Instytut Informatyki Politechniki Śląskiej

Zespół Mikroinformatyki i Teorii Automatów Cyfrowych

Rok akademicki:	Rodzaj studiów*: SSI/NSI/NSM	Przedmiot (Języki Asemblerowe/SMiW/BIAI):	Grupa	Sekcja
2016/2017	**SSI**	**BIAI**	**GKiO4**	**1**

Imię i nazwisko:	**Dominik Rączka**	**Prowadzący**: OA/JP/KT/GD/BSz/GB	**GB**
Imię i nazwisko:	**Piotr Wyleżałek**		

Raport końcowy

Temat projektu:

Rozpoznawanie języka na podstawie plików tekstowych z wykorzystaniem sieci neuronowej

Data oddania: dd/mm/rrrr	**15.09.2017**

1. Temat projektu i opis założeń.

Tematem projektu było stworzenie programu, który rozpozna język na podstawie fragmentu tekstu. Tekst jest analizowany pod kątem średniej ilości wystąpień danej litery w odniesieniu do łącznej liczby liter w tekście. Rozpoznanie odbywa się za pomocą sieci neuronowej, która jako wektor danych wejściowych przyjmuje 27 liczb (odpowiadającym literom ‘a’ – ‘z’ i znaki specjalne jako 27 element), a wektorem wyjściowym jest zakodowany na 5 bitach język. Program oprócz analizy porównawczej kilku języków może także rozpoznawać pojedynczy plik tekstowy. Sieć tak samo jak i pozostała część programu została zaimplementowana w języku C#, a uczenie sieci odbywa się z wykorzystaniem algorytmu wstecznej propagacji.

2. Analiza zadania.

Rozpoznawanie języków przez człowieka wydaje się bezproblemowe, ponieważ możemy użyć słownika lub wiedzy ogólnej o językach i wyłapać charakterystyczne konstrukcje gramatyczne, znaki specjalne lub charakterystyczny alfabet. Komputer jednak nie myśli jak my i potrzebuje konkretnej ścieżki działania do osiągnięcia rezultatu. W przypadku rozpoznawania języków dobrą drogą jest użycie podejścia statystycznego z wykorzystaniem średniej ilości wystąpień poszczególnych znaków. Alternatywą byłoby zaimplementowanie wszystkich słów świata do bazy danych programu i wyszukiwanie pasujących słów, jednak ta metoda posiada takie wady jak ogromna baza słów, którą trzeba dostarczyć do działania programu, konieczność posiadania wszystkich słów lub prawie wszystkich słów użytych w analizowanych tekstach oraz niejednoznaczność, gdyż wiele słów w różnych językach jest zapożyczona lub wspólna. W przypadku analizy statystycznej potrzebujemy jedynie reprezentatywnego fragmentu tekstu w interesujących nas językach do wytrenowania sieci. W przeprowadzonych przez nas badaniach używaliśmy książek dostępnych na stronie www.gutenberg.org.

Niestety metoda rozpoznawania języka na podstawie częstości wystąpień liter nie jest idealna i w niektórych przypadkach przestaje być skuteczna. Prawdopodobnie zwiększenie ilości danych w wektorze wejściowym spowodowałoby poprawę skuteczności sieci. Podczas analizy danych zaobserwowaliśmy, że każda książka prawie idealnie wpisuje się w rozkład statystyczny charakterystyczny dla danego języka, dlatego do uczenia sieci wykorzystujemy tylko 17 książek w każdym języku.

3. Struktura programu i najważniejsze jego funkcje.

Program podzielony jest na cztery klasy:

- *MainWindow* – obsługuje ona okno programu i interakcję z użytkownikiem oraz zarządza pozostałymi składowymi programu.
- *FileReader* – czyta pliki wejściowe oraz przetwarza zawarte w nich dane.
- *NeuralNetworkOperator* – obsługuje działanie sieci neuronowej
- *NeuralNetwork* – implementacja sieci neuronowej.

