Uniwersytet Zielonogórski

Wydział Informatyki, Elektrotechniki i Automatyki

Praca magisterska

Kierunek: Informatyka

Wykorzystanie głębokich sieci neuronowych do wspomagania diagnostyki medycznej

Krystian Dziędzioła

Promotor:
dr inż. Artur Gramacki

Zielona Góra, czerwiec 2019

Streszczenie

Głównym celem pracy jest przedstawienie wyników przeprowadzonych badań dotyczących możliwości zastosowania głębokich sieci neuronowych do wspomagania diagnostyki medycznej. Przykładowym zadaniem, które zostało wybrane jako obiekt badań jest diagnozowanie stanów padaczkowych na podstawie odczytów z elektroencefalogramu (EEG).

// todo: uzupełnić

Słowa kluczowe: Sieci neuronowe, diagnostyka medyczna, EEG, deep learning.

Spis treści

Spis rysunków

Spis tabel

Rozdział 1

Wstęp

1.1. Wprowadzenie

Współczesna technologia sztucznej inteligencji, której dużą część stanowią sztuczne sieci neuronowe, pozwala na wykonywanie przez maszyny zadań, które do niedawna byli w stanie wykonywać tylko ludzie. Jednym z przykładów jest diagnostyka medyczna, która wymaga specjalistycznej wiedzy oraz doświadczenia.

Dzięki sztucznym sieciom neuronowym maszyna jest w stanie nauczyć się pewnych reguł, na podstawie których podejmuje decyzje, które mogą symulować ludzką inteligencję. W niektórych problemach maszyna jest w stanie podejmować poprawne decyzje z dokładnością dorównującą lub nawet przewyższającą zdolności człowieka. Dodatkowo może to zrobić w znacznie krótszym czasie.

// todo: uzupełnić

1.2. Cel i zakres pracy

Celem pracy jest próba wykorzystania głębokich sieci neuronowych do wspomagania diagnostki medycznej dla wybranego problemu praktycznego oraz osiągnięcie możliwie najlepszych rezultatów.

Zakres pracy obejmuje:

- krótkie wprowadzenie w tematykę sieci głębokich oraz porównanie z klasycznymi sieciami neuronowymi,
- przegląd dostępnych bibliotek i narzędzi programistycznych dla sieci głębokich,
- omówienie praktycznego problemu z dziedziny diagnostyki medycznej i próba rozwiązania z wykorzystaniem sieci głębokich,
- przedstawienie szczegółów technicznych i implementacyjnych,
- wykonanie eksperymentów, ich ocena oraz sformułowanie wniosków końcowych.

1.3. Przegląd literatury

[2] [3] [4] // todo:

1.4. Struktura pracy

// todo:

Rozdział 2

Wprowadzenie do głębokich sieci neuronowych

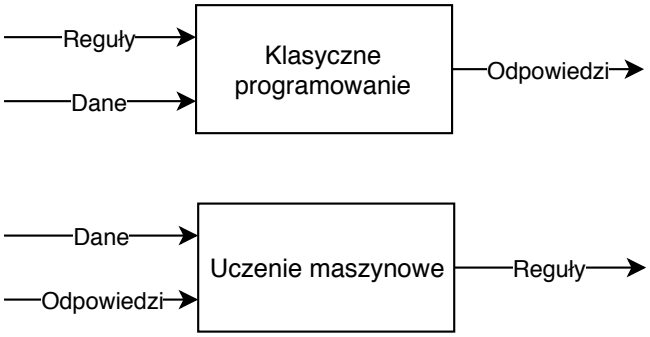
2.1. Sieci neuronowe a sztuczna inteligencja

W ciągu ostatnich kilku lat sztuczna inteligencja stała się tematem dużego zainteresowania w nauce, technologii oraz życiu codziennym. Ma to miejsce za sprawą szerokiego spektrum jej zastosowań oraz możliwości wykorzystania do zadań, które do tej pory były niemal nieosiągalne przez maszyny. Są to m.in. autonomiczne pojazdy, inteligentni wirtualni asystenci, wspomaganie diagnostyki medycznej oraz wiele innych.

Sztuczna inteligencja datuje swoje początki na lata 50-te XX wieku kiedy to zaczęto prowadzić badania nad tym czy komputer jest w stanie nauczyć się "myśleć". Bardziej ścisłą definicją celu prowadzonych badań mogłaby być "próba automatyzacji zadań intelektualnych, które normalnie wykonywane są przez człowieka". Początkowo próby te były poczynione poprzez zaprogramowanie szeregu reguł, którymi miała posługiwać się maszyna w podejmowaniu swoich własnych decyzji. Jest to tzw. symboliczna sztuczna inteligencja *(ang. symbolic artificial intelligence)*, czyli klasyczne podejście polegające na zaprogramowaniu określonego algorytmu działania. [5] Zauważono jednak, że zbiór określonych przez programistów reguł sprawdza się do rozwiązywania dobrze zdefiniowanych problemów logicznych takich jak np. gra w szachy. Takie podejście nie jest jednak wystarczające do osiągnięcia przez komputer umiejętności podejmowania decyzji, która mogłaby symulować ludzką inteligencję oraz wykonwania przez nią bardziej abstrakcyjnych zadań takich jak np. rozpoznawanie mowy, obrazu lub tłumacznie języków.

