Studium Magisterskie

Kierunek: Analiza danych – Big Data

Imię i nazwisko autora: Mikołaj Jaworski
Nr albumu: 63548

Analiza jakości predykcji klasyfikatora wykrywającego przerzutowe komórki nowotworowe na podstawie zdjęć histopatologicznych węzłów chłonnych z użyciem konwolucyjnych sieci neuronowych

Praca magisterska napisana w
Zakładzie Wspomagania i Analizy Decyzji
Instytutu Ekonometrii, Kolegium Analiz Ekonomicznych
pod kierunkiem naukowym
dr Grzegorza Kolocha

Warszawa 2019

Spis treści

I. Wstęp 4
II. Rozpoznawanie obrazów w medycynie 6
1. Zdjęcia histopatologiczne - detekcja komórek nowotworowych 6
2. Mammograficzne badania przesiewowe – diagnoza raka piersi. 8
3. Wymazy krwi – diagnoza malarii 9
4. Badanie RTG - diagnoza gruźlicy 10
5. Badanie MRI - diagnoza guza mózgu 11
III. Teoretyczne aspekty uczenia maszynowego 12
1. Przykład algorytmu uczenia maszynowego: regresja liniowa. 13
2. Optymalizacja wag – metoda najszybszego spadku 14
3. Generalizacja i podział danych 15
IV. Głębokie uczenie w kontekście rozpoznawania obrazu 18
1. Podstawy działania sieci skierowanych 18
2. Proces uczenia w sieciach neuronowych 22
3. Przewaga sieci nad innymi metodami uczenia maszynowego 24
4. Konwolucyjne sieci neuronowe 26
i. Operacja splotu 27
ii. Konstrukcja sieci 28
iii. Metody regularyzacji 30
V. Analiza właściwa 32
1. Analiza eksploracyjna i przygotowanie danych 32
2. Uczenie transferowe i opis wybranych architektur sieci 34
i. ResNet 35
ii. DenseNet 35
3. Omówienie rezultatów 36
VI. Podsumowanie 40
VII. Bibligrafia 41
VIII. Spis rysunków i tabel 44
IX. Streszczenie 45

I. Wstęp

Uczenie maszynowe stało się dziedziną, która z powodzeniem znajduje zastosowanie w każdym obszarze biznesu i nauki. Rozpoczynając od analiz na potrzeby podejmowania decyzji biznesowych, jak szacowania przychodów czy grupowanie klientów, poprzez systemy rekomendacyjne wspomagające proces zakupowy, aż po wirtualnych asystentów, autonomiczne pojazdy, czy przewidywanie trzęsień ziemi. Ilość aplikacji opartych na statystycznych regułach decyzyjnych jest bardzo długa i nie sposób wymienić je wszystkie. Równie dużo jest algorytmów, które z kolei można przyporządkować do trzech grup: uczenie ze wzmocnieniem, uczenie nienadzorowane i nadzorowane. Wśród metod, na których opierają się omawiane programy, można wyróżnić analizę regresji, drzewa decyzyjne, sieci neuronowe. Ta ostatnia znajduje zastosowanie w każdym z trzech obszarów uczenia maszynowego i zasłużyła sobie na swoją własną poddziedzinę, nazywaną głębokim uczeniem (*deep learning*). Staje się ona coraz bardziej popularna dzięki zwiększonemu dostępowi do mocy obliczeniowej oraz skuteczności w takich obszarach jak rozpoznawanie tekstu (*natural language processing*), dźwięku, czy obrazu (*computer vision*), ale również w klasycznych analizach danych tabelarycznych.

Branżą, która swój rozwój bardzo mocno opiera na zaawansowanej analityce jest ochrona zdrowia. Mowa tutaj o jej zastosowaniach przy testach klinicznych, szacowaniu ryzyka zachorowań i zgonów, diagnostyce na podstawie zdjęć różnego pochodzenia czy klasyfikacji dokumentacji.[1] Szacuje się, że w 2017 roku globalny rynek rozwiązań wykorzystujących zaawansowaną analitykę w służbie zdrowia był wart 14,25 miliardów USD (z czego 158,6 milionów obejmowały systemy analizy obrazu)[2], a do 2025 roku ma osiągnąć wartość prawie 70 miliardów USD.[3] To właśnie diagnostyka na podstawie zdjęć jest jednym z najbardziej medialnych zastosowań analityki w medycynie i właśnie na niej będzie się skupiać niniejsza praca.

Nadrzędnym celem badawczym było zbudowanie możliwie dokładnego klasyfikatora opartego na konwolucyjnych sieciach neuronowych (*convolutional neural networks - CNN*), który na podstawie zdjęć histopatologicznych (mikroskopowych) węzłów chłonnych zlokalizowanych w okolicach piersi pacjentek określa obecność lub brak tkanek przerzutowych nowotworu. Biorąc

[1] "Ascent of machine learning in medicine". Nature Materials 18, 407. 2019.
[2] "Global $4.5 Bn Computer Vision in Healthcare Market Outlook 2017-2019 & Forecast to 2026". businesswire.com. 2019.
[3] "Global Big Data in Healthcare Market: Analysis and Forecast, 2017-2025". bisresearch.com. 2018.

pod uwagę wysoką skuteczność zaprezentowanego rozwiązania, może ono znaleźć zastosowanie w zespołach lekarzy onkologów wspomagając i przyśpieszając diagnostyczny proces decyzyjny. Badania wykazały, że ilość informacji (zdjęć do analiz) otrzymywanych przez radiologów zwiększa się, a to, poprzez nadmiar pracy, może prowadzić do pomyłek i przeoczeń. To wskazuje na zapotrzebowanie na tego typu systemy.[4]

Do budowy narzędzia (trenowania, walidacji, oraz testowania modelu) wykorzystano konkursowy zbiór danych, który został udostępniony publicznie na stronie kaggle.com.[5] Jest to zmodyfikowana wersja (bez zduplikowanych plików) zbioru PatchCamelyon, który z kolei pochodzi z innego konkursowego zbioru, Camelyon16.[6,7] Warto zaznaczyć, że użyty w niniejszej pracy zbiór został już podzielony na mniejsze części (277 483), podczas gdy oryginał nie został jeszcze poddany takiej operacji i zawiera kilkaset pełnowymiarowych fotografii histopatologicznych (*whole-slide images*). Jest to pewnego rodzaju uproszczenie pod względem technicznym, ale utrudniające analizę danych, co zostanie wyjaśnione w dalszej części pracy. Do realizacji zadania wykorzystano język programowania Python, a ściślej biblioteki PyTorch oraz FastAI.[8] Ze względu na rozmiar zbioru danych, do przyśpieszenia złożonych obliczeń użyto instancji z kartami graficznymi w chmurze obliczeniowej Google Cloud. Teoretyczny opis wykorzystywanej metody, modyfikacji zdjęć, architektury sieci oraz rezultaty analizy zostaną zaprezentowane odpowiednio w drugim i trzecim rozdziale pracy. Szerszy opis zagadnienia jakim jest rozpoznawanie obrazów w medycynie wraz z omówieniem wybranych prac naukowych (zarówno ośrodków komercyjnych, jak np. Google[9] oraz akademickich, jak Uniwersytet Nowojorski[10]) znajdzie się w rozdziale pierwszym.

---

[4] McDonald, RJ. et. al. “The effects of changes in utilization and technological advancements of cross-sectional imaging on radiologist workload”. PubMed-NCBI. 2019.
[5] “Histopathologic Cancer Detection”. kaggle.com. 2019.
[6] Veeling, Basitan. et. al. "Rotation Equivariant CNNs for Digital Pathology". arXiv:1806.03962. 2018.
[7] Ehteshami Bejnordi et al. “Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer”. JAMA, 318(22). 2017.
[8] Howard, Jeremy. et. al. “Practical Deep Learning for Coders v3”. course.fast.ai. 2019.
[9] Liu, Yun. et. al. “Detecting Cancer Metastases on Gigapixel Pathology Images”. Google AI. 2017.
[10] Geras, Krzysztof, et. al. “High-Resolution Breast Cancer Screening with Multi-View Deep Convolutional Neural Networks”. arXiv:1703.07047. 2018.

II. Rozpoznawanie obrazów w medycynie

Koncepcja systemu analizującego swoje własne otoczenie za pomocą aparatu przypominającego ludzkie oko pojawiła się już w latach 60. i w ciągu następnej dekady zaprezentowano metody, które są podstawą dzisiejszych algorytmów, jak np. ekstrakcja krawędzi. Dalsze badania obejmowały rozległe matematyczne i ilościowe analizy, w wyniku których wprowadzono metody regularyzacji jak rozmycie czy cieniowanie obrazów. Dzięki wiedzy stojącej za aparatami fotograficznymi, fotogrametrią oraz grafiką komputerową, a metody analizy obrazów impuls do dalszego rozwoju.

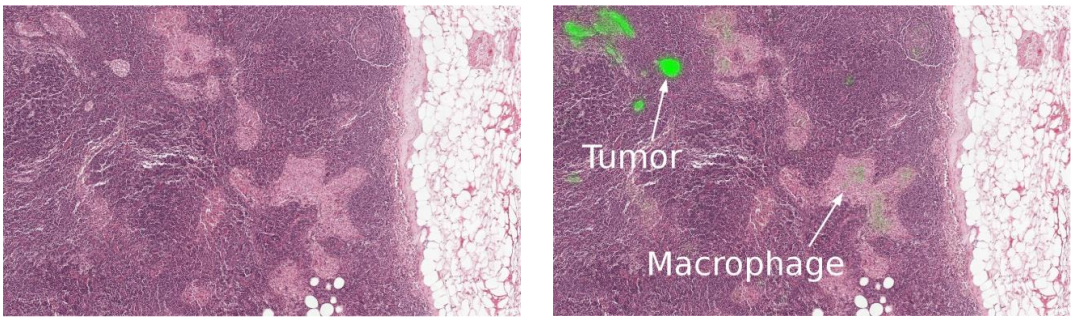
Historia zdjęć diagnostycznych rozpoczęła się w 1895 roku za sprawą wynalezienia promieni X Roentgena (nie licząc badań za pomocą mikroskopu). Od tamtego czasu medycyna przeszła bardzo długą drogę dokładając metody takie jak tomografia, USG (badanie ultrasonograficzne), czy MRI (rezonans magnetyczny). Stosowane są one w różnych badaniach, a ich interpretacja wymaga bardzo często konkretnej wiedzy eksperckiej. Na dzień dzisiejszy szacuje się, że obrazy obejmują 90% całości danych medycznych.[11] Mimo ich dużej ilości, dane są często trudno dostępne ze względu na ochronę danych osobowych, a zwykle zbiory są niskiej jakości. Wspomniane czynniki sprawiają, że analiza obrazu za pomocą oprogramowania jest trudnym zadaniem. Mimo to, jak wspomniano we wstępie, jest to jedno z najważniejszych zastosowań dla wizji komputerowej, w której pokłada się ogromne nadzieje. Poniżej zaprezentowano wyniki wybranych badań.

1. Zdjęcia histopatologiczne - detekcja komórek nowotworowych.

Na ten moment najpopularniejszym zbiorem danych jest Camelyeon16, którego pochodna jest wykorzystywana również w niniejszej pracy. Ze względu na konkursowy charakter zadania (aktualnie trwa konkurs Camelyeon17[12]) powstało wiele prac naukowych autorstwa drużyn z najlepszych firm i ośrodków naukowych na świecie. Udostępnione modele wykazywały się metryką AUC na poziomie od 0.556 do 0.994. Zbiorcza analiza wszystkich wyników wskazała na skuteczniejszą klasyfikację wykonaną przez algorytmy z najwyższymi metrykami

[11] Landi, Heather. "IBM Unveils Watson-Powered Imaging Solutions at RSNA". hcinnovationgroup.com. 2016.
[12] Ehteshami Bejnordi. et. al. "1399 H&E-stained sentinel lymph node sections of breast cancer patients: the CAMELYON dataset". GigaScience. 2018.

w porównaniu do zespołu 11 ekspertów biorących udział w badaniu, kiedy mieli oni ograniczenie czasowe oraz podobne rezultaty kiedy ograniczeń czasowych nie mieli.[13]



Rysunek 1. Zdjęcie histopatologiczne węzłów chłonnych: (od lewej) wersja surowa oraz wskazanie na podejrzane obszary przez model. Ukazane jest również wyzwanie rozróżniania nowotworu od makrofagów. Źródło: ai.googleblog.com.

Jedną z prac udostępnił zespół z Google Brain. Zaprezentował on model konwolucyjnych sieci neuronowych zbudowany na architekturze GoogLeNet wykrywający komórki nowotworowe na podstawie przeskalowanych i zmodyfikowanych (rotowanie i obracanie zdjęć, manipulacja jasnością, nasyceniem, kontrastem, dodanie rozmycia) zdjęć 100 × 100 pikseli. Do trenowania wykorzystano algorytm SGD (*stochastic gradient descent*), przy wielkości partycji (*batch size*) równej 32, oraz współczynniku uczenia (*learning rate*) na zmiennym poziomie między 0.05 do 0.002. Ze względu na niezbilansowaną liczebność klas używanego zbioru, oprócz AUC badacze zdecydowali się na użycie metryki FROC (*free-response receiver operating characteristic*), która wykorzystywana jest do oceny analiz obrazów medycznych wykonywanych głównie przez lekarzy radiologów. Odpowiada na pytanie jak dobrze radiolog lub program wykrywa podejrzane rejony na fotografiach. Oceną jest poziom ufności, że zaznaczony obszar jest nowotworem.[14] Ostatecznie osiągnięto bardzo wysokie wyniki metryk:

- na zbiorze walidacyjnym 99-100% AUC oraz 99% FROC,
- na zbiorze testowym 97.5% AUC oraz FROC na poziomie 88%.

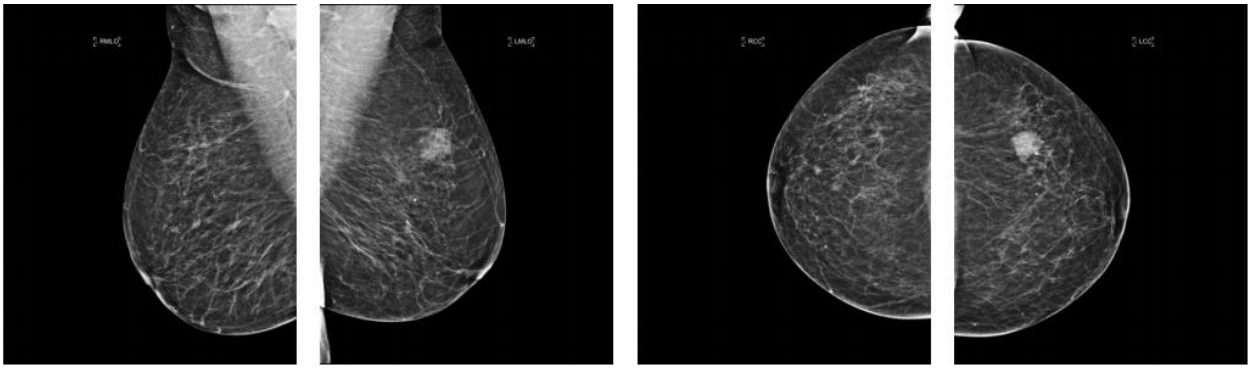
Dla porównania, lekarz diagnostyk przy zasobie czasowym 30 godzin osiągnął poziom 96.6 AUC oraz 73.3% FROC, a zwycięzcy konkursu Camelyeon 2016 odpowiednio 99.4% i 80.7% (na zbiorze testowym).[15]

---

[13] Ehteshami Bejnordi. et al. "Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer". JAMA, 318(22). 2017.
[14] Bandos, Andriy. et. al. "Area under the Free-Response ROC Curve (FROC) and a Related Summary Index". PubMed-NCBI. 2009.
[15] Liu, Yun. et. al. "Detecting Cancer Metastases on Gigapixel Pathology Images". Google AI. 2017.