4. Struktura danych wejściowych/testowych.

Wszystki pliki używane przez program znajdują się w folderze data. Są tam dwa pliki z danymi konfiguracyjnymi programu:

- ConfigFile.txt – są w nim języki, które mają być obsłużone aktualnie przez program. Muszą one być w formacie „lan xxxxx", gdzie „lan" jest trzyliterowym skrótem języka, a xxxxx unikalną pięciobitową sekwencją języka.

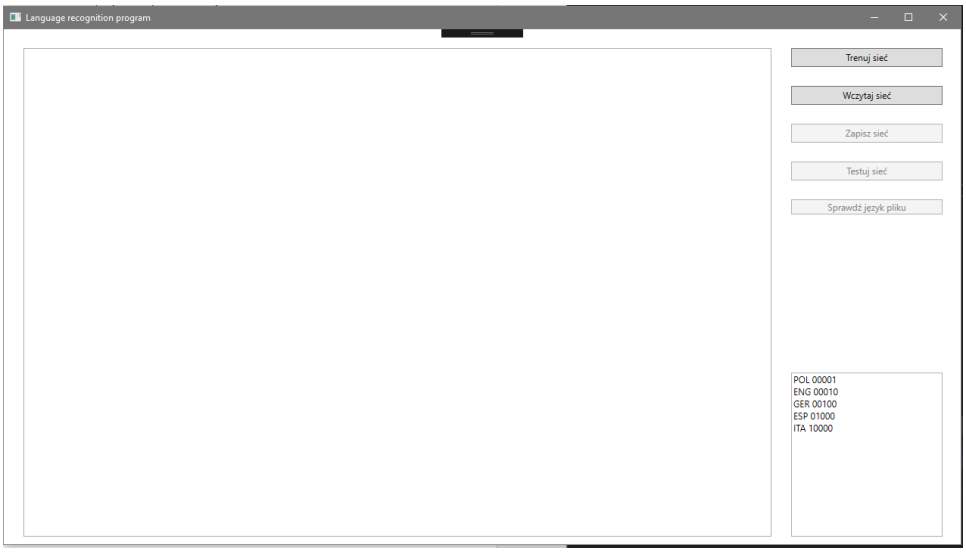
- Weights.txt – plik z wagami dla sieci neuronowej. Program może zaczytać stąd wagi dla swojej sieci lub zapisać wagi z aktualnie wytrenowanej sieci.

Dalej mamy foldery ze wszystkimi zgromadzonymi przez nas tekstami oraz foldery traindata i testdata, które zawierają foldery z plikami odpowiednio do trenowania i testowania sieci. Pliki tekstowe mogą być dowolnie przygotowane – program sam na podstawie tekstu stworzy sobie odpowiednie dane wejściowe. Jedynym wymogiem jest poprawność językowa zawartych tekstów.

Dane wyjsciowe są zapisane na 5 bitach co pozwala na zakodowanie 32 języków. Najlepsze wyniki uzyskuje się jednak dla 5 języków, gdy każdy z nich używa unikalnej kombinacji zakodowanej jako jeden z 5 bitów. Podczas kodowania wielu języków z użyciem tego samego bitu skutaczność sieci znacząco spada.

5. Uruchamianie i testowanie

W momencie włączenia programu wyświetla nam się okno główne programu.



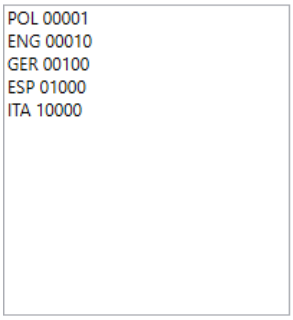
Rysunek 1: Widok początkowy programu

Przycisk „Trenuj sieć” spowoduje wytrenowanie sieci językami zdefiniowanymi w „ConfigFile.txt” plikami zawartymi w folderze „traindata”.

Przycisk „Wczytaj sieć” spowoduje wczytanie wag sieci z pliku „Weights.txt”.

Pozostałe opcje są zablokowane, gdyż po uruchomieniu programu nie są użyteczne.

