

Politechnika Warszawska

W Y D Z I A Ł   M E C H A N I C Z N Y  
E N E R G E T Y K I   I   L O T N I C T W A



Instytut Techniki Lotniczej i Mechaniki Stosowanej

# Praca dyplomowa inżynierska

na kierunku Automatyka i robotyka  
w specjalności Robotyka

Wykorzystanie sztucznych sieci neuronowych do rozpoznawania  
obrazów, na przykładzie automatycznego odczytywania pisma

Piotr Walędzik

Numer albumu 259692

promotor

dr hab. inż. Cezary Rzymkowski, prof. nadzw. PW

Warszawa, 2017

# Streszczenie

## WYKORZYSTANIE SZTUCZNYCH SIECI NEURONOWYCH DO ROZPOZNAWANIA OBRAZÓW, NA PRZYKŁADZIE AUTOMATYCZNEGO ODCZYTYWANIA PISMA.

Niniejsza praca ma na celu pokazanie przebiegu tworzenia algorytmu analizy danych, a dokładniej rozpoznawania obrazów na przykładzie automatycznego odczytywania pisma. Rozwiązanie problemu dotyczy sposobu automatyzacji procesu zaczytywania formularzy tabelarycznych i tłumaczenie ich na dane cyfrowe (podpis komputerowy).

Maszynowe rozpoznawanie kształtów jest zadaniem skomplikowanym, aczkolwiek użytecznym. Algorytm poniższej pracy opiera się na sztucznych sieciach neuronowych typu Multi-Layered Perceptron. Pomimo, że zaimplementowane zagadnienie dotyczy rozpoznawania obrazów na przykładzie odczytywania liter po ich kształcie, to w dokładnie ten sam sposób można efektywnie nauczyć sieć neuronową rozpoznawać inne obiekty, przykładowo kształty na taśmie produkcyjnej.

W rozdziałach po kolei omawiane są rysy historyczne sztucznych sieci neuronowych, budowa ludzkiego neuronu oraz budowa jego matematycznego odpowiednika. Następnie omówione są rodzaje sztucznych sieci neuronowych i metody ich nauczania, dokładny opis zagadnienia, użytego środowiska programistycznego oraz zbioru danych. Kończąc na opisie przebiegu implementacji oraz ostatecznym podsumowaniu uzyskanego rozwiązania.

Ostateczna wersja sztucznej sieci neuronowej kształtowana, testowana i analizowana była w różnych warunkach oraz z różnymi parametrami. Ostatecznie udało się uzyskać satysfakcjonujące wyniki sięgające poprawności 93%. Problem ten został rozwiązany przy użyciu sztucznych sieci neuronowych z grupy algorytmów uczenia nadzorowanego. Wyniki jakie można uzyskać pokazują, że sztuczne sieci neuronowe są dobrym narzędziem analizy danych. Pamiętać trzeba jednak, że do uzyskania najlepszych rezultatów, wymagany jest świadomy wybór i przygotowanie danych treningowych i testowych.

Do realizacji projektu wykorzystana została baza zdjęć „The Chars74K dataset”.

Dataset “The Chars74K dataset”, Character recognition in natural images (źródło: <http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/>, [dostęp 1.12.2016])

Jest to mieszany zbiór liter, utworzonych z czcionek komputerowych pisanych przy użyciu wariacji kursywy, pogrubienia i normalnie. Jest to ogólnodostępny zbiór danych, który w tym przypadku imitować miał odręcznie pisane litery, występujące w formularzach tabelarycznych.

Słowa kluczowe: Sztuczne sieci neuronowe, Analiza danych, Sztuczna inteligencja, Nauczanie maszynowe, MATLAB

# Summary

## **ARTIFICIAL NEURAL NETWORKS SUPPORTING RECOGNITION OF IMAGES, BY IMPLEMENTING THE AUTOMATIC RECOGNITION OF WRITTEN CHARACTERS.**

The main point of the below project is to present the process of implementing an algorithm of data analysis, but to be more precise it is about recognition of written characters. The solution of the given task refers to a way of automation of the process of loading tabular forms and its translation into digital data (computer signature).

Nevertheless, machine learning as the method of shapes recognition is a difficult task, once implemented can be extremely useful in the future. Algorithm of the above thesis is based on Multi-Layered Perceptron neural networks. Although, the implemented actions solve the problem of images recognition, by recognizing written characters, in the same way artificial neural network can be trained to recognize different objects, such as shapes on the conveyor belt.

In the following chapters, there is information about historical features of artificial neural networks, comparison of human neurons and its mathematical equivalent. Subsequently, the chapters are about the types of neural networks and their training methods, more accurate description of the problem, programming environment and dataset. Ending with the depiction of implementation process and final summary of the achieved solution.

In this case, final form of artificial neural network has been created, tested and analyzed in different conditions and with different parameters. Lastly, satisfying results has been reached with the accuracy up to 93%. The main aim of this project has been solved using artificial neural networks from the group of supervised machine learning algorithms. Outcomes that can be obtained show, that artificial neural networks can be a good tool for data analyzing. One thing to remember, is that, to achieve the best results, conscious choice and pre-preparation of training and test datasets are necessary.

To accomplish this project „The Chars 74K dataset” has been used.

Dataset “The Chars74K dataset”, Character recognition in natural images (source: <http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/>, [access 1.12.2016])

It is a mixed set of letters, created from computer fonts with variations of italic, bold and normal forms. It is a public dataset, which in this case has to imitate handwritten letters, which commonly appear in the tabular applications.

Key words: Artificial neural networks, Data analysis, Artificial intelligence, Machine learning, MATLAB

### **Oświadczenie autora (autorów) pracy:**

Świadom odpowiedzialności prawnej oświadczam, że przedstawiona praca dyplomowa:

- została napisana przeze mnie samodzielnie i nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami,
- nie była wcześniej przedmiotem procedur związanych z uzyskaniem tytułu zawodowego lub stopnia naukowego w wyższej uczelni.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

.....

data

.....

podpis autora (autorów) pracy

### **Oświadczenie**

Wyrażam zgodę na udostępnianie osobom zainteresowanym mojej pracy dyplomowej. Praca może być udostępniana w pomieszczeniach biblioteki wydziałowej. Zgoda na udostępnienie pracy dyplomowej nie oznacza wyrażenia zgody na jej kopiowanie w całości lub w części. Brak zgody nie oznacza ograniczenia dostępu do pracy dyplomowej osób:

- reprezentujących władze Politechniki Warszawskiej,
  - członków Komisji Akredytacyjnych,
  - funkcjonariuszy służb państwowych i innych osób uprawnionych, na mocy odpowiednich przepisów prawnych obowiązujących na terenie Rzeczypospolitej Polskiej, do swobodnego dostępu do materiałów chronionych międzynarodowymi przepisami o prawach autorskich.
- Brak zgody nie wyklucza także kontroli tekstu pracy dyplomowej w systemie antyplagiatowym.

.....

data

.....

podpis autora (autorów) pracy

# Spis Treści

<b>1. Wstęp</b>	<b>1</b>
1.1. Historia sztucznych sieci neuronowych	1
1.2. Budowa neuronu	2
1.2.1. Neuron, jako komórka nerwowa	2
1.2.2. Sztuczny neuron	4
1.3. Sztuczna sieć neuronowa	6
1.3.1. Rodzaje sztucznych sieci neuronowych	6
1.3.2. Sieci jednokierunkowe	7
1.3.3. Sieci rekurencyjne	9
1.3.4. Sieci komórkowe	10
1.3.5. Sieci hybrydowe	11
1.4. Metody nauczania sztucznych sieci neuronowych	11
1.4.1. Uczenie z nauczycielem	12
1.4.2. Uczenie bez nauczyciela	12
1.4.3. Back-Propagation metoda uczenia sztucznych sieci neuronowych	13
<b>2. Opis zagadnienia</b>	<b>14</b>
<b>3. Środowisko programistyczne</b>	<b>16</b>
3.1. Schemat pracy w programie MATLAB	16
3.2. MATLAB – Neural Network Toolbox	17
<b>4. Zbiór danych</b>	<b>18</b>
4.1. Opis zbioru	18
4.2. Sposób wczytania zdjęć do środowiska MATLAB	19
4.2.1. Opis funkcji LoadImages	19
4.2.2. Opis skryptu CreateAlphabet	19
4.2.3. Opis funkcji DummyLabels	20
4.3. Dane wykorzystane do treningu, walidacji i testu SSN	21

4.3.1.	Test na 1000 niezależnych próbkach .....	21
5.	<b>Stworzenie sztucznej sieci neuronowej (SSN)</b> .....	22
5.1.	Opis funkcji TestCheck.....	22
5.2.	Opis skryptu RegressionPlot .....	23
5.3.	Opis Confusion matrix .....	24
6.	<b>Implementacja rozwiązania problemu</b> .....	25
6.1.	SSN typu MLP z 1 warstwą ukrytą o 100 neuronach .....	25
6.1.1.	Dodatkowa walidacja zbiorem 1000 próbek .....	26
7.	<b>Rozwój i doskonalenie rozwiązania problemu</b> .....	28
7.1.	SSN typu MLP z 1 warstwą ukrytą o 717 neuronach .....	29
7.1.1.	Dodatkowa walidacja zbiorem 1000 próbek .....	30
7.2.	SSN typu MLP z 2 warstwami ukrytymi o 50 i 10 neuronach .....	31
7.2.1.	Dodatkowa walidacja zbiorem 1000 próbek .....	33
8.	<b>Przemodelowanie zbioru danych</b> .....	34
9.	<b>Wyniki końcowe</b> .....	35
9.1.	Confusion Matrix .....	35
9.2.	Dodatkowa walidacja zbiorem 1000 próbek .....	35
9.2.1.	Obserwacje .....	36
10.	<b>Rozwiązanie końcowe</b> .....	37
11.	<b>Opis funkcji i skryptów</b> .....	40
11.1.	CreateAlphabet .....	40
11.2.	LoadImages .....	40
11.3.	DummyLabels .....	40
11.4.	SiecNeuronowaAlphabet.....	40
11.5.	RegressionPlot.....	40
11.6.	TestCheck.....	40
11.7.	SymulacjaSSN .....	41

11.8.	PrepareImage .....	41
11.9.	Conversion.....	41
11.10.	ShowSignature .....	41
12.	<b>Podsumowanie</b> .....	42
13.	<b>Bibliografia</b> .....	43
14.	<b>Spis rysunków</b> .....	46





# 1. Wstęp

Rozdział ten opracowany został na podstawie dwóch źródeł tj. książki pt: „Sieci neuronowe” oraz historii sztucznych sieci neuronowych zaprezentowanej przez katedrę inżynierii komputerowej Politechniki Częstochowskiej.

[1] R. Tadeusiewicz, Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa 1993.

[2] Sieci neuronowe – historia, Katedra inżynierii komputerowej Politechniki Częstochowskiej, (źródło: <http://iisi.pcz.pl/nn/historia.php#home>, [dostęp 11.01.2017])

## 1.1. Historia sztucznych sieci neuronowych

Dziedzina sztucznych sieci neuronowych narodziła się dopiero wraz z wydaniem historycznej pracy McCulloch’a i Pitts’a w 1943 roku. Opublikowali oni bowiem opis mechanizmów działania mózgu oraz pojedynczych komórek układu nerwowego. Przedstawiony przez nich model odegrał wielką rolę w późniejszym rozwoju tej dziedziny. W roku 1949 Donald Hebb, dokonał odkrycia, że informacja może być przechowywana w strukturze połączeń pomiędzy neuronami, a następnie jako pierwszy zaproponował metodę uczenia sieci polegającą na zmianach wag połączeń między neuronami (reguła Hebba). Następnie w latach 50 XX wieku zaczęto budować pierwsze sieci neuronowe.

Wraz z pojawieniem się komputerów, badania nad sztucznymi sieciami neuronowymi, zaczęły rozwijać się w różnych kierunkach. Początkowo, gdy komputery nie dysponowały wystarczającą zdolnością obliczeniową, idea sztucznych sieci neuronowych nastawiona była głównie na zagadnienia teoretyczne. To spowodowało spowolnienie rozwoju tej dziedziny nauki. Z biegiem czasu, gdy komputery zaczęły mieć coraz większą moc obliczeniową, zainteresowanie sieciami neuronowymi ponownie wróciło do łask naukowców, a ilość typów sztucznych sieci neuronowych zwiększyła się zauważalnie.

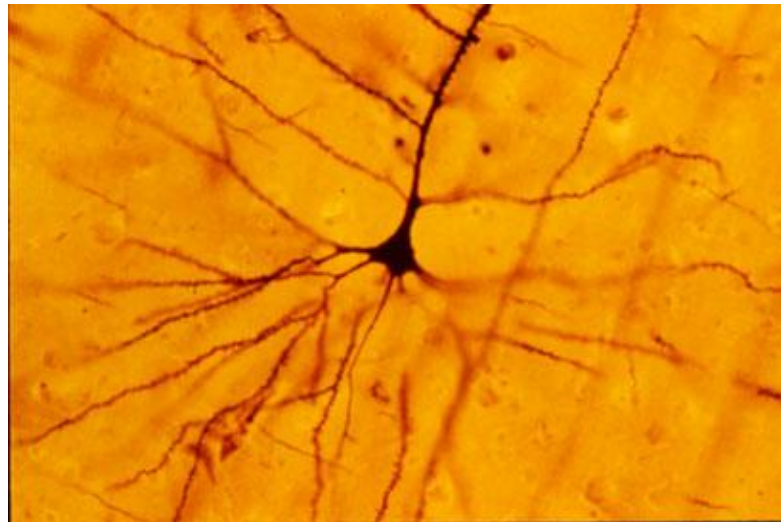
Fakt ten spowodował wzrost zainteresowania nimi w branży przemysłowej, militarnej, czy informatycznej. Współczesne systemy komputerowe zaczynają przetwarzać coraz większą ilość informacji, starając się je analizować i dostrzegać w nich różnego rodzaju zależności. Wiadome jest jedno, niezależnie od tego, jakiego charakteru są to dane, przejawiają się w nich nieprawidłowości i zakłócenia, najczęściej niezwiązane z tematem, którego dotyczą. Przykładami zakłóceń mogą być: warunki

otoczenia, w których pomiary zostały wykonane, różne parametry fizyczne, szумы, czy konfiguracja aparatury pomiarowej. Wszystko to może przyczyniać się negatywnie do jakości pracy systemów analizujących te dane. W celu redukcji ich niepożądanego efektu dokonuje się normalizacji danych, czyli takiego ich przekształcenia, aby pozbyć się niechcianych różnic.