2. Mammograficzne badania przesiewowe – diagnoza raka piersi.

W podobnej tematyce (raka piersi), ale na innych rodzajach zdjęć pracował zespół z Uniwersytetu Nowojorskiego, który analizował klasyczne zdjęcia mammograficzne. Z tym zadaniem związane były takie wyzwania jak: stworzenie odpowiednio dużego i opisanego zbioru danych oraz fakt, że na jedną osobę mogło przypadać kilka zdjęć wykonywanych pod różnymi kątami oraz w pewnych odstępach czasowych (przy zlecanych badaniach weryfikujących diagnozę oraz kontrolnych). Same zdjęcia mogły się różnić między sobą rozmiarami oraz cechami biologicznymi badanych jednostek. Natomiast o przynależności do danej klasy mogą decydować minimalne zmiany widoczne na fotografiach. Ostatecznie zebrano zbiór składający się z 201 689 badań (886 437 zdjęć), gdzie zadaniem było postawienie diagnozy w metodologii BI-RADS (*Breast Imaging-Reporting and Data System*) obejmującej 3 kategorie: badanie niekompletne, oraz rezultaty normalny i ze wskazaniem na nowotwór. Zastosowano model MV-DCN (*multi-view deep convolutional network*), w którym każde zdjęcie z danego badania było przepuszczane przez konkretne warstwy sieci (konwolucyjne i łączące), aby uzyskane reprezentacje zebrać w formie wektora i zbiorczo poddać warstwie gęstej (*fully-connected layer*) wraz z ostatnią (*softmax*) generującą rozkład prawdopodobieństwa przynależności do klas. Jako algorytm optymalizacyjny zastosowano SGD oraz zastosowano techniki regularyzacyjne takie jak losowe kadrowanie (*random cropping*) oraz dezaktywacja losowych neuronów w warstwach (*dropout*). Warto wspomnieć, że sieci były trenowane na zdjęciach o rozmiarach 2600 × 2000 pikseli. Współczynnik uczenia ustawiono na poziomie 0.00001, a sieć trenowano przez 100 cykli (*epoch*), co na jednej karcie NVIDIA Tesla V100 GPU zajmowało około 4 tygodnie.



Rysunek 2. Cztery różne zdjęcia mammograficzne tej samej osoby zakwalifikowanej przez model do powtórnego badania, które z kolei wykazało obecność nowotworu. Źródło: High-Resolution Breast Cancer Screening with Multi-View Deep Convolutional Neural Networks.

Ze względu na większą liczebność klas zamiast zwykłej miary AUC zastosowano tzw. macAUC, czyli średnią z wyliczonych dla każdej klasy miar pól pod krzywą (gdzie 1 – wybrana klasa, 0 – dwie pozostałe). W specjalnym badaniu model uzyskał wynik 0.688 macAUC. Działanie modelu skonfrontowano z zespołem lekarzy, którzy osiągnęli rezultat 0.704. Natomiast przy współpracy programu i diagnostyków zanotowano najwyższy wynik, tj. 0.735.[16]

3. Wymazy krwi – diagnoza malarii

Niedawno opublikowany został zbiór danych zawierający 27 588 zdjęć uzyskanych z rozmazów krwi (na podstawie krwinek czerwonych) od 150 osób zarażonych pasożytami z rodzaju *Plasmodium* (powodujących malarię) oraz 50 zdrowych. Zdjęcia zostały wykonane za pomocą smartfonów, a następnie ocenione i otagowane przez eksperta.[17] Wymieniony zbiór został natychmiast wykorzystany do badań przez wiele zespołów oraz indywidualnych badaczy.

W jednej z prac porównano kilka modeli konwolucyjnych sieci neuronowych zbudowanych na różnych architekturach: własna, VGG-19, SqueezeNet, InceptionResNet-V2. Osobno najlepszym z nich okazał się VGG-19, rozwinięty przez Oxford's Visual Geometry Group, który brał udział w konkursach takich jak ImageNet.[18] Natomiast w klasyfikacji generalnej najwyższe rezultaty osiągnęło połączenie VGG-19 oraz SqueezeNet. Modele były trenowane na zdjęciach przeskalowanych do rozmiaru 100 × 100 pikseli, gdzie zastosowano augmentacje takie jak rotacje, przewracanie, zbliżenia, obcinanie. Przy ocenie modeli wykorzystano walidację krzyżową w celu minimalizacji błędu związanego z losowym podziałem danych. Jako metryki wykorzystano m.in. trafność predykcji oraz AUC. Po przetestowaniu normalności rozkładu, homogeniczności wariancji (jej równości w grupach) oraz użyciu testów post-hoc Tukeya stwierdzono, że zbiorczy model jest istotnie lepszy od modeli opartych na jednej architekturze. Niniejszym osiągnięto 99.92% AUC oraz 99.51% dokładności oszacowań, co wskazuje na prawie nieomylny model.[19]

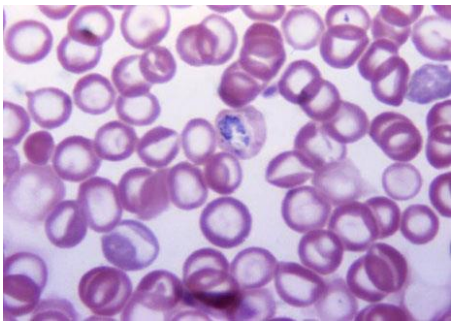
---

[16] Geras, Krzysztof, et. al. "High-Resolution Breast Cancer Screening with Multi-View Deep Convolutional Neural Networks". arXiv:1703.07047. 2018.

[17] Rajaraman, Sivaramakrishnan. et. al. "Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images". PubMed-NCBI. 2018.

[18] Simonyan, Karen, Zisserman, Andrew. "Very Deep Convolutional Networks for Large-Scale Image Recognition". arXiv:1409.1556. 2015.

[19] Rajaraman, Sivaramakrishnan. et. al. "Performance evaluation of deep neural ensembles toward malaria parasite detection in thin-blood smear images". PubMed-NCBI. 2019.



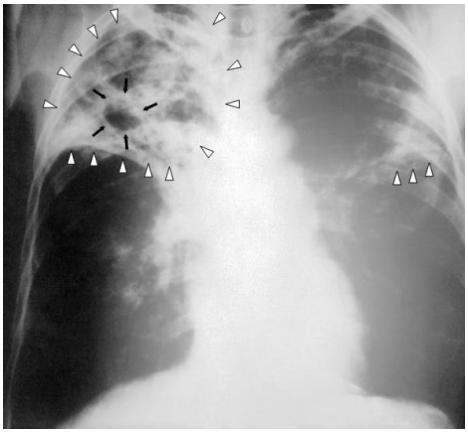
Rysunek 3. Wymaz krwi osoby zarażonej malarią (widoczne pasożyty Plasmodium). Źródło: kaggle.com.

4. Badanie RTG - diagnoza gruźlicy

Kolejne zastosowanie konwolucyjnych sieci neuronowych w analizie obrazów medycznych zaproponowano dla diagnostyki gruźlicy. W tym przypadku zbiór danych był znacznie mniejszej objętości w porównaniu do wcześniej omawianych przypadków i zawierał zaledwie 1007 zdjęć (492 zarażonych i 515 zdrowych). Dane pochodziły z różnych ośrodków badawczych zlokalizowanych w Chinach, USA i na Białorusi. Oryginalnie zdjęcia różniły się od siebie wielkością, więc zostały ustandaryzowane do rozmiaru 256 × 256. Ponadto zmodyfikowano je za pomocą kadrowań (do 227 × 227 pikseli), rotacji, normalizacji, przewrotów. Tak jak w poprzednich przypadkach użyto koncepcji przenoszenia uczenia (*transfer learning*) i zastosowano połączenie przetrenowanych architektur AlexNet oraz GoogLeNet. Modele trenowano przez 120 cykli, gdzie współczynnik uczenia ustawiono na poziomie 0.001. Jako optymalizator wykorzystano SGD.

Zbiór został losowo podzielony na trzy podzbiory: testowy 14.9%, walidacyjny 17.08%, treningowy 68.02%. Jako metrykę wykorzystano AUC, która dla wybranego, łączonego modelu wyniosła 99%, przy czym parametr czułości (*true positive rate*) wyniósł 97.3%, a specyficzność (*true negative rate*) 100%.[20]

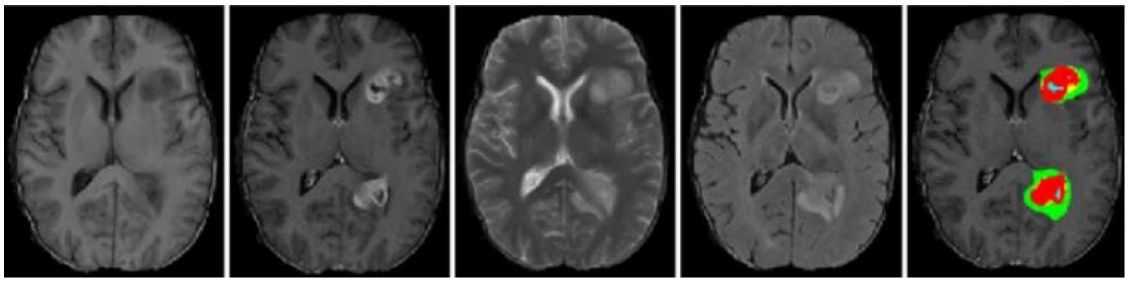
[20] Lakhani, Paras. Sundaram, Baskaran. “Deep Learning at Chest Radiography: Automated Classification of Pulmonary Tuberculosis by Using Convolutional Neural Networks”. RSNA Radiology. 2017.



Rysunek 4. Zdjęcie RTG osoby chorej na gruźlicę. Źrodło: wikipedia.org

5. Badanie MRI - diagnoza guza mózgu

Zdjęcia MRI mózgu mają podobną złożoność co omawiane wcześniej dane z mammografii. W tym przypadku zadaniem jest segmentacja, co przy trójwymiarowym charakterze badania oraz relewantności zmiennych takich jak położenie, kształt, rozmiar patologicznych zmian, które mogą się różnić dla każdego pacjenta, stanowi spore wyzwanie dla analityków. Sam nowotwór niekoniecznie musi mieć wyraźne granice, co w pewnym sensie dyskwalifikuje klasyczne metody oparte na wyróżnianiu krawędzi. Sam proces pozyskiwania zdjęć również może być inny w zależności od osoby wykonującej badanie, co skutkuje np. różnym poziomem intensywności i nasycenia barw.



Rysunek 5. Przykłady segmentacji na zdjęciach MRI głowy. Źródło: www.med.upenn.edu

Od kilku lat rozwijany jest zbiór danych BRATS[21], który w wersji z 2015 roku zawierał 274 zestawów zdjęć osób ze stwierdzonym glejakiem (nowotworem mózgu) w różnym stadium rozwoju. Ponadto 110 oddzielnych zestawów udostępniono jako zbiór testowy. BRATS, jako

[21] Menze, Bjoern. et. al. "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)". IEEE Transactions on Medical Imaging, 34(10). 2015.

pewnego rodzaju punkt odniesienia pozwala na porównywanie różnych metod i podejść. W analizach wykorzystano 3 metryki: czułość, specyficzność (pomocniczo) oraz współczynnik podobieństwa Dice:

$$Dice(P,T) = \frac{|P_1 \wedge T_1|}{(|P_1| + |T_{1|})}/2 \quad (1)$$

gdzie:

P – wyróżniony rejon guza przez wybraną metodę,

T – faktyczny rejon guza,

∧ – operator logiczny AND,

| . | – rozmiar zestawu (liczba elementów graficznych).

Współczynnik obliczono dla 3 obszarów nowotworu: całości, części głównej (bez obrzęku), oraz obszaru z tylko aktywnymi komórkami. Zestawienie metod[22] wyróżniło w pełni automatyczną metodę segmentacji nowotworów opartą na CNN, która dla wersji zbioru z 2013 roku osiągnęła najwyższy wynik odpowiednio: (0.88, 0.83, 0.77) oraz dla wersji z 2015 roku drugie miejsce z wynikami: (0.78, 0.65, 0.75). Metoda jednak wymaga ulepszenia (np. poprzez zmiany w architekturach sieci lub dodaniu informacji z innych źródeł), aby mogła być akceptowalna w warunkach klinicznych.[23]

III. Teoretyczne aspekty uczenia maszynowego

W celu ułatwienia zrozumienia sposobu działania głębokich konwolucyjnych sieci neuronowych oraz budowy finalnego modelu, w tym rozdziale zostaną przedstawione podstawy szerszego pojęcia od głębokiego uczenia, czyli uczenia maszynowego.

Z definicji, program uczący się to taki, który w oparciu o doświadczenie Y i miarę jakości Z, dla danej kategorii zadań X, wraz z przyrostem doświadczenia Y poprawia jakość wykonywanego zadania X mierzoną przez miarę Z. Ogólnie zadania można zdefiniować jako problemy klasyfikacji (zmienna celu skategoryzowana) lub regresyjne (zmienna celu ciągła). Podejście bardziej szczegółowe wyróżnia np. detekcję anomalii, grupowanie, szacowanie funkcji gęstości.

[22] Isin, Ali. et. al. "Review of MRI-based Brain Tumor Image Segmentation Using Deep Learning Methods". Procedia Computer Science 102. 2016.

[23] Pereira, Sergio. et. al. "Brain Tumor Segmentation Using Convolutional Neural Networks in MRI Images". IEEE Transactions on Medical Imaging, 35(5). 2016.

Najprostszymi miarami oceny modelu dla klasyfikatorów mogą być jego dokładność oszacowań (*accuracy*) czy pole pod krzywą ROC (szerzej omówione będą w kolejnych rozdziałach), a dla regresorów błąd średniokwadratowy. Zdobywane doświadczenie różni się w zależności od problemu i dzielimy je na nadzorowane (zbiór zawierający oznakowane dane treningowe i testowe), nienadzorowane (wyszukiwanie zależności w nieoznakowanych danych), ze wzmocnieniem (optymalizacja na podstawie kontaktów agenta z otoczeniem).[24] Zbudowany i opisany w kolejnym rozdziale model rozwiązuje problem klasyfikacyjny za pomocą uczenia nadzorowanego z wykorzystaniem wyżej wymienionych metryk.

1. Przykład algorytmu uczenia maszynowego: regresja liniowa.

Do zbudowania algorytmu uczenia należy połączyć komponenty takie jak sparametryzowany algorytm optymalizacyjny, funkcję straty, model oraz zbiór danych. Dla zobrazowania procesu, wykorzystamy wielowymiarową regresję liniową, którą można zapisać wzorem:

$$\hat{y} = w^T x \tag{2}$$

gdzie y oznacza szacowaną wartość, x oznacza zbiór parametrów, a w odpowiada wektorowi wag przyporządkowywanych każdemu parametrowi. Miarą skuteczności modelu jest błąd średniokwadratowy liczony na zbiorze testowym za pomocą wzoru:

$$MSE_{test} = \frac{1}{m}\sum_i (y_{test} - \hat{y}_{test})^2 \tag{3}$$

Celem jest znalezienie algorytmu, który będzie modyfikował wagi parametrów minimalizując MSE_{test}, co z kolei zostanie osiągnięte poprzez minimalizację MSE_{train} (błąd średniokwadratowy na zbiorze treningowym). Następuje to w momencie, kiedy gradient funkcji będzie się równał 0:

$$\nabla_w MSE_{train} = 0 \tag{4}$$

czyli innymi słowy należy przyrównać pochodne cząstkowe (dla wag w, obserwacji i oraz cechy j) do 0:

$$\frac{dMSE_{train}}{dw_j} = \sum_i 2(y_i - w^T x_i)(-x_{ij}) = 0 \tag{5}$$

co sprowadza się do układu równań z elementami wyrażonymi wzorem:

[24] Mitchell, Thomas. *Machine Learning*. 1st ed. McGraw-Hill, 1997.