Z komentarzem [DR1]:

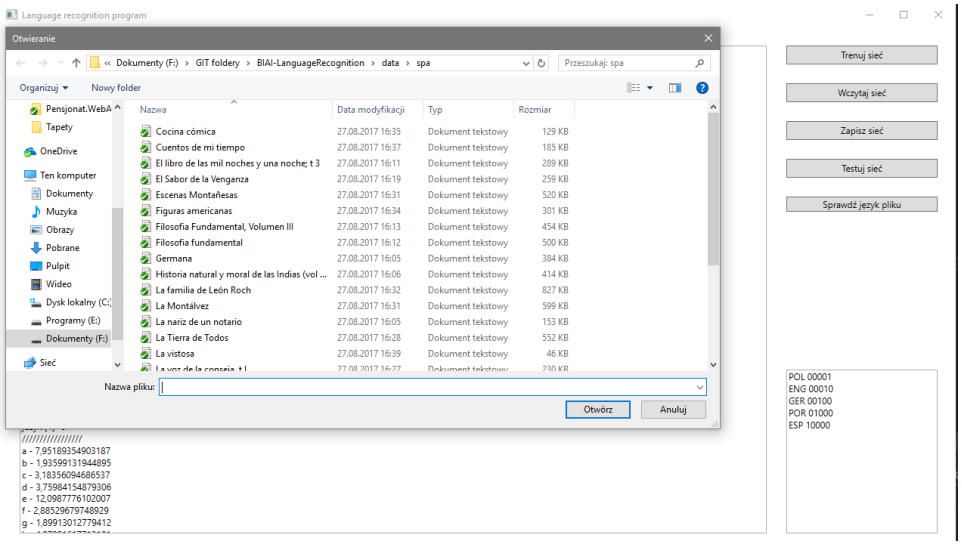


Rysunek 2: Pole z aktualnie używanymi językami

Po wytrenowaniu lub wczytaniu sieci możemy przeprowadzić testy sieci.

Przycisk „Zapisz sieć” spowoduje zapisanie wag aktualnej sieci neuronowej.

Przycisk „Testuj sieć” spowoduje przetestowanie sieci plikami z folderu „testdata”.



Rysunek 3: Okno programu wraz z okienkiem dialogowym do wyboru pliku

Przycisk „Sprawdź język pliku” spowoduje wyskoczenie okienka z wyborem pliku oraz późniejszym sprawdzeniu języku pliku.

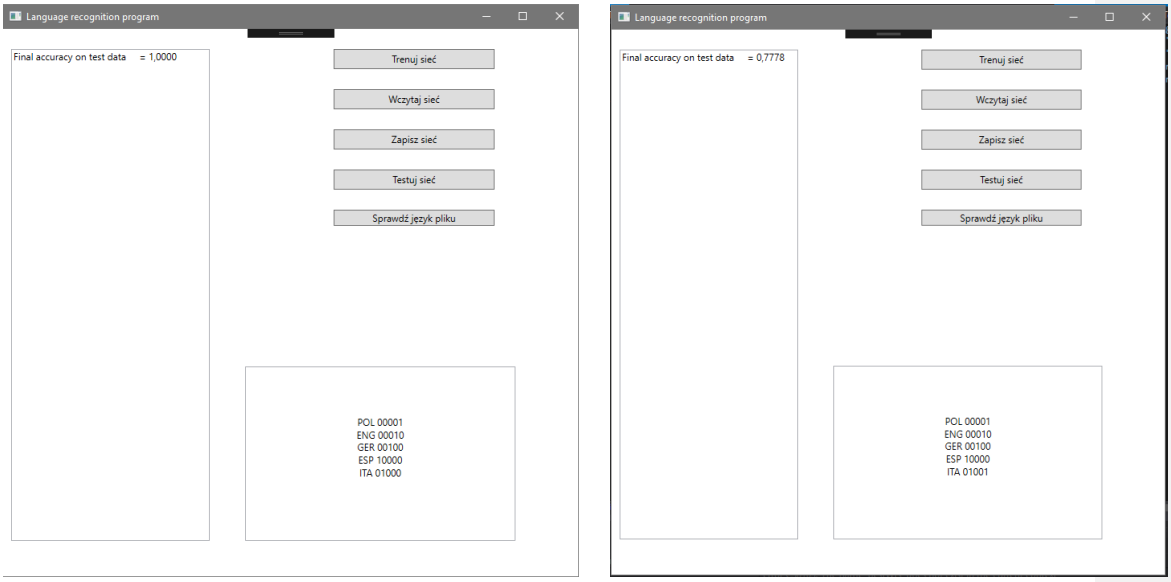
Wyniki naszych działań pojawiają się w głównym polu tekstowym programu.