Nasuwa się pytanie czy komputer jest w stanie wyjść poza ramy ściśle zdefiniowanych reguł i samemu nauczyć się w jaki sposób wykonywać określone zadania? Odpowiedzią na to pytanie oraz niedoskonałości klasycznego programowania było zastosowanie podejścia zwanego dziś uczeniem maszynowym *(ang. machine learning)*. Polega ono na tym, że maszynie nie jest dostarczany ściśle określony algorytm działania, a jedynie przedstawiane są dane oraz wyniki jakie powinny powstać po przetworzeniu tychże danych. Maszyna ma za zadanie stworzyć swoje własne reguły, które będą odzwierciedlać sposób w jaki prawidłowo należy wykonać konkretne zadanie. Można więc powiedzieć, że w uczeniu maszynowym, maszyna jest uczona, a nie programowana. Pomimo tego, że uczenie maszynowe zaczęło się roz-

wijać dopiero w latach 90-tych bardzo szybko stało się popularne za sprawą tego, że przynosiło najlepsze wyniki spośród wszystkich podejść w dziedzinie sztucznej inteligencji. Różnicę pomiędzy klasycznym programowaniem a uczeniem maszynowym zaprezentowano na rysunku 2.1.



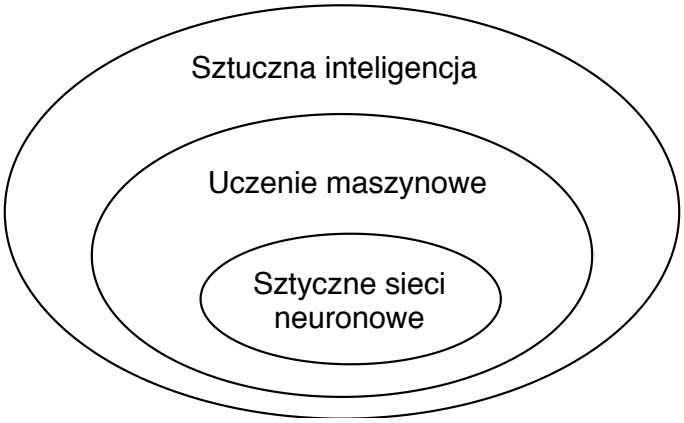
Rys. 2.1. Porównanie klasycznego programowania z metodą uczenia maszynowego

Mnogości zastosowań uczenia maszynowego dorównuje również ilość algorytmów, które są wykorzystywane w tym podejściu. Dostępnych jest wiele modeli, poczynając od bardzo prostych, kończąc na dosyć skomplikowanych. Kilka najczęściej używanych, to m.in.:

- **drzewa decyzyjne** *(ang. decision trees)*,
- **algorytm k-średnich** *(ang. k-means)*,
- **sztuczne sieci neuronowe** *(ang. artificial neural networks)*. [6]

Przedmiotem tej pracy jest ostatnia z wymienionych metody, czyli sztuczne sieci neuronowe. Jest to zestaw algorytmów, który został zaprojektowany w sposób inspirowany działaniem ludzkiego mózgu. W rzeczywistości jednak, ich działanie nie jest do końca zgodne z tym, w jaki sposób działa umysł człowieka. Sieci neuronowe potrafią przyjmować dane w formie wektorów liczbowych, które mogą zawierać takie informacje jak np. obrazy, dźwięki, tekst lub szeregi czasowe. Na ich podstawie są w stanie dokonywać klasyfikacji, klasteryzacji oraz predykcji. Jednym z typów sztucznych sieci neuronowych są tzw. głębokie sieci neuronowe *(ang. deep neural networks)*, które znacząco zwiększają możliwości klasycznych sieci neuronowych. Zostaną one omówione w rozdziale 2.3.