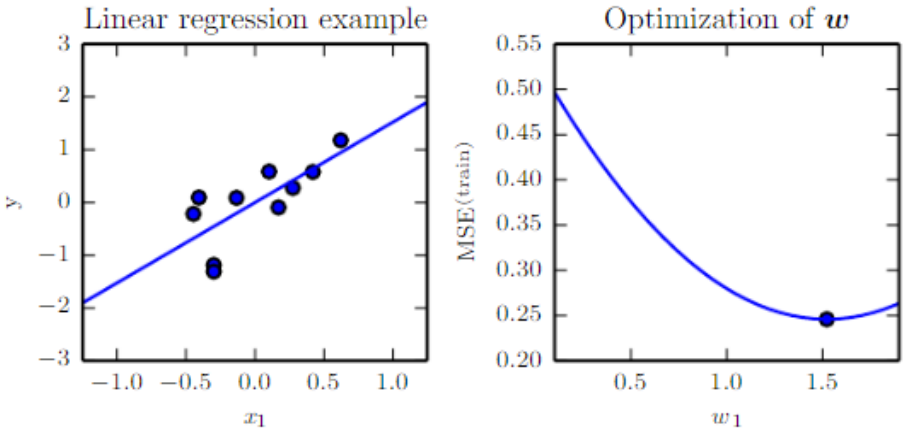
$$w^T \sum_i x_i x_{ij} = \sum_i y_i x_{ij} \quad (6)$$

Po transformacji do formy macierzowej i transponowaniu wektora wag otrzymujemy kolejno:

$$w^T(X^T X) = y^T X \quad (7)$$

$$w = (X^T X)^{-1} y^T X \quad (8)$$

Ostatnie równanie jest szukanym wektorem wag minimalizującym błąd średniokwadratowy.[25] Na wykresach poniżej graficznie zaprezentowano regresję logistyczną dla jednej cechy i odpowiadającej jej wagi.



Rysunek 6. Regresja liniowa oraz optymalizacja wag. Źródło: deeplearningbook.org

Za pomocą regresji liniowej szacujemy potencjalną wartość na skali ciągłej. Natomiast do problemów klasyfikacyjnych stosuje się regresję logistyczną (stosując funkcję logitową), która nakreśla granicę dzielące zmienną na konkretne kategorie. Podsumowując, powyższy przykład jest bardzo trywialny w porównaniu do bardziej złożonych problemów rozwiązywanych przez algorytmy głębokiego uczenia, ale pozwala zrozumieć zasady, na których się opierają.[26]

2. Optymalizacja wag – metoda najszybszego spadku

Wcześniej zaprezentowano rozwiązanie optymalizacji przyrównując gradient do zera, jednak przy bardziej złożonych problemach jest to zbyt skomplikowane obliczeniowo, tj. wymagałoby od programisty tworzenia dużej ilości kalkulacji (8). Dlatego zwykle do znajdowania minimum lub maksimum funkcji używa się metody najszybszego spadku (*gradient descent*). Zamiast

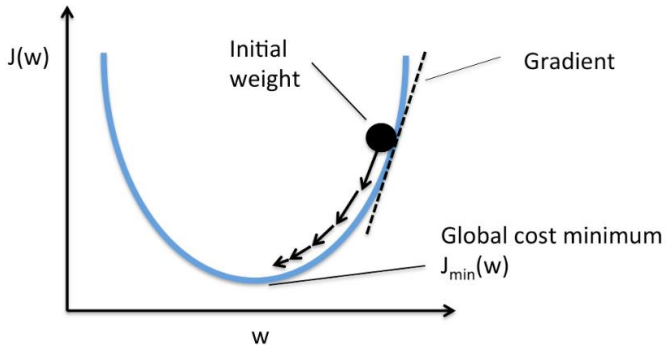
[25] Lazy Programmer (pseudonim). "Deep Learning Prerequisites: Linear Regression in Python". Udemy.com. 2019.
[26] Lazy Programmer (pseudonim). "Data Science: Deep Learning in Python". Udemy.com. 2019.

rozwiązywania problemu analitycznie, stosuje się podejście iteracyjne, gdzie początkowe w wybiera się losowo:

$$w = w - \eta \nabla_w J(w) \tag{9}$$

η to wybierany arbitralnie parametr uczenia, $J(w)$ to funkcja straty. Przykładowo dla wartości: $J(w) = w^2$, $\nabla_w J(w) = \frac{\partial J}{\partial w} = 2w$, początkowego $w = 20$ oraz $\eta = 0.1$ proces będzie przebiegał jak poniżej:

1) Iteracja: $w = 20 - 0.1 \times 40 = 16$,
2) Iteracja: $w = 16 - 0.1 \times 32 = 12.8$,
3) Iteracja: $w = 12.8 - 0.1 \times 25.6 = 10.24$, itd.



Rysunek 7. Proces wyszukiwania minimum funkcji za pomocą metody najszybszego spadku. Źródło: Lazy Programmer Inc.

Problem pojawia się w momencie, kiedy wymiary i ilość obserwacji rosną. Operacja obliczania gradientu w najgorszym przypadku może osiągnąć złożoność obliczeniową $O(m)$, co oznacza, że ilość wykonywanych kalkulacji rośnie liniowo proporcjonalnie do rozmiaru zbioru danych. Dlatego dla ciężkich przypadków używa się wspominanego w drugim rozdziale SGD, który opiera się na założeniach stochastycznych. Oznacza to, że algorytm losowo wybiera próbkę ze zbioru danych na której oblicza gradienty.[27]

3. Generalizacja i podział danych

Gdyby minimalizację funkcji straty przeprowadzano na pełnym zbiorze danych, to uczenie maszynowe nie różniłoby się od problemu optymalizacji. Jednak nadrzędnym zadaniem każdego

[27] Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. "Deep Learning". MIT Press (www.deeplearningbook.org), 2016.

modelu jest generalizacja, czyli odpowiednie szacowanie i minimalizacja błędów na nowych, niedostępnych wcześniej dla modelu obserwacjach. Z tym pojęciem wiążą się kolejne, tj. zbiory treningowe, walidacyjne, testowe oraz przeuczenie i niedouczenie modelu. Role poszczególnych części podziałowych prezentują się następująco:

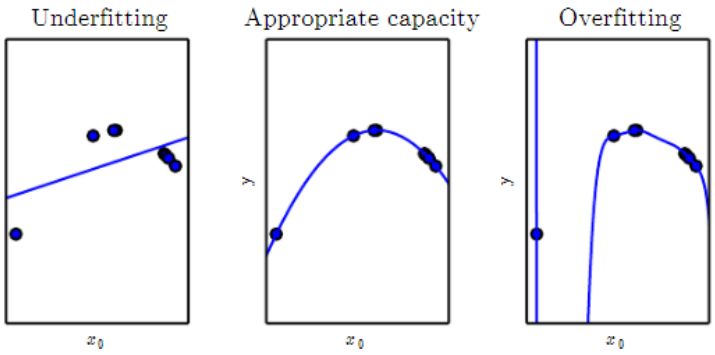
1) Zbiór treningowy ma za zadanie dopasowanie funkcji aproksymacji do danych.
2) Zbiór walidacyjny stosuje się do wygenerowania metryk i porównania działania różnych modeli oraz hiperparametrów.
3) Zbiór testowy potwierdza działanie modelu wybranego na podstawie metryk ze zbioru walidacyjnego.

Teoretycznie można pominąć zbiór walidacyjny i zasilić zbiór treningowy większą ilością danych, ale brak metryk kontrolnych generowanych w trakcie procesu uczenia zwiększa ryzyko przeuczenia (o czym poniżej).[28] Jeżeli otrzymane dane nie są od razu podzielone, należy zastosować proces generowania oparty na założeniach i.i.d (*independent and identically distributed*), czyli zapewniających podobieństwo rozkładów prawdopodobieństwa oraz niezależność obserwacji między grupami.

Zatem dobry model powinien spełniać dwa warunki:

1) Zachować niski błąd na zbiorze treningowym, czyli unikać niedouczenia.
2) Zapewnić odpowiednio niską różnicę między błędami na zbiorach testowym, walidacyjnym, treningowym.

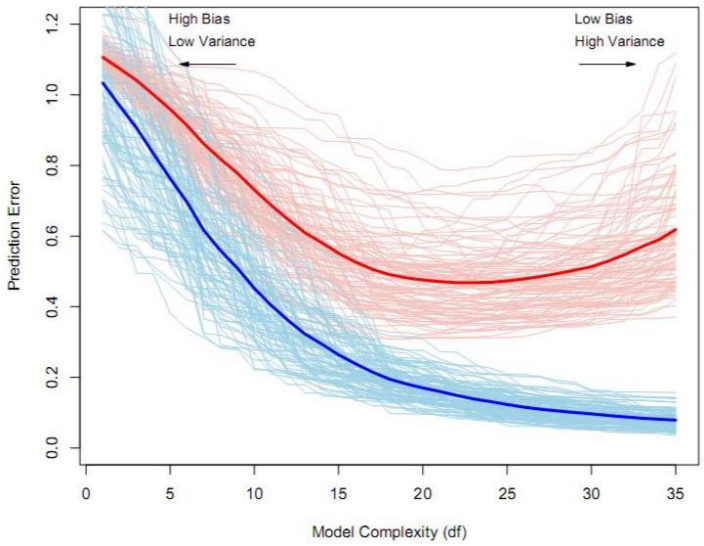
A tym samym zagwarantować odpowiedni poziom generalizacji na świeżych danych. Idea została zaprezentowana na poniższych wykresach.



Rysunek 8. Wykresy prezentujące problemy niedouczenia i przeuczenia w porównaniu do odpowiedniego dopasowania. Źródło: deepleaningbook.org.

[28] Thomas, Rachel. "How (and why) to create a good validation set". Fast.ai. 2017.

Istnieje kilka sposobów radzenia sobie z niedouczeniem, z czego głównym jest zwiększenie ilości informacji (wymiarów) danych lub ilości obserwacji (złożoności modelu). Można zastosować również techniki wzbogacania danych (*feature engineering*) polegające np. na uszczeguławianiu zmiennych czasowych poprzez rozbijanie ich na większą ilość zmiennych (pora dnia, kwartały, miesiące itd.). Natomiast w przypadku przeuczenia stosuje się głównie techniki regularyzacji, czyli techniki mające na celu obiżenie błędu na zbiorze testowym bez naruszania błędu treningowego. Innymi słowy dzięki temu unika się nadawania zbyt dużych wag dla zmiennych. Są różne sposoby w zależności od stosowanych metod, np. nakładanie kar, przycinanie głębokości przy drzewach decyzyjnych oraz zamrażanie neuronów lub wyłączanie warstw w sieciach (metody w kontekście sieci zostaną opisane w następnym rozdziale). Można również manipulować podziałem zbioru danych (np. walidacja krzyżowa, czyli budowanie wielu modeli na różnych konfiguracjach podziału) czy hiperparametrami.[29] Celem jest znalezienie złotego środka (*bias-variance tradeoff*) pomiędzy niedoszacowaniami, a zbytnim dostosowaniem się modelu do szumu w danych, tak jak jest to ukazane na poniższym wykresie.



Rysunek 9. Zależność między błędem predykcji a złożonością modelu dla zbiorów testowego i treningowego. Źródło: Statystyczne Reguły Decyzyjne (materiały dydaktyczne).

[29] Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. "Deep Learning". MIT Press (www.deeplearningbook.org), 2016.

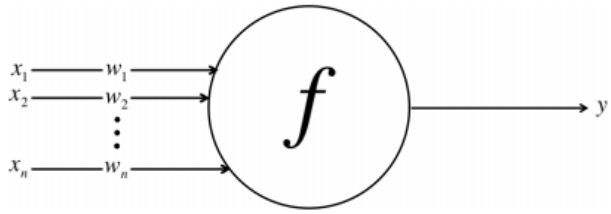
Poruszona tematyka jest o wiele szersza, lecz zaprezentowane podstawy uczenia maszynowego powinny wystarczyć aby zrozumieć zasady, które zostaną rozwinięte w odniesieniu do głębokich sieci neuronowych w kolejnym rozdziale.

IV. Głębokie uczenie w kontekście rozpoznawania obrazu

Zanim poruszone zostaną kwestie związane bezpośrednio z metodami stosowanymi w przypadku rozpoznawania obrazu, konieczne jest zaprezentowanie podstawowych architektur oraz algorytmów na których opierają się sieci neuronowe.

1. Podstawy działania sieci skierowanych

W tym rozdziale przedstawione zostaną podstawowe pojęcia i mechanizmy mające miejsce w sieciach skierowanych, co sprowadza się do procesu tworzenia predykcji na podstawie posiadanych danych.



Rysunek 10. Prezentacja działania neuronu w sztucznych sieciach. Źródło: Fundamentals of Deep Learning.

Główna przewaga sieci neuronowych opiera się na zdolności wychwytywania skomplikowanych interakcji również między zmiennymi objaśniającymi. Podstawowym elementem składowym sieci jest neuron, którego działanie polega na przemnożeniu danych wejściowych przez odpowiadające im wagi, a następnie wygenerowaną (10) użyć w wybranej funkcji generującej daną wyjściową.

$$z = \sum x_i w_i + b \tag{10}$$

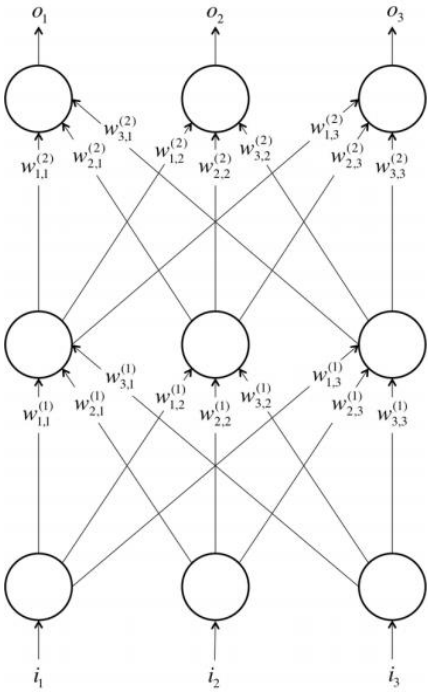
Innymi słowy należy obliczyć produkt (*dot product*) z danych wejściowych i wag, dodać wyraz wolny, a następnie zaaplikować wybraną funkcję aktywacji, która w przypadku regresji logistycznej będzie sigmoidalna (11).

$$p_i = \frac{1}{1 - e^{-z}} \tag{11}$$

Formuła (x) jest inaczej nazywana perceptronem, czyli jednowarstwową siecią neuronową. Zatem każdy neuron można traktować jako prostą operację na wektorach. Sieć neuronowa jest niczym innym jak złożeniem wielu funkcji (12) połączonych ze sobą do rozwiązania jednego problemu.[30]

$$y = f^1\left(f^2\left(f^3(x)\right)\right) \tag{12}$$

Najbardziej podstawową architekturą sieci neuronowych stosowanym w głębokich sztucznych sieciach neuronowych jest wielowarstwowy perceptron należący do rodziny sieci skierowanych (*feedforward*), w których informacja przebiega tylko w jednym kierunku, od danych wejściowych, poprzez kolejne warstwy sieci, aż do warstwy wyjściowej. Do tej kategorii należą również sieci konwolucyjne omawiane w kolejnych rozdziałach. Jeżeli sieć umożliwia przysyłanie wyników wstecz jako dane wejściowe, to będzie to sieć rekrusywna, natomiast jeżeli neurony są połączone w tej samej warstwie, to będzie to sieć rekurencyjna. Te pierwsze są zwykle wykorzystywane do problemów optymalizacyjnych, natomiast drugie do rozpoznawania mowy lub tekstu.[31] Tego typu sieci są poza zakresem niniejszej pracy.