6. Analiza sieci neuronowej

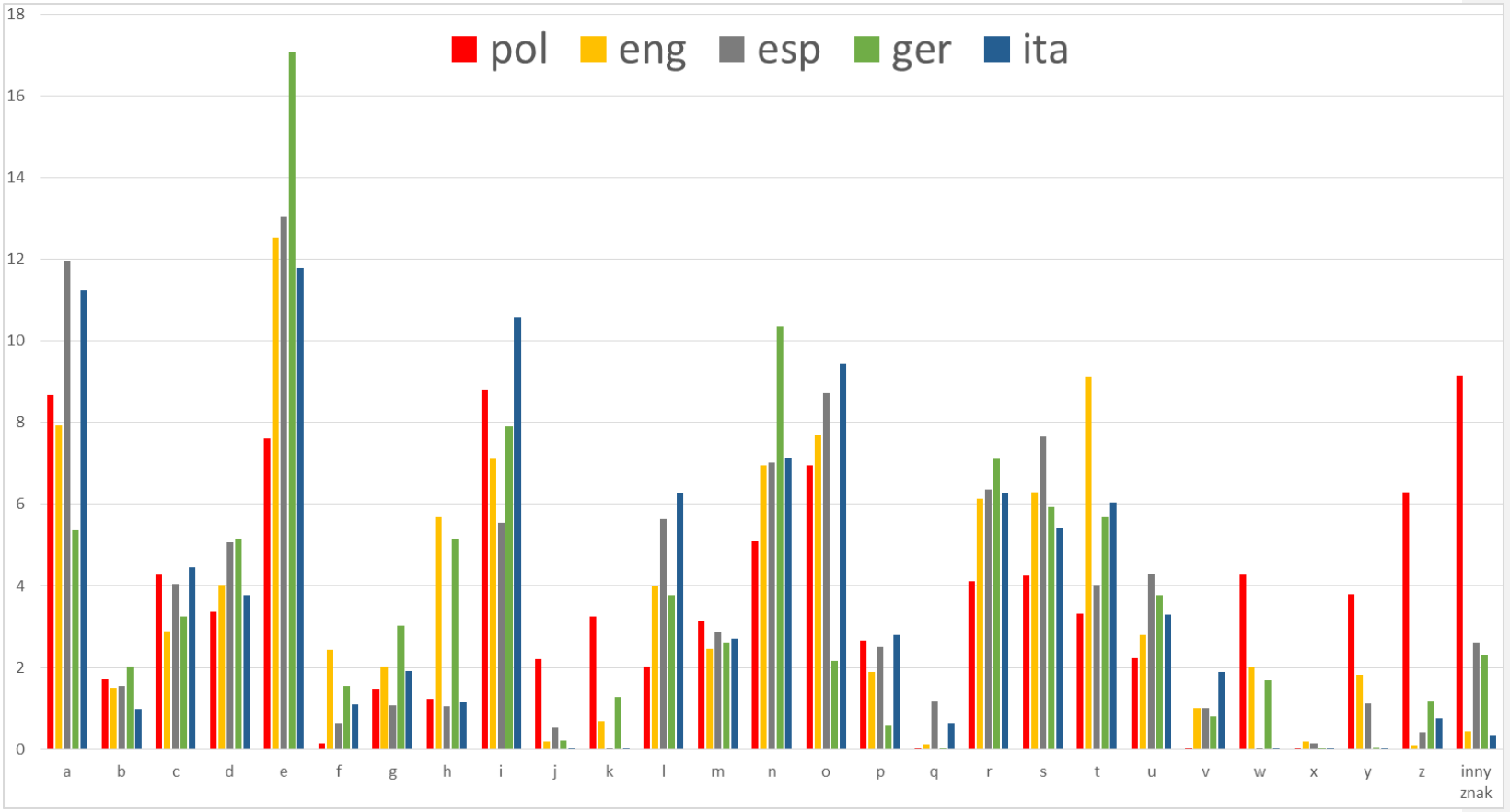
Parametry sieci neuronowej zostały przez nas dobrane na podstawie testów i poradników znalezionych w Internecie. Nasza sieć składa się z 27 neuronów warstwy wejściowej, 7 warstwy ukrytej i 5 warstwy wyjściowej. Współczynnik uczenia (learning rate) ustawiliśmy na 0,05, a wagę rozkładu na 0,01. Ustawiliśmy ilość epok uczenia na 1000 i spośród nich sieć wybiera epokę o najmniejszym współczynniku błędu. Najczęściej są to dość młode epoki (do ok. 150 epoki), starsze epoki albo nie poprawiają wyników, albo powodują ogromny wzrost błędów sieci. Błąd liczymy porównując wynik obliczony przez sieć z oczekiwanym rezultatem. Wynik zaokrąglamy od 0.7 do wartości 1, a wyniki niższe do wartości 0. Parametry nie mają zbyt dużego wpływu na naszą sieć. Wyniki są bardziej zależne od doboru języków oraz ich kodowania.