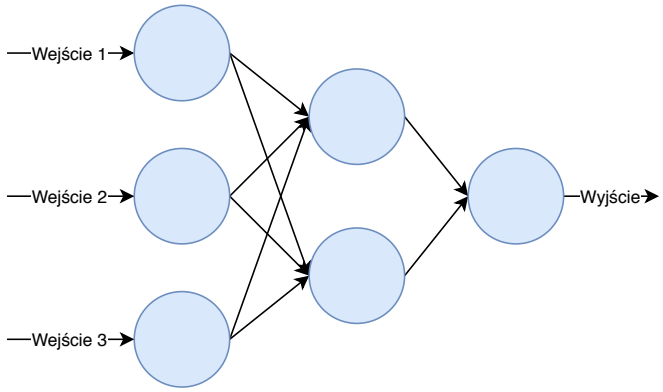
Można zauważyć, że sztuczna inteligencja jest pojęciem bardzo szerokim. W jej skład wchodzą różne metody m.in. uczenie maszynowe. Z kolei specyficznym podejściem w uczeniu maszynowym są sieci neuronowe. Zależności pomiędzy tymi pojęciami przedstawia rysunek 2.2.



Rys. 2.2. Zależność między sztuczną inteligencją, uczeniem maszynowym i sieciami neuronowymi

2.2. Klasyczne sieci neuronowe

Sieci neuronowe to dziedzina uczenia maszynowego, która jako model matematyczny podlegający uczeniu wykorzystuje sturukturę sieci składającą się z wielu jednostek obliczeniowych zwanych neuronami. Neurony wykonują podstawowe obliczenia i ich wyniki przekazują do kolejnych neuronów. Operacje, które są wykonywane przez neurony to przeważnie sumowanie otrzymanych informacji oraz zastowanie prostej, nieliniowej funkcji. W większości sieci neuronowych, neurony zgrupowane są w tzw. warstwy. Obliczenia wykonane przez jedną warstwę są przekazywane do kolejnej, która z kolei wykonuje obliczenia na otrzymanych wynikach. Ostatnia warstwa zwraca końcowy wynik, który jest interpretowany w różny sposób w zależności od wykonywanej operacji np. klasyfikacja lub regresja. Schemat prostej sieci neuronowej przedstawiony został na rysunku 2.3



Rys. 2.3. Struktura prostej sieci neuronowej

Pierwszą koncepcję neuronu datuje się na rok 1943, w którym to powstał model neuronu McCulloch-Pitts. Był on bardzo prosty w porównaniu do współczesnych sieci neuronowych, gdyż pozwalał jedynie na wykorzystanie wartości binarnych na wyjściu z neuronów. Każdy z nich sumował wartości wejściowe i przyrównywał je do zera. Dodatkowo nie istniała żadna reguła aktualizacji wewnętrznych wartości neuronów (tzw. wag), która jest niezbędna do prawidłowego przebiegu procesu adaptacji neuronu do nowych informacji. Bez takiej reguły wszystkie wagi neuronów

musiałyby być ustawiane ręcznie.

W latach 50-tych przedstawiony został perceptron, który jest najprostszą siecią neuronową składającą się wielu neuronów McCullocha-Pittsa. Implementuje on algrotym uczenia nadzorowanego klasyfikatorów binarnych. Jest funkcją przynależności potrafiąca przydzielić jedną z dwóch klas do danych parametrów wejściowych. W perceptronie została przedstawiona prosta reguła służąca do aktualizacji wag dla kolejnych iteracji. Wzór na wagę w kroku $t + 1$ przedstawia się następująco:

$$w_i(t+1) = w_i(t) + (d_j - y_j(t))x_{j,i} \tag{2.1}$$

gdzie w_i to waga i-tego neuronu, d_j to oczekiwana wartość dla j-tego wejścia, $y_i(t)$ oznacza wartość obliczoną z j-tego wejścia, natomiast $x_{j,i}$ jest i-tą wartością j-tego wejścia. Oznacza to, że przyszła wartość wagi neuronu obliczona jest przez dodanie błędu (różnicy między wartością oczekiwaną a obliczoną) pomnożonego przez wartość rzeczywistą do wartości aktualnej wagi. Reguła ta może być stosowana jedynie do uczenia jednowarstwowych sieci neuronowych, co znacznie ogranicza zakres jej zastosowań. Perceptron może więc być używany jedynie do problemów liniowo separowalnych.[3]

W latach 60-tych zostało matematycznie udowodnione, że sieć z pojedynczą warstwą nie posiada możliwości klasyfikowania problemów, które nie są liniowo separowalne. Przykładem jest funkcja alternatywy wykluczającej - XOR. Kolejne, bardziej skomplikowane problemy, których się podejmowano, takie jak m.in. rozpoznawanie mowy, kończyły się więc niepowodzeniem. Możliwość uczenia sieci wielowarstwowych była więc konieczna w dalszym rozwoju sieci neuronowych. W tej samej publikacji [4] zostało również udowodnione, że 2-warstwowa sieć jest w stanie zamodelować niemal każdą funkcją. W praktyce jest to jednak ciężkie do osiągnięcia.

