

**WYDZIAŁ ZARZĄDZANIA**

Katedra Informatyki Biznesowej i Inżynierii Zarządzania

Projekt dyplomowy

*Aplikacja mobilna do nauki matematyki dla szkoły podstawowej z wykorzystaniem machine learningu do rozpoznawania odpowiedzi pisanych ręcznie*

*A mobile math learning app for elementary school using machine learning to recognize handwritten answers*

Autor: *Monika Helena Etrych*

Kierunek studiów: Informatyka i Ekonometria

Opiekun pracy: *dr Beata Basiura*

Kraków, 2023 r.

Spis treści

[Wstęp 3](#_Toc131004316)

[1. Rozpoznawanie cyfr i ich interpretacja 4](#_Toc131004317)

[1.1. Interpretacja obrazów przez komputer 4](#_Toc131004318)

[1.2. Metody uczenia maszynowego do rozpoznawania cyfr 4](#_Toc131004319)

[1.3. Obróbka obrazów z cyframi 6](#_Toc131004320)

[1.3.1. Wstępne przetwarzanie obrazów 6](#_Toc131004321)

[1.3.2. Augmentacja 6](#_Toc131004322)

[1.4. Weryfikacja modelu do rozpoznawania cyfr 7](#_Toc131004323)

[2. Model aplikacji do nauki uczenia matematyki 11](#_Toc131004324)

[2.1. Założenia modelu aplikacji 11](#_Toc131004325)

[2.2. Wybrane technologie zastosowane do budowy aplikacji 12](#_Toc131004326)

[2.3. Wybrane metody uczenia maszynowego zastosowane w aplikacji 13](#_Toc131004327)

[2.4. Korzystanie z modelu ML poprzez API HTTP 14](#_Toc131004328)

[3. Opis implementacji i działania aplikacji 15](#_Toc131004329)

[3.1. Ekran startowy i podstawowe funkcjonalności 15](#_Toc131004330)

[3.2. Schematy, opisy poszczególnych funkcjonalności, opis kodów 16](#_Toc131004331)

[3.3. Przykłady zastosowania aplikacji w konkretnych zadaniach 17](#_Toc131004332)

[Zakończenie 18](#_Toc131004333)

[Spis tabel, wykresów i rysunków 19](#_Toc131004334)

[Wykaz rysunków 19](#_Toc131004335)

[Bibliografia 20](#_Toc131004336)

[[4] M. Javed, *The Best Machine Learning Algorithm for Handwritten Digits Recognition,* 2020, 20](#_Toc131004337)

[Aneks (kod, tabelka z danymi, treść ankiety) 22](#_Toc131004338)

# Wstęp

1-2 stron, cel pracy, ważny problem, może być hipoteza badawcza, napisać na koniec!

Problem rozpoznawania cyfr – problematyczne, bo każdy ma inny charakter, pochyłość, duże małe. Różnice kulturowe. Pisanie cyfry od góry do dołu lub od dołu.

# Rozpoznawanie cyfr i ich interpretacja

## Interpretacja obrazów przez komputer

Obrazy reprezentowane przez komputer składają z pikseli - kwadratów, które posiadają kilka wartości określających kolor. Wyróżnia się 4 podstawowe reprezentacje: binarną, monochromatyczną (ang. grayscale), RGB i RGBA.

Reprezentacja binarna oraz monochromatyczna składają się tylko z jednej wartości dla danego pixela. W przypadku reprezentacji binarnej wartość jest zerojedynkowa, gdzie 0 oznacza, że pixel ma kolor czarny, a 1 to kolor biały. W pozostałych reprezentacjach wartość jest z zakresu od 0 do 255. W reprezentacji monochromatycznej 0 oznacza kolor czarny, 255 - biały, natomiast wartości pośrednie, to odcienie szarości.