7. Wyniki

Poniżej przedstawiamy testy różnych konfiguracji oraz komentarze do nich.



Rysunek 4: Różnice w skuteczności sieci po zmianie kodowania (język włoski ITA z 01000 na 01001)



Wykres 1: Porównanie statystyk rozkładu częstości występowania liter dla wszystkich języków w teście

Jak widać pomiędzy językami widoczne są różnice. Można zauważyć, że każdy z języków ma przynajmniej jeden znak szczytowy, który pozwala mu się odróżnić od pozostałych języków. Dla człowieka taki wykres jest ciężki w interpretacji, jednak dla sieci neuronowej takie dane są potrzebne do przeprowadzenia identyfikacji języka.

Dane treningowe

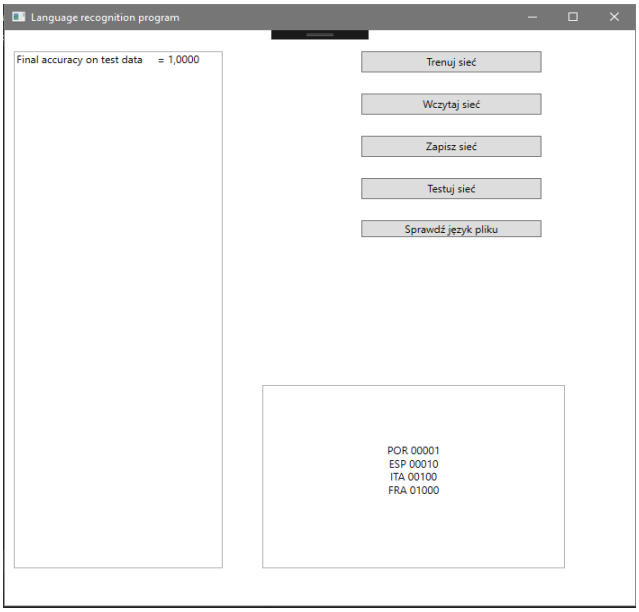
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	inny znak
pol	8.67	1.71	4.27	3.36	7.62	0.14	1.47	1.23	8.79	2.21	3.24	2.02	3.14	5.10	6.96	2.66	0.00	4.11	4.26	3.31	2.22	0.01	4.26	0.01	3.80	6.29	9.15
eng	7.92	1.51	2.89	4.03	12.53	2.44	2.01	5.68	7.12	0.19	0.68	4.00	2.46	6.96	7.70	1.88	0.11	6.13	6.28	9.14	2.81	1.01	2.00	0.19	1.81	0.09	0.43
ger	5.35	2.02	3.24	5.16	17.08	1.56	3.01	5.16	7.91	0.21	1.27	3.77	2.62	10.36	2.17	0.56	0.02	7.10	5.92	5.67	3.78	0.80	1.69	0.02	0.05	1.18	2.30
ita	11.24	0.98	4.45	3.77	11.79	1.09	1.90	1.17	10.58	0.02	0.01	6.27	2.70	7.13	9.45	2.80	0.64	6.26	5.40	6.04	3.30	1.88	0.00	0.03	0.01	0.76	0.35
esp	11.94	1.56	4.06	5.06	13.03	0.63	1.07	1.05	5.55	0.52	0.01	5.64	2.87	7.01	8.72	2.49	1.18	6.36	7.65	4.02	4.29	1.00	0.01	0.14	1.11	0.41	2.62

