SIAMESE NETWORKS

Maciej Dobrzański,Grzegorz Sroka

Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

24 stycznia 2022

Plan prezentacji

Cifar-10

Zbiór Cifar-10 składa się z 60 000 małych, kolorowych obrazów (rozmiaru 32x32), które dzielą się na 10 rozłącznych klas. Dane podzielone są na zbiór treningowy składający się z 50 000 obrazów oraz testowy zawierający ich 10 000.

Problemy

W naszej pracy będziemy rozpatrywali 2 zagadnienia:

- **Klasyfikacja** - chcemy przyporządkowywać obrazom odpowiednią klasę.
- **Rozróżnianie** - dla pary obrazów chcemy odpowiedzieć na pytanie czy są pochodzą one z jednej klasy. W tym przypadku zbiorem testowym są pary obrazów powstałe poprzez połączenie podstawowego zbioru testowego oraz jego losowej permutacji.

Tak naprawdę obydwa te problemy są problemami klasyfikacji (pierwszy wieloklasowej, drugi binarnej).

Metryki

Podstawową metryką której będziemy używać jest standardowa dokładność oznaczająca po prostu stosunek poprawnych klasyfikacji do liczby wszystkich predykcji ($accuracy = \frac{(correct\ \ classifications)}{(all\ \ classifications)}$)
Dla ustalonej klasy (a) możemy również określić precyzję oraz swoistość:
$precision(a) = \frac{TP(a)}{TP(a)+FP(a)}$
$recall(a) = \frac{TP(a)}{TP(a)+FN(a)}$
TP(a) - prawidłowe przyporządkowanie do klasy a.
FP(a) - nieprawidłowe przyporządkowanie do klasy a.
FN(a) - nieprawidłowe przyporządkowanie do klasy innej niż a.

Metryki cd.

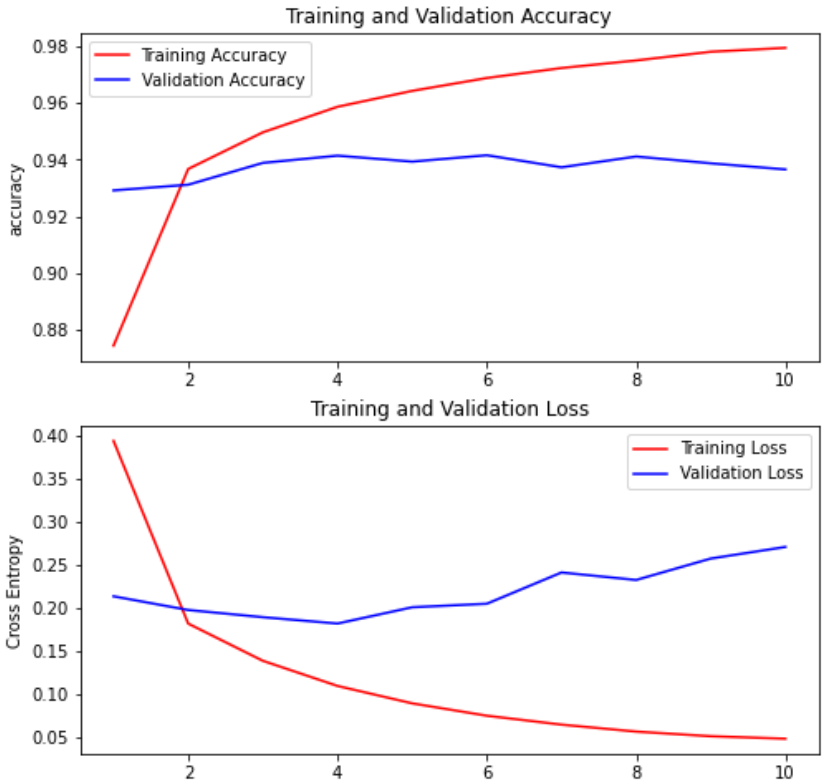
Na podstawie precyzji oraz swoistości możemy wyznaczyć metrykę F1-score dla danej klasy a ($F1(a) = \frac{2precision(a)recall(a)}{precision(a)+recall(a)}$)
F1-score jest jedną z powszechnie używanych miar oceny jakości klasyfikatorów, jest szczególnie pomocna w przypadku niezbalansowanych liczności klas.
F1-score dla całego zbioru będziemy wyznaczali jako średnią wartość tej miary po wszystkich klasach.
$F1 = \frac{\sum_{a \in classes} F1(a)}{|classes|}$

