Wprowadzenie do algorytmów ewolucyjnych
Projekt - Raport końcowy

Paulina Dziadkiewicz
Piotr Wolski
Informatyka FMSI1
PW MiNI

1 Opis problemu

Tytuł: Generowanie złośliwych danych

Zastosować algorytm ewolucyjny do generowania (możliwie niewielkich) szumów, które będą zakłócały działanie sieci neuronowych wytrenowanych do klasyfikacji zdjęć.

2 Koncepcja rozwiązania

2.1 Zmiany względem idei z dokumentacji wstępnej

Według dokumentacji wstępnej program powinien tak przerabiać zdjęcia, aby nie tylko były one w stanie oszukać klasyfikatory zdjęć w rozumieniu "Klasyfikator się myli, ponieważ dla zdjęcia dotychczas oznaczanego jako psa, zaczął zwracać inną - dowolną - klasę", ale także by można było kontrolować to, jaką konkretnie klasę ma teraz oszukany klasyfikator wskazywać.

Model sieci neuronowej Inception-V3[3] obsługuje 1000 różnych klas, podczas testów można było zauważyć z jaką trudnością programowi ewolucyjnemu przychodziło "wstrzelenie się" w wybraną przez nas jedną z tysiąca klas. Z powodu bardzo długiego czasu liczenia zdecydowaliśmy, że nasz program będzie starał się znaleźć taką modyfikację zdjęcia, by wynik zwracany przez sieć był jak najdalszy od pierwotnie przypisanej klasy, ale zmieniony w dowolnym kierunku, a nie skojarzonym z jakąś konkretną klasą. To podejście znacznie przyspieszyło liczenie i pozwoliło na uzyskanie wymiernych wyników.

Działanie programu wciąż pokrywa się w pełni z zadaniem generowania szumów zakłócających działanie sieci klasyfikacyjnej, ale zastosowaliśmy kompromis między jakością wyników, a czasem jego pracy.

2.1.1 Wykorzystana baza danych

Jako dane wejściowe zostało wykorzystanych kilka zdjęć dostępnych publicznie, przedstawiających obiekty, które sieć Inception-v3 jest zdolna rozpoznać.

2.2 Opis implementacji

2.2.1 Algorytm ewolucyjny

Parametry związane z problemem:

- p_{max} - maksymalne odchylenie wartości piksela od wartości oryginalnej
- p_{min} - minimalne odchylenie wartości piksela od wartości oryginalnej
- *fake_class_prob_to_get* - wartość prawdopodobieństwa, poniżej której ma spaść odpowiedź sieci neuronowej przy ocenie przynależności danego zdjęcia do swojej oryginalnej klasy

Parametry algorytmu ewolucyjnego:

- N - rozmiar populacji

- G - liczba generacji
- k - procent najlepiej dostosowanych osobników wziętych z poprzedniej populacji do następnej
- p - prawdopodobieństwo krzyżowania
- m - prawdopodobieństwo mutacji
- f - procent pikseli obrazka, które będą modyfikowane przez algorytm (fenotyp)

Powtarzaj poniższą sekwencję kroków maksymalnie G razy lub aż do znalezienia osobnika w populacji, który spełnia *fake_class_prob_to_get*:

1. Przygotowanie populacji wstępnej:
   - Powtórz N razy:
     - Wybranie losowych f pikseli z obrazka
     - Do każdego z wybranych pikseli dodaj liczbę z przedziału $[p_{min}, p_{max}]$
     - Przytnij piksele, które przekroczyły wartość obsługiwaną przez Inception-V3
     - Dodaj nowo utworzony obrazek do populacji
2. Wybranie najlepszych osobników
   - Sprawdzenie i zapisanie wyniku prawdopodobieństwa przynależności do oryginalnej klasy zwracanego przez sieć dla każdego obrazu z populacji
   - Wybór najlepszych k osobników z populacji
3. Krzyżowanie
   - Wybór dwóch losowych osobników z populacji i skrzyżowanie wartości pikseli dla p procent całkowitej liczby wybranych pikseli
   - Krzyżowanie dwóch wartości pikseli na tych samych pozycjach pomiędzy dwoma odpowiadającymi sobie obrazkami jako obliczenie średniej arytmetycznej między nimi.
4. Mutacja
   - Stworzenie kopii dla m procent obrazków z populacji
   - W każdej kopii do każdego zmienianego piksela dodanie losowej wartości z przedziału $[p_{min}, p_{max}]$ i dołączenie uzyskanej nowej wersji kopii do populacji
   - Populacja posiada teraz więcej elementów niż na starcie, ale nie stanowi to problemu, ponieważ operacja selekcji wybiera zawsze k najlepszych osobników, co pozwala na zachowanie stałego rozmiaru populacji.