Dane testowe

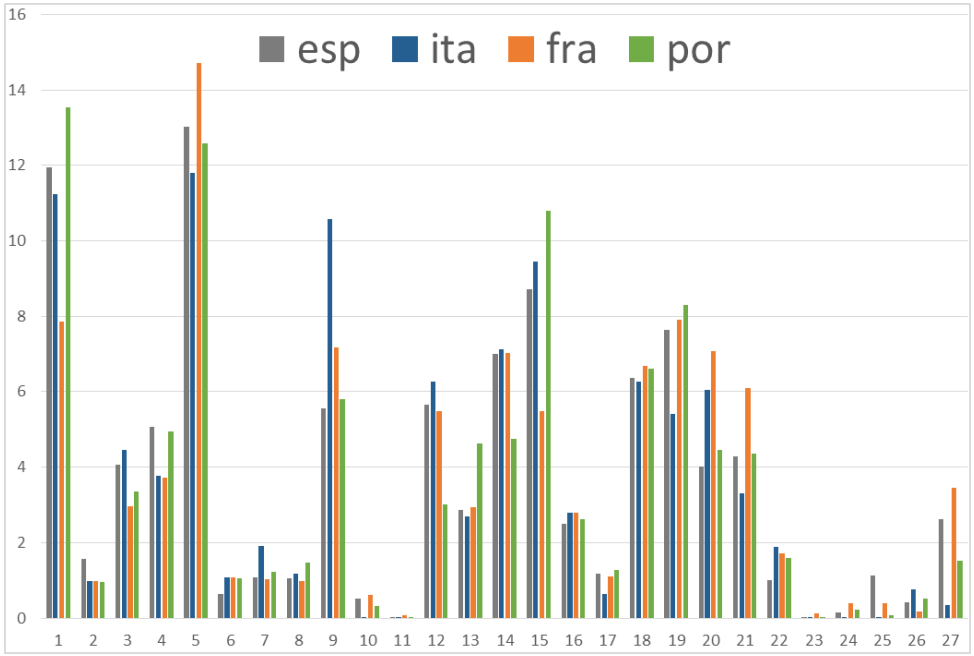
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	inny znak
pol	8.21	1.91	4.29	3.51	8.22	0.10	1.44	1.26	8.51	2.70	3.25	2.12	3.16	4.89	7.00	2.71	0.00	4.14	4.37	3.49	2.13	0.45	3.85	0.06	3.26	6.03	8.94
eng	7.92	1.53	2.71	4.37	12.42	2.30	2.01	5.83	7.02	0.17	0.69	3.82	2.55	6.98	7.69	1.94	0.12	6.07	5.98	9.29	2.86	0.96	2.14	0.19	1.94	0.09	0.38
ger	5.61	1.96	3.16	5.24	16.64	1.55	2.85	5.12	7.55	0.23	1.38	3.77	2.55	10.41	2.36	0.65	0.02	6.89	6.33	5.69	3.86	0.69	1.82	0.02	0.04	1.13	2.48
ita	11.09	0.98	4.32	3.77	12.07	1.07	1.90	1.31	10.71	0.02	0.02	6.32	2.93	6.90	9.50	2.70	0.57	6.35	5.36	6.05	3.30	1.79	0.01	0.02	0.02	0.71	0.21
esp	11.99	1.51	4.12	4.88	12.90	0.65	1.10	1.01	5.76	0.47	0.01	5.57	2.87	7.11	8.91	2.55	1.08	6.49	7.78	4.16	4.26	1.07	0.01	0.12	1.09	0.40	2.14