Próba zawarcia w systemie wiedzy o tym, jak wygląda pewna klasa analizowanych danych, jest nauczanie sztucznej sieci neuronowej.

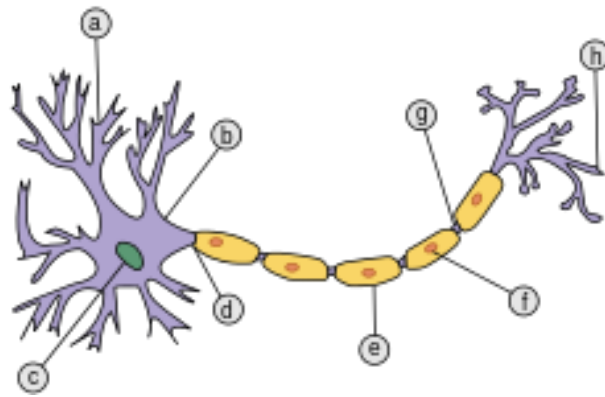
## **1.2. Budowa neuronu**

### **1.2.1. Neuron, jako komórka nerwowa**



1. Neuron piramidowy z kory mózgowej człowieka

Neuron jest rodzajem komórki pobudzanej impulsem elektrycznym, zdolnej do przetwarzania i przewodzenia informacji w postaci sygnału elektrycznego. Jest to podstawowy budulec układu nerwowego.



2. Schemat budowy neuronu ludzkiego

Typowa komórka nerwowa zbudowana jest z następujących składników:

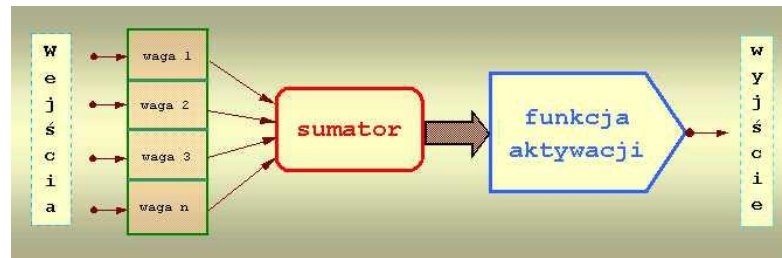
- a) Dendryt
- b) Ciało komórki
- c) Jądro komórkowe
- d) Akson
- e) Otoczka mielinowa
- f) Komórka Schwanna
- g) Przewężenie Renviera
- h) Zakończenia aksonu

[3] Neuron – Wikipedia, the free encyclopedia. (źródło: <https://pl.wikipedia.org/wiki/Neuron>, [dostęp 1.12.2016])

Neurony połączone w grupy tworzą sieci neuronowe, kontaktując się między sobą poprzez synapsy, czyli połączenia błony kończące jeden akson z błoną komórkową drugiej komórki nerwowej lub komórki efektor (na przykład narządu wykonawczego tj. mięśni lub gruczołu). Mechanizm odbierania informacji od innych neuronów polega na odbieraniu impulsu przez synapsy położone na dendrytach, przewodzeniu go wzdłuż neuronu i przekazywaniu sygnału dalej do synaps na zakończeniach aksonu.

### 1.2.2. Sztuczny neuron

Sztuczny neuron zaprojektowany został na podobieństwo neuronu naturalnego. Jest on prostym system przetwarzającym wartości sygnałów podane na jego wejścia w pojedynczą wartość wyjściową, która wysyłana jest na jego jedynym wyjściu. Wejścia to odpowiedniki dendrytów, wagi to cyfrowe odpowiedniki modyfikacji dokonywanych na sygnałach przez synapsy. Blok sumujący to odpowiednik jądra, blok aktywacji to wzgórek aksonu, a wyjście to akson.



3. Budowa sztucznego neuronu

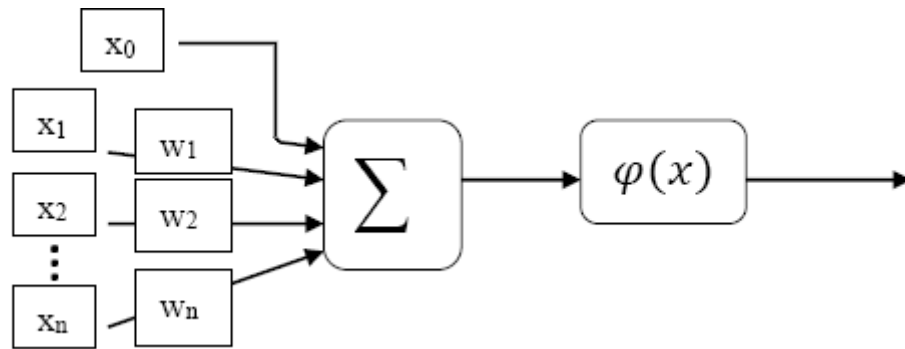
[4] Sieci neuronowe – C-Labtech. (źródło: <http://www.ai.c-labtech.net/sn/index.html>, [dostęp 1.12.2016])

Mechanizm przetwarzania sygnału w sztucznym neuronie można opisać następująco:

*Wejścia dostarczają sygnał, który jest wymnażany poprzez współczynniki wag, następnie w bloku sumującym występuje sumowanie pomnożonych sygnałów. Wynikiem czego jest sygnał zwany potencjałem membranowym. Następnie sygnał jest przetwarzany w bloku aktywacji, który w zależności od przeznaczenia może być opisany różnymi funkcjami – zwanymi funkcjami aktywacji. Wartość funkcji aktywacji jest sygnałem wyjściowym neuronu i propagowana jest do neuronów warstwy następnej.*

[5] Budowa neuronu – Instytut informatyki, Politechnika Poznańska. (źródło: <http://www.cs.put.poznan.pl/rklaus/assn/neuron.htm>, [dostęp 1.12.2016])

Tak zaimplementowany neuron jest podstawowym elementem budującym sztuczne sieci neuronowe, które są jedną z metod implementacji sztucznej inteligencji.



4. Neuron - schemat matematyczny

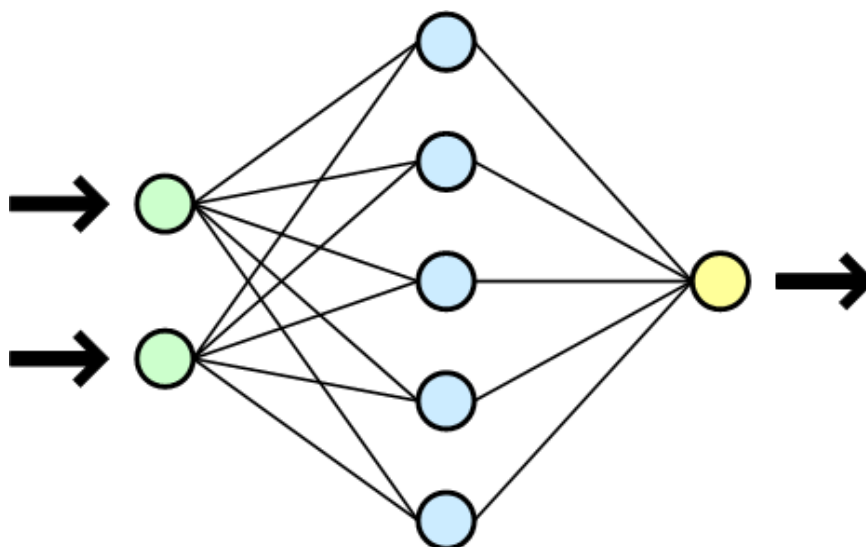
Matematyczny opis działania neuronu:

- $X_0$  – bias neuronu
- $X_1, X_2, \dots, X_n$  – sygnały wejściowe
- $W_1, W_2, \dots, W_n$  – wagi odpowiadające sygnałom wejściowym
- $\varphi(x)$  – funkcja aktywacji
- $y$  – sygnał wyjściowy

$$y = \varphi\left(\sum_{i=1}^n x_i \cdot w_i + x_0\right)$$

### 1.3. Sztuczna sieć neuronowa

Sztuczna sieć neuronowa to uogólniona nazwa struktur matematycznych oraz ich pochodnych modeli sprzętowych lub programowych, realizujących obliczenia lub przetwarzanie sygnałów poprzez rzędy elementów, które poznaliśmy już do tej pory tj. sztuczne neurony. Czasem nazwą sztuczne sieci neuronowe określana jest interdyscyplinarna dziedzina nauki zajmująca się budową, trenowaniem i badaniem możliwości tego rodzaju sieci.



5. Uproszczony schemat jednodukierunkowej sieci neuronowej (kółka oznaczają sztuczne neurony)

[6] Sieć Neuronowa – Wikipedia, the free encyclopedia. (źródło: [https://pl.wikipedia.org/wiki/Sieć\\_neuronowa](https://pl.wikipedia.org/wiki/Sieć_neuronowa), [dostęp 1.12.2016])

#### 1.3.1. Rodzaje sztucznych sieci neuronowych

Choć istnieje wiele typów sztucznych sieci neuronowych, ich cechą wspólną jest to, że na ich strukturę składają się neurony połączone ze sobą synapsami. Rodzaj sieci neuronowej zależy od sposobu, w jaki neurony są ze sobą połączone oraz od kierunku przepływu sygnałów. Każdy typ sieci ma własne metody doboru wag, czyli tak zwanego uczenia się.

Istnieje wiele rodzajów sztucznych sieci neuronowych, jednak do najbardziej podstawowych, przedstawiających budowę oraz sposób działania zaliczamy:

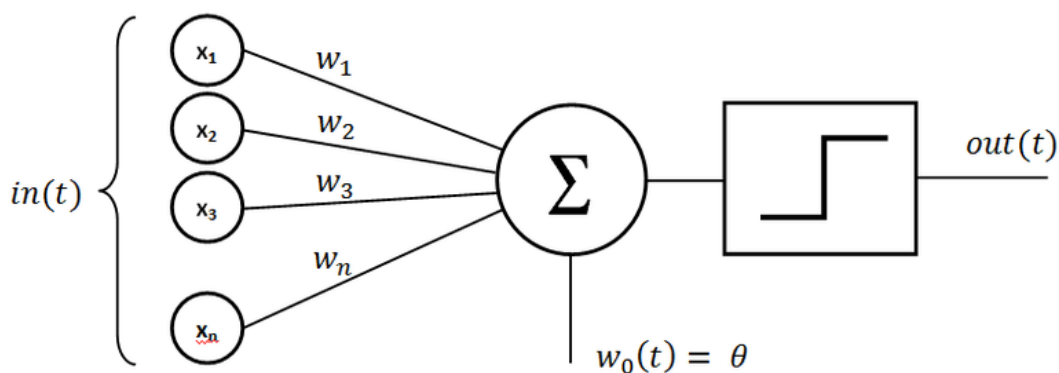
- Sieci jednokierunkowe
  - ❖ Jednowarstwowe
  - ❖ Wielowarstwowe
- Sieci rekurencyjne
- Sieci komórkowe
- Sieci hybrydowe

### 1.3.2. Sieci jednokierunkowe

Sieci jednokierunkowe, czyli takie, w których nie występuje sprzężenie zwrotne, co oznacza, że pojedynczy wzorzec lub sygnał przechodzi przez każdy z neuronów dokładnie raz w swoim cyklu.

#### Sieci jednowarstwowe:

Najprostszą siecią jednowarstwową jest pojedynczy perceptron progowy, który został opracowany przez McCullocha i Pittsa w roku 1943.



6. Perceptron złożony z jednego neuronu McCullocha-Pittsa

[7] Perceptron – Wikipedia, the free encyclopedia (źródło: <http://pl.wikipedia.org/wiki/Perceptron>, [dostęp 1.12.2016])

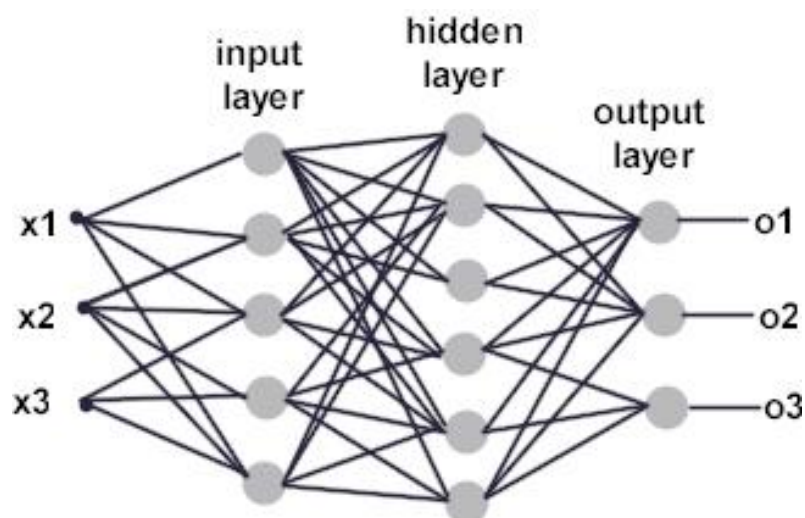
#### Sieci wielowarstwowe:

Jednokierunkowe, wielowarstwowe sieci neuronowe, czyli tak zwane sieci typu MLP (ang. Multi-Layered Perceptron – Perceptron wielowarstwowy), są najpopularniejszą i najchętniej wykorzystywaną w zastosowaniach praktycznych architekturą neuronową. Upowszechnienie tego rodzaju sieci związane jest z opracowaniem

algorytmu wstecznej propagacji błędów (ang. Error Back Propagation), który umożliwił skuteczne trenowanie tego typu sieci w stosunkowo prosty sposób.

Sieć typu MLP jest w stanie przybliżać dowolnie złożone i skomplikowane odwzorowanie. Podczas, gdy użytkownik nie musi znać lub z góry zakładać żadnej formy występujących w poszukiwanym modelu zależności. Ta funkcjonalność w połączeniu z samodzielną metodą uczenia się sieci neuronowej, czyni z niej niezwykle użyteczne i wygodne narzędzie do wszelkiego rodzaju zastosowań, przykładowo związanych z prognozowaniem, klasyfikacją, czy automatycznym sterowaniem. Ponadto, kolejną zaletą jest duża tolerancja systemu neuronowego na uszkodzenia pojedynczych elementów, co podnosi niezawodność w sposób praktycznie nieosiągalny w przypadku tradycyjnego oprogramowania, np.: klasycznych systemów ekspertowych. Lista zalet tego rodzaju sieci jest długa, równoległy model przetwarzania sygnałów wejściowych i stosunkowo prosta struktura nauczonej sieci neuronowej pozwala osiągnąć bardzo szybki czas odpowiedzi na sygnał, pozwalający na wykorzystywanie jej w problemach wymagających reagowania w czasie rzeczywistym.