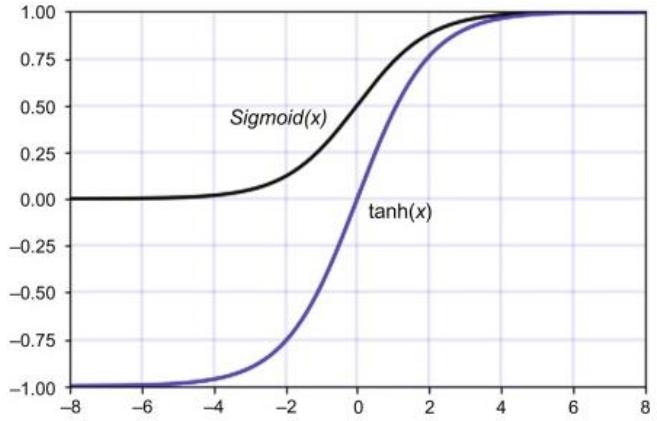


Rysunek 11. Przykład w pełni połączonej, sieci skierowanej z trzema warstwami: wejściową, ukrytą, wyjściową. Źródło: Fundamentals of Deep Learning.

[30] Lazy Programmer (pseudonim). "Data Science: Deep Learning in Python". Udemy.com. 2019.
[31] Russel, Stuart. Norvig, Peter. "Artificial Intelligence: A Modern Approach". 3rd ed. Pearson Education, 2009.

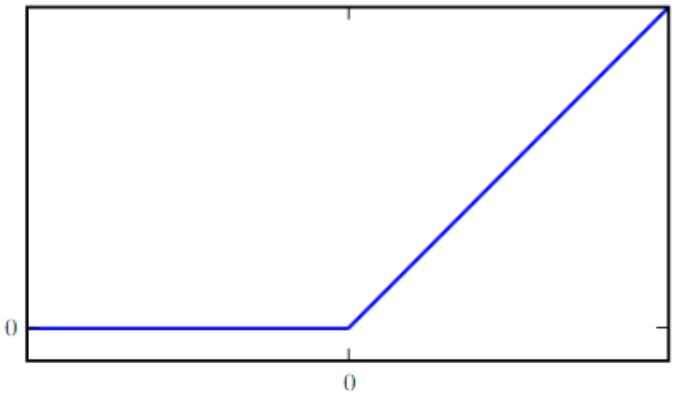
Zaletą sieci neuronowych jest ich umiejętność wykrywania nieliniowych zależności między zmiennymi. W tym celu używa się transformacji funkcji liniowej za pomocą wybranej funkcji aktywacji, która zwykle w równaniach jest oznaczana jako σ (sigma). Najpopularniejsze z nich to:

1) Sigmoid, o którym wspominano wcześniej (11), sprowadza wynik do przedziału $(0; 1)$.
2) Tangens hiperboliczny jest zbliżony kształtem do sigmoidu, ale jego przedział jest szerszy, tj. $(-1; 1)$.



Rysunek 12. Porównanie funkcji aktywacji: sigmoidu i tanh. Źródło: Deep Neural Networks for Natural Language Processing.

3) ReLU (*Rectified Linear Unit*) jest prostym przekształceniem zawierającym się w przedziale $(0; x)$.



Rysunek 13. Funkcja aktywacji ReLU. Źródło: deeplearningbook.com

4) Softmax (13) jest stosowany głównie w warstwach wyjściowych sieci i ma na celu konwersję surowych liczb z przedziału $(-\infty; \infty)$ do postaci prawdopodobieństw przynależności do danej klasy w zakresie $(0; 1)$.

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{13}$$

Dwie pierwsze opcje szczególnie narażone są na tzw. nasycenie lub zanikanie (*vanishing gradient*), czyli osiąganie wartości skrajnie bliskich przedziałom. W tamtych okolicach gradient funkcji jest bardzo mały, co drastycznie spowalnia proces aktualizacji wag, czyli uczenia. Rozwiązaniem może być ReLU, który eliminuje problem nasycenia, dzięki czemu przyśpiesza konwergencję algorytmu optymalizującego. Jedynym problemem może być ryzyko tzw. martwego neuronu, czyli sytuacji kiedy wartość funkcji trafi do 0, z której to nie ma możliwości ucieczki. Taki neuron staje się bezużyteczny. Mimo tej wady, ta funkcja aktywacji jest bardzo popularna przede wszystkim przy problemach rozpoznawania obrazu. Z kolei softmax jest rozszerzeniem sigmoidu o szacowanie prawdopodobieństwa dla więcej niż dwóch klas. To znaczy, że przy klasyfikacji binarnej mogą być stosowane zamiennie, ale w innych przypadkach już tylko softmax.[32]

	Kategoria 0.	**Kategoria 1.**	**Kategoria 2.**
Obserwacja $X_1 = y_1$	$P(Kat.0 \mid X_1)$	$P(Kat.1 \mid X_1)$	$P(Kat.2 \mid X_1)$
Obserwacja $X_2 = y_2$	$P(Kat.0 \mid X_2)$	$P(Kat.1 \mid X_2)$	$P(Kat.2 \mid X_2)$
Obserwacja $X_3 = y_3$	$P(Kat.0 \mid X_3)$	$P(Kat.1 \mid X_3)$	$P(Kat.2 \mid X_3)$
Obserwacja $X_4 = y_4$	$P(Kat.0 \mid X_4)$	$P(Kat.1 \mid X_4)$	$P(Kat.2 \mid X_4)$

Tabela 1. Macierz wyjściowa transformacji softmax dla 4 obserwacji i 3 możliwych kategoriach. Źródło: Lazy Programmer Inc.

Powyższa tabela ukazuje schemat obliczania prawdopodobieństwa dla każdej zmiennej i możliwych kategorii. Przykładowo dla obserwacji X_1 prawdopodobieństwo przynależności zmiennej celu y_1 do kategorii 0 będzie wynosiło $P(Kat.0 \mid X_1)$. Następnie wektory są porównywane z wektorami rzeczywistymi.

Predykcje			Rzeczywiste wartości		
0.2	0.7	0.1	0	1	0
0.4	0.3	0.3	0	0	1

Tabela 2. Przykład prawidłowego (zielony) i nieprawidłowego (czerwony) przyporządkowania do kategorii. Źródło: Lazy Programmer Inc.

[32] Jadon, Shruti. "Introduction to Different Activation Functions for Deep Learning" Medium.com. 2018.

2. Proces uczenia w sieciach neuronowych

Idea trenowania sieci neuronowych w pojęciu ogólnym polega na modyfikacji wag, które zwykle na samym początku wybierane są losowo. Proces ten opiera się na zastosowaniu algorytmu propagacji wstecznej (*backpropagation*), który wbrew nowej nazwie, nie różni się koncepcyjnie od metody znanej z regresji liniowej i logistycznej. Właściwie jest on metodą najszybszego spadku zastosowanej na funkcji straty w odniesieniu do wszystkich wag w sieci. Jest jedynie bardziej skomplikowany ze względu na złożenia funkcji. Co ważne, propagacja wsteczna stosowana jest we wszystkich rodzajach sieci, nie tylko skierowanych, a nawet innych metodach uczenia maszynowego.[33]

Działanie całego mechanizmu można zawrzeć w kilku punktach:

1) Pseudolosowe zainicjowanie wag dla sieci.
2) Obliczenie wartości predykcyjnej dla obserwacji.
3) Wykonanie propagacji wstecznej i znalezienie gradientu funkcji straty.
4) Znaleziony gradient wykorzystywany jest do optymalizacji wag (np. za pomocą SGD i danego parametru uczenia).
5) Powtarzanie aż do momentu, w którym wynik nie będzie ulegał poprawie.[34]

Przy omawianiu regresji liniowej jako funkcję kosztu wybrano błąd średniokwadratowy, który jest często stosowaną miarą w problemach regresyjnych. Natomiast dla klasyfikacji zwykle stosuje się entropię krzyżową (*cross-entropy*):

$$L = -\sum_{n=1}^{N} (t_n \log y_n + (1 - t_n) \log(1 - y_n)) \tag{14}$$

Gdzie t_n oznacza rzeczywistą wartość, a y_n predykcję. W przypadku wielu kategorii (*categorical cross-entropy*) można ją zapisać w postaci ujemnego logarytmu wiarygodności (*negative log-likelihood*):

$$L = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \log y_{nk} \tag{15}$$

[33] Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. "Deep Learning". MIT Press (www.deeplearningbook.org), 2016.

[34] Becker, Dan. "Deep Learning in Python". Datacamp.com. 2017.

W stosowaniu propagacji wstecznej należy pamiętać, że nie minimalizujemy bezpośrednio funkcji straty, ponieważ znajduje się ona w relacji do warstwy wyjściowej, która z kolei jest zależna od warstwy ukrytej. Kolejny przykład odnosi się do sieci z jedną warstwą ukrytą. Do rozwiązania tego problemu stosuje się wzór na pochodną funkcji złożonej (16) (*chain rule*).

$$J = f(z);\ z = g(y); y = h(x)$$
$$\frac{dJ}{dx} = \frac{dJ}{dz}\frac{dz}{dy}\frac{dy}{dx} \tag{16}$$

Gdzie za odpowiednie zmienne możemy postawić prezentowane wcześniej wzory kolejno na transformacje wewnątrz warstw ukrytych (17), softmax na warstwie wyjściowej (18) oraz funkcję straty (19). Tą ostatnią dla ułatwienia zmodyfikowano usuwając minus, co zmienia jedynie optymalizację z minimalizacji na maksymalizację (*gradient ascent*).

$$z = \sigma(W^T x + b) \tag{17}$$
$$y = softmax(V^T z + c) \tag{18}$$
$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \log y_{nk} \tag{19}$$

Gdzie celem jest aktualizacja wag V, W. Następnie wnętrze funkcji aktywacji i softmax przypisano do zmiennych (20).

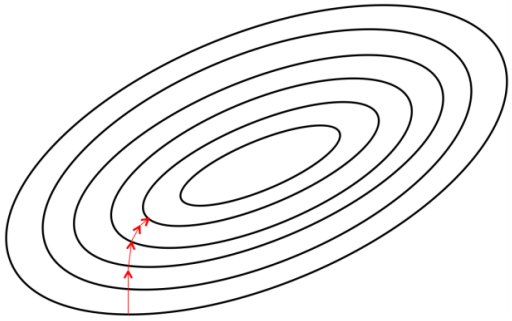
$$\alpha = W^T x + b$$
$$a = V^T z + c \tag{20}$$

Jednak obliczenie pochodnej złożenia funkcji w tym przypadku nie wystarczy, dlatego konieczne jest uwzględnienie pochodnej zupełnej od parametrów wag. Ostatecznie należy obliczyć gradient funkcji straty:

$$\frac{dJ}{dW_{dm}} = \sum_{k=1}^{K}\sum_{n=1}^{N}\sum_{k'=1}^{K} \frac{dyJ_{nk'}}{dy_{nk'}}\frac{dy_{nk'}}{da_{nk}}\frac{da_{nk}}{dz_{nm}}\frac{dz_{nm}}{d\alpha_{nm}}\frac{d\alpha_{nm}}{dW_{dm}} \tag{21}$$

Podobny mechanizm należy zastosować dla wyrazu wolnego. Warto wspomnieć, że rozmiar równania rozrasta się wraz ze zwiększaniem ilości warstw sieci neuronowych. Jednak dzięki właściwości rekursywnej, tj. każdy kolejny krok wstecz opiera się na kroku go poprzedzającym (tzw. parametr δ), algorytm jest oszczędny i efektywny.[35]

[35] Lazy Programmer (pseudonim). "Data Science: Deep Learning in Python". Udemy.com. 2019.



Rysunek 14. Wizualizacja działania metody najszybszego spadku. Źródło: Fundamentals of Deep Learning.

Jak wspomniano wcześniej, gradient przekazywany jest optymalizatorowi. Poza SGD istnieją jeszcze jego dalsze modyfikacje. Szczególnie efektywny jest ADAM, który jest połączeniem dwóch innych metod. Pierwszą z nich jest momentum, oparte na wykładniczo ważonej średniej ruchomej:

$$S_t = \alpha g + \sum (1-\alpha)^n S_{t-n}$$
$$W = W - \eta S_t \tag{22}$$

W algorytmie wartość kolejnego kroku obliczana jest na podstawie znanej z klasycznej wersji SGD pochodnej g, ale z uwzględnieniem wartości poprzednich kroków S_{t-n}. Tym zmiennym nadane są wagi za pomocą stałej α, która najczęściej przyjmuje wartość 0.9. W oznacza wartość aktualizowanej wagi. Kolejną modyfikacją jest *RMSProp*:

$$S_t = \alpha g + \sum (1-\alpha) S_{t-n}^2$$
$$W = W - \frac{\eta g}{\sqrt{S_t}} \tag{23}$$

Zasada działania jest ta sama, jednak wartości kroków podnoszone są do kwadratów, co zmienia równanie wag względem poprzedniej wersji. Dzięki temu ADAM jest bardzo szybkim i skutecznym optymalizatorem, który zwykle jest wykorzystywany do trenowania sieci zamiast klasycznego SGD.[36]

3. Przewaga sieci nad innymi metodami uczenia maszynowego

Po objaśnieniu fundamentów głębokiego uczenia w tej części przedstawione zostaną teoria i twierdzenia mówiące o przewadze sieci neuronowych nad klasycznymi metodami uczenia

[36] Howard, Jeremy. et. al. "Practical Deep Learning for Coders v3". course.fast.ai. 2019.

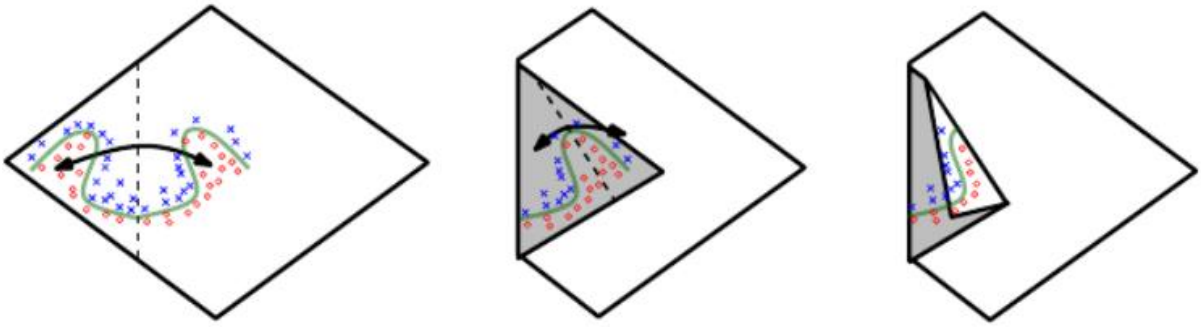
maszynowego jak drzewa decyzyjne i regresje logistyczne. Warto jednak nadmienić, że w niektórych przypadkach klasyczne metody mogą okazać się lepszym wyborem dla mniej złożonych problemów, również dlatego, że zwykle są łatwiejsze w interpretacji.