RGB zawiera trzy wartości określające kolor: czerwony, zielony i niebieski. RGBA posiada dodatkowo jedną wartość, która określa poziom przezroczystości. [1]

## Metody uczenia maszynowego do rozpoznawania cyfr

Metody uczenia maszynowego są coraz częściej wykorzystywane do rozpoznawania cyfr w różnych dziedzinach, takich jak bankowość czy usługi pocztowe. Uczenie maszynowe pozwala na automatyczne i precyzyjne rozpoznawanie cyfr. Wiele papierowych formularzy w sposób szybki i dokładny może zostać przeniesiona do komputera. Natomiast rozpoznawanie kodów pocztowych znacząco przyspiesza proces sortowania listów i przesyłek [3].

Do rozpoznawania cyfr wykorzystuje się algorytmy uczenia nadzorowanego. W tym podejściu, dane uczące zawierają cyfry wraz z odpowiadającymi im etykietami, czyli poprawnymi wartościami cyfr. Algorytm uczony na takich danych jest w stanie nauczyć się rozpoznawać cyfry na podstawie cech wizualnych, takich jak kształt i proporcje. [1, 5]

Wśród popularnych algorytmów uczenia maszynowego stosowanych do rozpoznawania cyfr znajdują się między innymi konwolucyjne sieci neuronowe (CNN), wielowarstwowe sieci neuronowe (MLP) oraz algorytm maszyn wektorów nośnych (SVM). Zastosowanie tych algorytmów wraz z odpowiednio przygotowanymi danymi uczącymi, pozwala na uzyskanie skuteczności rozpoznawania cyfr na poziomie nawet 99% [4, 5].

Algorytm maszyn wektorów nośnych polega na maksymalizacji odległości pomiędzy hiperpłaszczyzną, a marginesem przynależności do poszczególnych klas, poprzez zmianę położenia hiperpłaszczyzny. Istnieje wiele możliwych hiperpłaszczyzn, a jej kształt zależy od złożoności problemu - liczby klas do zaklasyfikowania. Kiedy istnieją dwie klasy, to hiperpłaszczyzna jest prostą, dla trzech danych wejściowych jest płaszczyzną 2D itd. [6].

Wielowarstwowy perceptron składa się z trzech podstawowych warstw: warstwy wejściowej, warstw ukrytych oraz warstwy wyjściowej. Liczba warstw ukrytych zależy od złożoności problemu. Warstwy składają się z neuronów oraz funkcji aktywacji. Wraz z kolejnymi iteracjami modelu, wagi są aktualizowane tak, aby zminimalizować błąd predykcji. Wagi te określają poziom wpływu każdego neuronu na wynik sieci. Funkcje aktywacji pozwalają na nieliniowe modelowanie złożonych zależności, co pozwala na lepszą predykcję [9].

W konwolucyjnych sieciach neuronowych można wyróżnić sześć warstw: wejściową, konwolucyjną, aktywacji, łączenia, w pełni połączoną oraz wyjściową. Warstwa konwolucyjna składa się z filtrów i map cech. Filtry są to kwadratowe wycinki obrazu, które wyodrębniają cechy charakterystyczne dla różnych obrazów. Mapa cech jest zbiorem wyodrębnionych cech. Warstwa aktywacji przekształca dane z mapy cech tak aby dodać nieliniowość, używa wybranej funkcji np. ReLU. Warstwa łączenia (ang. pooling) generalizuje cechy, aby zapobiec przeuczeniu. Warstwa w pełni połączona jest używana do stworzenia końcowej nieliniowej kombinacji cech oraz dokonywania przewidywań przez sieć [5, 9, 10].

Wymienione metody zostały porównane w 2021r. przez zespół R. Dixita. W procesie wstępnego przetwarzania obraz wykorzystano tylko podstawowe przekształcenia takie jak normalizacja wartości pikseli, co pozwoliło na zamianę wartości pikseli z przedziału 0-255 na wartości z przedziału 0-1. Ponadto, wartość liczby przedstawionej na obrazie została zamieniona na zmienną kategoryczną. Następnie wybrane metody zostały zestawione pod kątem czasu wykonania oraz dokładności dla danych treningowych i testowych. Najwyższy wynik dokładności dla danych treningowych osiągnął SVM - 99,98%, a dla danych testowych - CNN - 99,31% [5].