[8] Multilayer perceptron – Wikipedia, the free encyclopedia (źródło: [https://en.wikipedia.org/wiki/Multilayer\\_perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron), [dostęp 1.12.2016])



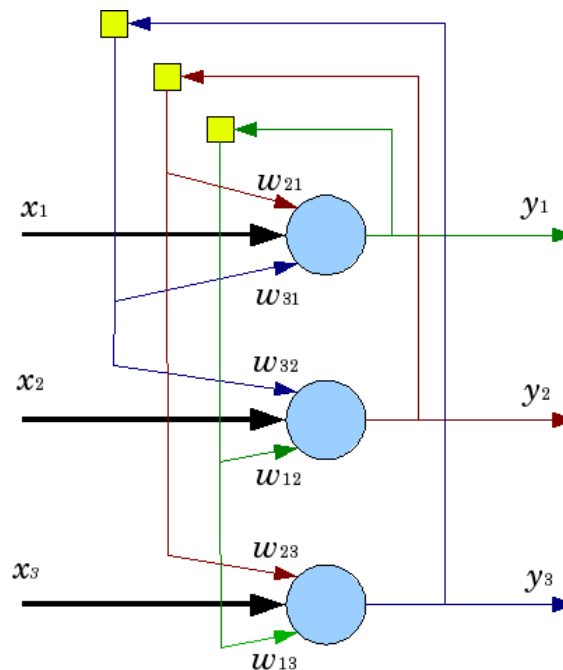
7. Multi-Layered Perceptron z jedną warstwą ukrytą

[9] Multi-Layered Perceptron, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])



### 1.3.3. Sieci rekurencyjne

Sieci rekurencyjne są modelem, w którym występuje przynajmniej jedno sprzężenie zwrotne. Oznacza to, że sygnały wyjściowe warstwy podawane są na jej wejścia, co skutkuje pewną dynamiką w pracy sieci. Sygnały wejściowe w takiej sieci zależą zarówno od aktualnego stanu wejścia, lecz także od sygnałów wyjściowych w poprzednim cyklu. Ogólną strukturę takiej sztucznej sieci neuronowej przedstawia poniższa grafika.



8. Sieć rekurencyjna Hopfielda

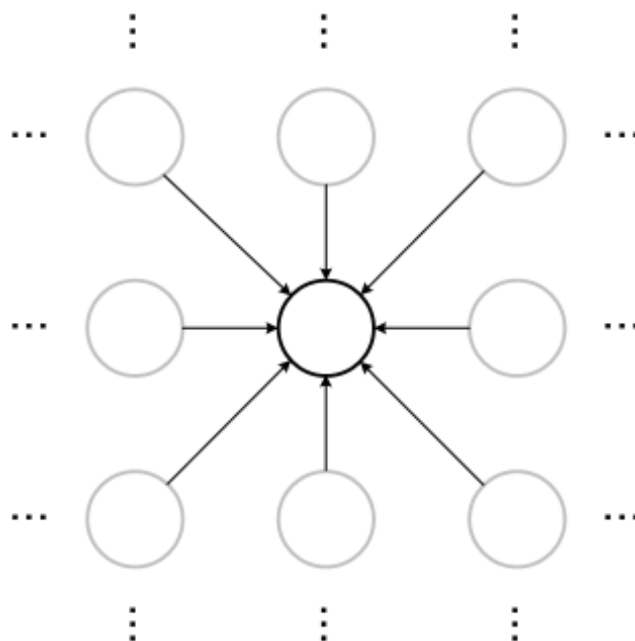
[10] Sieć Hopfielda, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])

Sieć Hopfielda opracowana przez amerykańskiego fizyka Johna Hopfielda w 1982 roku. Ten rodzaj sztucznej sieci neuronowej wykorzystywany jest do modelowania pamięci skojarzeniowej. Taka sieć daje możliwość rekonstrukcji i rozpoznawania wcześniej zapamiętanych wzorców na podstawie skojarzeń, bazując na dostępnym fragmencie wzorca lub wzorca podobnego do niego. Strukturę sieci Hopfielda w prosty sposób można opisać jako układ wielu identycznych elementów połączonych ze sobą każdy z każdym. Sieć ta używana jest najczęściej jako struktura jednowarstwowa. W odróżnieniu od sieci wielowarstwowych typu MLP, sieć Hopfielda jest siecią rekurencyjną, gdzie neurony są wielokrotnie pobudzane w jednym cyklu rozpoznawania poprzez użycie pętli sprzężenia zwrotnego.

[11] Sieć asocjacyjna – Wikipedia, the free encyclopedia (źródło: [https://pl.wikipedia.org/wiki/Sieć\\_asocjacyjna](https://pl.wikipedia.org/wiki/Sieć_asocjacyjna), [dostęp 1.12.2016])

#### 1.3.4. Sieci komórkowe

W sieciach komórkowych sprzężenia wzajemne między elementami przetwarzającymi, dotyczą jedynie najbliższego sąsiedztwa. Połączenia te są w ogólności nieliniowe i opisane poprzez układ równań różniczkowych. Podstawową trudność w stosowaniu tego typu sieci stanowi opracowanie skutecznej, efektywnej i uniwersalnej metody projektowania. Typowym przykładem sieci komórkowej może być sieć typu mapa Kohonena.



9. Schemat sztucznej sieci komórkowej

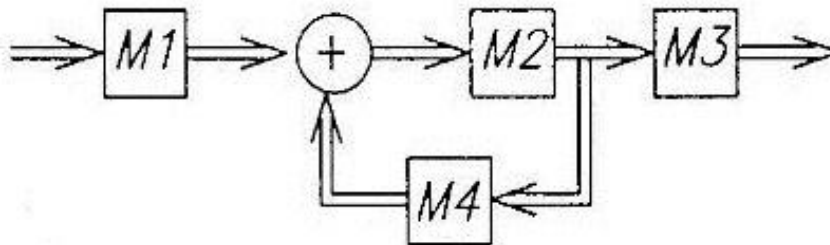
[12] Sztuczna sieć komórkowa, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])

Sieć Kohonena, nazwa pochodzi od nazwiska fińskiego uczonego Teuvo Kohonena. Rodzaj sztucznej sieci neuronowej uczonej w trybie bez nauczyciela w celu wytworzenia niskowymiarowej (przeważnie dwuwymiarowej) dyskretyzowanej reprezentacji przestrzeni wejściowej. Sieć Kohonena różni się tym od innych sieci, że zachowuje odwzorowanie sąsiedztwa obszaru wejściowego. Wynikiem działania sieci jest klasyfikacja przestrzeni w sposób grupujący zarówno przypadki ze zbioru uczącego, jak i wszystkie inne wprowadzenia po procesie uczenia.

[13] Rodzaje sztucznych sieci neuronowych – Instytut informatyki, Politechnika Poznańska. (źródło: <http://www.cs.put.poznan.pl/rklaus/assn/neuron.htm>, [dostęp 1.12.2016])

### 1.3.5. Sieci hybrydowe

Sieci hybrydowe, czyli połączenia sieci warstwowych z rekurencyjnymi.



10. Schemat sieci hybrydowej

[14] Sieć hybrydowa, (źródło: <http://www.twins.pk.edu.pl/~pkowal/SSN/rodzaje.htm/> [dostęp 1.12.2016])

## 1.4. Metody nauczania sztucznych sieci neuronowych

Podrozdział ten głównie bazuje na artykule dotyczącym sztucznych sieci neuronowych dostępnym na stronie <http://www.poltynk.pl/marcin/uczenie.html>.

[15] Sztuczne sieci neuronowe, (źródło: <http://www.poltynk.pl/marcin/uczenie.html>, [dostęp 1.12.2016])

Sposób w jaki działają sieci neuronowe podzielić można na etap nauki, kiedy sieć gromadzi informacje potrzebne jej do określenia, co i w jaki sposób ma robić, oraz na etap normalnego działania (nazywany czasem także egzaminem), kiedy w oparciu o zdobytą wiedzę sieć musi rozwiązywać konkretne nowe zadania. Możliwe są dwa warianty procesu uczenia:

- Uczenie z nauczycielem
- Uczenie bez nauczyciela

#### **1.4.1. Uczenie z nauczycielem**

Uczenie z nauczycielem jest algorytmem polegającym na tym, że sztucznej sieci neuronowej podaje się przykłady poprawnego działania, które powinna ona potem naśladować w swoim bieżącym działaniu (w czasie egzaminu). Co należy rozumieć w ten sposób, że nauczyciel podaje konkretne sygnały wejściowe i wyjściowe. Pokazując w ten sposób sieci, jaka jest wymagana odpowiedź dla pewnej konfiguracji danych wejściowych. Mamy w tym wypadku do czynienia z parą wartości tj. przykładowym sygnałem wejściowym i pożądanym (oczekiwanym) wyjściem, czyli wymaganą odpowiedzią sieci na ten sygnał wejściowy. Zbiór przykładów zgromadzonych w celu ich wykorzystania w procesie uczenia sieci nazywa się zwykle ciągiem uczącym, lub zbiorem treningowym. Zatem w typowym procesie uczenia, sieć otrzymuje od nauczyciela ciąg uczący i na jego podstawie uczy się prawidłowego działania, stosując jedną z wielu znanych dziś strategii uczenia.

#### **1.4.2. Uczenie bez nauczyciela**

Oprócz przedstawionego wyżej procesu uczenia z nauczycielem, występuje też szereg metod tak zwanego uczenia bez nauczyciela (albo samouczenia sieci). Metoda nauki bez nauczyciela polega na podawaniu na wejście sieci wyłącznie szeregu przykładowych danych wejściowych, bez podawania jakiegokolwiek informacji dotyczącej pożądaných czy chociażby tylko oczekiwanych sygnałów wyjściowych. Odpowiednio zaprojektowana sieć neuronowa potrafi wykorzystać wyłącznie obserwacje sygnałów wejściowych i zbudować na ich podstawie sensowny algorytm swojego działania - najczęściej polegający na tym, że automatycznie wykrywano są klasy powtarzających się sygnałów wejściowych i sieć uczy się (spontanicznie, bez jawnego nauczania) rozpoznawać te typowe wzorce sygnałów.

Uczenie bez nauczyciela jest też bardzo interesujące z punktu widzenia zastosowań, ponieważ nie wymaga żadnej uprzednio podawanej do sieci neuronowej zewnętrznej wiedzy, a sieć w procesie samouczenia zgromadzi wszystkie potrzebne informacje oraz wiadomości.

### **1.4.3. Back-Propagation metoda uczenia sztucznych sieci neuronowych**

Poniżej opisana została przykładowa, najczęściej stosowana metoda uczenia, wykorzystująca uczenie z nauczycielem. Algorytm ten został przeze mnie użyty do opracowywania rozwiązania problemu, na którym skupia się niniejsza praca.

#### Metoda Back Propagation

Pierwszą czynnością w tym procesie nauczania jest przygotowanie dwóch zbiorów danych: uczącego i weryfikującego (testowego). Ciąg uczący składa się z danych, które dokładnie charakteryzują dany problem. Jednorazowa porcja danych nazywana jest wektorem uczącym. W jego skład wchodzi wektor wejściowy, czyli dane podawane na wejścia sztucznej sieci neuronowej oraz wektor wyjściowy, czyli dane oczekiwane, jakie sieć powinna wygenerować na swoich wyjściach. Po przetworzeniu wektora wejściowego, nauczyciel porównuje wartości otrzymane z wartościami oczekiwanymi. Następnie informuje sieć, czy odpowiedź jest poprawna, a jeżeli nie, to jaki powstał błąd odpowiedzi. Błąd ten jest propagowany do sieci „wstecznie”, następuje to bowiem w odwrotnej kolejności niż wektor wejściowy (od warstwy wyjściowej do wejściowej), na jego podstawie następuje taka korekcja wag w każdym neuronie, aby ponowne przetworzenie tego samego wektora wejściowego spowodowało zmniejszenie błędu odpowiedzi. Proces ten powtarza się do momentu wygenerowania przez sieć błędu mniejszego niż założony. Wtedy na wejście sieci podaje się kolejny wektor wejściowy i powtarza te czynności.

Jeżeli sieć została już nauczona, potrzebna jest weryfikacja jej działania. W tym momencie na wejście sieci podawane są wzorce spoza zbioru treningowego, w celu zbadania czy sieć może efektywnie generalizować zadanie, którego się nauczyła. Do tego stosowany jest ciąg testowy, który ma te same cechy co ciąg uczący, tzn. dane dokładnie charakteryzują problem i znamy dokładne odpowiedzi. Ważne jest jednak, aby dane nie były używane do wcześniejszego nauczania. W tym przypadku prezentujemy ciąg weryfikujący, z tą różnicą, że nie rzutujemy błędów wstecz, jedynie porównujemy ilość odpowiedzi poprawnych i na tej podstawie orzekamy, czy sieć spełnia nasze wstępne założenia.

[16] Sieci neuronowe – C-Labtech. (źródło: <http://www.ai.c-labtech.net/sn/sneuro.html>, [dostęp 1.12.2016])

## 2. Opis zagadnienia

Celem pracy było opracowanie algorytmu służącego do automatyzacji procesu rozpoznawania odręcznego pisma ludzkiego z formularzy tabelarycznych oraz tłumaczenie go na pismo cyfrowe, które później może zostać wykorzystane do ułatwienia i usprawnienia przebiegu procesów występujących w życiu codziennym. Mianowicie, algorytm ma szereg zastosowań w takich miejscach jak np.:

- Banki
- Urzędy Pocztowe
- Instytucje Państwowe

The image shows a blank Polish bank transfer form (Polecenie przelewu). The form is divided into several sections with red borders and labels. At the top, there are fields for 'nazwa odbiorcy' (recipient name) and 'nazwa odbiorcy od' (recipient address). Below these are fields for 'nr rachunku odbiorcy' (recipient account number) and 'nr rachunku zlecającego (przelew) / kwota słownie (wpłata)' (sender account number / amount in words). The central part of the form contains fields for 'nazwa zlecającego' (sender name) and 'nazwa zlecającego od' (sender address). Below these are fields for 'tytułem' (for) and 'tytułem od' (from). The bottom section contains a large box for 'pieczęć, data i podpis(y) zlecającego' (stamp, date, and signature(s) of the sender). To the right of this box is a circular stamp area and a small box for 'Opłata' (fee). The form also features a vertical label on the left side: 'Polecenie przelewu / wpłata gotówkowa' and a vertical label on the right side: 'odcinek dla banku zlecającego'. The currency 'PLN' is printed in the center, and the number '06' is visible in the bottom right corner.