W późniejszym okresie przedstawiony został algorytm wstecznej propagacji błędów *(ang. backpropagation)*, który umożliwiał uczenie wielowarstwowych sieci neuronowych. Oparty jest on na minimalizacji wartości tzw. funkcji straty z wykorzystaniem optymalizacyjnej metody największego spadku. Błędy obliczane są na warstwie wyjściowej i przekazywane są wstecz, do warstw poprzedzających. Jest to możliwe dzięki wykorzystaniu tzw. metody łańcuchowej *(ang. chain rule)*. Szczegóły dotyczące tego algorytmu zostały opisane w wielu publikacjach m.in. w [7]. Dzięki zastosowaniu tego typu podejścia osiągnięty został cel, który zakładał umożliwienie uczenia wielowarstwowych sieci neuronowych. Pozwoliło to w znaczącym stopniu przyspieszyć rozwój w tej dziedzinie.

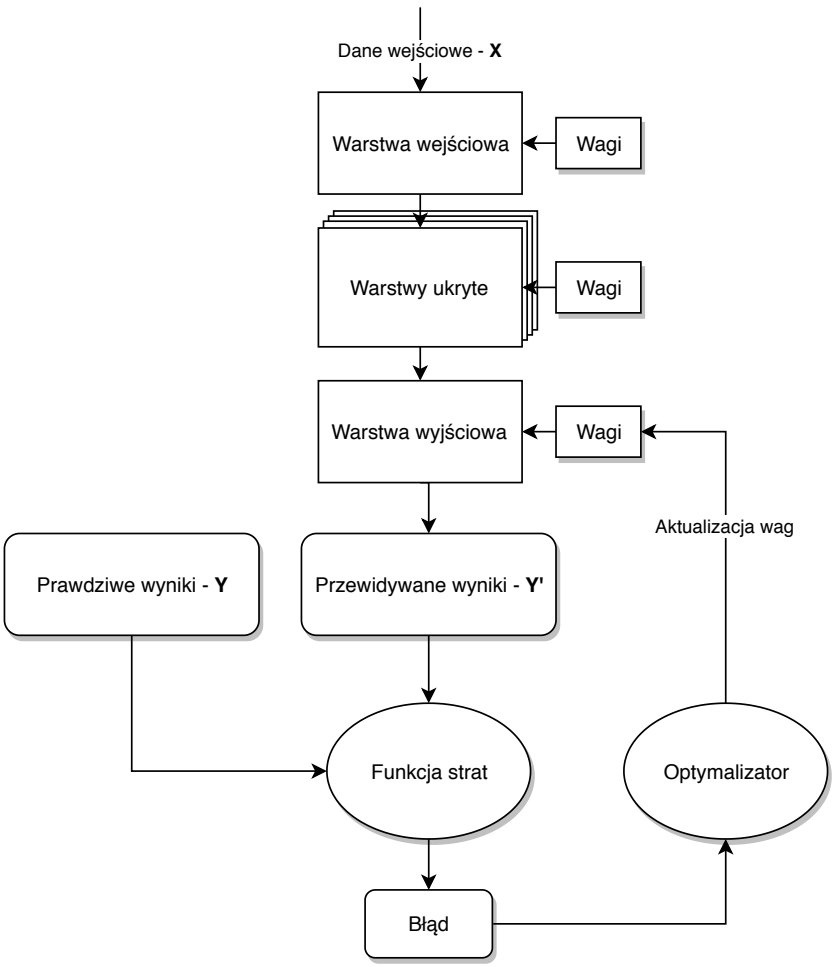
Algorytm wstecznej propagacji błędów stał się najpopularniejszym i najbardziej skutecznym algorytmem do nauki sieci neuronowych. Ma on właśnie zastosowanie w głębokich sieciach neuronowych, które zostały opisane w rozdziale 2.3.

2.3. Deep learning, czyli głębokie sieci neuronowe

Deep learning to dziedzina uczenia maszynowego, która zakłada nowe podejście do nauki wzorców w danych. Kładzie ona nacisk przede wszystkim na naukę kolejnych warstw reprezentacji, które z każdą warstwą są coraz bardziej konkretne. Słowo "głębokie" *(ang. deep)* odnosi się głównie do ilości warstw, które składają się na model sieci. Ta cecha jest zwykle nazywana głębokością sieci *(ang. depth)*. Nowoczesne

modele głębokich sieci neuronowych często uczą się dziesiątek, a czasem nawet setek kolejnych warst reprezentacji danych. Każda z nich uczy się samodzielnie na podstawie przedstawionych im informacji.

