**Sieci neuronowe i neurosterowniki**

**Projekt nr.2**

|  |  |
| --- | --- |
| **Prowadzący:**  Dr. Inż. Krzysztof Halawa | **Termin:**  Środa 13:15 TP |

**Autorzy:**

Armand Piecyk 235293

Radosław Bugiel 235118

1.Wstęp

Zadanie polegało na stworzeniu głębokiej sieci neuronowej- CNN(convolutional neural network). Dane pobrano ze strony https://www.cs.toronto.edu/~kriz/cifar.html, oznaczone jako Cifar-10. Dane składają się z 60 000 obrazków o wielkości 32x32 podzielonych na 10 klas po 6 000 obrazków każda. Są to: samolot,samochód,ptak,kot,sarna,pies,żaba,koń,statek i ciężarówka. Zestaw danych zawiera 50 000 instancji treningowych i 10 000 instancji testowych. Zestaw jest podzielony na 5 zestawów treningowych i 1 zestaw testowy, każdy po 10 000 obrazów. Zestaw testowy składa się z 1000 losowo wybranych zdjęć każdej klasy.

2.Program i testy

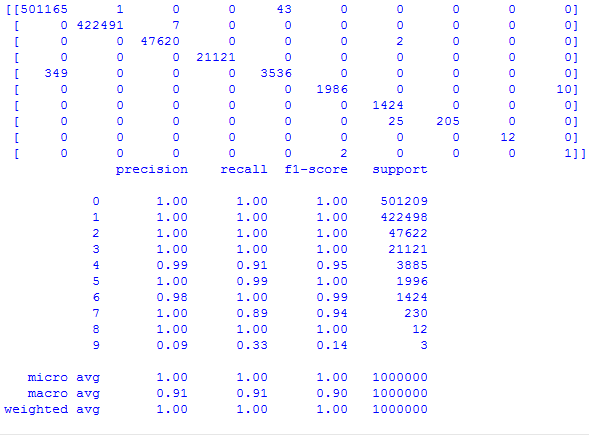
Do stworzenia sieci wykorzystano język Python i bibliotekę Keras.

//wrzucic kod

*1.Kod programu w języku Python*

Głównym celem testów było uzyskanie zadowalającej precyzji ocenianych danych. Wyniki wahają się na granicy 85- 92% poprawnej klasyfikacji. Czas pracy programu wynosi około 80 minut. Z tak dużym czasem wykonywania ciężko było zrobić dużą ilość testów, więc postanowiliśmy opisać argumenty wpływające na wynik.

/\*Macierz znajdująca się na poniżej pokazuje jak sieć klasyfikował dane. Każdy wiersz odpowiada wartości układu pokerowego tworzonego przez zestaw 5 kart(0-9), a każda kolumna oznacza ilość sklasyfikowanych tam układów. Pierwszy wiersz pokazuje, że z 501209 zestawów danych o wartości 0 sieć poprawnie sklasyfikowała 501165 zestawów, 1 zestaw sklasyfikowała z wartością 1, a 43 zestawom przypisał wartość 4. Następnie w drugim wierszu sieć przypisała złe wartości tylko 7 zestawom, gdzie 422491 zostało ocenione poprawnie. Najmniejsza precyzja jest dla układu o wartości 9, gdzie w całym zestawie danych są 3 takie instancje i tylko jedna została poprawnie sklasyfikowana.\*/ nie wiem czy chcemy to, mozemy wrzucic zdj 1 macierzy i znowu opisac co tam sie dzieje, dlatego tego nie wywalam

****

*2.Wyniki działania programu.*

Nasz program uczy się na 125 „epokach”, przy pierwszych 75 learning rate(lrate) wynosi 0.001, dla następnych 25 epok mamy lrate równe 0.0005, a ostatnie 25 epok wykonuje się z lrate na poziome 0.0003. Jeden test polegał na ujednoliceniu lrate i po testach, gdzie 125 epok wykonało się z lrate równym 0.001 nie było zmian, ani w precyzji, ani w czasie wykonywania.

Zmieniając ilość epok nie zauważyliśmy żadnych znaczących zmian precyzji, zmianie uległ tylko czas wykonywania programu. Dla 80 epok jest on równy 50 minut, a dla 170 epok 105 minut.

Zmiana parametru VGG Blocks z 3 na 2 daje wyraźny spadek precyzji o ponad 3%.

Jako funkcji aktywującej użyliśmy ELU(Exponential Linear Unit), funkcji bardzo podobnej do Relu.

3. Wnioski

/\*Największy wpływ na poprawność predykcji ma wybór funkcji aktywacji. Najlepsze wyniki otrzymano dla funkcji ReLU. Kolejnym ważnym elementem była ilość ukrytych warstw oraz liczba neuronów w poszczególnej warstwie. Najlepsza okazały się 3 warstwy po 70 neuronów. Zwiększanie lub zmniejszanie obu wartości skutkowało pogarszaniem efektywności. Pozostałe współczynniki dla szerokich zakresów wartości miały nieznaczny wpływ poprawność predykcji. Mogło to wynikać z ilości ukrytych neuronów.\*/ nie wiem co tu napisać