2.2.2 Poszukiwanie optymalnych parametrów

Opracowano skrypt, który dla określonych zakresów parametrów dotyczących algorytmu ewolucyjnego testuje wszystkie ich kombinacje i pozwala na wybranie takiego zestawu parametrów, dla którego zadowalający jest kompromis pomiędzy czasem działania a uzyskanym wynikiem.

Wybrane wyniki przetestowanych różnych zestawów parametrów są zaprezentowane w kolejnej sekcji.

3 Przeprowadzone eksperymenty

3.1 Klasa "Pies"

3.1.1 Oryginalny obraz

3.1.2 Testowane zestawy parametrów

	Zestaw 0	Zestaw 1	Zestaw 2	Zestaw 3
N - rozmiar populacji	100	100	150	150
G - maksymalna liczba generacji	500	500	500	500
k - zachowywane osobniki	20%	40%	20%	20%
p - prawdopodobieństwo krzyżowania	40%	40%	40%	40%
m - prawdopodobieństwo mutacji	10%	80%	10%	80%
f - piksele podlegające zmianie	40%	40%	40%	40%

3.1.3 Uzyskane obrazy i wyniki dla różnych zestawów parametrów

Zestaw 0

czas obliczeń: 35 min 12 sec

wynik: 97% pies

Zestaw 1

czas obliczeń: 3h 25 min 36 sec

wynik: 19% pies

Zestaw 2

czas obliczeń: 46 min

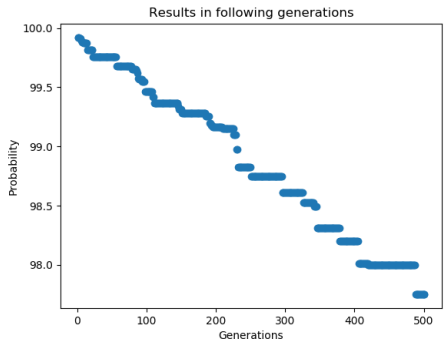
wynik: 98% pies

Zestaw 3

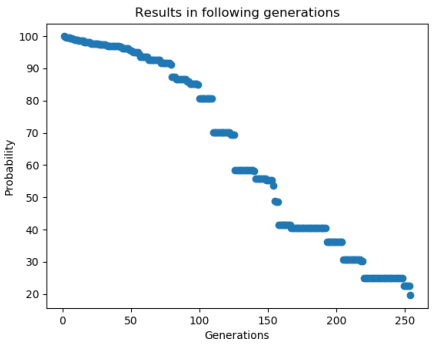
czas obliczeń: 1h 27 min 45 sec

wynik: 18% pies

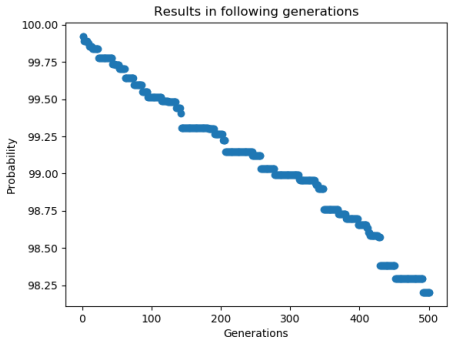
3.1.4 Zmiany wartości najlepszego wyniku w kolejnych iteracjach