Inne podejścia, takie jak np. klasyczne uczenie maszynowe skupia się na nauce jedynie jednej lub dwóch warstw reprezentacji danych, dlatego tego typu metody są często nazywane "płytkim uczeniem" *(ang. shallow learning)*. W deep learning'u jako model matematyczny prawie zawsze używa się sieci neuronowych, które składają się z nałożonych na siebie wielu warstw neuronów.



Rys. 2.4. Schemat procesu uczenia głębokiej sieci neuronowej

Każda z warstw posiada zestaw tzw. wag *(ang. weights)*, które określają sposób w jaki interpretowane są dane wejściowe. Z technicznego punktu widzenia wagi są parametrami transformacji, które są wykonywane przez warstwy na danych wejściowych. W tym kontekście uczenie oznacza poszukiwanie odpowiedniego zestawu wag, który pozwoli na poprawne przekształcanie danych wejściowych w oczekiwane rezultaty. W głębokich sieciach neuronowych mogą istnieć miliony tego typu parametrów, więc znalezienie odpowiedniej wartości dla każdego z nich może być trudnym zadaniem, biorąc pod uwagę, że zmiana jednego parametru może mieć wpływ na zachowanie innych.

W celu poprawnej modyfikacji parametrów sieć musi obserwować wyniki wyjściowe i porównywać je z wynikami docelowymi, które dostarczone zostały w procesie

uczenia. Pomiarem tego jak bardzo uzyskane wyniki różnią się od tych prawdziwych zajmuje się tzw. funkcja strat *(ang. loss function)*. Zwraca ona konkretną wartość liczbową popełnionego błędu, który następnie może być minimalizowany w celu uzyskiwania coraz to lepszych wyników. Odbywa się to poczynając od warstwy ostatniej, kończąc na warstwie wejściowej stosując algorytm wstecznej propagacji błędów, który implementowany jest przez tzw. optymalizator *(ang. optimizer)*. [7]

Początkowo wagi posiadają losowe wartości, które dostosowywane są w ramach wielu iteracji zwanych epokami uczącymi *(ang. epochs)*. Przeważnie proces powtarzany jest dziesiątki razy na tysiącach przykładowych danych. Wybierany jest zestaw wag sieci, który posiada najmniejszą wartość funkcji strat, czyli jest najlepiej dostosowany do poprawnego przetwarzania danych wejściowych na wyjściowe. Cały proces uczenia głęgokiej sieci neuronowej został zaprezentowany na rysunku 2.4.

Możliwość rozwoju głębokich sieci neuronowych była więc uwarunkowana powstaniem sprzętu o odpowiedniej mocy obliczeniowej, która umożliwiłaby wykonywanie tak dużej ilości obliczeń w skończonym czasie. Obecnie używane są do tego procesory kart graficznych *(GPU - ang. graphics processing unit)*, które swoimi możliwościami znacznie przewyższają zwykłe procesory *(CPU - central processing unit)*.

2.4. Architektury głębokich sieci neuronowych

Wyróżnia się 5 najczęściej stosowanych architektur głębokich sieci neuronowych. Są to:

- **Sieci splotowe** *(CNN - ang. Convolutional Neural Networks)*
- **Rekurencyjne sieci neuronowe** *(ang. Recurrent Neural Networks)*
- **ResNets** *(Residual Networks)*
- **Autoenkodery** *(ang. Autoencoders)*
- **GAN** *(ang. Generative Adversarial Networks)*

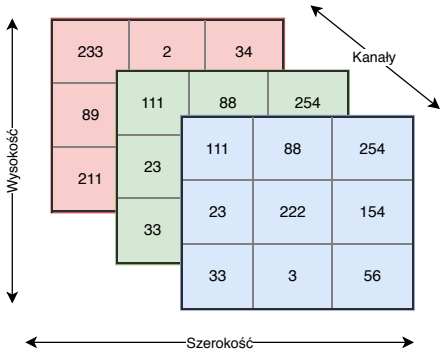
Zostaną omówione jedynie dwie pierwsze architektury, gdyż to one znalazły zastosowanie w niniejszej pracy.

2.4.1. Splotowe sieci neuronowe

Splotowe sieci neuronowe są niewątpliwie najpopularniejszą architekturą głębokich sieci neuronowych. Są one przeważnie używane do zadań związanych z rozpoznawaniem obrazów, jednak równie dobrze sprawdzają się w wielu innych dziedzinach.