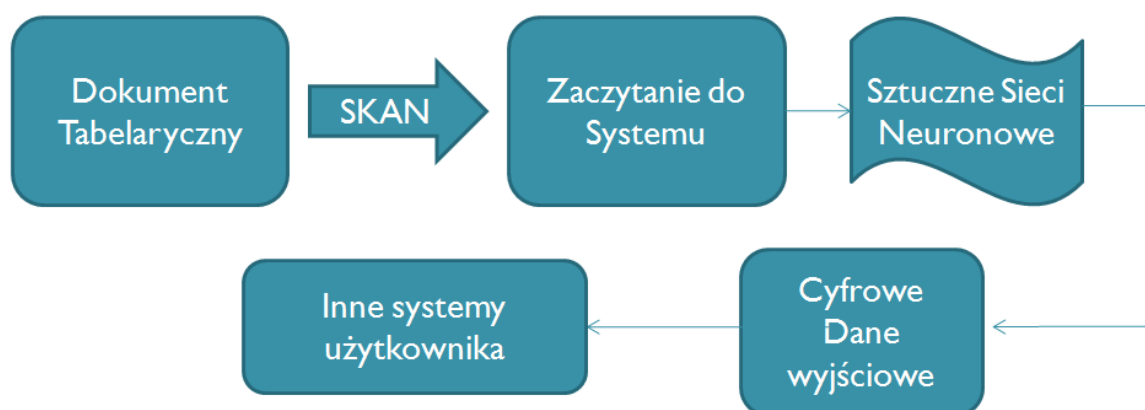
11. Polecenie przelewu

[17] Polecenie przelewu– Wikipedia, the free encyclopedia. (źródło: [https://pl.wikipedia.org/wiki/Polecenie\\_przelewu](https://pl.wikipedia.org/wiki/Polecenie_przelewu), [dostęp 1.12.2016])

12. Przesyłka pocztowa

[18] Przesyłka pocztowa, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])

Przybliżając zastosowanie, algorytm ma na celu zautomatyzowanie procesu pobierania danych z powszechnie znanych dokumentów tabelarycznych: ich skan, wczytanie do utworzonego systemu, analizę poprzez sztuczne sieci neuronowe i wykorzystanie uzyskanych wyjściowych danych cyfrowych w innych systemach użytkownika.



13. Algorytm automatyzacji procesu

### 3. Środowisko programistyczne

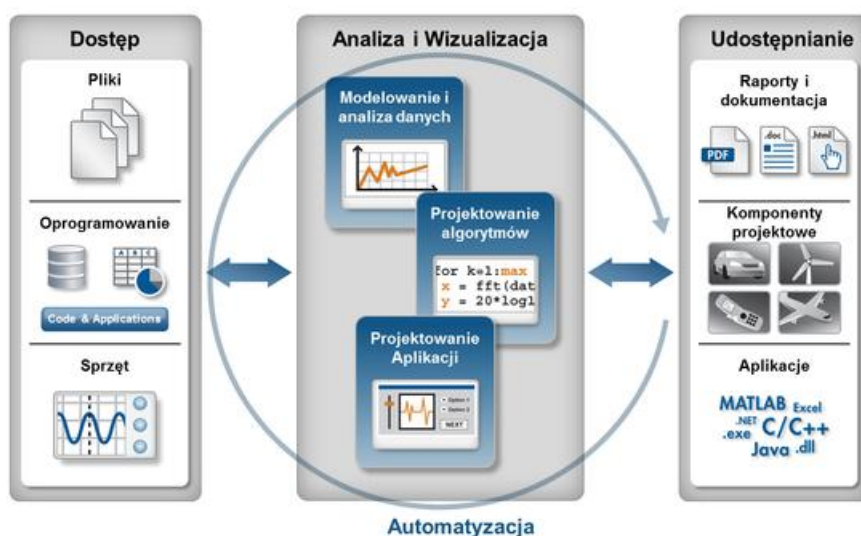
Przy tworzeniu algorytmu autor posłużył się środowiskiem MATLAB R2016a, do którego od roku 2016 dostępna jest darmowa licencja studencka. W głównej mierze wykorzystany został Neural Network Toolbox, czyli dodatek wspierający użycie sztucznych sieci neuronowych.

MATLAB to środowisko programistyczne przeznaczone do rozwijania algorytmów, wizualizacji i analizy danych oraz prowadzenia obliczeń numerycznych. Dzięki programowi MATLAB problemy można rozwiązywać szybciej, niż przy wykorzystaniu tradycyjnych języków programowania takich jak C, C++ czy Fortran. MATLAB może mieć wiele zastosowań. Przykładowe obszary, w których program jest często podstawowym narzędziem pracy to:

- Przetwarzanie sygnałów
- Przetwarzanie obrazów
- Telekomunikacja
- Projektowanie układów sterowania
- Matematyka finansowa

#### 3.1. Schemat pracy w programie MATLAB

##### Schemat pracy w programie MATLAB



14. Schemat pracy w programie MATLAB

[19] Oprogramowanie naukowo techniczne (źródło:  
<http://www.ont.com.pl/produkty/lista-produktow/matlab/>, [dostęp 1.12.2016])



### **3.2. MATLAB – Neural Network Toolbox**

Neural Network Toolbox zapewnia algorytmy, funkcje i aplikacje do tworzenia, trenowania, wizualizacji i symulacji sztucznych sieci neuronowych. Użytkownik ma możliwość łatwej implementacji takich procesów jak:

- Klasyfikacja (ang. Classification)
- Regresja (ang. Regression)
- Grupowanie (ang. Clustering)
- Redukcja wymiarowości (ang. Dimensionality reduction)
- Prognozowanie szeregów czasowych (ang. Time-series forecasting)
- Modelowanie i kontrola systemów dynamicznych (ang. Dynamic system modeling and control)

## 4. Zbiór danych

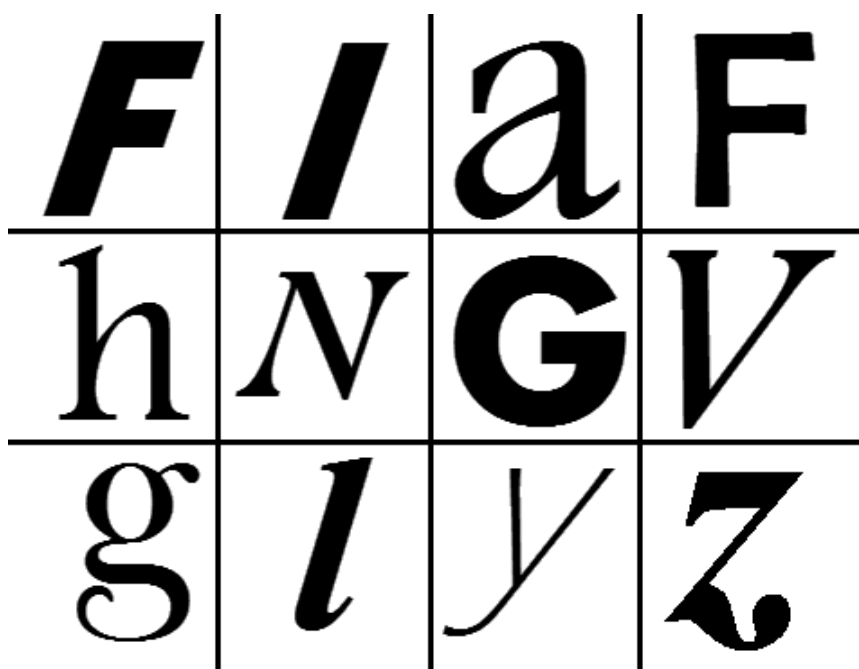
### 4.1. Opis zbioru

Zbiór danych jest podstawową rzeczą podczas treningu sieci neuronowych. Dobrze zorganizowany dataset, bez szumów i innych złośliwych czynników wpływających na trening to podstawa. Zdobycie, bądź własnoręczne stworzenie odpowiedniego zbioru liter dużych i małych, z alfabetu od A do Z (bez polskich znaków), to dosyć trudne i czasochłonne zajęcie. Na potrzeby pracy skorzystano z dostępnego zbioru liter, który został utworzony z 1016 różnych czcionek, a następnie każda z liter została zapisana w formacie pliku .png w rozmiarze 128x128 pixeli. Wynika z tego, że do dyspozycji jest 1016 różnych zdjęć każdej z liter, co daje 26 x 1016 litery duże A-Z oraz 26 x 1016 litery małe a-z.

Łączny rozmiar tak uzyskanego zbioru to:

$$(2 \times 26) \times 1016 = 52832$$

Przykładowe obrazy ze zbioru:



15. Litery ze zbioru danych

## 4.2. Sposób wczytania zdjęć do środowiska MATLAB

Na potrzeby wczytania tak dużego zbioru zdjęć do środowiska MATLAB, stworzono skrypt o nazwie CreateAlphabet oraz funkcje LoadImages i DummyLabels.

### 4.2.1. Opis funkcji LoadImages

Funkcja jest zbudowana w następujący sposób:

```
function [ Letter ] = LoadImages( fullFileName, img_nr )
```

Pobierane są dwie wartości:

- fullFileName: czyli ścieżka do folderu, w którym znajduje się 1016 próbek danej litery. Przykład:
  - ❖ C:\\Users\\Piotrek\\Desktop\\Praca inzynierska\\DataSet\\Sample0%d
- img\_nr: czyli numer litery z folderu zawierającego foldery 52 liter, numerowane od 11 do 62, w celu uproszczenia kodu i pobierania liczb z przedziału liczb dwucyfrowych, aniżeli z przedziału 1-52. Przykład:
  - ❖ img\_nr=11 jest przekazywany do ścieżki z punktu „a)” i wygląda tak:  
C:\\Users\\Piotrek\\Desktop\\Praca inzynierska\\DataSet\\Sample011

Po podaniu odpowiednich parametrów do funkcji, jej zadanie polega na iteracji po wszystkich zdjęciach w danym folderze (1016). Podczas iteracji zmniejszeniu ulega każde zdjęcie z rozmiaru 128x128 pikseli do 32x32 piksele, a następnie obrazy zaczytywane są do jednej dużej macierzy, która ma rozmiar 1024x1016, gdzie każda kolumna to zdjęcie w postaci wektora 1x1024 pikseli, powstałego ze spłaszczenia macierzy 32x32.

### 4.2.2. Opis skryptu CreateAlphabet

Skrypt CreateAlphabet bazuje na funkcji LoadImages. Automatyzuje on proces wczytywania wszystkich obrazów tj. iteruje po 52 folderach i po 1016 plikach w każdym z nich. W ten sposób tworzony jest zbiór 52832 zdjęć wczytanych do jednej macierzy. Skrypt ten dodaje również etykietę do każdej z fotografii. Etykieta to wartość liczbową z przedziału 1-52, gdzie liczby 1-26 odpowiadają kolejno literom dużym A-Z, a liczby 27-52 literom małym a-z.

Oznacza to, że stworzony alfabet ma daną budowę:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52

Etykiety są doklejane do ostatniego wiersza w każdej kolumnie, po czym litery w alfabecie zostają ustawione losowo tak, aby stworzyć losowy zbiór danych. Na koniec etykiety są odczepiane z ostatniego wiersza każdej z kolumny i zapisywane są do samodzielnego wektora. Sztuczna sieć neuronowa na wyjściach, potrzebować będzie wektora 52 elementowego (rozpoznanie etykiety), w celu podjęcia decyzji o rozpoznaniu danej litery. Do tego utworzona została funkcja `DummyLabels`.

#### 4.2.3. Opis funkcji `DummyLabels`

Funkcja jest zbudowana w następujący sposób:

```
function [ targetsd ] = DummyLabels( labels )
```

Pobierana wartość:

- `labels`: czyli wektor alfabetu o rozmiarze 1x52832

Po podaniu odpowiedniego parametru do funkcji, jej zadanie polega na utworzeniu macierzy zerojedynkowej o rozmiarze 52x52832. Każda kolumna wypełniana jest zerami, a jedynka pojawia się na miejscu wiersza o numerze etykiety. Stąd sztuczna sieć neuronowa, dostając wektor z tej macierzy przykładowo:

0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

wie, że jest to litera „C”.

### **4.3. Dane wykorzystane do treningu, walidacji i testu SSN**

Dzięki skryptowi CreateAlphabet do dyspozycji otrzymano losowy zbiór danych 52832 obrazów wraz z ich etykietami. W celu treningu, walidacji oraz testu sieci neuronowych wykorzystano 45000 z tych obrazów. Środowisko MATLAB dzieli dane treningowe w stosunku:

- 70% dane treningowe
- 15% dane walidacyjne
- 15% dane testowe

#### **4.3.1. Test na 1000 niezależnych próbkach**

Dodatkowo wyodrębniono 1000 obrazów, które nie pojawiły się w zbiorze 45000 jednostek treningowych, w celu uzupełniającego testu już wytrenowanej i dostępnej sieci neuronowej.

## 5. Stworzenie sztucznej sieci neuronowej (SSN)

W poniższym rozdziale przedstawiony zostanie proces kształtowania sztucznej sieci neuronowej.

Każda z sieci bazuje na tym samym wektorze treningowym 45000 zdjęć, dzielonym przez sieć w stosunku: 70% danych treningowych, 15% danych walidacyjnych oraz 15% danych testowych.

Do dodatkowego testu użyty jest również stały wektor 1000 zdjęć, wcześniej niewykorzystywanych.

Każda z sieci posiada 1024 wejścia w celu zaczytania zdjęć w formacie 32x32 piksele oraz 52 wyjścia do podjęcia decyzji o literze.

Funkcja używana do treningu to „trainscg”, czyli skalowalna metoda gradientu sprzężonego (ang. Scaled conjugate gradient backpropagation) używana podczas treningów z nauczycielem przy uczeniu sieci typu Feedforward Neural Networks.

Przeprowadzana jest także dodatkowa walidacja funkcją TestCheck, która analizuje rezultaty na 1000 próbkach wcześniej niewykorzystywanych.

