote-ref-56)
57. Aurelien Geron, op.cit, 440. [↑](#footnote-ref-57)
58. Dulari Bhatt, Chirag Patel, Hardik Talsania, Jigar Patel, Rasmika Vaghela, Sharnil Pandya, Kirit Mody, Hemant Ghayvat, *CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope*,Electronics 2021, MDPI, s.2. [↑](#footnote-ref-58)
59. *Deep Learning – AlexNet*, materiał zaczerpnięty ze strony: https://www.doc.ic.ac.uk/~bkainz/teaching/DL/notes/AlexNet.pdf (dostęp 15.04.2023) [↑](#footnote-ref-59)
60. Andrey Arsenov, Igo Ruban, Kyrylo Smelyakov, Anastasiya Chupryna, Evolution of Convolutinoal Neural Network Architecture in Image Classification Problems, International Conference on Intelligent Tutoring Systems, 2018, s.35-44. [↑](#footnote-ref-60)
61. Ibidem. [↑](#footnote-ref-61)
62. Ibidem. [↑](#footnote-ref-62)
63. Deep Learning in Medical Imaging materiał zaczerpnięty ze strony: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6945006/ (dostęp 01.05.2023) [↑](#footnote-ref-63)
64. Faster CNN-based vehicle detection and counting strategy for fixed camera scenes materiał zaczerpnięty ze strony: https://link.springer.com/article/10.1007/s11042-022-12370-9 (dostęp 01.05.2023) [↑](#footnote-ref-64)
65. Applications of Convolutional Neural Networks (SCNN) materiał zaczerpnięty ze strony: https://www.analyticsvidhya.com/blog/2021/10/applications-of-convolutional-neural-networkscnn/ (dostęp 01.05.2023) [↑](#footnote-ref-65)
66. Convolutional Neural Network: Benefits, Types, and Applications materiał zaczerpnięty ze strony: https://datagen.tech/guides/computer-vision/cnn-convolutional-neural-network/ (dostęp 01.05.2023) [↑](#footnote-ref-66)
67. Dulari Bhatt, Chirag Patel, Hardik Talsania, Jigar Patel, Rasmika Vaghela, Sharnil Pandya, Kirit Mody, Hemant Ghayvat,op.cit, s.2. [↑](#footnote-ref-67)
68. Aurelien Geron, op.cit, s.441. [↑](#footnote-ref-68)
69. Convolutional Neural Networks (CNN) and Computer Vision 101 fundamentals Part 1 materiał zaczerpnięty ze strony: https://medium.com/data-science-bootcamp/convolutional-neural-networks-cnn-and-computer-vision-101-fundamentals-part-1-a5d39a1faf8f [↑](#footnote-ref-69)
70. Aurelien Geron, op.cit, s.444. [↑](#footnote-ref-70)
71. Aurelien Geron, op.cit, s.442. [↑](#footnote-ref-71)
72. Aurelien Geron, op.cit, s.443. [↑](#footnote-ref-72)
73. Keiron O’Shea and Ryan Nash, An Introduction to Convolutional Neural Network, arXiv, 2015, s.6-8. [↑](#footnote-ref-73)
74. Ibidem [↑](#footnote-ref-74)
75. Aurelien Geron, op.cit, s.443. [↑](#footnote-ref-75)
76. Keiron O’shea and Ryan Nash, op.cit, s.6. [↑](#footnote-ref-76)
77. Aurelien Geron, op.cit, s.445. [↑](#footnote-ref-77)
78. Ibidem. [↑](#footnote-ref-78)
79. Frank Millstein, op.cit, s.34. [↑](#footnote-ref-79)
80. Shallu Sharma and Rajesh Mehra, *Implications of Pooling Strategies in Convolutional Neural Networks: A Deep Insight*, Foundations of computing and decision sciences vl.44 No.3, sciendo, 2019, s.305-308. [↑](#footnote-ref-80)
81. Ibidem [↑](#footnote-ref-81)
82. Hossein Gholamalinezhad, Hossein Khosravi, Pooling Methods in Deep Neural Networks, a Review, arXiv, 2020, s.2.3. [↑](#footnote-ref-82)
83. Ibidem. [↑](#footnote-ref-83)
84. A Gentle Introduction to Pooling Layers for Convolutional Neural Networks materiał zaczerpnięty ze strony: https://machinelearningmastery.com/pooling-layers-for-convolutional-neural-networks/ (dostęp 22.04.2023) [↑](#footnote-ref-84)
85. Hossein Gholamalinezhad, Hossein Khosravi, Pooling Methods in Deep Neural Networks, a Review, arXiv, 2020, s.2.3. [↑](#footnote-ref-85)
86. Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, Kaori Togashi, Convolutional neural networks: an overview and application in radiology, Insights Imaging 9, 2018, s.611-629. [↑](#footnote-ref-86)
87. *Softmax Activation Function: Everything you need to know* materiał zaczerpnięty ze strony: https://www.pinecone.io/learn/softmax-activation/ (dostęp 23.04.2023) [↑](#footnote-ref-87)