Twierdzenie o uniwersalnym aproksymatorze (*universal approximation theorem*) mówi, że dowolna sieć skierowana z co najmniej jedną ukrytą warstwą neuronów z np. sigmoidem czy ReLU, jako funkcją aktywacji może z żądaną dokładnością dopasować do danych dowolną funkcję borelowską (ciągłą i nieliniową). Założeniem jest, że sieć będzie zawierała odpowiednią liczbę neuronów w warstwie ukrytej.[37] Jednak nie ma gwarancji, że algorytm będzie w stanie znaleźć odpowiednie parametry danej funkcji, a w przeciwnym wypadku może wystąpić problem przeuczenia. Liczba potrzebnych neuronów też nie jest znana. Innymi słowy nie istnieje uniwersalny przepis na wytrenowanie sieci z odpowiednią generalizacją dla nowych danych.[38] Twierdzenie Barrona odnosi się do powyższego, ustanawiając granicę maksymalnego błędu aproksymacji na poziomie $O(1/n)$, gdzie n oznacza ilość neuronów w warstwie ukrytej jednowarstwowej sieci. Mając nałożone powyższe ograniczenie na wartość błędu, który jest zależny od ilości wykorzystanych węzłów, można stwierdzić, że sieci nie dotyczy tzw. klątwa wielowymiarowości (*curse of dimensionality*), która polega na małej skuteczności klasycznych metod uczenia maszynowego przy dużej ilości zmiennych (wymiarów).[39]

Powyższe twierdzenia odwołują się do struktur o pojedynczej warstwie ukrytej. Z kolei twierdzenie Montufara odnosi się do relacji pomiędzy głębokością sieci a jej zdolnością do aproksymacji. Niniejszym udowodniono, że sieć może dopasowywać przedziałami liniową funkcję

z wykładniczo rosnącą dokładnością wraz ze wzrostem liczby warstw ukrytych.

[37] Cybenko, George. "Approximation by superpositions of a sigmoidal function". Mathematics of Control, Signals, and Systems. 1989.
[38] Nielsen, Michael. "Neural Networks and Deep Learning". Determination Press. 2015.
[39] Barron, Andrew. "Universal Approximation Bounds for Superpositions of a Sigmoidal Function". IEEE Transactions on Information Theory, 39(3). 1993.



Rysunek 15. Zobrazowanie działania kolejnych warstw sieci neuronowych na przykładzie narysowanej na kartce funkcji. Źródło: deeplearningbook.org.

Powyższy schemat obrazuje mechanizm tego procesu. Każdy neuron w warstwie ukrytej wskazuje punkt zgięcia hiperpłaszczyzny w celu stworzenia lustrzanego odbicia danego rejonu funkcji (nie musi obejmować jej całości). Sekwencje takich zgięć pozwalają efektywniej odwzorować zachodzące w danych prawidłowości.[40] Na tej podstawie również można stwierdzić, że im bardziej dana funkcja jest symetryczna, tym dokładniejszy będzie zbudowany na niej model.[41]

Ze statystycznego punktu widzenia, przy klasycznych algorytmach zakłada się pewien kształt aproksymowanej funkcji. Zatem przewagą sieci może być brak takich założeń, ponieważ wiadomo jedynie, że aproksymanta będzie złożeniem kilku prostszych funkcji, których rezultaty przekazywane są w kolejnych krokach do następnych funkcji.[42]

4. Konwolucyjne sieci neuronowe

Zwykłe głębokie sieci neuronowe nie zdają egzaminu w przypadku analizy obrazu, ponieważ standardowy model skierowany łączy każdy piksel z każdym neuronem. To oznacza, że przykładowo dla zdjęć o rozmiarze 28×28 otrzymamy 784 wagi na połączeniach dla jednego węzła. W przypadku rozmiaru 200×200 wag będzie już 120 000. Ze względu na fakt, że piksele zazwyczaj nie są niezależne od siebie, tj. są skorelowane przestrzennie, konieczne jest zastosowanie operacji splotu, inaczej konwolucji, w celu efektywnej analizy danych. Wagi nadawane są dla każdego elementu w filtrze, co względem zwykłej sieci oznacza znaczną redukcję

[40] Montúfar, Guido. Pascanu, Razvan. Cho, Kyunghyun. Bengio, Yoshua. "On the number of linear regions of deep neural networks". NIPS'2014. 2014.
[41] Kamiński, Bogumił. Pankratz, Bartosz. "Startystyczne Reguły Decyzyjne". Szkoła Główna Handlowa w Warszawie. 2019.
[42] Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. "Deep Learning". MIT Press (www.deeplearningbook.org), 2016.

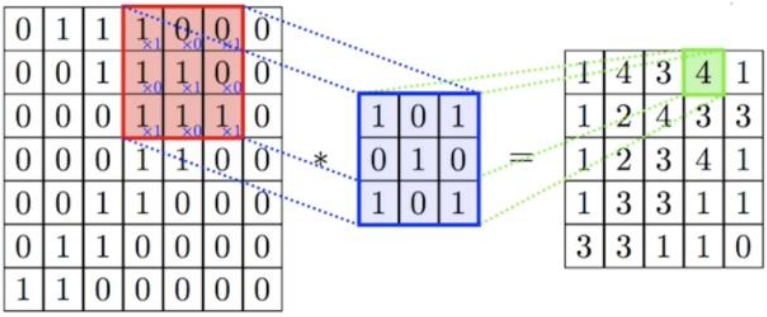
ich ilości. Oczywiście splot nie jest stosowany tylko w głębokim uczeniu, ponieważ wszelkie efekty i modyfikacje stosowane w programach do obróbki fotografii się na nim opierają. Też zdjęcia nie są jedynym obszarem jego zastosowania. Mowa tutaj o aplikacjach sygnałowych, np. do wideo, dźwięków, danych sejsmologicznych i genomicznych.

i. Operacja splotu

Najprościej rzecz ujmując, konwolucją nazywamy operację, która przy wykorzystaniu dwóch oddzielnych funkcji $f(t)$ i $g(t)$ oraz operacji dodawania i mnożenia zwraca trzecią funkcję (znak $*$ oznacza operację splotu):

$$s(t) = (f * g)(t) = \int_0^t f(x)g(t-x)dx \tag{24}$$

Zmienna x jest zmienną tymczasową i znika w trakcie rozwiązywania całki. W przypadku obrazów działanie tej operacji zawiera trzy elementy: daną wejściową (I), filtr (K - *kernel*), daną wyjściową.[43]

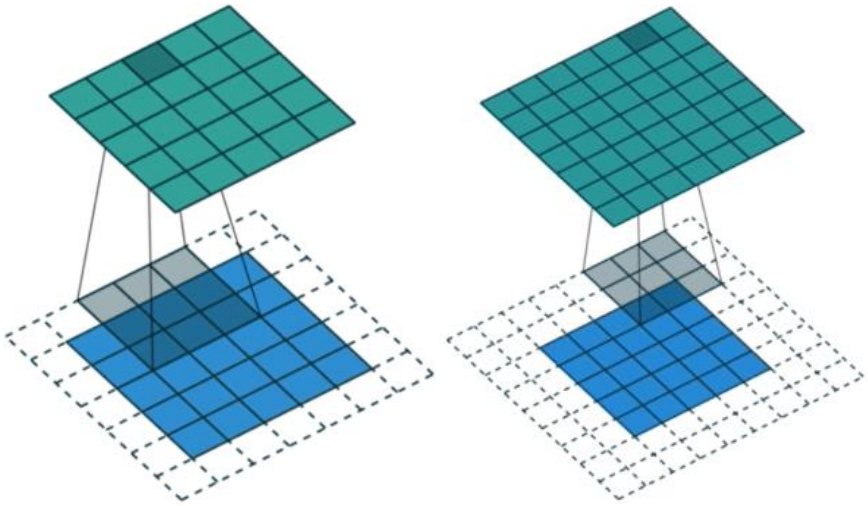


Rysunek 16. Operacja splotu na przykładzie zdjęcia (pikseli). Od lewej: dane wejściowe, filtr, dane wyjściowe. Źródło: Lazy Programmer Inc.

Okno filtra przeskakuje o jeden piksel od lewej do prawej transformując wybrane wartości na dane wyjściowe. Przykładowo dla zaznaczonego obszaru mnożymy każdy element macierzy wejściowej przez odpowiadający mu element filtra. Na czterech pozycjach zostanie zwrócona wartość 1, a na pozostałych 0. Suma elementów zatem będzie równa 0. Zwracaną macierz nazywa się mapą cech (*feature map*), która to zawiera lokalizacje łączące cechy reprezentowane przez nałożony filtr. Można modyfikować wielkość obrazu wyjściowego poprzez dodawanie wyzerowanych pikseli (*padding*) dookoła macierzy wejściowej. Przykładowo jeden dodatkowy

[43] Sopyła, Krzysztof. ”Przetwarzanie obrazu z wykorzystaniem splotu funkcji”. Ksopyla.com. 2016.

rząd zer (*half-padding*) będzie oznaczał taki sam rozmiar macierzy. Dodanie dwóch rzędów (*full-padding*) pozwoli na dosięgnięcie każdego piksela z warstwy wejściowej.



Rysunek 17. Wizualizacja przykładów dla half-padding oraz full-padding. Źródło: Lazy Programmer Inc.

Można zmienić również długość kroku przesuwania filtra po macierzy (*stride*). Ostatecznie wielkość obrazu wyjściowego można opisać wzorem:

$$Size = \frac{I - K + 2p}{s} + 1 \tag{25}$$

Gdzie p oznacza dodatkowe rzędy zer, a s rozmiar kroku.[44]

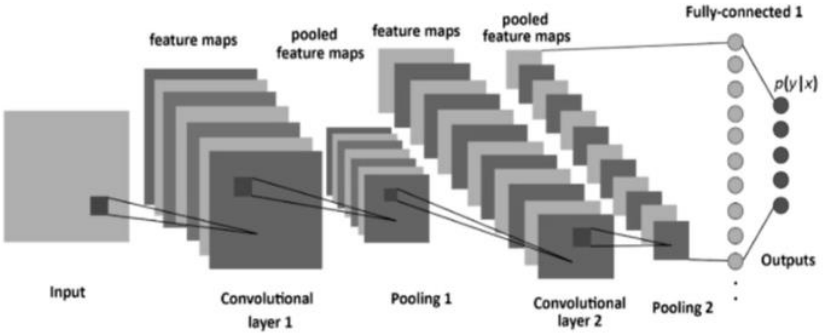
ii. Konstrukcja sieci

Oprócz interakcji przestrzennych, decydujących o mniejszej ilości przekazywanych parametrów, sieci mają jeszcze dwie właściwości decydujące o ich skuteczności. Kolejną są połączone wagi (*tied weights*). Oznacza „dzielenie się" wagami przez wszystkie neurony w ramach jednej mapy cech. Oznacza to, że zamiast tworzyć zbiór parametrów i optymalizować go oddzielnie dla każdego regionu zdjęcia, tworzony jest tylko jeden zestaw. To z kolei tworzy zjawisko ekwiwariancji, to znaczy, że gdy dane wejściowe się zmieniają, to tak samo zmienia się rezultat. W przypadku rozpoznawania obrazów będzie to zależność kształtu zmapowanych cech (np. krawędzi) od wprowadzonego oryginału. Zwykle konstrukcja jednej warstwy ukrytej sieci neuronowej obejmuje 3 etapy:

1) przeprowadzenie operacji splotu,
2) użycie wybranej funkcji aktywacji (zwykle ReLU),

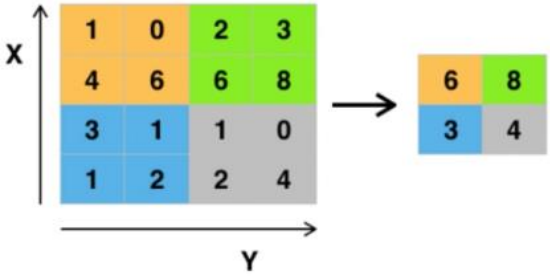
[44] Rokem, Ariel. "Convolutional Neural Networks for Image Processing". Datacamp.com. 2018.

3) użycie warstwy łączącej (*pooling*),
4) redukcja wymiaru danych i użycie warstwy gęstej (*dense layer*).



Rysunek 18. Przykład prostej architektury konwolucyjnych sieci neuronowych. Źródło: Lazy Programmer Inc.

Oznacza to, że po zaaplikowaniu funkcji aktywacji, w warstwie łączącej, mapa cech jest poddawana operacji próbkowania (*downsampling*), która na celu redukcję ilości parametrów za pomocą wybranej funkcji (np. maksymalizacja, uśrednianie). Wymaga deklaracji wielkości okna, gdzie np. okno 2×2 redukuje wielkość do $^1/_4$ oryginalnej mapy cech. Dzięki wymienionym wcześniej właściwościom oraz wykorzystaniu warstwy łączącej sieć staje się niejako niewrażliwa na zmiany położenia i kształtu poszukiwanych obiektów, co oznacza lepszą generalizację.



Rysunek 19. Przykład działania warstwy łączącej - max pooling. Źródło: Lazy Programmer Inc.

Na samym końcu spłaszcza się parametry do postaci jednowymiarowej (wektorowej), a następnie wprowadza się do standardowej sieci skierowanej z maksymalną ilością połączeń w celu dokończenia predykcji. Zasada odkrywania zależności jest podobna jak wcześniej, czyli poziom szczegółowości reprezentacji zwiększa się wraz z głębokością sieci. Od namierzania prostych krawędzi przechodzi się aż po złożone kształty jak np. twarze. [45]

[45] Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. "Deep Learning". MIT Press (www.deeplearningbook.org), 2016.

iii. Metody regularyzacji

Można wyróżnić kilka sposobów na unikanie przeuczenia w modelach sieci neuronowych: zwiększanie ilości obserwacji, zmniejszanie złożoności architektury (liczby warstw i neuronów) lub ilości zmiennych. Pierwsza opcja często jest poza kontrolą analityka, natomiast spłycając sieci i rezygnując z niektórych wymiarów traci się bardzo duży potencjał. Dlatego wyróżnia się kilka metod pozwalających na efektywniejszą kontrolę błędu na zbiorze testowym. Istotnym jest, że poniższe metody zwykle można zaaplikować do różnych rodzajów sieci, nie tylko konwolucyjnych.

Pierwszą jest metoda nakładania kary *L2*, zwana również metodą rozpadu wag (*weight decay*). Jej idea polega na pomniejszaniu wag aż do wartości bliskich zeru. Do przykładowej funkcji straty należy dodać sumę kwadratów wszystkich wag pomnożoną przez arbitralnie wybrany parametr (wd).