### 5.1. Opis funkcji TestCheck

Funkcja jest zbudowana w następujący sposób:

```
function [ right ] = TestCheck( RealOutputs, TargetOutputs )
```

Pobierane są dwie wartości:

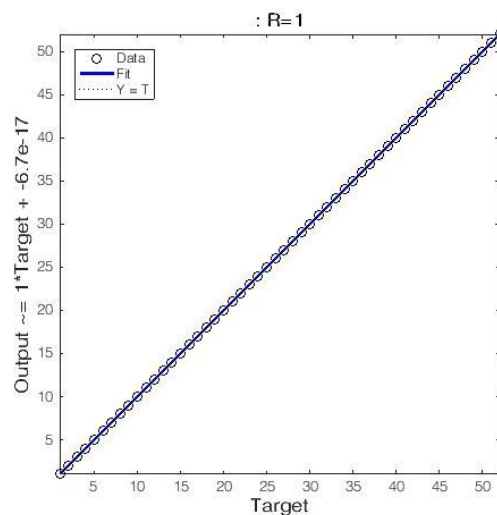
- RealOutputs: czyli rzeczywiste etykiety zdjęć
- TargetOutputs: czyli przewidziane przez sieć etykiety zdjęć

Funkcja TestCheck została wykorzystana w skrypcie RegressionPlot, do którego zwraca wartość „right”, czyli procentowy stosunek etykiet rozpoznanych prawidłowo do całego zbioru danych.

$$right = \frac{\text{etykiety rozpoznane prawidłowo}}{1000}$$

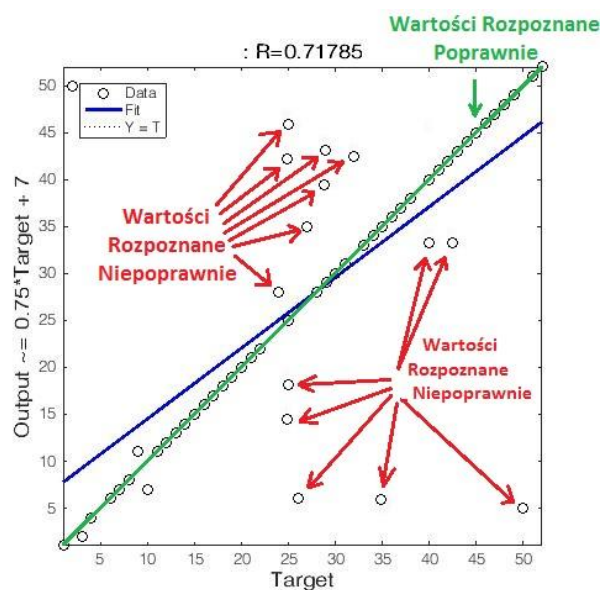
## 5.2. Opis skryptu RegressionPlot

W skrypcie tym sieć jest ponownie testowana, zbiorem 1000 próbek niewykorzystanych wcześniej do treningu. Następnie rysowany jest poglądowy wykres regresji dla wartości rzeczywistych oraz wartości przewidzianych. W 100% poprawny wykres miałby odwzorowanie 1:1, tj. wykres liniowy  $y = x$ .



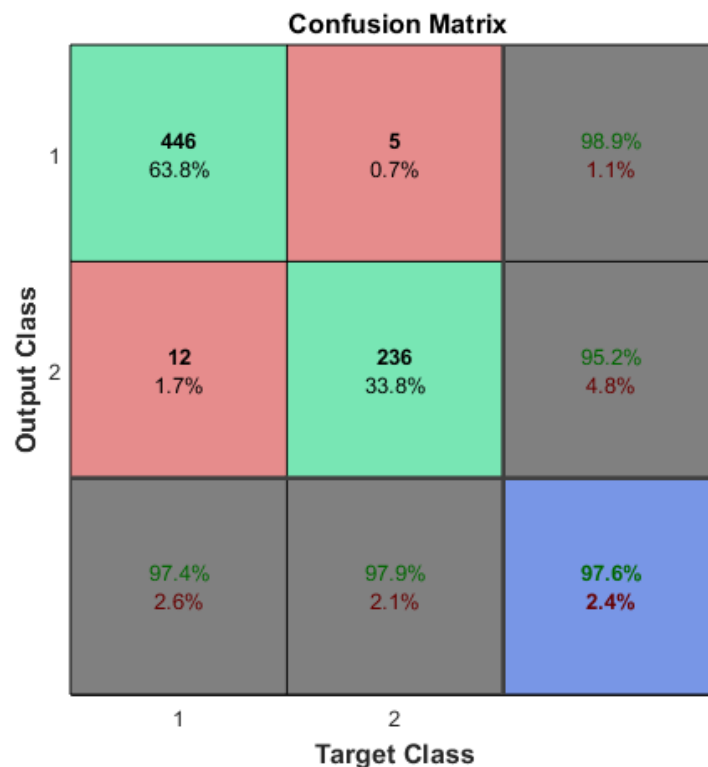
16. Regression plot R=1.00

Niestety zważając na to, że sztuczne sieci neuronowe, będą przewidywały na poziomie 90-95% na wykresie mogą pojawić się spore odchylenia, które przeanalizowane i wyjaśnione są w dalszej części pracy.



17. Objaśnienia regression plot

### 5.3. Opis Confusion matrix



18. Confusion matrix

Output Class – etykiety wyjściowe ze sztucznej sieci neuronowej

Target Class – etykiety prawidłowe

Dwie pierwsze komórki na diagonalnej (zielone) pokazują ilość i procent poprawności przyporządkowania danej klasy elementu do etykiety przez wytrenowaną sieć.

Wyjaśnienie:

- Pierwsza zielona komórka na diagonalnej: 446 ze wszystkich 699 przypadków (446+236+5+12=699) zostało poprawnie rozpoznane jako etykieta nr 1.
- Druga zielona komórka na diagonalnej: 236 ze wszystkich 699 przypadków zostało poprawnie rozpoznane jako etykieta nr 2.
- Prawy dolny róg (komórka niebieska): łączna sprawność sieci dla całego zbioru danych, tj.  $\frac{446+236}{699} \times 100\% = 97.6\%$

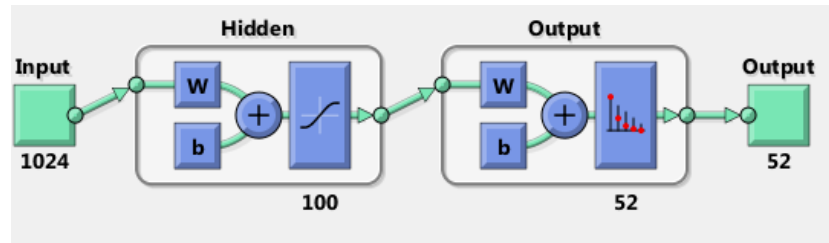
W dalszych rozważaniach ze względu na rozmiar i nieczytelność Confusion matrix dla 52 etykiet, prezentowane będą tylko wyjściowe dane z prawej dolnej komórki (komórka niebieska) bez jej graficznej reprezentacji.



## 6. Implementacja rozwiązania problemu

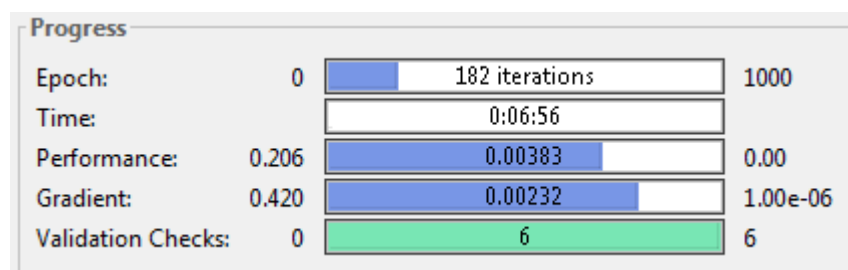
### 6.1. SSN typu MLP z 1 warstwą ukrytą o 100 neuronach

Oto uzyskana sieć neuronowa:



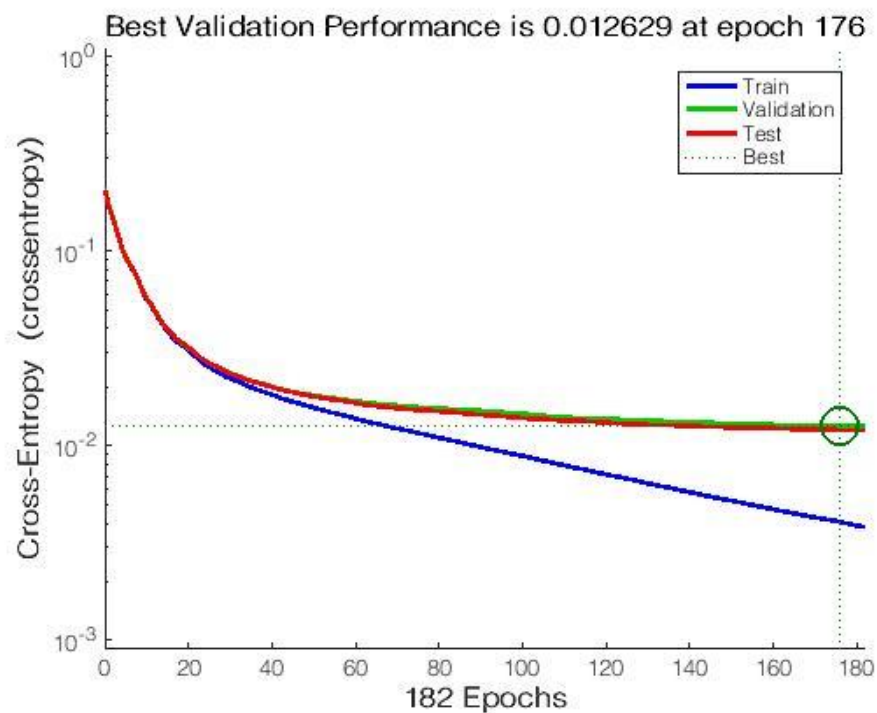
19. SSN typu MLP z 1 warstwą ukrytą o 100 neuronach

Czas trenowania tak zaprojektowanej sieci jest satysfakcjonujący:



20. Czas trenowania

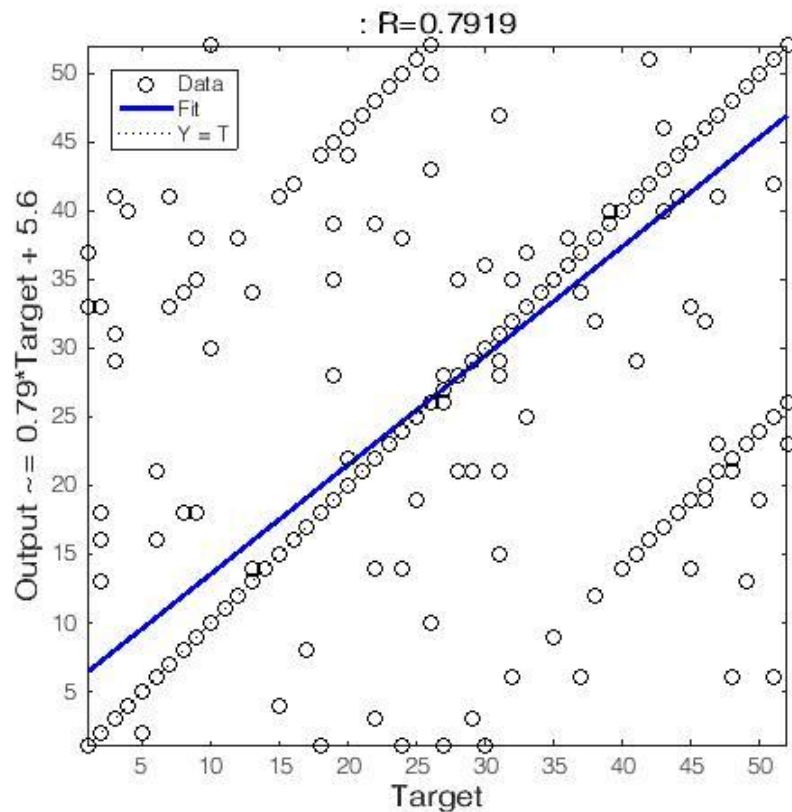
Wykres wydajności:



21. Wykres wydajności treningu

Dane odczytane z rubryki finalnej (prawy dolny róg) Confusion Matrix to: 90.4% oraz 9.6%. Co mówi o tym, że podczas treningu: 90.4% wartości została przewidziana poprawnie, a jedynie 9.6% niepoprawnie.

#### 6.1.1. Dodatkowa walidacja zbiorem 1000 próbek



22. Wykres regresji dla SSN o 100 neuronach w warstwie ukrytej

Używając funkcji TestCheck:

826 próbek ze zbioru 1000 próbek zostało rozpoznane prawidłowo, 174 próbek ze zbioru 1000 próbek zostało rozpoznane nieprawidłowo.

Wykres nie wygląda najlepiej, jednak funkcja dopasowania wręcz przeciwnie. Użytkownik może być wprowadzony w zakłopotanie, gdyż z obu stron wykresu dopasowania pojawiają się szumy układające się w rzędy tworząc swojego rodzaju linie proste. Spowodowane jest to nieprawidłowym rozpoznaniem, przez sieć neuronową, szeregu liter, które na wykresie niefortunnie układają się w linie proste.

Współczynnik  $R=0.79$  świadczy, że dopasowanie jest dość duże. Tak jak już wcześniej zostało wspomniane, odwzorowanie poprawne to linia przebiegająca przez punkty odwzorowane 1:1. Rozstrzelenie punktów rozpoznanych nieprawidłowo (szumy na wykresie) świadczy o tym, że ilość punktów rozpoznanych prawidłowo musi wielokrotnie je przewyższać. Wartość  $R=0.79$  jest wynikiem relatywnie dobrym.

## 7. Rozwój i doskonalenie rozwiązania problemu

W celu usprawnienia działania użytej sztucznej sieci neuronowej typu MLP autor natknął się na kilka reguł mówiących o optymalnej liczbie neuronów w warstwie ukrytej.

Przykładowe reguły:

- „Jak duża powinna być warstwa ukryta? Jedna z reguł mówi, że powinna być nie większa niż ilość wejść warstwy wejściowej pomnożona przez dwa”.  
(ang. „How large should the hidden layer be? One rule of thumb is that it should never be more than twice as large as the input layer.”)  
(Berry and Linoff, 1997, p. 323).

- „Jedna z reguł mówi, że rozmiar warstwy ukrytej to coś pomiędzy rozmiarem warstwy wejściowej i rozmiarem warstwy wyjściowej”  
(ang. „A rule of thumb is for the size of this [hidden] layer to be somewhere between the input layer size ... and the output layer size ...”)  
(Blum, 1992, p. 60).