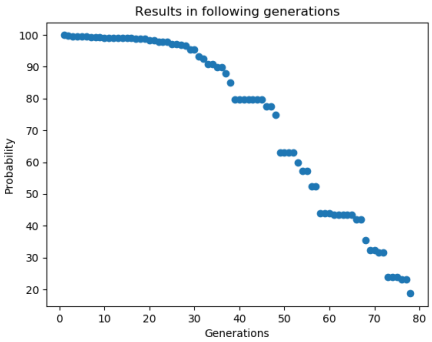
Zestaw 0



Zestaw 1

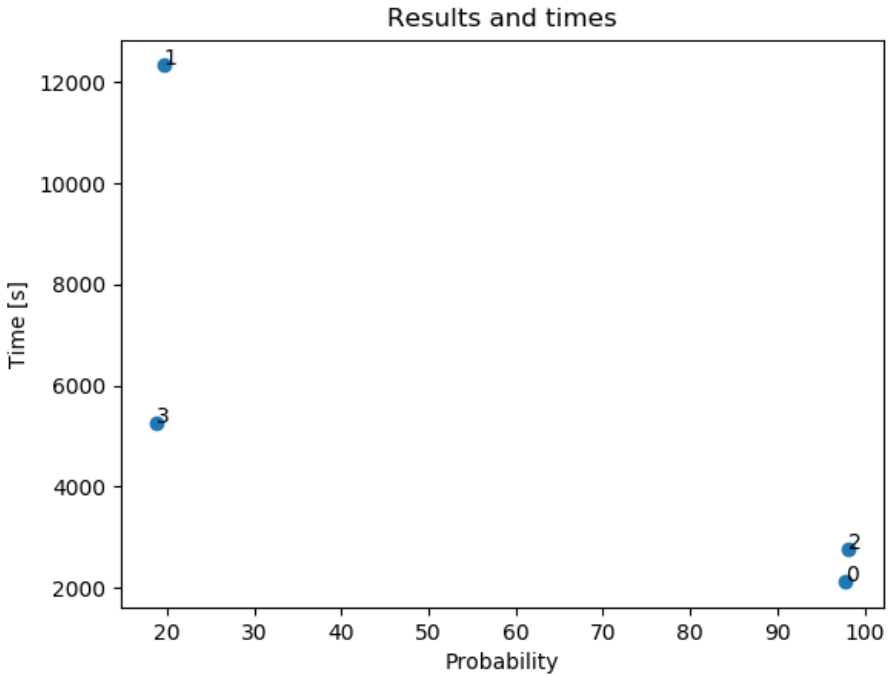


Zestaw 2



Zestaw 3

3.1.5 Zestawienie czasów i rezultatów dla wszystkich zestawów



3.1.6 Obserwacje

Zestaw nr 3 dał najlepszy wynik w najkrótszym czasie. Warto wybrać wysokie prawdopodobieństwo mutacji przy zachowanym niższym procencie osobników selekcjonowanych do następnej iteracji.

3.2 Klasa "Pizza"

3.2.1 Oryginalny obraz

3.2.2 Testowane zestawy parametrów

	Zestaw 0	Zestaw 1	Zestaw 2	Zestaw 3
N - rozmiar populacji	200	200	250	250
G - maksymalna liczba generacji	500	500	500	500
k - zachowywane osobniki	20%	40%	20%	20%
p - prawdopodobieństwo krzyżowania	40%	40%	40%	40%
m - prawdopodobieństwo mutacji	10%	80%	10%	80%
f - piksele podlegające zmianie	40%	40%	40%	40%