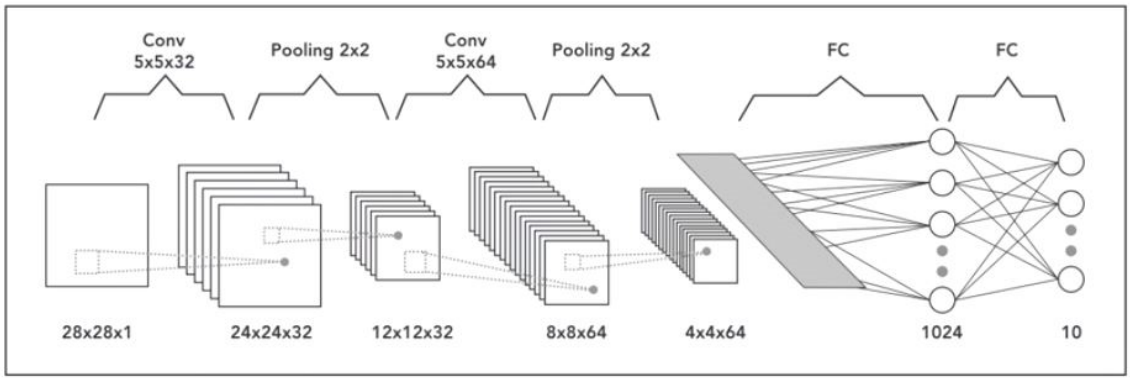
Przewagą tego typu sieci nad klasycznymi sieciami neuronowymi jest możliwość rozpoznawania tymczasowych oraz przestrzennych zależności pomiędzy danymi. Jest to możliwe, dzięki temu, że dane które trafiają do sieci mogą posiadać strukturę wielowymiarową. Przykładowo, obrazki zapisane w formacie RGB są 3-wymiarową macierzą przechowującą informację o wartościach koloru każdego z pikseli 2.5.

Na danych przestrzennych wykonywane są operacje takie jak: konwolucja *(ang. convolution)*, zmniejszanie rozmiaru danych *(ang. subsampling)* oraz aktywacja *(ang.*



Rys. 2.5. Dane liczbowe obrazka w formacie RGB

activation). Znalezione cechy są następnie podawane na wejście klasycznej sieci, gdzie neurony połączone są każdy z każdym *(ang. fully connected)*, która wykonuje ostateczne operacje i zwraca wynik końcowy. Działanie sieci splotowej polega na tworzeniu kolejnych tzw. map cech *(ang. feature map)*, które są abstrakcyjnymi reprezentacjami danych umożliwiającymi wychwycenie podobieństw i zależności pomiędzy danymi. Szczegółowy opis sposobu działania tych operacji został przedstawiony m.in. w [2] oraz [7]. Przykładowy schemat splotowej sieci neuronowej składający się z kliku warstw konwolucyjnych połączonych z warstwami zmniejszającymi rozmiar danych (tzw. *pooling*'iem), która została zakończona dwoma warstwami sieci *FC - fully connected* przedstawiony jest na rysunku 2.6.



Rys. 2.6. Schemat działania sieci splotowej [1]

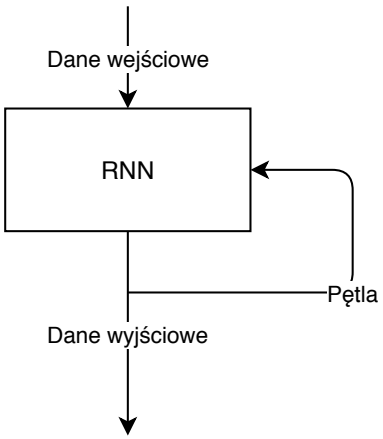
Zaletą tego typu sieci jest bez wątpienia możliwość przetwarzania wielowymiarowych danych, dzięki czemu są one w stanie odnajdywać przestrzenne zależności pomiędzy nimi, co przekłada się na ich wysoką skuteczność i wszechstronność.

2.4.2. Rekurencyjne sieci neuronowe

Rekurencyjne sieci neuronowe są często używane do rozwiązywania problemów, gdzie wykorzystywane są dane sekwencyjne. W praktyce znajdują one zastosowanie w takich problemach jak np. przetwarzanie języka naturalnego, synteza mowy czy automatyczne tłumaczenie języków.

Idea działania rekurencyjnych sieci neuronowych polega na zapamiętywaniu pewnego ciągu sekwencji i dokonywanie wyborów nie tylko na podstawie aktualnych

danych, ale również tych, które były przetwarzane poprzednio. Jest to realizowane za pomocą wewnętrznej pętli, która polega na przekazywaniu wyników z danej warstwy jako dodatkowych danych wejściowych podczas kolejnej iteracji. Schemat pętli zaprezentowany został na rysunku 2.7.



Rys. 2.7. Pętla w rekurencyjnej sieci neuronowej

Zaletą rekurencyjnych sieci neuronowych jest to, że współdzielą one jeden zestaw parametrów we wszystkich krokach, dzięki temu pozwala to zredukować ich ilość. Mogą być on również używane w połączeniu z sieciami splotowymi.