- „W celu obliczenia liczby węzłów warstwy ukrytej używamy ogólnej reguły: (Liczba wejść + wyjść) \* (2/3).”  
(ang. „To calculate the number of hidden nodes we use a general rule of: (Number of inputs + outputs) \* (2/3).”)

$$\text{Hidden} = \frac{2}{3} \times (\text{liczba wejść} + \text{liczba wyjść})$$

([www.neuralware.com](http://www.neuralware.com))

- W celu obliczenia rozmiaru warstwy ukrytej używamy ogólnej reguły średniej geometrycznej z ilości wejść i wyjść:

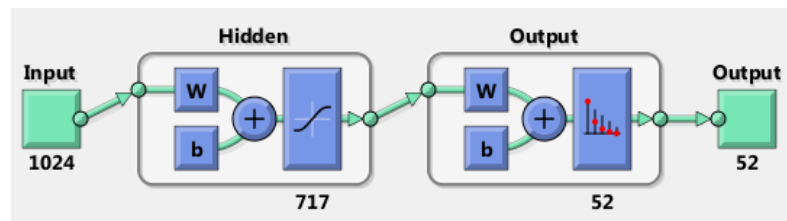
$$\text{Hidden} = \sqrt[2]{\text{ilość wejść} \times \text{ilość wyjść}}$$

Wykorzystano regułę z podpunktu nr 3. Czyli liczbę neuronów w warstwie ukrytej obliczono na podstawie wzoru:

$$\text{Warstwa ukryta} = \frac{2 \times (1024 + 52)}{3} \approx 717$$

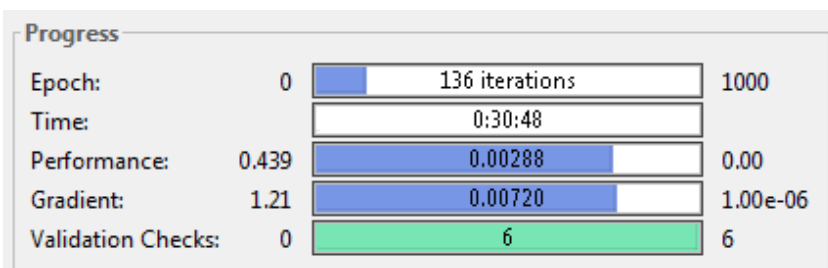
## 7.1. SSN typu MLP z 1 warstwą ukrytą o 717 neuronach

Oto uzyskana sieć neuronowa:



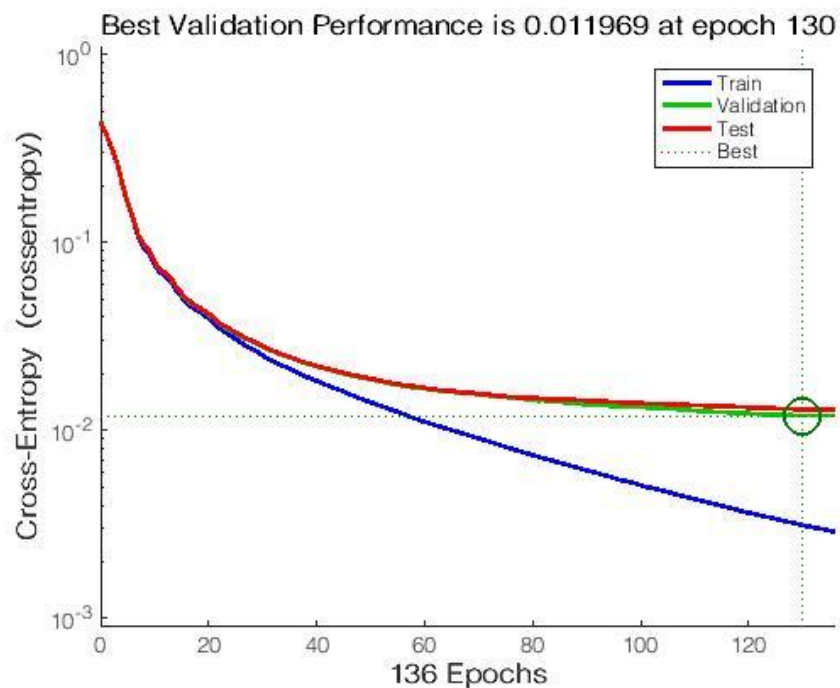
23. SSN typu MLP z jedną warstwą ukrytą o 717 neuronach

Czas trenowania tak zaprojektowanej sieci jest dosyć długi:



24. Czas trenowania

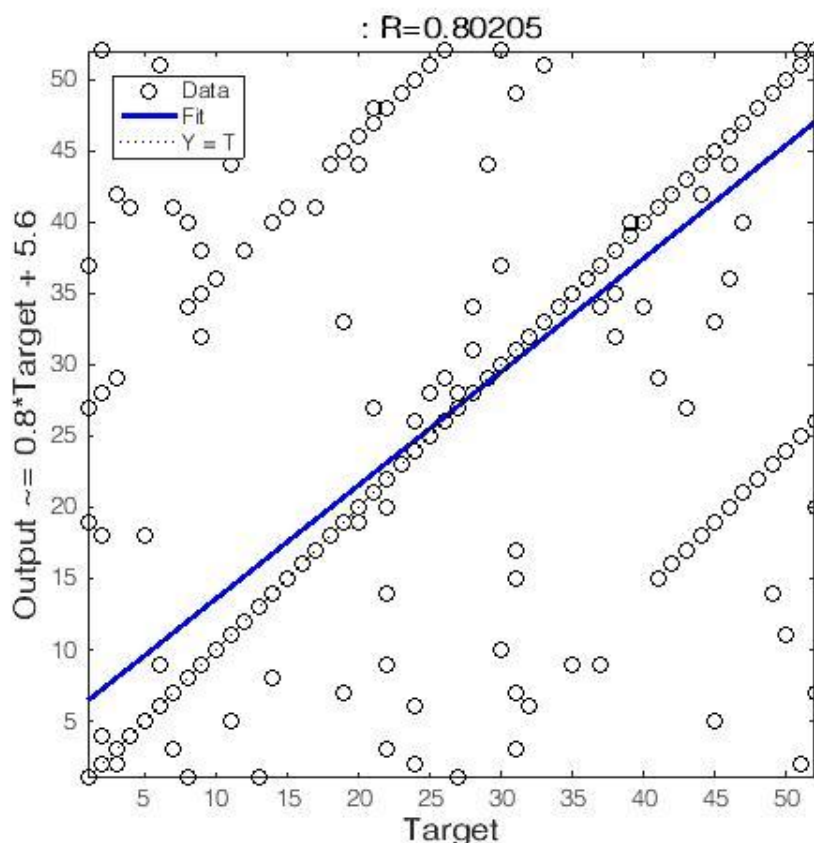
Wykres wydajności:



25. Wykres wydajności treningu

Dane odczytane z rubryki finalnej(prawy dolny róg) Confusion Matrix to: 91.5% oraz 8.5%. Co mówi nam o tym, że podczas treningu: 91.5% wartości została przewidziana poprawnie, a jedynie 8.5% niepoprawnie.

### 7.1.1. Dodatkowa walidacja zbiorem 1000 próbek



26. Wykres regresji dla SSN o 717 neuronach w warstwie ukrytej

Używając funkcji TestCheck:

845 próbek z datasetu 1000 próbek zostało rozpoznane prawidłowo  
155 próbek z datasetu 1000 próbek zostało rozpoznane nieprawidłowo.

Po raz kolejny wykres nie wygląda najlepiej, jednak funkcja dopasowania wręcz przeciwnie. Użytkownik może ponownie być wprowadzony w zakłopotanie, gdyż z obu stron wykresu dopasowania pojawiają się szumy układające się w rzędy tworząc swojego rodzaju linie proste. Spowodowane jest to nieprawidłowym rozpoznaniem, przez sieć neuronową, szeregu liter, które na wykresie niefortunnie układają się w linie proste.

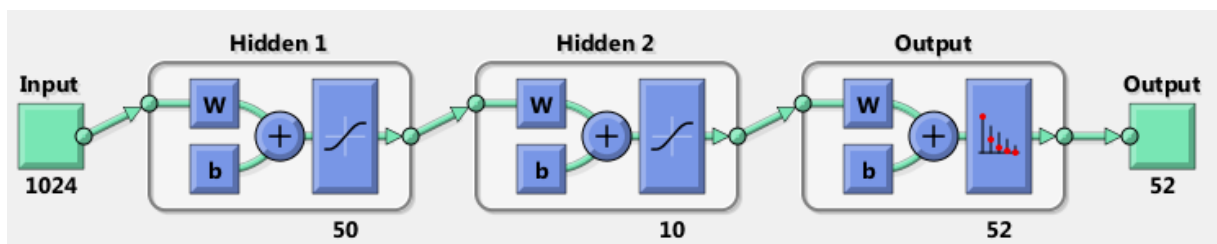
Współczynnik  $R=0.80$  świadczy, że dopasowanie jest dość duże. Tak jak już wcześniej zostało wspomniane, odwzorowanie poprawne to linia przebiegająca przez punkty odwzorowane 1:1. W tym przypadku również ilość punktów rozpoznanych prawidłowo wielokrotnie przewyższa ilość punktów rozpoznanych nieprawidłowo. Wykres może wprowadzać w zakłopotanie, jednak  $R=0.80$  jest ponownie wynikiem dobrym.

#### Obserwacje:

Zauważalne jest, że zastosowana reguła dotycząca ilości neuronów w warstwie ukrytej podwyższyła sprawność naszej sieci.

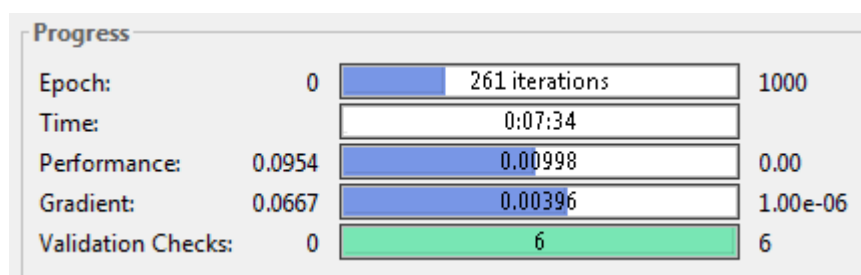
## **7.2. SSN typu MLP z 2 warstwami ukrytymi o 50 i 10 neuronach**

Oto uzyskana sieć neuronowa:



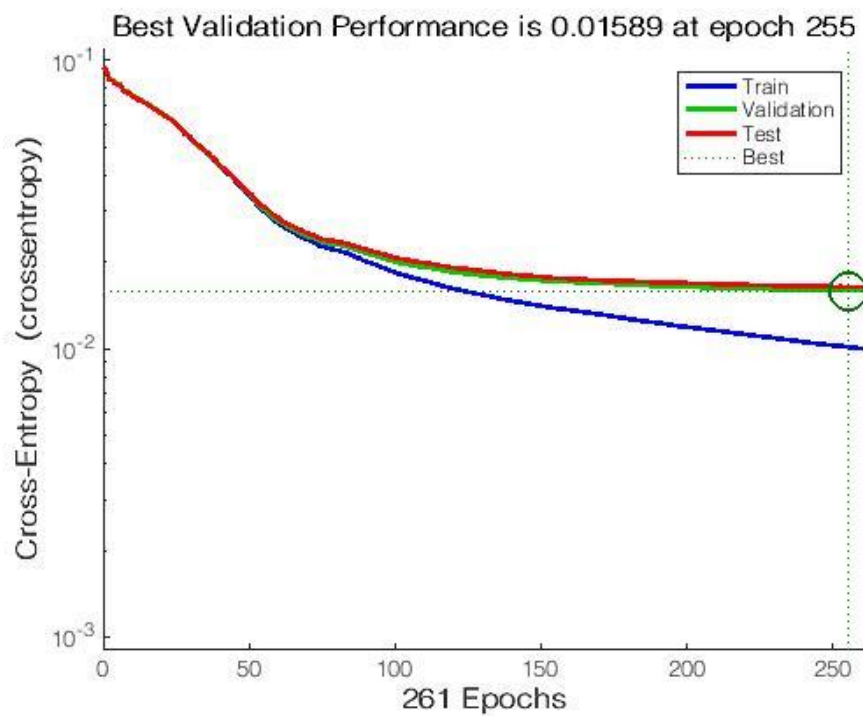
27. SSN typu MLP z dwiema warstwami ukrytymi o 50 i 10 neuronach

Czas trenowania tak zaprojektowanej sieci jest porównywalny z czasem sieci o 1 warstwie ukrytej i 100 neuronach:



28. Czas trenowania

Wykres wydajności:

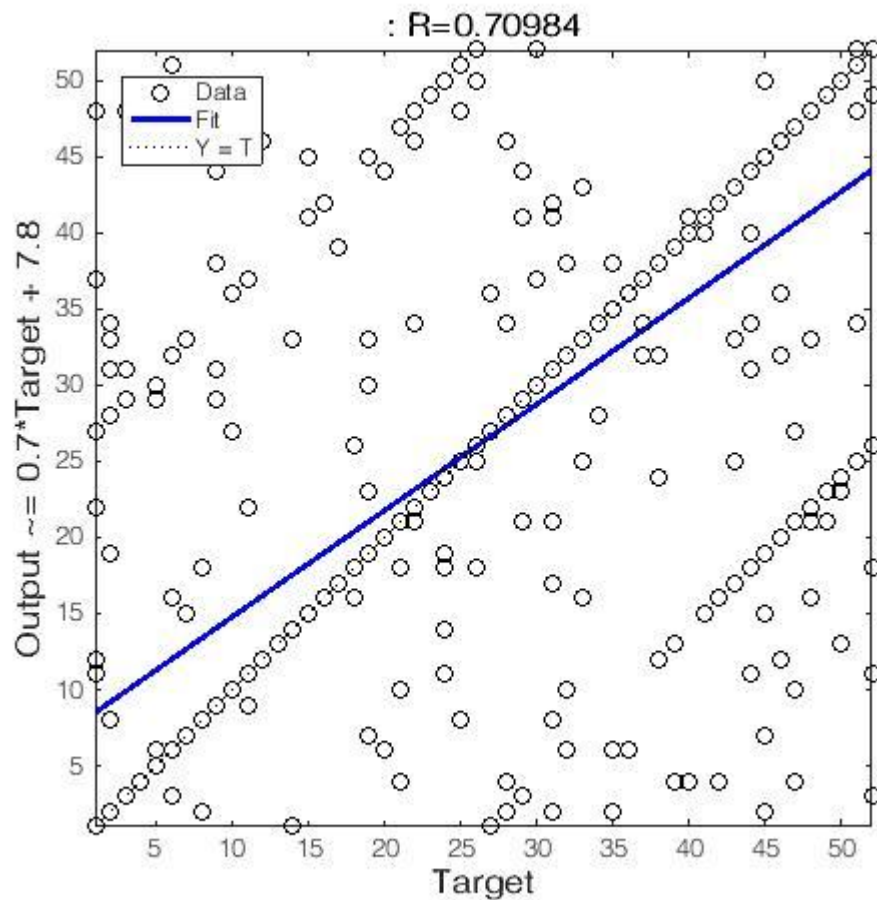


29. Wykres wydajności treningu

Dane odczytane z rubryki finalnej(prawy dolny róg) Confusion Matrix to: 82.1% oraz 17.9%. Co mówi nam o tym, że podczas treningu: 82.1% wartości została przewidziana poprawnie, a 17.9% niepoprawnie.