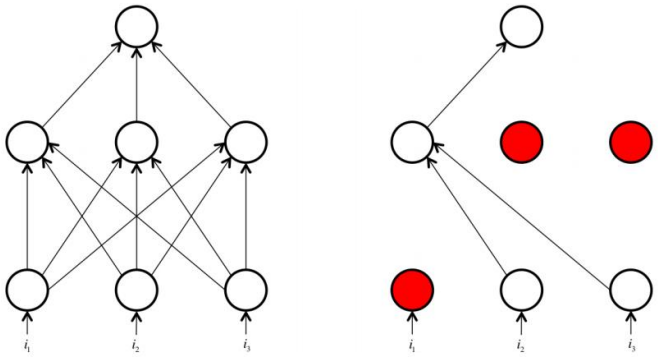
$$L = -\sum_{n=1}^{N}(t_n \log y_n + (1 - t_n)\log(1 - y_n)) + wd \sum_{w} w^2 \qquad (26)$$

Biorąc pod uwagę, że do aktualizacji wag wykorzystujemy gradient z kosztu, okazuje się, że należy również obliczyć gradient części regularyzacyjnej i będzie wynosił $2wd \cdot w$. Oznacza to, że dla każdej porcji danych pomniejszamy wagi o konkretny procent. Nazewnictwo różni się dla powyższych form. Kiedy mówimy o dodawaniu sumy kwadratów wag do funkcji straty, jest to regularyzacja typu *L2*, natomiast formę gradientową, gdzie jest operacja odejmowania, nazywamy rozpadem wag.[46]

Powszechnie stosowany jest *dropout*, który został odkryty w 2014 roku. Ta metoda polega na wprowadzeniu prawdopodobieństwa (p) zachowania neuronu jako aktywnego elementu sieci lub jego dezaktywacji $(1 - p)$. To generuje wymóg dopasowywania się algorytmu do danych po utracie pewnych informacji zawartych w wyłączonych węzłach, przez co zapobiega uzależnieniu się sieci od konkretnych wag. Innymi słowy, za każdym razem trenowana jest sieć o innej strukturze na różnych częściach zbioru danych. Jeżeli jakaś część sieci staje się zbyt wrażliwa na szum w danych, inne części to zrekompensują. Ponadto struktury zabezpieczone są przed zbytnią korelacją neuronów w ich aktywności, np. jeżeli jeden neuron uczy się poziomych orientacji, inny

[46] Howard, Jeremy. et. al. "Practical Deep Learning for Coders v3". course.fast.ai. 2019

będzie specjalizował się w pionowych. *Dropout* działa zarówno dla propagacji postępującej jak i wstecznej.[47,48]



Rysunek 20. Prezentacja pełnej sieci oraz z wyłączanymi neuronami. Źródło: Fundamentals of Deep Learning

Następną metodą regularyzacji jest *batch normalization* z 2015 roku. W skrócie polega ona na przeskalowaniu danych wyjściowych z konkretnej warstwy, tak aby średnia arytmetyczna zawsze wynosiła 0, a odchylenie standardowe 1 w każdej porcji danych treningowych. Zatem *BN* jest kolejną warstwą łączącą. Rozwiązuje ona problem zróżnicowania rozkładów prawdopodobieństwa w warstwach pomiędzy różnymi częściami zbioru. Algorytm działa następująco:

1) oblicza się średnią arytmetyczną z wartości zwracanej przez funkcje aktywacji,
2) podobnie jak powyżej, oblicza się odchylenie standardowe,
3) stosuje się normalizację:

$$\widehat{x_i} = \frac{x_i - \bar{x}}{\sigma} \tag{27}$$

4) tworzy się funkcje zawierającą wyraz wolny oraz współczynnik kierunkowy, które poddaje się optymalizacji w procesie uczenia.[49]

$$y_i = \gamma \widehat{x_i} + \beta \tag{28}$$

Ostatnią metodą wpływającą na dopasowanie do danych jest augmentacja danych wejściowych, która zostanie zaprezentowana na przykładach w ostatnim rozdziale.

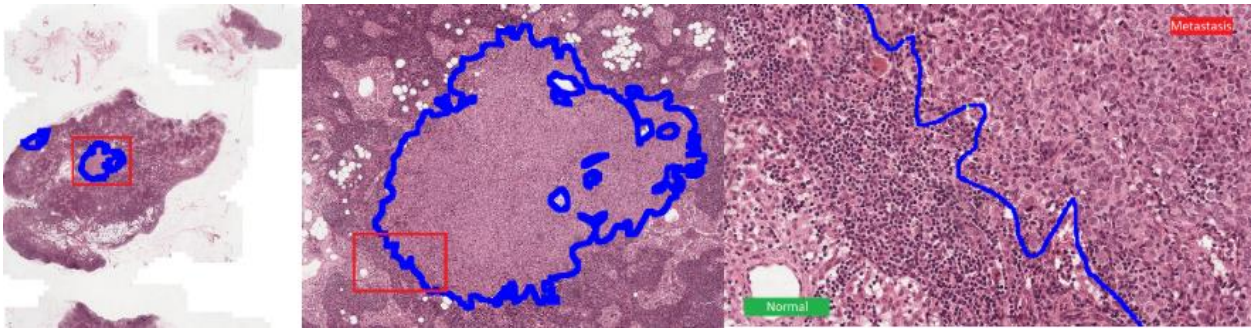
[47] Becker, Dan. "Deep Learning in Python". Datacamp.com. 2017.
[48] Howard, Jeremy. et. al. "Practical Deep Learning for Coders v3". course.fast.ai. 2019
[49] Ioffe, Sergey, Szegedy, Christian. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". arXiv:1502.03167. 2015.

V. Analiza właściwa

W tym rozdziale zostanie przedstawiony proces tworzenia i walidacji modelu predykcyjnego mającego za zadanie klasyfikowanie zdjęć histopatologicznych. Jest to problem klasyfikacji binarnej, gdzie celem jest identyfikacja zdjęć, na których widnieją nowotworowe komórki przerzutowe. Do obliczeń wykorzystywano instancję maszyny wirtualnej w chmurze Google Cloud Platform z jednym procesorem graficznym NVIDIA Tesla P4, co pozwalało na osiągnięcie prędkości trenowania około jednej epoki na godzinę. Przy użyciu biblioteki FastAI (PyTorch) wykonano ponad 50 modeli z różnymi konfiguracjami parametrów oraz architektur, z których w dalszej części pracy zostaną zaprezentowane wybrane z nich.

1. Analiza eksploracyjna i przygotowanie danych

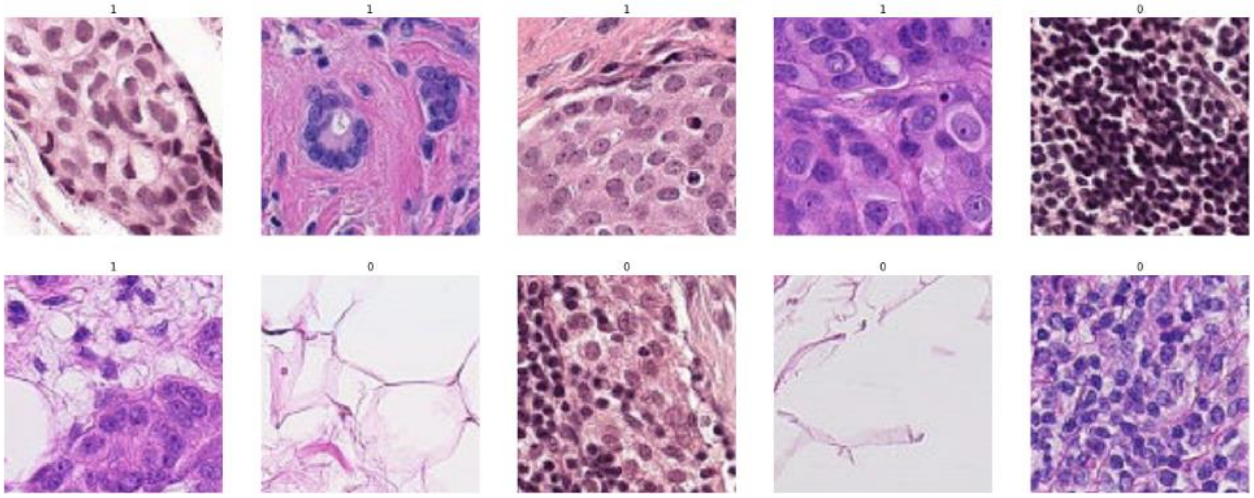
Zbiór danych zawiera zdjęcia tkanek pozyskanych operacyjnie lub za pomocą biopsji, które następnie były poddane spłaszczeniu do grubości mikrometrów oraz barwieniu hematoksyliną i eozyną (czemu zawdzięczają swój różowo-fioletowy kolor). Dzięki technologi digitalizowania szklanych próbek, otrzymywane są bardzo szczegółowe zdjęcia o rozmiarach nawet 200000 × 100000 pikseli. Te z kolei zostają podzielone na mniejsze części. W przypadku użytego zestawu jest to 96 × 96 pikseli.



Rysunek 21. Przykład zdjęć histopatologicznych: (od lewej) w niskiej, średniej, wysokiej rozdzielczości. Źródło: camelyon17.grand-challenge.org

Wykorzystywany zbiór danych pochodzi z konkursu Kaggle, który został udostępniony w formie podzielonej na zbiór testowy (57 458) i treningowy (220 025). Ze względu na zasady wyzwania, użytkownicy nie mają dostępu do prawidłowych oznaczeń zbioru testowego. Kaggle API po załadowaniu pliku CSV z rezultatami modelu automatycznie zwraca jedynie miarę pola pod krzywą *ROC* (*AUC*). Analiza zbioru treningowego wskazuje na pewną dysproporcję pomiędzy liczebnością kategorii: 130908 negatywnych oraz 89117 pozytywnych. Przynależności do grupy

dyktowane są obecnością nowotworu w centralnym obszarze zdjęcia o rozmiarze 32 × 32 pikseli.[50] Co widać po poniższych zdjęciach, kategoryzacja wykonana przez niewykwalifikowanego medycznie badacza może być problematyczna.

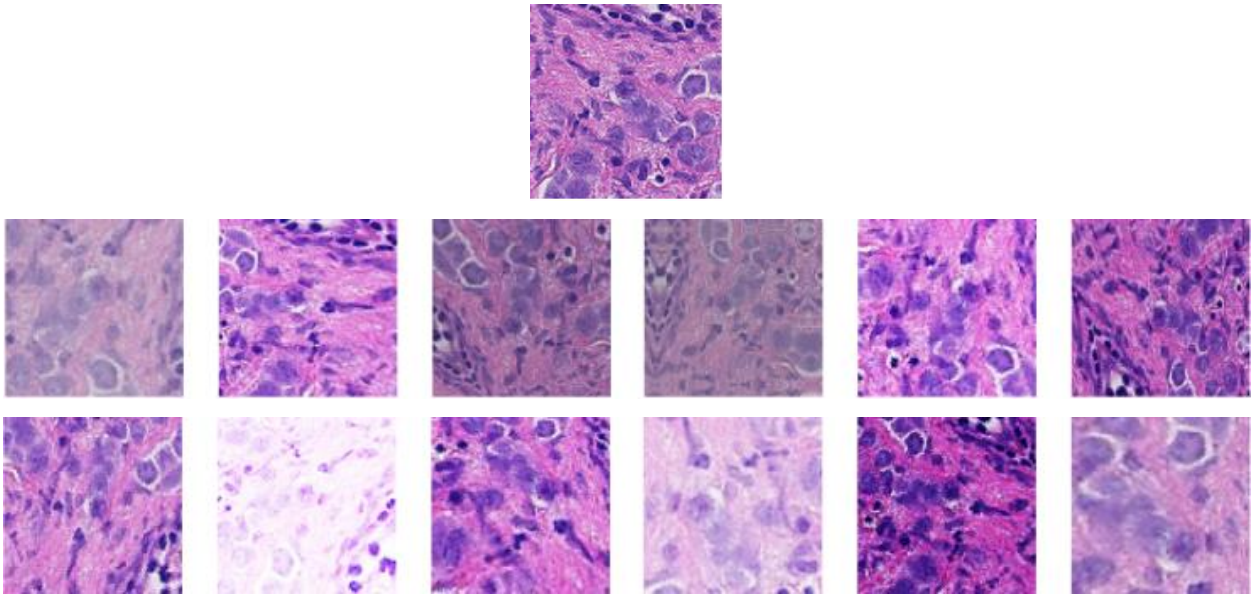


Rysunek 22. Przykładowe zdjęcia z treningowego zbioru danych z etykietami. Źródło: opracowanie własne.

Kolejne testy modelu wykazały, że przeskalowanie danych wejściowych do rozmiaru 224 × 224 poprawiło rezultaty. Istotnym krokiem była augmentacja, czyli transformacja danych na wybrane sposoby. Pozwoliła ona na regularyzację modelu bez wydłużania czasu trenowania. Zamiast przepuszczania przez model tych samych zdjęć, z wybranym prawdopodobieństwem, wykonywane są na nich konkretne modyfikacje. Metodą prób i błędów zdecydowano się na poniższe opcje, gdzie wszystkie z wyjątkiem pierwszej aplikowano z prawdopodobieństwem 0.75:

1) przewrócenie obrazu horyzontalne i wertykalne z prawdopodobieństwem 0.5
2) rotacja od -80 do 80 stopni,
3) przybliżenie od 1. do 1.75,
4) zmiana jasności i kontrastu na poziomie 0.2, gdzie 0. oznacza czarny obraz, a 1. Biały,
5) zmiana perspektywy zdjęcia na poziomie 0.3,
6) dodatkowa zmiana jasności (pomiędzy 0.3 i 0.9) i kontrastu (0.5 i 2.),
7) rozmazanie obrazu z intensywnością na poziomie 0.01.

[50] "Histopathologic Cancer Detection". kaggle.com. 2019.



Rysunek 23. Porównanie oryginalnego zdjęcia (rząd pierwszy) sklasyfikowanego pozytywnie z zaaplikowanymi wybranymi modyfikacjami. Źródło: opracowanie własne

Ponadto zdecydowano się na znormalizowanie danych przy użyciu statystyk (odchyleń standardowych oraz średnich) pochodzących ze zbioru ImageNet, co nieznacznie poprawiło moc predykcyjną.

2. Uczenie transferowe i opis wybranych architektur sieci

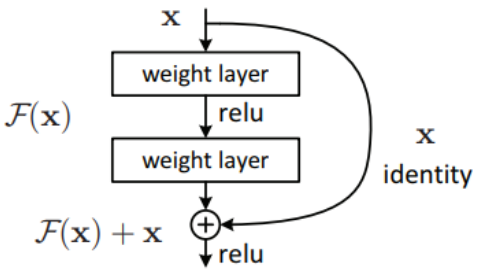
W analizie obrazu powszechne jest stosowanie wytrenowanych wcześniej sieci na większych i bardziej zróżnicowanych zbiorach danych. Zwykle są to modele trenowane na zbiorze danych ImageNet (ILSVRC 2012) zawierającym 1000 klas i prawie 1.3 miliona zdjęć. Dzięki temu takie struktury są w stanie rozpoznawać szeroki wachlarz kształtów i obiektów. W rezultacie prawie każdy model wytrenowany z użyciem transferu wiedzy jest lepszy od modelu trenowanego od zera, a zwłaszcza wtedy kiedy badany zbiór jest małych rozmiarów.[51] Aplikacja tego rozwiązania obejmuje wa elementy sieci i dwa etapy procesu. Sieć dzieli się na korpus zawierający najważniejsze warstwy konwolucyjne oraz głowę zawierającą warstwy dopasowujące korpus do konkretnego zadania. W pierwszym etapie trenowana jest tylko głowa, a korpus pozostaje „zamrożony". W drugim, opcjonalnym etapie rozmraża się sieć i dopasowuje się wagi na całości.[52]

[51] Kornblith, Simon. Shlens, Jonathon. Le, Quoc. "Do Better ImageNet Models Transfer Better?". arXiv:1805.08974. 2019.
[52] Howard, Jeremy. et. al. "Practical Deep Learning for Coders v3". course.fast.ai. 2019.