W innym badaniu w 2012r. przez zespół D.C. Ciresana porównano duże modele wielowarstwowe sieci neuronowych (MLP) z różnymi konfiguracjami liczby warstw oraz neuronów. W rezultacie najlepszy wynik uzyskano dla MLP składającej się z ośmiu warstw i liczb neuronów: 2500, 2000, 1500, 1000, 500, 10. Proces uczenia modelu obejmował aż 2000 epok, a osiągnięta dokładność wyniosła 99,65%. Warto zauważyć, że cyfry, które zostały źle sklasyfikowane przez model, były trudne do określenia nawet dla człowieka [11].

W 2020r. zespół S. Ahlawata podjął próbę dopasowania hiperparametrów do modelu konwolucyjnych sieci neuronowych. W badaniu sprawdzono wpływ liczby warstw, rozmiaru kroku (liczby pixeli o jaką przesuwa się filtr), pól recepcyjnych (wycinka obrazu odpowiadającego danej cesze), rozmiaru filtrów (rozmiaru macierzy z wagami), dopełnienia (paddingu), rozcieńczenia (dilution) oraz rodzajów klasyfikatorów. W ramach badań zastosowano kilka metod przygotowania danych takie jak: skalowanie, centrowanie, redukcje szumów. Najlepszy wynik, który udało się osiągnąć to dokładność na poziomie 99,89% używając optymalizatora Adam [13].

## Obróbka obrazów z cyframi

### Wstępne przetwarzanie obrazów

W procesie rozpoznawania cyfr przez model, kluczowe znaczenie ma odpowiednie przekształcenie obrazów przed ich przekazaniem do modelu. Wstępne przetwarzanie obrazów ma na celu przygotowanie danych do dalszej analizy. Ponadto zmniejsza czas potrzebny do wytrenowania modelu i zwiększa szybkość otrzymania wyniku. Na przykład, kiedy wejściowy obraz jest stosunkowo duży, zmniejszenie rozmiaru znacząco zmniejsza czas trenowania, bez znacznej utraty dokładności modelu.

Sieci neuronowe wymagają, aby dane wejściowe miały ten sam rozmiar, ponieważ sieci te korzystają z macierzy wag, które mają ustaloną liczbę kolumn i wierszy. Przetwarzanie obrazów przez ustawienie stałych wymiarów jest jednym ze sposobów na dostosowanie rozmiaru obrazów do wymogów sieci neuronowej. Zmiany rozmiaru obrazu można dokonać na dwa sposoby. Pierwszy z nich polega na zmniejszeniu obrazu poprzez centrowanie oraz wycinanie pustego obszaru. Drugi sposób polega na zwiększeniu rozmiaru przez dodanie pustych pikseli po bokach.

Złożoność modelu można uprościć za pomocą zamiany reprezentacji obrazu na mniej złożoną. Dla kolorowych zdjęć stosuje się zamianę na skalę monochromatyczną, a dla skali szarości zamianę na reprezentację binarną.

### Augmentacja

W sytuacji, kiedy danych treningowych jest zbyt mało, model głębokiego uczenia może ulec przeuczeniu, czyli zbytniemu dopasowaniu do danych trenujących. Przeuczenie skutkuje niskimi wynikami na nowych danych. Aby zapobiec temu problemowi, konieczna jest dostarczenie modelowi odpowiedniej liczby danych trenujących. Można to osiągnąć stosując augmentację, czyli przekształcenia istniejących obrazów za pomocą np. losowych zmian rotacji lub jasności. Na (rys. 1.1.) w pierwszym rzędzie przedstawiono obrazy cyfr w normalnym położeniu oraz w drugim rzędzie po zastosowaniu zmian rotacji o losowy kąt z zakresu ± 30 stopni. Augmentacja jest łatwym sposobem na zwiększenie różnorodności danych trenujących i poprawę jakości modelu.