3.2.3 Uzyskane obrazy i wyniki dla różnych zestawów parametrów

Zestaw 0

czas obliczeń: 1h 1 min 36 sec

wynik: 42% pizza

Zestaw 1

czas obliczeń: 2h 4 min 57 sec

wynik: 18% pizza

Zestaw 2

czas obliczeń: 1h 14 min 56 sec

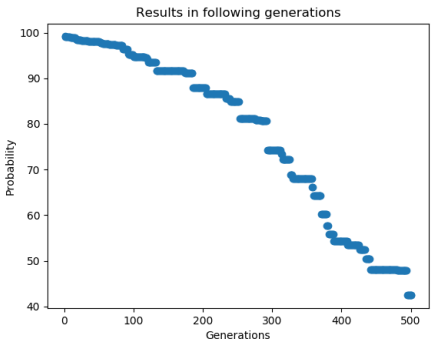
wynik: 52% pizza

Zestaw 3

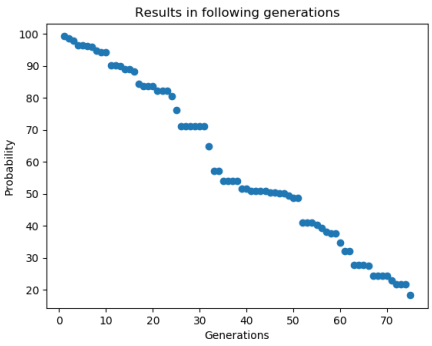
czas obliczeń: 6h 12 min 52 sec

wynik: 19% pizza

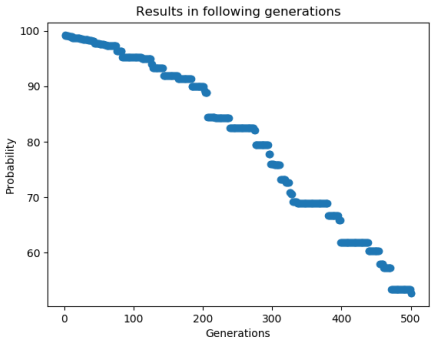
3.2.4 Zmiany wartości najlepszego wyniku w kolejnych iteracjach



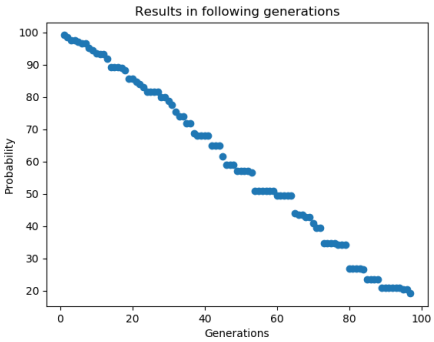
Zestaw 0



Zestaw 1



Zestaw 2



Zestaw 3

3.2.5 Obserwacje

Tym razem, na osiągnięcie najlepszych wyników w najkrótszym czasie pozwoliło zastosowanie zestawu nr 1 - podobnie jak w poprzednim przykładzie, dobry rezultat został osiągnięty przy wysokim prawdopodobieństwie mutacji.

Widać również, że choć zestaw nr 3 zawiera podobne parametry i zwraca podobny wynik, to zwiększenie populacji znacząco wydłużyło czas działania programu.

4 Wnioski

Przy użyciu algorytmu ewolucyjnego (genetycznego) możliwa jest realizacja zadania, którego celem jest generowanie niezauważalnego dla człowieka szumu na zdjęciu, który jednak powoduje, że sieć neuronowa wytrenowana do rozpoznawania przedmiotu na zdjęciu zaczyna zwracać słabe rezultaty.

W programie zastosowano parametry typowe dla implementacji algorytmu ewolucyjnego. Po serii wykonanych testów można przyjąć, że zwiększenie prawdopodobieństwa mutacji pozytywnie wpływa na jakość uzyskanego wyniku - prawdopodobnie dlatego, że operacja mutacji w dużym stopniu zmienia wartości pikseli. Choć spowalnia to również obliczenia, to wciąż pozostaje zabiegiem opłacalnym. Innym zabiegiem spowalniającym, nieprzynoszącym jednak pozytywnych efektów, jest zwiększanie populacji. Jeśli jednak celem jest jak najmniejsza zmiana wartości pikseli obrazu i jednoczesne uzyskanie dobrego wyniku, to lepiej jest zwiększyć populację kosztem czasu obliczeń.

Zauważalne jest również, że dla zestawów parametrów o tych samych krytycznych parametrach (np. prawdopodobieństwo mutacji) uzyskuje się zbliżone wyniki.

Załączniki

Kod źródłowy programu. Opis uruchomienia znajduje się w pliku README.md.

Literatura

[1] `https://blog.sicara.com/getting-started-genetic-algorithms-python-tutorial-81ffa1dd72f9` (dostęp 3.06.2018r.)

[2] `https://medium.com/@ageitgey/machine-learning-is-fun-part-8-how-to-intentionally-trick-n` 3.06.2018r.)

[3] `https://arxiv.org/abs/1512.00567` (dostęp 8.06.2018r.)