Do wad należy m.in. to, że nie są one w stanie zapamiętywać zbyt długich sekwencji. Dodatkowo ich głębokość nie może być zbyt duża, ponieważ używane funkcje aktywacji powodują, że gradient zmian zanika po przejściu przez kilka warstw.

Rozdział 3

Omówienie wybranego problemu diagnostyki medycznej

3.1. Przedstawienie problemu

W celu sprawdzenia możliwości zastosowania głębokich sieci neuronowych do diagnostyki medycznej został wybrany problem **wykrywania napadów padaczki na podstawie odczytów z elektroencefalogramu** (EEG).

Padaczka (zwana również epilepsją) to grupa przewlekłych zaburzeń neurologicznych, które charakteryzują się napadami padaczkowymi. Napady są powodowane przejściowymi zaburzeniami pracy mózgu polegającymi na samorzutnych wyładowaniach bioelektrycznych w komórkach nerwowych mózgu. Występują one ze zróżnicowaną siłą, od krótkich i ledwo zauważalnych, aż do długich, silnych wstrząsów. Może towarzyszyć im utrata świadomości oraz drgawki. Epilepsja jest nieuleczalna, jednak w wielu przypadkach może być kontrolowana lekami. Cierpi na nią ok. 1% ludzi na całym świecie, czyli ok. 65 milionów [8].

Podczas standardowego badania aktywność mózgu monitorowana jest za pomocą elektroencefalogramu. Odczyty fal mózgowych tworzone są na podstawie sygnałów odbieranych przez elektrody umieszczane na głowie pacjenta. Aktualnie odczyty muszą być analizowane przez wykwalifikowanego neurologa, który jest w stanie rozpoznać charakterystyczne wzorce wskazujące na występowanie choroby. Wizualna diagnostyka odczytów jest jednak niezwykle pracochłonna i wymaga specjalistycznej wiedzy. Skutkiem tego mogą być limity pacjentów, których badania mogą zostać przeanalizowane. Ograniczenia te mogą być rozwiązane poprzez próbę automatycznego wykrywania napadów padaczkowych przez maszynę.

Problem automatycznego wykrywania ataków był już wielokrotnie badany. Często do rozwiązania problemów używane były algorytmy, które używały specjalnie przygotowanych zestawów cech charakteryzujących ataki. Znanym faktem jest jednak to, że napady padaczki mogą być niezwykle zróżnicowane. Tyczy się to zarówno przypadków, w których porównywane są charakterystki ataków różnych pacjentów, jak i tych, które występują podczas badania jednego chorego. Zostały podejmowane również próby wykorzystania głębokich sieci neuronowych, które byłyby w stanie wykryć w danych bardziej generyczne wzorce pozwalające na klasyfikację z większą dokładnością i odpornością na zróznicowanie danych. Sposób, oraz wyniki przeprowadzonych w ten sposób badań można znaleźć m.in. w [9], [10] oraz [11].

3.2. Dostępne dane

Badania opisane w niniejszej pracy zostały przeprowadzone na danych udostępnionych przez Szpital Wojewódzki w Zielonej Górze. Dostępne są odczyty 104 chorych pacjentów w różnym wieku, każde trwające około 15 minut. Zostały one przeanalizowane przez neurologa, który oznaczył punkty w czasie, w których doszło do ataków.

Odpowiednio przygotowane odczyty wraz z wynikami badań posłużą jako dane, na podstawie których zostaną przeprowadzone eksperymenty polegające na próbie nauki głębokiej sieci neuronowej do automatycznego rozpoznawania stanów padaczkowych.

3.3. Oczekiwane rezultaty

Wymienione publikacje naukowe przedstawiają obiecujące wyniki. Niektóre z nich prezentują rezultaty na poziomie nawet ok. 90% skuteczności. Badania te zostały jednak przeprowadzone na zupełnie innych danych. Często używana jest ogólnie dostępnea baza CHB-MIT (Children's Hospital of Boston-Massachusetts Institute of Technology). Nie są znane również szczegóły implementacyjne, dlatego opracowane zostanie własne rozwiązanie analizowanego problemu.

Rezultatem oczekiwanym w tej pracy jest osiągnięcie wyników świadczących o tym, że technologię deep learning'u oraz zaproponowane rozwiąznie można z powodzeniem zastosować również do posiadanej bazy odczytów. Wyniki przewyższające 70% skuteczności powinny być dobrym wyznacznikiem pomyślności wykonanych badań.

Rozdział 4

Przegląd dostępnych bibliotek oraz narzędzi programistycznych

4.1. Dostępne narzędzia

Istnieje niezliczona ilość narzędzi umożliwiających tworzenie oraz naukę głębokich sieci neuronowych. Do tych najbardziej popularnych zalicza się m.in.:

- **TensorFlow** - najczęściej używana biblioteka do deep learning'u stworzona przez Google. Jest ona używana m.in. przez takie firmy jak Airbus, Twitter oraz IBM. Google wykorzystuje ją również w swoich produktach np. tłumacz Google zbudowany jest na bazie TensorFlow. Rekomendowanym językiem jest Python, jednak istnieje możliwość używania języków Java, C oraz Go.