W niniejszej analizie dostosowano niżej opisane architektury do zadania klasyfikacji binarnej oraz zrezygnowano z drugiego etapu ze względu na pogorszenie mocy predykcyjnej.

i. ResNet

Prawdopodobnie najbardziej popularną siecią używaną do uczenia z transferem jest ResNet (*deep residual neural network*), który zajął pierwsze miejsce w konkursie rozpoznawania obrazu ILSVRC 2015. Jest on rezultatem prób poprawienia działania głębokich sieci, które bardzo często osiągały zbyt słabą generalizację i w rezultacie miały gorszy wynik od płytszych sieci. Polega ona na przekazywaniu informacji do warstwy z dalszych warstw poprzedzających, a nie tylko z nią poprzedzającej. Takie połączenie „na skróty" rozwiązuje problem nasycenia gradientu.



Rysunek 24. Jedna z części składowych sieci ResNet. Źródło: Deep Residual Learning for Image Recognition.

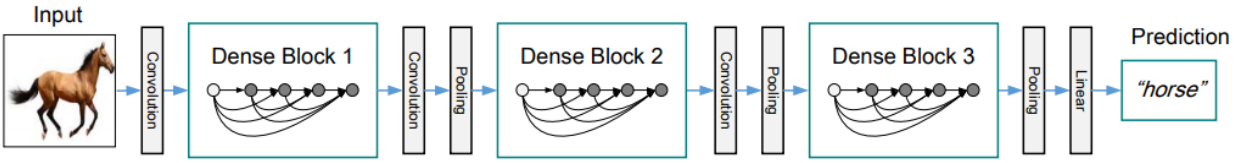
Przykładowa sieć z 34 warstwami zakłada wykorzystanie filtrów o rozmiarze 3×3 z długością kroku przesuwania o wartości 2. Każda warstwa konwolucyjna obejmuje operację normalizacji i wykorzystuje ReLU jako funkcję aktywacji. Warstwy łączące (kolejno *max pooling* i *average pooling*) znajdują się na początku i końcu sieci.[53,54] ResNet występuje w różnych rozmiarach. W tej analizie sprawdzono działanie wersji z 50 i 101 warstwami.

ii. DenseNet

Architektura DenseNet opisana w 2018 roku jeszcze bardziej zwiększa możliwości pogłębiania sieci i redukcji problemu nasycenia, a ponadto pozwala poprawić przekazywanie informacji. Jest to poniekąd rozszerzenie sieci typu ResNet, ponieważ autorzy połączyli wszystkie warstwy z takimi samymi rozmiarami map cech ze sobą w danych częściach. Ponadto w tym przypadku w punktach łączących cechy nie są dodawane do siebie, lecz wiązane ze sobą. Oprócz głównych elementów konwolucyjnych z filtrami 3×3 i krokiem równym 2, wykorzystywane są elementy normalizujące i ReLU. Zestawy takich warstw łączone są w bloki (*dense blocks*), pomiędzy

[53] He, Kaiming. et. al. "Deep Residual Learning for Image Recognition". Microsoft Research. 2015.
[54] Ruiz, Pablo. "Understanding and visualizing ResNets". Medium.com. 2018.

którymi jest dodatkowa warstwa konwolucyjna oraz łącząca (na początku *max pool*, następnie *average pool*).



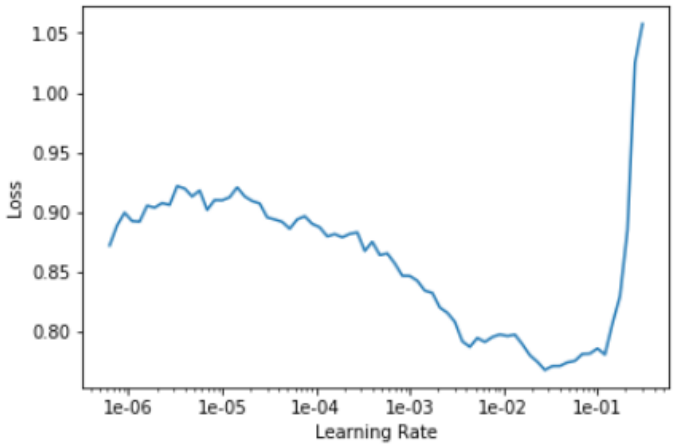
Rysunek 25. Uproszczona wizualizacja architektury DenseNet z podziałem na bloki i warstwy łączące. Źródło: Densely Connected Convolutional Networks.

Ten typ sieci również występuje w różnych wersjach.[55] Tutaj przetestowano głębokości na poziomach 169 oraz 201.

3. Omówienie rezultatów

Proces budowy i trenowania modeli zostanie zaprezentowany na przykładzie najlepszego modelu zbudowanego na najgłębszej z testowanych architektur – DenseNet201. Ze zbioru treningowego losowo wydzielono 20% (44 005) danych na poczet walidacji. Jako optymalizator wybrano ADAM, a ponadto ustawiono parametry dla rozpadu wag na poziomie 0.01 oraz prawdopodobieństwo 0.5 dla *dropout*. Aby zmieścić się w limicie pamięci, procesowano 32 zdjęcia w danym momencie (*batch* size). Do modelu standardowo dołączono warstwy łączące (*pooling*), normalizujące, ReLU, *dropout* oraz warstwą wyjściową z dwoma neuronami. Jako metrykę kontrolną wybrano błąd na zbiorze (*error rate*). Trenowanie przeprowadzano w trzech blokach: najpierw 3 epoki, następnie kolejne 3, i na końcu jeszcze dwie. Pomiędzy blokami dostosowywano (zmniejszano) parametr uczenia, który z kolei wybierano sugerując się wykresem tworzonym po rozpoczęciu testowej epoki, którą szybko przerywano. W rezultacie uwidocznione zostały rejony, gdzie koszt spada najmocniej.

[55] Huang, Gao. et. al. "Densely Connected Convolutional Networks". arXiv:1608.06993. 2018.



Rysunek 26. Wygenerowany wykres służący do wyboru parametru uczenia przed rozpoczęciem trenowania. Źródło: opracowanie własne.

Pomiędzy każdym blokiem zapisywano rezultat i generowano predykcje, aby sprawdzić optymalną ilość epok. Ze względu na zasady wyzwania, użytkownicy nie posiadają dostępu do listy prawidłowych predykcji na części testowej, a Kaggle API zwraca jedynie metrykę pola pod krzywą ROC (AUC). Dlatego AUC na zbiorze testowym jest głównym miernikiem działania modelu. Miara ta jest najpopularniejsza w klasyfikacji binarnej i jest ściśle powiązana z macierzą pomyłek. Na osiach ukazane są wartości dwóch miar (odpowiednio y i x):

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{29}$$

$$1 - Specifity = 1 - \frac{True\ Negatives}{True\ Negatives + False\ Positives} \tag{30}$$

Gdzie czułość można interpretować jako procent w jakim obserwowana klasa pozytywna została faktycznie pokryta przewidywaniem pozytywnym. Natomiast specyficzność określa w jakim procencie klasa negatywna została pokryta szacowaniem negatywnym. Te wartości pochodzą z macierzy pomyłek:

		Stan faktyczny	
	Kategorie	1	0
Predykcja	1	*True Positives*	*False Positives*
	0	*False Negatives*	*True Negatives*

Tabela 3. Macierz pomyłek. Źródło: opracowanie własne.

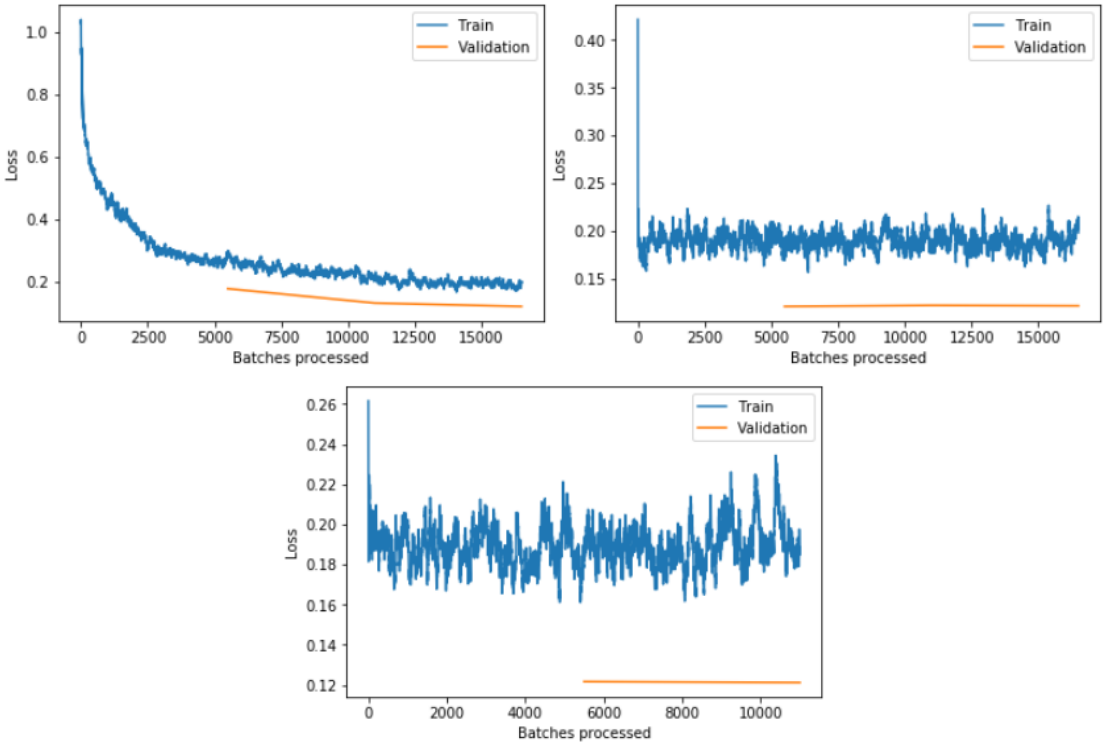
Zatem dla kolejnych bloków wybrano współczynnik uczenia odpowiednio: *1e-03, 1e-07, (1e-07, 1e-06)*. Ostatni zapis oznacza, że warstwy trenowano za pomocą różnych parametrów z danego

przedziału rozłożonych równomiernie między warstwami. Sprawdzono, czy dalsze uczenie poprawi wynik, jednak rezultat był negatywny. Różnica pomiędzy 6 a 8 jest minimalna, więc teoretycznie można było skrócić czas trenowania.

Liczba epok	Parametr uczenia	Wynik na 51% zbioru testowego	Wynik na 49% zbioru testowego
3	*1e-03*	97,11%	96,85%
3	*1e-07*	97,53%	97,08%
2	***(1e-07, 1e-06)***	**97,56%**	**97,08%**
2	*1e-06*	97,53%	96,99%

Tabela 4. Rezultaty różnej długości trenowania sieci dla różnych parametrów. Źródło: opracowanie własne.

Zatem największe zyski osiągnięto na samym początku trenowania. Ukazują to również poniższe wykresy stosunku błędów na zbiorach treningowym i walidacyjnym wykonywanych pomiędzy blokami.



Rysunek 27. Stosunki błędów na zbiorach walidacyjnym i testowym w trakcie uczenia odpowiednio dla 1-3, 4-6, 7-8 epok. Źródło: opracowanie własne.

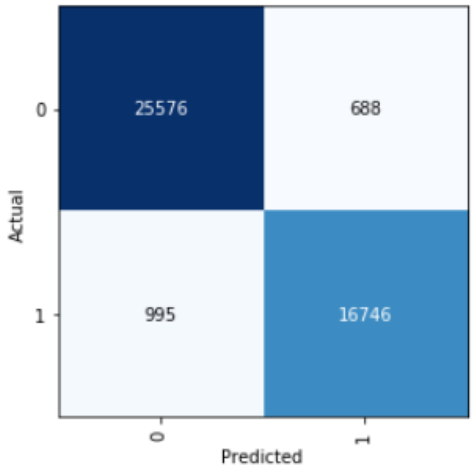
W porównaniu do innych, płytszych architektur model zbudowany na DenseNet201 osiągnął najwyższy wynik. Jak widać rezultat na zbiorze testowym jest przeciwnie proporcjonalny do

rezultatu na zbiorze walidacyjnym. Może to wynikać z faktu, że model zbyt dobrze dopasowuje się do danych treningowych, a następnie słabo generalizuje na zbiór docelowy.

Rodzaj sieci	Wynik na 51% zbioru testowego	Wynik na 49% zbioru testowego	Wynik na zbiorze walidacyjnym
ResNet 50	97,18%	96,83%	**99,27%**
ResNet 101	97,11%	96,92%	99,21%
DenseNet 169	97,15%	97,01%	99,16%
DenseNet 201	**97,56%**	**97,08%**	99,08%

Tabela 5. Porównanie rezultatów dla różnych rodzajów sieci. Źródło: opracowanie własne.

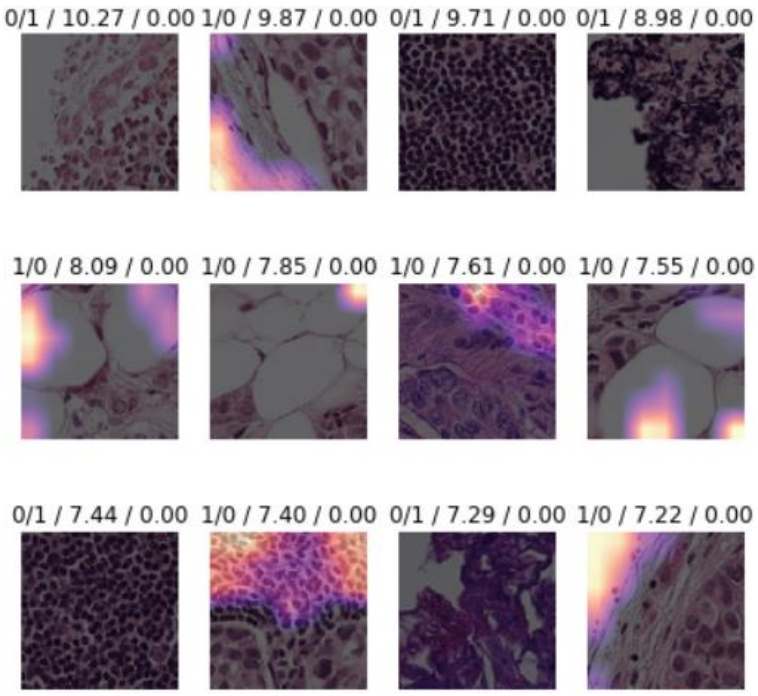
Na podstawie zbioru walidacyjnego wygenerowano macierz pomyłek. Jej kształt wskazuje na większe trudności przy namierzaniu faktycznych przypadków pozytywnych, niż negatywnych. Jednak biorąc pod uwagę rozmiar zbioru, można tą proporcję uznać za stosunkowo zbilansowaną.



Rysunek 28. Macierz pomyłek na zbiorze walidacyjnym dla modelu opartego na DenseNet 201. Źródło: opracowanie własne.

Sprawdzono również największe pomyłki, tj. gdy prawdopodobieństwo przynależności do danej kategorii wynosiło 0 i była to ocena błędna. Podświetlone obszary wskazują na jakiej podstawie model wskazywał na obecność nowotworu. Obrazy wygenerowano na za pomocą metody Grad-CAM[56].

[56] Selvaraju, Ramprasaath. et. al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization". arXiv:1610.02391. 2017.



Rysunek 29. Wyróżnione największe pomyłki modelu. Nad każdym zdjęciem numery wskazują na: predykcję/wartość rzeczywistą/wartość straty/prawodpodobieństwo. Źródło: opracowanie własne.