Tabela 1: Porównanie rozkładu dla danych treningowych i testowych

Jak widać po wynikach z tabel, rozkłady statystyczne częstości występowania liter w poszczególnych językach są stałe i różnice rozkładu między różnymi tekstami są minimalne. Wyniki poszczególnych liter mogą się minimalni różnić pomiędzy zbiorami, jednak dzięki zachowaniu wartości charakterystycznych (liter, których występowanie znacznie różni się pomiędzy językami) sieć neuronowa nie ma problemów z identyfikacją tych języków.



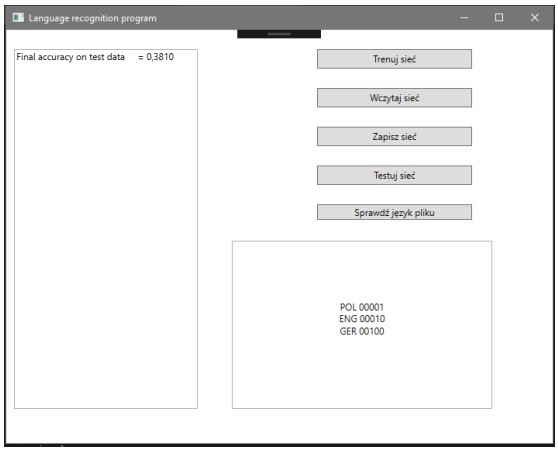
Rysunek 5: Porównanie języków romańskich: francuskiego, włoskiego, hiszpańskiego i portugalskiego.



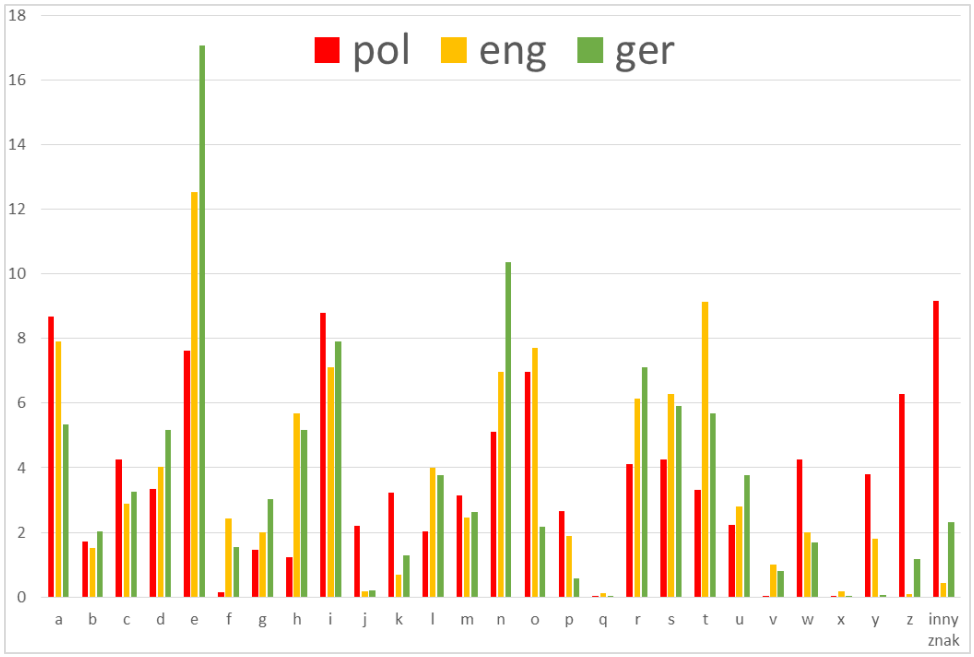
Wykres 2: Porównanie języków romańskich: francuskiego, włoskiego, hiszpańskiego i portugalskiego.