Innymi sposobami zwiększenia liczby danych są losowe odbicia obrazów wzdłuż osi OX lub OY, losowa zmiana jasności i naświetlenia, dodawanie pixeli w losowych miejscach.

Rysunek 1.1 Losowy obrót obrazów

Obraz zawierający tekst

Opis wygenerowany automatycznie

*źródło: opracowanie własne*

Te same zabiegi przetwarzania danych mogą być stosowane zarówno w ramach wstępnego przetwarzania danych, jak i augmentacji. Jednakże, wstępne przetwarzanie danych jest stosowane zarówno na danych treningowych, jak i testowych, podczas gdy augmentacja jest stosowana wyłącznie na danych treningowych po wstępnym przetworzeniu [2].

## Weryfikacja modelu do rozpoznawania cyfr

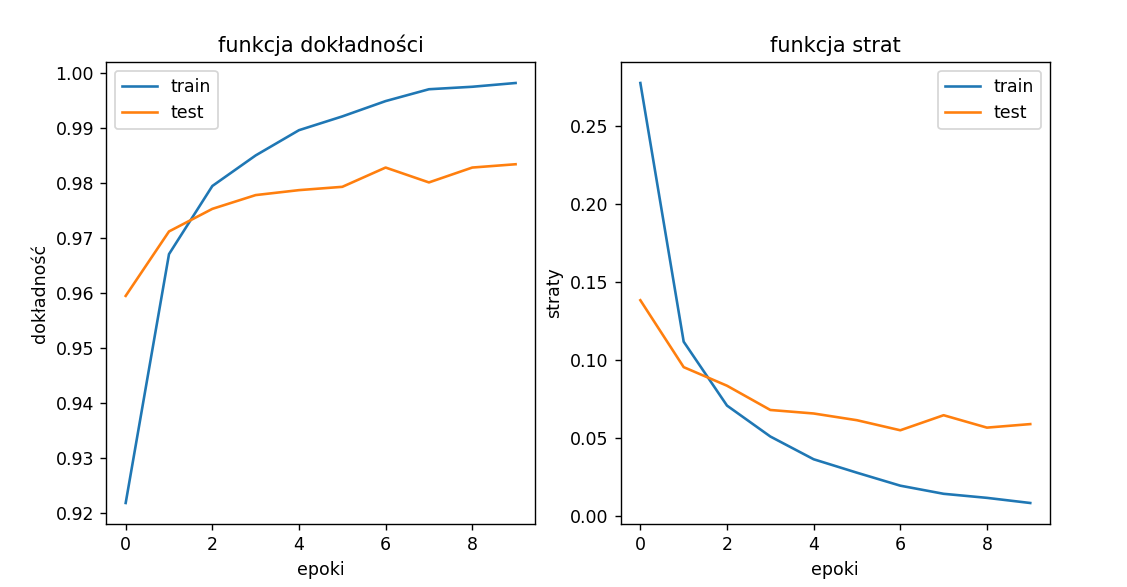
Poziom dokładności stanowi jedną z najprostszych metod oceny jakości modelu. Określa on stosunek poprawnie sklasyfikowanych danych do liczby wszystkich danych. Miara ta jest adekwatna tylko, gdy zbiór danych jest zbilansowany, czyli zawiera równą liczbę próbek dla każdej klasy [12].

Innym sposobem weryfikacji modelu uczenia maszynowego jest sprawdzenie na jakim etapie nauczenia znajduje się. Model pod wpływem czasu może być kolejno: niedouczony, wyuczony odpowiednio i przeuczony. Jednak, jeśli model jest źle skonstruowany może nie dochodzić do późniejszych etapów.