Porównując ostateczny wynik modelu z rezultatami biorących udział w konkursie, wybrany model znalazłby się wśród pierwszych 100 wyników na 1157 drużyn. Na podstawie zaproponowanych rozwiązań można stwierdzić, że powyższy model mógłby zostać ulepszony za pomocą kilku zabiegów. Pierwszym z nich jest łączenie dwóch lub więcej rodzajów modeli (*ensemble*). Niektórzy manipulowali również funkcją straty i stosowali np. *focal loss.* Trzeci sposób na poprawienie wyników zakłada korzystanie z informacji zewnętrznej, jaką jest przynależność zdjęć zbioru do konkretnego *WSI*, co następnie jest wykorzystywane do podziału zbioru na walidacyjny i treningowy. Według użytkowników takie zabiegi pozwalały osiągnąć AUC w okolicach 98% i więcej.[57] Mimo to, uzyskany wynik przez znacznie prostszy model opisany w niniejszej pracy można zakwalifikować jako dobry.

VI. Podsumowanie

Głębokie uczenie i metody analizy obrazów wciąż są w trakcie bardzo intensywnego rozwoju. Nawet patrząc po zastosowanej tutaj literaturze i poszczególnych datach publikacji prac, można stwierdzić, że nowe rozwiązania pojawiają się bardzo szybko i trudno oszacować, czy przełomowe

[57] "Histopathologic Cancer Detection". kaggle.com. 2019.

odkrycia poprawiające skuteczność systemów sztucznej inteligencji nie pojawią się w przeciągu miesięcy lub lat. Jak wspominano na poprzednich stronach, medycyna jest jednym z najważniejszych obszarów wykorzystania wizji komputerowej i już teraz rozwiązania proponowane przez badaczy mają bardzo wysoką skuteczność. Cel niniejszej pracy, czyli analiza predykcyjna modelu konwolucyjnych sieci neuronowych wskazuje, że do stworzenia rozwiązania w obszarze medycznej diagnostyki obrazowej o potencjalnie akceptowalnej jakości klasyfikacji na poziomie 97,33% mogą wystarczyć stosunkowo niewielkie zasoby obliczeniowe, odpowiednia wiedza z zakresu uczenia maszynowego oraz dostęp do narzędzi open-source. Największym ograniczeniem może być sam dostęp do danych, który bywa limitowany poprzez prawo (ochrona wrażliwych danych osobowych) lub przez interesy firm będących w ich posiadaniu. Nawet po przezwyciężeniu tych ograniczeń mogą się pojawić kwestie jakości, np. brak etykiet lub błędy w nich, niejednolite formaty, czy też samej ilości danych. Mimo to, powstaje coraz więcej repozytoriów zawierających różnego rodzaju dobrej jakości zdjęcia, z których można korzystać przy budowaniu modeli rozpoznawania obrazów. Oprócz danych konkursowych tutaj wykorzystanych, przykładem mogą być tutaj ogłoszone w ostatnim czasie dwa wyzwania z przewidzianymi nagrodami pieniężnymi: diagnoza retinopatii cukrzycowej na podstawie zdjęć siatkówki oka, jak również identyfikacja odmy opłucnowej na podstawie zdjęć RTG klatki piersiowej.[58,59] Takie inicjatywy demokratyzują obszar głębokiego uczenia obniżając próg wejścia dla badaczy oraz pozwalają na dalszy rozwój systemów analizy obrazów w medycynie.

VII. Bibligrafia

Książki:

1. Goodfellow, Ian. Bengio, Yoshua. Courville, Aaron. “Deep Learning”. MIT Press (www.deeplearningbook.org), 2016.
2. Russel, Stuart. Norvig, Peter. “Artificial Intelligence: A Modern Approach”. 3rd ed. Pearson Education, 2009.
3. Mitchell, Thomas. “Machine Learning”. 1st ed. McGraw-Hill, 1997.
4. Buduma. Nikhil. Locascio, Nikhil. “Fundamentals of Deep Learning: Designing Next-Generation Machine Intelligence Algorithms”. O’Reilly Media. 2017.
5. Nielsen, Michael. “Neural Networks and Deep Learning”. Determination Press. 2015.

[58] “SIIM-ACR Pneumothorax Segmentation”. Kaggle.com. 2019.
[59] “APTOS 2019 Blindness Detection”. Kaggle.com. 2019.

Materiały dydaktyczne:

1. Howard, Jeremy. et. al. “Practical Deep Learning for Coders v3”. course.fast.ai. 2019.
2. Lazy Programmer (pseudonim). “Data Science: Deep Learning in Python”. Udemy.com. 2019.
3. Lazy Programmer (pseudonim). “Deep Learning Prerequisites: Linear Regression in Python”. Udemy.com. 2019.
4. Becker, Dan. “Deep Learning in Python”. Datacamp.com. 2017.
5. Rokem, Ariel. “Convolutional Neural Networks for Image Processing”. Datacamp.com. 2018.
6. Kamiński, Bogumił. Pankratz, Bartosz. ”Startystyczne Reguły Decyzyjne”. Szkoła Główna Handlowa w Warszawie. 2019.

Artykuły naukowe:

1. Praca zbiorowa. "Ascent of machine learning in medicine". Nature Materials 18: 407. 2019.
2. McDonald, RJ. et. al. “The effects of changes in utilization and technological advancements of cross-sectional imaging on radiologist workload”. PubMed-NCBI. 2019.
3. Fathi, Ehsan. Babak, Maleki. “Deep Neural Networks for Natural Language Processing”. Science Direct. 2018.
4. Veeling, Basitan. et. al. "Rotation Equivariant CNNs for Digital Pathology". arXiv:1806.03962. 2018.
5. Ehteshami Bejnordi et al. “Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer”. JAMA, 318(22). 2017.
6. Liu, Yun. et. al. “Detecting Cancer Metastases on Gigapixel Pathology Images”. Google AI. 2017.
7. Geras, Krzysztof, et. al. “High-Resolution Breast Cancer Screening with Multi-View Deep Convolutional Neural Networks”. arXiv:1703.07047. 2018.
8. Ehteshami Bejnordi. et. al. “1399 H&E-stained sentinel lymph node sections of breast cancer patients: the CAMELYON dataset”. GigaScience. 2018.
9. Bandos, Andriy. et. al. “Area under the Free-Response ROC Curve (FROC) and a Related Summary Index”. PubMed-NCBI. 2009.
10. Rajaraman, Sivaramakrishnan. et. al. “Pre-trained convolutional neural networks as feature extractors toward improved malaria parasite detection in thin blood smear images”. PubMed-NCBI. 2018.
11. Simonyan, Karen, Zisserman, Andrew. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. arXiv:1409.1556. 2015.
12. Rajaraman, Sivaramakrishnan. et. al. “Performance evaluation of deep neural ensembles toward malaria parasite detection in thin-blood smear images”. PubMed-NCBI. 2019.
13. Lakhani, Paras. Sundaram, Baskaran. “Deep Learning at Chest Radiography: Automated Classification of Pulmonary Tuberculosis by Using Convolutional Neural Networks”. RSNA Radiology. 2017.
14. Menze, Bjoern. et. al. “The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)”. IEEE Transactions on Medical Imaging, 34(10). 2015.
15. Isin, Ali. et. al. “Review of MRI-based Brain Tumor Image Segmentation Using Deep Learning Methods”. Procedia Computer Science 102. 2016.

16. Pereira, Sergio. et. al. “Brain Tumor Segmentation Using Convolutional Neural Networks in MRI Images”. IEEE Transactions on Medical Imaging, 35(5). 2016.
17. Cybenko, George. “Approximation by superpositions of a sigmoidal function”. Mathematics of Control, Signals, and Systems. 1989.
18. Barron, Andrew. “Universal Approximation Bounds for Superpositions of a Sigmoidal Function”. IEEE Transactions on Information Theory, 39(3). 1993.
19. Montúfar, Guido. Pascanu, Razvan. Cho, Kyunghyun. Bengio, Yoshua. “On the number of linear regions of deep neural networks”. NIPS'2014. 2014.
20. Ioffe, Sergey, Szegedy, Christian. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. arXiv:1502.03167. 2015.
21. Kornblith, Simon. Shlens, Jonathon. Le, Quoc. “Do Better ImageNet Models Transfer Better?”. arXiv:1805.08974. 2019.
22. He, Kaiming. et. al. “Deep Residual Learning for Image Recognition”. Microsoft Research. 2015.
23. Huang, Gao. et. al. “Densely Connected Convolutional Networks”. arXiv:1608.06993. 2018.
24. Selvaraju, Ramprasaath. et. al. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization”. arXiv:1610.02391. 2017.

Strony internetowe:

1. Praca zbiorowa. “Global $4.5 Bn Computer Vision in Healthcare Market Outlook 2017-2019 & Forecast to 2026”. businesswire.com. 2019. (dostęp 13.05.2019)
2. Praca zbiorowa. “Global Big Data in Healthcare Market: Analysis and Forecast, 2017-2025”. bisresearch.com. 2018. (dostęp 13.05.2019)
3. “Histopathologic Cancer Detection”. kaggle.com. 2019. (dostęp 20.02.2019)
4. “Malaria Cell Images Dataset”. kaggle.com. 2019. (dostęp 05.05.2019)
5. “Gruźlica człowieka”. wikipedia.org. (dostęp 05.05.2019)
6. Stumpe, Martin. Peng, Lily. “Assisting Pathologists in Detecting Cancer with Deep Learning”. Google AI Blog. 2019. (dostęp 15.05.2019)
7. Landi, Heather. “IBM Unveils Watson-Powered Imaging Solutions at RSNA”. hcinnovationgroup.com. 2016. (dostęp 01.06.2019)
8. Thomas, Rachel. “How (and why) to create a good validation set”. Fast.ai. 2017. (dostęp 30.05.2019)
9. Jadon, Shruti. “Introduction to Different Activation Functions for Deep Learning” Medium.com. 2018. (dostęp 15.06.2019).
10. Sopyła, Krzysztof. ”Przetwarzanie obrazu z wykorzystaniem splotu funkcji”. Ksopyla.com. 2016. (dostęp 30.06.2019)
11. Ruiz, Pablo. “Understanding and visualizing ResNets”. Medium.com. 2018. (dostęp 15.07.2019)
12. “SIIM-ACR Pneumothorax Segmentation”. Kaggle.com. 2019. (dostęp 20.07.2019)
13. “APTOS 2019 Blindness Detection”. Kaggle.com. 2019. (dostęp 20.07.2019)

VIII. Spis rysunków i tabel

Rysunek 1. Zdjęcie histopatologiczne węzłów chłonnych.
Rysunek 2. Cztery różne zdjęcia mammograficzne tej samej osoby zakwalifikowanej przez model do powtórnego badania.
Rysunek 3. Wymaz krwi osoby zarażonej malarią.
Rysunek 4. Zdjęcie RTG osoby chorej na gruźlicę.
Rysunek 5. Przykłady segmentacji na zdjęciach MRI głowy.
Rysunek 6. Regresja liniowa oraz optymalizacja wag.
Rysunek 7. Proces wyszukiwania minimum funkcji za pomocą metody najszybszego spadku.
Rysunek 8. Wykresy prezentujące problemy niedouczenia i przeuczenia.
Rysunek 9. Zależność między błędem predykcji a złożonością modelu dla zbiorów testowego i treningowego.
Rysunek 10. Prezentacja działania neuronu w sztucznych sieciach.
Rysunek 11. Przykład w pełni połączonej, sieci skierowanej z trzema warstwami: wejściową, ukrytą, wyjściową.
Rysunek 12. Porównanie funkcji aktywacji: sigmoidu i tanh.
Rysunek 13. Funkcja aktywacji ReLU.
Rysunek 14. Wizualizacja działania metody najszybszego spadku.
Rysunek 15. Zobrazowanie działania kolejnych warstw sieci neuronowych na przykładzie narysowanej na kartce funkcji.
Rysunek 16. Operacja splotu na przykładzie zdjęcia (pikseli).
Rysunek 17. Wizualizacja przykładów dla half-padding oraz full-padding.
Rysunek 18. Przykład prostej architektury konwolucyjnych sieci neuronowych.
Rysunek 19. Przykład działania warstwy łączącej - max pooling.
Rysunek 20. Prezentacja pełnej sieci oraz z wyłączanymi neuronami.
Rysunek 21. Przykład zdjęć histopatologicznych.
Rysunek 22. Przykładowe zdjęcia z treningowego zbioru danych z etykietami.
Rysunek 23. Porównanie oryginalnego zdjęcia z zaaplikowanymi wybranymi modyfikacjami.
Rysunek 24. Jedna z części składowych sieci ResNet.
Rysunek 25. Uproszczona wizualizacja architektury DenseNet.
Rysunek 26. Wygenerowany wykres służący do wyboru parametru uczenia przed rozpoczęciem trenowania.
Rysunek 27. Stosunki błędów na zbiorach walidacyjnym i testowym w trakcie uczenia odpowiednio dla 1-3, 4-6, 7-8 epok.
Rysunek 28. Macierz pomyłek na zbiorze walidacyjnym dla modelu opartego na DenseNet 201.
Rysunek 29. Wyróżnione największe pomyłki modelu.

Tabela 1. Macierz wyjściowa transformacji softmax dla 4 obserwacji i 3 możliwych kategoriach.
Tabela 2. Przykład prawidłowego i nieprawidłowego przyporządkowania do kategorii.
Tabela 3. Macierz pomyłek.
Tabela 4. Rezultaty różnej długości trenowania sieci dla różnych parametrów.
Tabela 5. Porównanie rezultatów dla różnych rodzajów sieci.

IX. Streszczenie

Diagnostyka medyczna jest aktualnie jednym z głównych obszarów zastosowań systemów analizy obrazów, które mają wspierać lekarzy w ocenie stanu zdrowia pacjenta. Równolegle rozwijane są algorytmy klasyfikujące fotografie pochodzące m.in. z badań histopatologicznych i mammograficznych, wymazów krwi, prześwietleń RTG, badań USG, rezonansu magnetycznego. Na podstawie opisanej literatury można stwierdzić, że jakość klasyfikacji tych rozwiązań przekracza, lub jest bardzo bliska, dokładności diagnoz lekarzy. Jednocześnie są o wiele bardziej efektywne czasowo. Tego rodzaju wsparcie byłoby bardzo przydatne biorąc pod uwagę stale rosnącą ilość informacji, które muszą przetwarzać diagnostycy. Celem ninejszej pracy była analiza jakości predykcji jednego z takich właśnie modeli. Przeprowadzono klasyfikację binarną zdjęć histopatologicznych węzłów chłonnych pod kątem obecności nowotworowych komórek przerzutowych. Dane pochodziły z konkursowego repozytorium opublikowanym na Kaggle.com zawierającego ponad 275 tysięcy zdjęć. Do tego zadania wykorzystano konwolucyjne sieci neuronowe oparte na transferze wiedzy z bardziej złożonych modeli oraz na wybranych technikach regularyzacji, w tym augmentacji zdjęć. Wykorzystując architekturę DenseNet o głębokości 201 warstw osiągnięto wynik metryki AUC na poziomie 97.56% na 51% zbioru testowego oraz 97.08% na pozostałych 49%. Powyższe rezultaty pozycjonują zaproponowany model wśród najlepszych 100 w wyżej wymienionym konkursie. Biorąc pod uwagę prostotę zastosowanego kodu programu i relatywnie krótki czas trenowania ostatecznej wersji sieci (~8 godzin) można uznać zaproponowane rozwiązanie jako potencjalnie użyteczne we wspieraniu procesów decyzyjnych przeprowadzanych przez lekarzy diagnostyków na oddziałach onkologicznych.