Jak widać na powyższych przykładach mimo pochodzenia języków francuskiego, hiszpańskiego, włoskiego i portugalskiego z jednej rodziny języków – języków romańskich, różnice statystyczne między nimi są wyraźnie zauważalne. Jest to przypadek, w którym uzycie sieci neuronowej jest bardzo pomocne, ponieważ języki są do siebie tak podobne, że nie da się ich „na oko” od siebie odróżnić.

Na koniec zostawiliśmy najciekawsze naszym zdaniem porównanie, które dotyczy języków polskiego, angielskiego i niemieckiego.



Rysunek 6: Porównanie języków: polskiego, niemieckiego i angielskiego.



Wykres 3: Porównanie języków: polskiego, niemieckiego i angielskiego.

Porównanie to jest o tyle wyjątkowe, że w grupie 5 języków razem z włoskim i hiszpańskim sieć jest w stanie zawsze rozróżnić od siebie te języki, jednak gdy chcemy porównać je tylko we trzy to sieć nie jest sobie w stanie z tym poradzić. Jest to bardzo dziwna i zaskakująca sytuacja, tym bardziej, że na wykresie widać, że języki te mają charakterystyczne dla siebie częstości występowania liter. Podczas sprawdzania pojedynczych plików okazało się, że problem jest z rozróżnieniem języków angielskiego i niemieckiego – sieć nie klasyfikuje ich jako żaden z języków. Jest to prawdopodobnie spowodowane tym, że język polski ma 8 (c, f, j, k, w, y, z, inny znak) charakterystycznych liter, język angielski tylko jedną natomiast język niemiecki tylko dwie takie litery. Prawdopodobnnie powoduje to dominację wag języka polskiego i inne języki nie są w stanie wyliczyć dla siebie odpowiednio wysokich wartości dla neuronów wyjściowych.

8. Wnioski

Przed rozpoczęciem projektu sieci neuronowe były dla nas tylko ogólnie rozumianą ideą. Dzięki zaimplementowaniu własnej sieci oraz późniejszym przeprowadzniu na niej badań lepiej poznalismy zasady jej działania oraz jej mocne i słabe strony. Spory problem stanowiło odpowiednie zaimplementowanie algorytmów odpowiadających za działanie sieci neuronowej, jednak dzięki licznym źródłom udało nam się zaimplementować własne rozwiązanie, a następnie podczas testów zmienić niektóre funkcje zgodnie z naszymi wymaganiami. Jest to kolejna, obok poszerzenia własnej wiedzy, zaleta używania własnego rozwiązania. Gotowej biblioteki nie moglibysmy tak łatwo dostosować do własnyh potrzeb. Sporo czasu poświęciliśmy także na badania języków - najpierw na dobranie odpowiednich przykładów, a następnie na przeprowadzenie dokładnych badań. Wyniki były dla nas dość zaskakujące. Nie spodziewaliśmy się, że sieć tak dobrze poradzi sobie z rozróżnaniem wielu różnych języków (rysunek 4) oraz języków z tej samej rodziny (rysuunek 5). Zaskoczył nas także wynik testów widocznych na rysunku numer 6. Wydawało nam się, że będzie to najłatwiejsza sytuacja dla siei neuronowej, jednak po analizie wyników doszlismy do wniosku, że dominacja charakterystycznych liter jednego języka jest dla sieci gorszą sytuacją od wielu podobnych języków z pojedynczą literą charakterystyczną.

9. Źródła

- http://www.theprojectspot.com/tutorial-post/introduction-to-artificial-neural-networks-part-2-learning/8
- http://fann.sourceforge.net/fann_pl.pdf

10. Projekt GIT

- https://github.com/xXhariboXx/BIAI-Projekt-LanguageRecognition