Problem niedouczenia polega na tym, że hipotezy modelu są nieodpowiednie i zbyt proste w porównaniu do złożoności danych. Przez to model nie opisuje wystarczająco dobrze danych. Natomiast przeuczenie polega na zbytniej złożoności modelu i w konsekwencji zbyt dokładnym opisywaniu danych treningowych i problemami w dopasowaniu nowych elementów [7].

Dwa wykresy dokładności i strat, mogą dostarczyć informacji o stanie wyuczeniu modelu. Funkcja dokładności przedstawia poziom dokładności modelu w kolejnych epokach. Tak długo jak te krzywe dla danych uczących i testujących rosną model polepsza swoją skuteczność – model nie jest jeszcze wystarczająco wyuczony. Kiedy wykres stabilizuje się, nie rośnie tak gwałtownie mamy model odpowiednio wyuczony. Jednak, gdy funkcja na zestawie danych testujących po pewnym czasie zacznie maleć – oznacza to, że model został przeuczony. Celem jest maksymalizacja współczynnika dokładności i minimalizacja strat, błędu [8].

Rysunek 1.2 Funkcja dokładności i funkcja strat



*źródło: opracowanie własne*

Na (rys. 1.2.) został przedstawiony model odpowiednio wyuczony. Funkcja dokładności osiąga wyższe wartości dla danych uczących niż dla danych testowych. Funkcja strat maleje wraz z upływem czasu, jednak odległość pomiędzy prostymi dla danych uczących i testowych nie zmniejsza się, co świadczy o zbyt małej reprezentatywności danych treningowych. Model zawiera zbyt mało danych uczących [8].

Podczas klasyfikacji można rozróżnić cztery przypadki zaklasyfikowania:

1. poprawnego jako pozytywne – TP (ang. true positive)
2. poprawnego jako negatywne – FN (ang. false negative)
3. niepoprawnego jako pozytywne – FP (ang. false positive)
4. niepoprawnego jako negatywne – FN (ang. false negative)

Można wyróżnić procentowe wskaźniki, które bazują na wartościach są to: czułość, specyficzność oraz precyzja. W zależności od oczekiwań danego modelu powinno się maksymalizować wybrany współczynnik. Nie jest możliwe otrzymanie wszystkich współczynników na wysokim poziomie lub na niskim poziomie.

Czułość jest to stosunek poprawnie zaklasyfikowanych wartości jako pozytywne (TP) do sumy wartości prawdziwie pozytywnych (TP) i fałszywie negatywnych (FN): TP/(TP+FN). Jest wykorzystywana, gdy niepożądane jest wystąpienie nieprawdziwie negatywnych wyników klasyfikacji.

Specyficzność to stosunek wyników prawdziwie negatywnych (TN) do sumy prawdziwie negatywnych (TN) i fałszywie pozytywnych (FP): TN/(TN+FP). Jest używana, kiedy ważne jest zwrócenie uwagi na prawdziwie negatywne przypadki.

Precyzja to stosunek prawdziwie pozytywnych (TP) do sumy prawdziwie pozytywnych (TP) i fałszywie pozytywnych (FP): TP/TP+FP. Znajduje ona zastosowanie, gdy ważne są prawdziwie pozytywne wyniki sklasyfikowania [14, 15].