### 7.2.1. Dodatkowa walidacja zbiorem 1000 próbek



30. Wykres regresji dla SSN o 2 warstwach ukrytych

Używając funkcji TestCheck:

764 próbek ze zbioru 1000 próbek zostało rozpoznane prawidłowo

236 próbek ze zbioru 1000 próbek zostało rozpoznane nieprawidłowo

Obserwacje:

Tak zaprojektowana sieć, nie spełnia swoich założeń, a wydajność na poziomie 82% nie jest satysfakcjonująca.

## 8. Przemodelowanie zbioru danych

Początkowo interpretowane były zdjęcia, gdzie tło jest koloru białego, a litera koloru czarnego tj. :

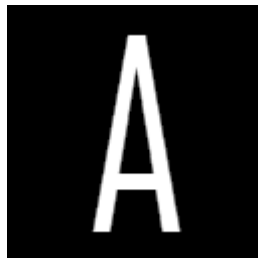


31. Litera z początkowego zbioru danych

Ponadto, wartości pikseli w macierzy zdjęcia przechowywane były w formacie liczby ze zbioru 0-255, gdzie 0 oznaczało w pełni nasycony kolor czarny, a wartość 255 kolor biały.

Liczby te zostały zamienione na wartości, ze zbioru 0-1. Kolory zostały także odwrócone. Zabiegi te miały na celu zmniejszenie wartości liczbowych na jakich operują funkcje aktywacji w sztucznych sieciach neuronowych.

Przy kolejnych próbach zdjęcia wyglądały tak:



32. Litera z przemodelowanego zbioru danych

## 9. Wyniki końcowe

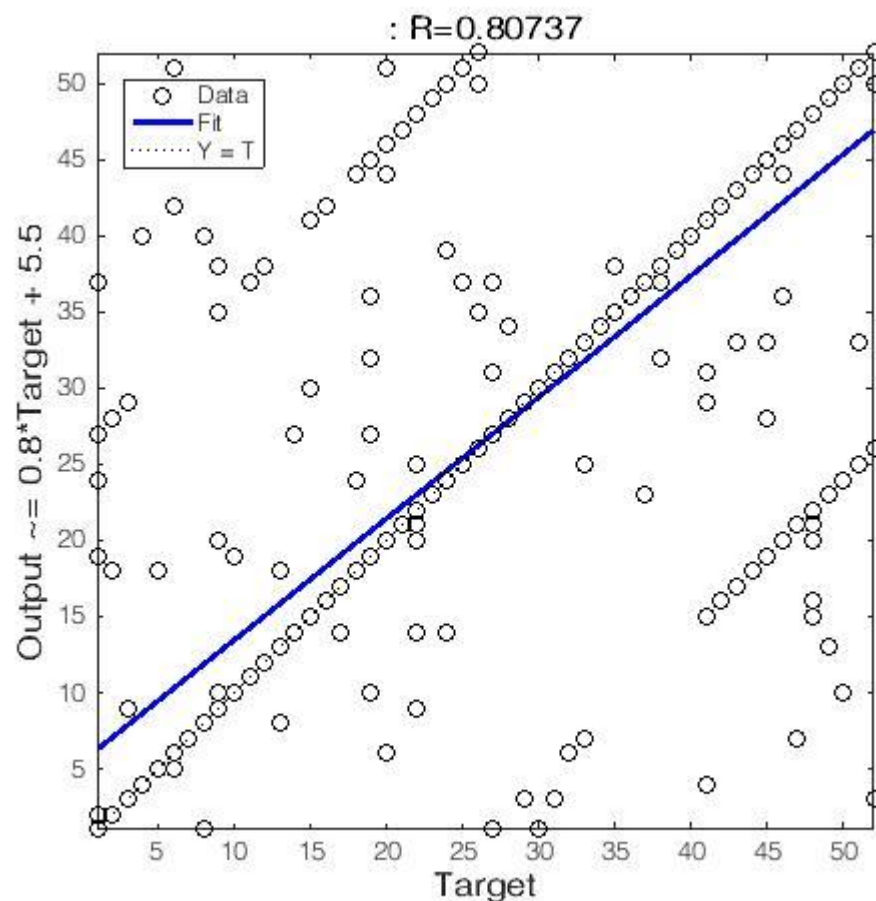
Tak przemodelowane dane zostały ponownie użyte do treningu. Trenowana była sieć, która wykazała najlepszą wydajność tj. SSN typu MLP z 1 warstwą ukrytą o 717 neuronach.

Rezultaty zostały zaprezentowane oraz omówione poniżej.

### 9.1. Confusion Matrix

Dane odczytane z rubryki finalnej(prawy dolny róg) Confusion Matrix to: 92.6% oraz 7.4%. Co mówi nam o tym, że podczas treningu: 92.6% wartości została przewidziana poprawnie, a 7.4% niepoprawnie. Wynik ten jest najbardziej satysfakcjonujący.

### 9.2. Dodatkowa walidacja zbiorem 1000 próbek



33. Końcowy wykres regresji

Używając funkcji TestCheck:

847 próbek ze zbioru 1000 próbek zostało rozpoznane prawidłowo  
153 próbki ze zbioru 1000 próbek zostało rozpoznane nieprawidłowo.

Współczynnik  $R \approx 0.81$  świadczy, że dopasowanie jest duże. W tym przypadku ilość punktów rozpoznanych prawidłowo jest największa oraz, tak samo jak poprzednio, wielokrotnie przewyższa ilość punktów rozpoznanych nieprawidłowo.

#### **9.2.1. Obserwacje**

Naniesione modyfikacje usprawniły działanie sztucznej sieci neuronowej. Wyniki testów wskazują na to, że sztuczna sieć neuronowa razem z przemodelowanym zbiorem danych to sieć działająca najlepiej.

## 10. Rozwiązanie końcowe

Użytkownik tak zaprojektowanego prototypu systemu ma możliwość zacytywania skanów formularzy tabelarycznych oraz automatyczną ich analizę. W celu zaprezentowania działania prototypu, stworzone zostały dwa skany podpisów, utworzone na podstawie danych użytych do treningu sztucznych sieci neuronowych.

Wniosek tabelaryczny:

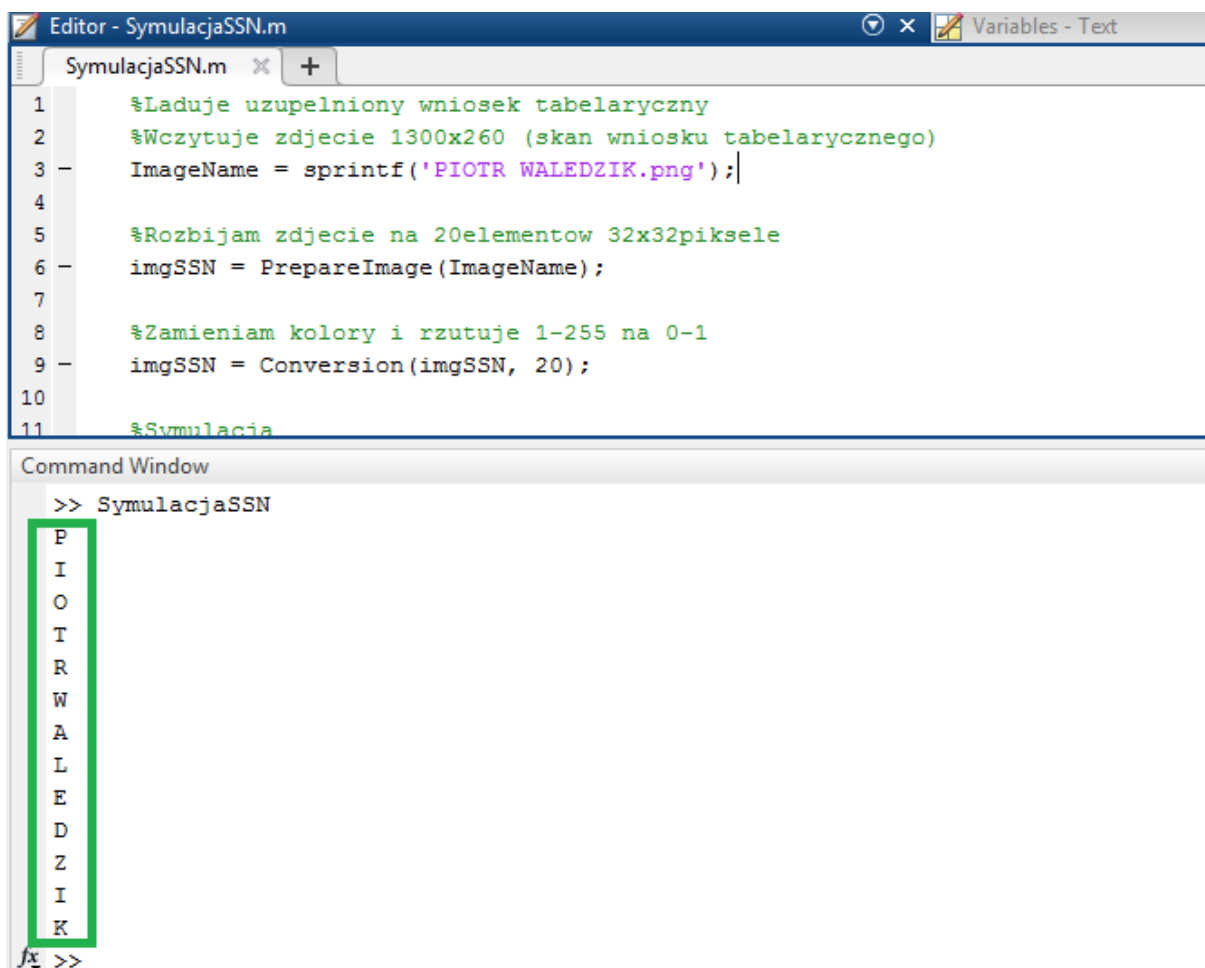

34. Formularz tabelaryczny

Uzupełniony formularz tabelaryczny:

<i><b>P</b></i>	<i><b>I</b></i>	<i><b>O</b></i>	<i><b>T</b></i>	<i><b>R</b></i>					
<i><b>W</b></i>	<i><b>A</b></i>	<i><b>L</b></i>	<i><b>E</b></i>	<i><b>D</b></i>	<i><b>Z</b></i>	<i><b>I</b></i>	<i><b>K</b></i>		

35. Uzupełniony formularz tabelaryczny nr 1

Wartości wyjściowe z systemu:



```
Editor - SymulacjaSSN.m
Variables - Text

SymulacjaSSN.m
1 %Laduje uzupełniony wniosek tabelaryczny
2 %Wczytuje zdjecie 1300x260 (skan wniosku tabelarycznego)
3 - ImageName = sprintf('PIOTR WALEDZIK.png');
4
5 %Rozbijam zdjecie na 20elementow 32x32piksele
6 - imgSSN = PrepareImage(ImageName);
7
8 %Zamieniam kolory i rzutuje 1-255 na 0-1
9 - imgSSN = Conversion(imgSSN, 20);
10
11 %Symulacja

Command Window
>> SymulacjaSSN
P
I
O
T
R
W
A
L
E
D
Z
I
K
>>
```

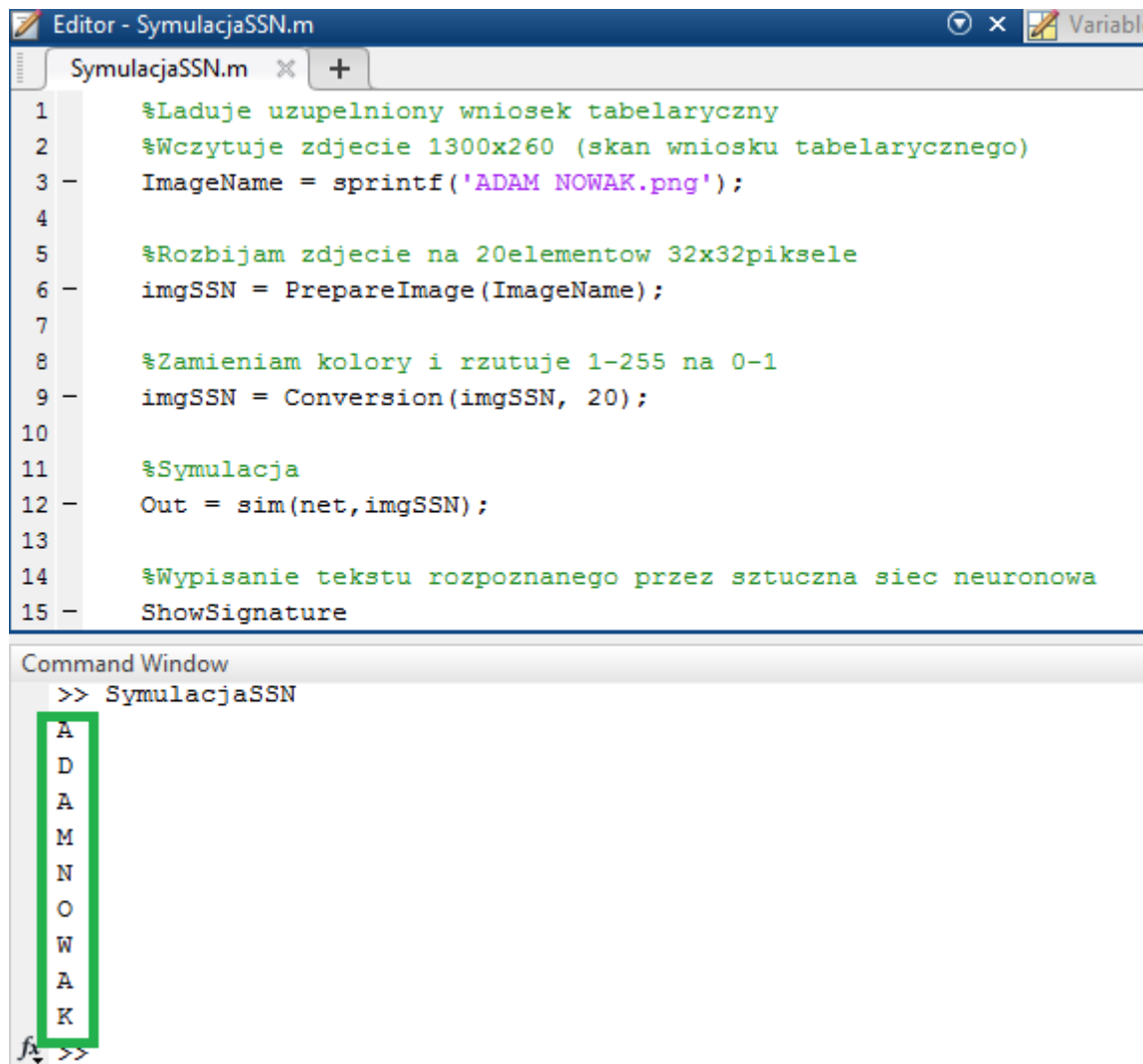
36. Wartości wyjściowe z systemu

Uzupełniony formularz tabelaryczny (litery małe):