- **Keras** - biblioteka dostarczająca wysokopoziomowe API umożliwiające tworzenie sieci neuronowych w prosty sposób. Keras sam w sobie nie posiada silnika umożliwiającego tworzenia sieci neuronowych, jednak używa jednej z wybranych biblitek (TensorFlow, CNTK lub Theano). Biblioteka jest napisana w Pythonie i ten język jest zalecany do użytku, jednak istnieje możliwość użycia również języka R.

- **Theano** - biblioteka dedykowana dla Pythona umożliwiająca tworzenie sieci neuronowych może być używana w połączeniu z biblioteką Keras.

- **Caffe** - biblioteka stworzona przez zespół Berkeley AI Research (BAIR). Została napisana w C++, jednak umożliwa również korzystanie z Matlab'a oraz Python'a.

- **Deeplearning4j** - biblioteka napisana w Javie, kompatybilna ze wszystkimi językami uruchamianymi na wirtualnej maszynie Javy (JVM) m.in. Java, Groovy, Scala czy Kotlin. Jest bardziej ukierunkowana na użycie w aplikacjach biznesowych, niż naukowo-badawczych.

Wszystkie wymienione biblioteki są dostępne za darmo.

4.2. Wybrany stos technologiczny

Stos technologiczny wybrany do wykonania badań w tej pracy to:

- **Python** - najbardziej popularny język programowania do deep learning'u, bardzo dobrze przystosowany do zastosowań naukowych.
- **Keras/TensorFlow** - najczęściej wykorzystywany zestaw bibliotek posiadający bardzo dobrą dokumentację, opisywany w wielu poradnikach. Wspiera również możliwość wykonywania obliczeń na karcie graficznej. Dostarcza dodatkowy zestaw narzędzi umożliwiających monitorowania sieci m.in. TensorBoard.
- **Nvidia cuDNN** - bilbioteka umożliwiająca wykonywanie obliczeń na karcie graficznej podczas nauki sieci neuronowych. Jej używanie jest bardzo zalecane, gdyż umożliwia o wiele szybsze przetwarzanie danych. Wymagane jest posiadania odpowiedniej karty graficznej wspierającej tę technologię.
- **Jupyter Notebook** - środowisko programistyczne umożliwiające pracę w językach Julia, Python oraz R.
- **Docker** - narzędzie służące do konteneryzacji. Umożliwia uruchamianie całego środowiska w odizolowanych kontenerach za pomocą dostępnych publicznie obrazów bez konieczności instalacji w systemie żadnych dodatkowych narzędzi.

Rozdział 5

Próba rozwiązania problemu

Rozdział 6

Zakończenie

Dodatek A

Płyta DVD

Do tesktu pracy załączona została płyta DVD z następującą zawartością:

- plik **/praca-dyplomowa.pdf** - tekst pracy dyplomowej,
- katalog **/projekt/** - zawiera wszystkie pliki wykonanego projektu,
- katalog **/oprogramowanie/** - zawiera oprogramowanie wymagane do uruchomienia projektu. W szczególności są to:

  // todo: uzupełnić

  - katalog **/program/** - program,

Bibliografia

[1] engmrk.com. *Module 22 – Implementation of CNN Using Keras*. https://engmrk.com/module-22-implementation-of-cnn-using-keras/.

[2] Francois Chollet. *Deep Learning with Python*. 2018.

[3] Jacob M. Williams. *Deep Learning and Transfer Learning in the Classification of EEG Signals*. 2017.

[4] Marvin Minsky and Seymour Paper. *Perceptrons*. 1969.

[5] futureoflife.org. *Benefits & risks of artificial intelligence*. https://futureoflife.org/background/benefits-risks-of-artificial-intelligence.

[6] searchenterpriseai.techtarget.com. *Machine learning (ML)*. https://searchenterpriseai.techtarget.com/definition/machine-learning-ML.

[7] Yoshua Bengio Ian Goodfellow and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016.

[8]

[9] Andrew Lim Pierre Thodoroff, Joelle Pineau. *Learning Robust Features using Deep Learning for Automatic Seizure Detection*. 2016.

[10] Emad-ul-Haq Qazi Ihsan Ullah, Muhammad Hussain and Hatim Aboalsamh. *An Automated System for Epilepsy Detection using EEG Brain Signals based on Deep Learning Approach*.

[11] Nick Hershey. *Detecting Epileptic Seizures in Electroencephalogram Data*.