Rysunek 1.3 Macierz błędu

Obraz zawierający stół

Opis wygenerowany automatycznie

*źródło: opracowanie własne*

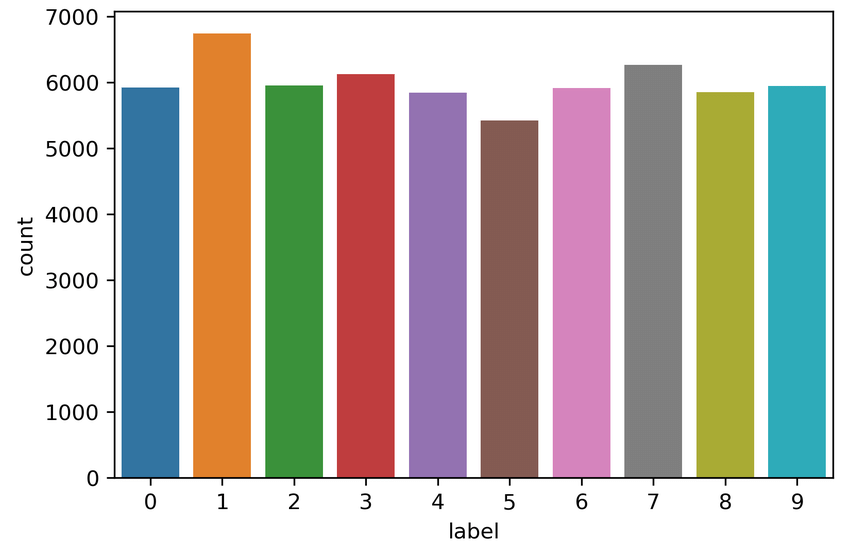
# Model aplikacji do nauki uczenia matematyki

## Założenia modelu aplikacji

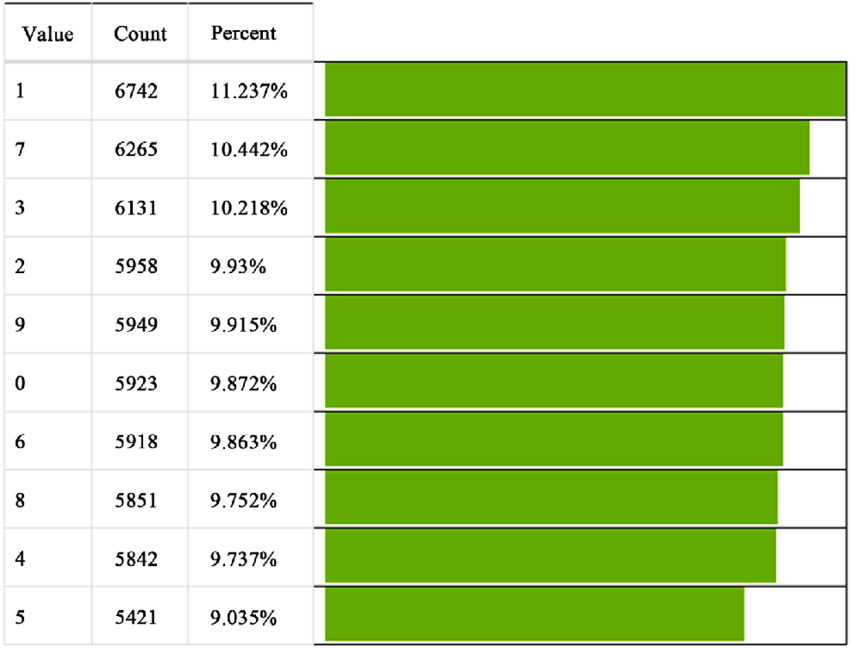
## Wybrane technologie zastosowane do budowy aplikacji

## Wybrane metody uczenia maszynowego zastosowane w aplikacji

Confiusion matrix, accuracy, czułość, specyficzność, precyzja



MNIST DISTRIBUTION



## Korzystanie z modelu ML poprzez API HTTP

# Opis implementacji i działania aplikacji

## Ekran startowy i podstawowe funkcjonalności

## Schematy, opisy poszczególnych funkcjonalności, opis kodów

## Przykłady zastosowania aplikacji w konkretnych zadaniach

# Zakończenie

Wyniki, rezultaty, udowodnienie tezy, co na przyszłość rozwinąć

# Spis tabel, wykresów i rysunków

# Wykaz rysunków

Rysunek 1.1: opracowanie własne, stworzone przy użyciu programu Python

Rysunek 1.2: opracowanie własne, stworzone przy użyciu programu Python

Rysunek 1.3: opracowanie własne, stworzone przy użyciu programu Excel

# Bibliografia

[1] M. [Candocia](https://www.linkedin.com/in/max-candocia-a270b574/)*, A Simple Explanation of how Computers Recognize Images,* [Analysis for Many Audiences,](https://maxcandocia.com/)  2016, url: <https://maxcandocia.com/article/2016/Apr/06/how-computers-recognize-images/>, dostęp: 23.02.2023

[2] J. Nelson*, What is Image Preprocessing and Augmentation?,* Roboflow, 2020, url: <https://blog.roboflow.com/why-preprocess-augment/>, dostęp: 28.02.2023r.