<i>A</i>	<b>d</b>	<i>a</i>	<i>m</i>					
<b>N</b>	<i>o</i>	<i>w</i>	<b>a</b>	<i>k</i>				

37. Uzupełniony formularz tabelaryczny nr 2

Wartości wyjściowe z systemu:



The image shows a MATLAB environment with two windows. The top window, titled 'Editor - SymulacjaSSN.m', contains a script with 15 lines of code. The code performs the following steps: 1. Loads a complete tabular application form. 2. Reads a 1300x260 pixel image (scanned form). 3. Formats the filename as 'ADAM NOWAK.png'. 4. Resizes the image to 20 elements by 32x32 pixels. 5. Prepares the image for the neural network. 6. Converts colors from 1-255 to 0-1. 7. Performs the simulation. 8. Outputs the result. 9. Displays the recognized text through a neural network. 10. Shows the signature. The bottom window, titled 'Command Window', shows the command '>> SymulacjaSSN' being executed, followed by the output of the simulation: 'A', 'D', 'A', 'M', 'N', 'O', 'W', 'A', 'K'. The output is displayed vertically, with each character on a new line, and is enclosed in a green rectangular box.

```
1 %Laduje uzupelniony wniosek tabelaryczny
2 %Wczytuje zdjecie 1300x260 (skan wniosku tabelarycznego)
3 - ImageName = sprintf('ADAM NOWAK.png');
4
5 %Rozbijam zdjecie na 20elementow 32x32piksele
6 - imgSSN = PrepareImage(ImageName);
7
8 %Zamieniam kolory i rzutuje 1-255 na 0-1
9 - imgSSN = Conversion(imgSSN, 20);
10
11 %Symulacja
12 - Out = sim(net,imgSSN);
13
14 %Wypisanie tekstu rozpoznanego przez sztuczna siec neuronowa
15 - ShowSignature
```

```
>> SymulacjaSSN
A
D
A
M
N
O
W
A
K
>>
```

38. Wartości wyjściowe z systemu

## 11. Opis funkcji i skryptów

### 11.1. CreateAlphabet

Skrypt zaczytujący zdjęcia (pliki o rozszerzeniu .png) do macierzy, gdzie każde zdjęcie zostaje spłaszczone do wektora przechowywanego w kolumnach macierzy. W skrypcie podawana jest dokładna ścieżka wskazująca na położenie folderu ze zdjęciami. Używane są tu również dwie funkcje: LoadImages i DummyLabels. Wartościami wyjściowymi skryptu są:

- Alphabet\_Training – zbiór 52832 obrazów o rozmiarze 32x32 piksele, ułożonych losowo, zaczytanych do macierzy 1024x52832
- dTraining – zbiór labeli 52832 obrazów w postaci macierzy zerojedynkowej o wymiarach 52x52832

### 11.2. LoadImages

Funkcja umożliwiająca wczytywanie obrazów do macierzy w skrypcie CreateAlphabet. Iteruje po 1016 zdjęć, w każdym z podfolderów.

### 11.3. DummyLabels

Funkcja przyjmuje wektor etykiet rozmiarów 1x52832, następnie zwraca macierz zerojedynkową rozmiarów 52x52832. Wartość 1.0 przechowywana jest w komórce macierzy o numerze z wektora wejściowego, reszta (51 wartości) to zera.

### 11.4. SiecNeuronowaAlphabet

Skrypt tworzy i trenuje sztuczną sieć neuronową o 717 neuronach.

### 11.5. RegressionPlot

Skrypt testuje utworzoną sieć na 1000 niezależnych nowych próbek (1000 wcześniej nieużywanych zdjęć). Dla uzyskanych wyników rysuje wykres regresji oraz, używając funkcji TestCheck, zwraca ilość próbek rozpoznanych prawidłowo.

### 11.6. TestCheck

Funkcja używana w skrypcie RegressionPlot, zwraca ilość próbek rozpoznanych prawidłowo.



### **11.7. SymulacjaSSN**

Skrypt zaczytuje gotowy skan uzupełnionego formularza tabelarycznego poprzez podanie nazwy i rozszerzenia pliku (na przykład 'ADAM NOWAK.png'). Następnie poprzez funkcję `PrepareImage`, obraz jest zmniejszany i rozbijany na 20 elementów o rozmiarze 32x32 piksele. Używając funkcji `Conversion`, w 20 elementach zostają odwrócone kolory i wartości liczbowe są rzutowane z 0-255 na 0-1. Tak przemodelowane dane wejściowe trafiają do wcześniej wytrenowanej sztucznej sieci neuronowej rozpoznającej litery. Na koniec skrypt `ShowSignature` wyświetla w konsoli rozpoznany podpis.

### **11.8. PrepareImage**

Funkcja pobiera nazwę i rozszerzenie pliku tj. "Nazwa.png". Następnie zwraca macierz 1024x20, czyli 20 zdjęć o rozmiarze 32x32, które mogą być podane do SSN w celu rozpoznania.

### **11.9. Conversion**

Funkcja pobiera zbiór danych i ilość obrazów, po których ma iterować. Odwraca kolory i rzutuje wartości liczbowe z 0-255 na 0-1.

### **11.10. ShowSignature**

Skrypt wyświetlający w konsoli podpis rozpoznany przez sztuczną sieć neuronową.

## 12. Podsumowanie

Cele niniejszej pracy, jakimi były stworzenie i zautomatyzowanie algorytmu analizy danych oraz ich porównanie i optymalizacja, zostały osiągnięte. Zbudowany system w zależności od konfiguracji charakteryzuje się dobrymi, bądź bardzo dobrymi osiągnięciami.

Tak zaimplementowane podejście może z powodzeniem zostać uogólnione i przeniesione na dowolny inny przypadek rozpoznawania tekstu, bądź innych kształtów. Algorytm zależy od użytego zbioru danych treningowych. Posiadając rzeczywisty zbiór liter pisanych odręcznie, system stanowi łatwą metodę zacytywania tych danych ze skanów prawdziwych dokumentów.

Temat mógłby być dalej rozwijany, ponieważ z pewnością nie został jeszcze wyczerpany. Ilość podobnych publikacji jest stosunkowo niewielka, a potencjalna liczba możliwych modyfikacji jest niemal nieograniczona. Zagłębiając się jeszcze dokładniej w dziedzinę sztucznych sieci neuronowych, prawdopodobnie można znacznie poprawić skuteczność rozpoznawania pisma.

### 13. Bibliografia

- [1] R. Tadeusiewicz, Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa 1993.
- [2] Sieci neuronowe – historia, Katedra inżynierii komputerowej Politechniki Częstochowskiej, (źródło: <http://iisi.pcz.pl/nn/historia.php#home>, [dostęp 11.01.2017])
- [3] Neuron – Wikipedia, the free encyclopedia. (źródło: <https://pl.wikipedia.org/wiki/Neuron>, [dostęp 1.12.2016])
- [4] Sieci neuronowe – C-Labtech. (źródło: <http://www.ai.c-labtech.net/sn/index.html>, [dostęp 1.12.2016])
- [5] Budowa neuronu – Instytut informatyki, Politechnika Poznańska. (źródło: <http://www.cs.put.poznan.pl/rklaus/assn/neuron.htm>, [dostęp 1.12.2016])
- [6] Sieć Neuronowa – Wikipedia, the free encyclopedia. (źródło: [https://pl.wikipedia.org/wiki/Sieć\\_neuronowa](https://pl.wikipedia.org/wiki/Sieć_neuronowa), [dostęp 1.12.2016])
- [7] Perceptron – Wikipedia, the free encyclopedia (źródło: <http://pl.wikipedia.org/wiki/Perceptron>, [dostęp 1.12.2016])
- [8] Multilayer perceptron – Wikipedia, the free encyclopedia (źródło: [https://en.wikipedia.org/wiki/Multilayer\\_perceptron](https://en.wikipedia.org/wiki/Multilayer_perceptron), [dostęp 1.12.2016])
- [9] Multi-Layered Perceptron, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])
- [10] Sieć Hopfielda, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])
- [11] Sieć asocjacyjna – Wikipedia, the free encyclopedia (źródło: [https://pl.wikipedia.org/wiki/Sieć\\_asocjacyjna](https://pl.wikipedia.org/wiki/Sieć_asocjacyjna), [dostęp 1.12.2016])
- [12] Sztuczna sieć komórkowa, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])

- [13] Rodzaje sztucznych sieci neuronowych – Instytut informatyki, Politechnika Poznańska. (źródło: <http://www.cs.put.poznan.pl/rklaus/assn/neuron.htm>, [dostęp 1.12.2016])
- [14] Sieć hybrydowa,(źródło: <http://www.twins.pk.edu.pl/~pkowal/SSN/rodzaje.htm/> [dostęp 1.12.2016])
- [15] Sztuczne sieci neuronowe, (źródło: <http://www.poltynk.pl/marcin/uczenie.html/>, [dostęp 1.12.2016])
- [16] Sieci neuronowe – C-Labtech. (źródło: <http://www.ai.c-labtech.net/sn/sneuro.html>, [dostęp 1.12.2016])
- [17] Polecenie przelewu– Wikipedia, the free encyclopedia. (źródło: [https://pl.wikipedia.org/wiki/Polecenie\\_przelewu](https://pl.wikipedia.org/wiki/Polecenie_przelewu), [dostęp 1.12.2016])
- [18] Przesyłka pocztowa, Google grafika (źródło: <https://www.google.pl/>, [dostęp 1.12.2016])
- [19] Oprogramowanie naukowo techniczne (źródło: <http://www.ont.com.pl/produkty/lista-produktow/matlab/>, [dostęp 1.12.2016])
- [20] Synapsa – Wikipedia, the free encyclopedia. (źródło: <https://pl.wikipedia.org/wiki/Synapsa>, [dostęp 1.12.2016])
- [21] J. Żurada, M. Barski, W. Jędruch, Sztuczne sieci neuronowe – podstawy teorii i zastosowania, Wydawnictwo Naukowe PWN, Warszawa 1996.
- [22] S. Osowski, Sieci neuronowe w ujęciu algorytmicznym, WNT, Warszawa 2006.
- [23] MATLAB Help (źródło: <https://es.mathworks.com/help/>, [dostęp 1.12.2016])
- [24] Implementacja i wykorzystanie wielowarstwowej sieci perceptronowej w modelowaniu makroekonomicznym – [www.roszczak.com](http://www.roszczak.com) (źródło: <http://www.roszczak.com/mlp/ai.html>, [dostęp 1.01.2017])

- [25] P. Staszewski, Sieć neuronowa backpropagation + MNIST (źródło: <http://www.pawelstaszewski.pl/2014/07/13/siec-neuronowa-backpropagation-mnist-rozpoznawanie-cyfr-0-9/>, [dostęp 1.01.2017])
- [26] Dataset, Character recognition in natural images (źródło: <http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/>, [dostęp 1.12.2016])

## 14. Spis rysunków

1. Neuron piramidowy z kory mózgowej człowieka .....	2
2. Schemat budowy neuronu ludzkiego.....	3
3. Budowa sztucznego neuronu .....	4
4. Neuron - schemat matematyczny .....	5
5. Uproszczony schemat jednokierunkowej sieci neuronowej (kółka oznaczają sztuczne neurony) .....	6
6. Perceptron złożony z jednego neuronu McCullocha-Pittsa .....	7
7. Multi-Layered Perceptron z jedną warstwą ukrytą.....	8
8. Sieć rekurencyjna Hopfielda.....	9
9. Schemat sztucznej sieci komórkowej.....	10
10. Schemat sieci hybrydowej.....	11
11. Polecenie przelewu .....	14
12. Przesyłka pocztowa.....	15
13. Algorytm automatyzacji procesu.....	15
14. Schemat pracy w programie MATLAB .....	16
15. Litery ze zbioru danych.....	18
16. Regression plot $R=1.00$ .....	23
17. Objaśnienia regression plot.....	23
18. Confusion matrix .....	24
19. SSN typu MLP z 1 warstwą ukrytą o 100 neuronach .....	25
20. Czas trenowania.....	25
21. Wykres wydajności treningu .....	25
22. Wykres regresji dla SSN o 100 neuronach w warstwie ukrytej.....	26
23. SSN typu MLP z jedną warstwą ukrytą o 717 neuronach.....	29
24. Czas trenowania.....	29
25. Wykres wydajności treningu .....	29

26. Wykres regresji dla SSN o 717 neuronach w warstwie ukrytej.....	30
27. SSN typu MLP z dwiema warstwami ukrytymi o 50 i 10 neuronach .....	31
28. Czas trenowania.....	31
29. Wykres wydajności treningu .....	32
30. Wykres regresji dla SSN o 2 warstwach ukrytych .....	33
31. Litera z początkowego zbioru danych .....	34
32. Litera z przemodelowanego zbioru danych .....	34
33. Końcowy wykres regresji .....	35
34. Formularz tabelaryczny .....	37
35. Uzupełniony formularz tabelaryczny nr 1.....	37
36. Wartości wyjściowe z systemu .....	38
37. Uzupełniony formularz tabelaryczny nr 2.....	38
38. Wartości wyjściowe z systemu .....	39