[3] S. M. Shamim, M. B. A. Miah, A. Sarker, M. Rana, A. A. Jobair, 2018, *Handwritten Digit Recognition Using Machine Learning,* Algorithms Indonesian Journal of Science & Technology (1) vol. 3, str.: 29-39 ISSN 2528-1410, ISSN 2527-8045

[4] M. Javed, The Best Machine Learning Algorithm for Handwritten Digits Recognition, 2020, url:<https://towardsdatascience.com/the-best-machine-learning-algorithm-for-handwritten-digits-recognition-2c6089ad8f09>, dostęp: 1.03.2023

[5] R. Dixit, R. Kushwah, S. Pashine, *Handwritten Digit Recognition using Machine and  
Deep Learning Algorithms,* Artykuł Naukowy 2021, url: <https://arxiv.org/pdf/2106.12614.pdf>, dostęp: 1.03.2023

[6] R. Gandhi, *Support Vector Machine — Introduction to Machine Learning Algorithms*, 2018, url: <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>, dostęp: 1.03.2023

[7] D. Nikolaiev *Overfitting and Underfitting Principles,* 2021,url: [*https://towardsdatascience.com/overfitting-and-underfitting-principles-ea8964d9c45c*](https://towardsdatascience.com/overfitting-and-underfitting-principles-ea8964d9c45c)*,* dostęp: *14.03.2023*

[8] J. Brownlee, *How to use Learning Curves to Diagnose Machine Learning Model Performance, 2019,* url: <https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/>, dostęp: 9.03.2023

[9] J. Brownlee, *Develop Deep Learning Models on Theano and TensorFlow Using Keras,* 2016, str: 37-42*,* 116 – 134

[10] M. Mamczur, *Jak działają konwolucyjne sieci neuronowe (CNN)?,* 2021, url: <https://miroslawmamczur.pl/jak-dzialaja-konwolucyjne-sieci-neuronowe-cnn/>, dostęp: 14.03.2023

[11] D.C. Ciresan, U. Meier, L. M. Gambardella, J. Schmidhuber, *Deep Big Multilayer Perceptrons For Digit Recognition,* 2012, w: G. Montavon, G.B. Orr, KR. Müller, *Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science*, vol 7700, str: 581-598, Springer, Berlin, Heidelberg

[12] N. Bressler, *How to Check the Accuracy of Your Machine Learning Model,* 2022, url: <https://deepchecks.com/how-to-check-the-accuracy-of-your-machine-learning-model/>, dostęp: 23.03.2023

[13] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, B. Yoon, 2020, *Improved Handwritten Digit Recognition Using Convolutional Neural Networks (CNN)*

[14] A Lekhtman, 2019, *Data Science in Medicine — Precision & Recall or Specificity & Sensitivity?,* url: <https://towardsdatascience.com/should-i-look-at-precision-recall-or-specificity-sensitivity-3946158aace1>, dostęp: 29.03.2023

[15] S. Ghoneim, 2019, *Accuracy, Recall, Precision, F-Score & Specificity, which to optimize on?,* url: <https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124>, dostęp: 29.03.2023

# Aneks (kod, tabelka z danymi, treść ankiety)

(NOTATKI)

? Dorobić augmentację - jak starczy czasu – obracanie obrazków

? Może wycinanie obrazków cyfr z obrazu liczby

? Napisać o zbiorze danych MNIST

? Wykres, ile jest jakiej klasy danych, histogram ile jedynek ile dwójek itd.

? MAE, RMSE, RAE, RRAE, ROC i AUC