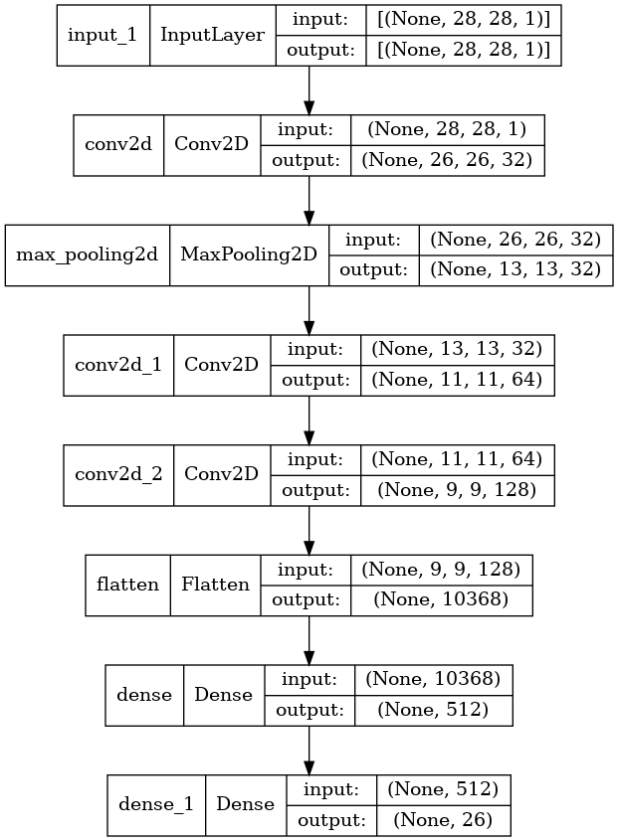
Benchmarki dla Cifar-10

Aktualnym modelem SOTA dla problemu klasyfikacji zbioru Cifar-10 jest model **VIT-H/14** (632 miliony parametrów) osiągający na nim dokładność rzędu **99.5%** .
Znane model osiągały na tym zbiorze następujące wyniki:
LaNet - 99.03
PyramidNet - 99.03
ResNet50 - 98.3
VGG-19 with GradInit - 94.71
ResNet - 92.3

Emnist (toy example)

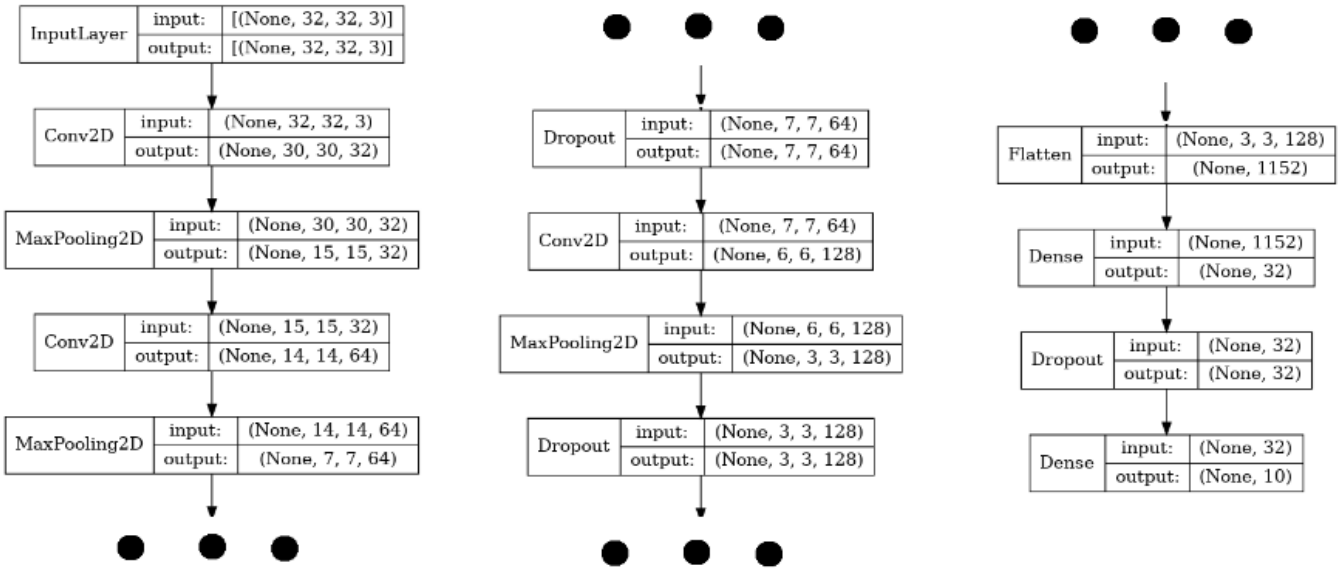
Przy uczenia na zbiorze Emnist użyliśmy prostej sieci konwolucyjnej. Wyniki poniżej:





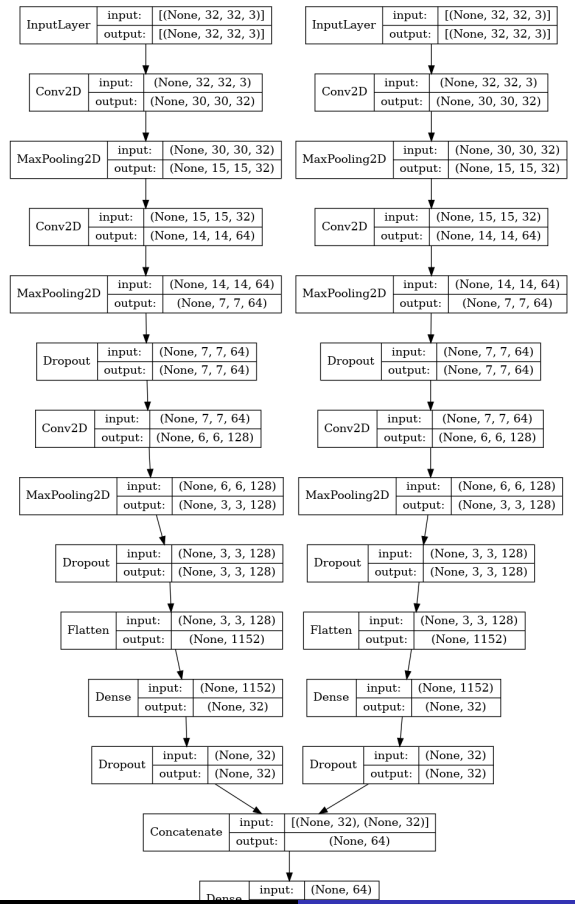
Vanilla network - opis architektury

Naszą sieć typu vanilla stworzyliśmy z myślą o szybkim uczeniu, przy zachowaniu sensownych możliwości reprezentacji danych. Składa się ona przede wszystkim z 3 warstw konwolucyjnych (o odpowiednio 32, 64 oraz 128 neuronach) połączonych blokami max-pooling.



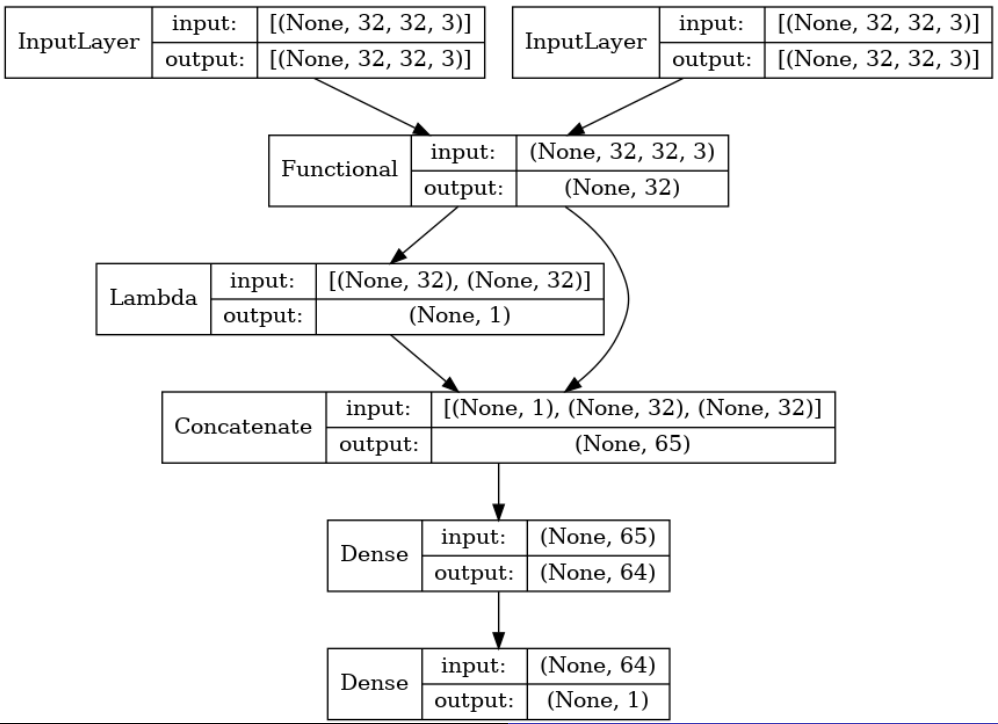
Siamese vanilla network - opis architektury

Sieć typu siamese stworzona z dwóch przedstawionych wcześniej sieci typu vanillla network, połączonych przy pomocy warstwy gęstej zawierającej 64 neurony.



Siamese vanilla network dla rozróżniania klas - opis architektury

Sieć typu siamese stworzona w celu sprawdzania czy dwa obrazy należą do tej samej klasy.
W elemencie oznaczonym "Lambda" jest wyliczana odległość euklidesowa pomiędzy wejściowymi wektorami.



Wyniki vanilla network dla problemu klasyfikacji

Sieć była uczona na zbiorze treningowym na przestrzeni 200 epok.
Do nauki używaliśmy straty categorical crossentropy.
Na zbiorze testowym osiągnęliśmy następujące rezultaty:

- Accuracy: 0.7890
- Macro f1 score: 0.7843
- Weighted f1 score: 0.7936

Wyniki siamese vanilla network dla problemu klasyfikacji

Odpowiadająca powyższej sieci sieć syjamska była uczona na duplikatach zbioru treningowego oraz dwóch zbiorach powstałych przez połączenie tego zbioru z jego permutacją zachowującą klasy. Na zbiorze testowym (gdzie podawane były na wejście duplikaty obrazów) osiągnęła ona rezultaty:

- Accuracy: 0.8133
- Macro f1 score: 0.8116
- Weighted f1 score: 0.8149

Wyniki vanilla network dla problemu klasyfikacji - zmniejszone zbiory treningowe

Wyniki dla zmniejszającego się zbioru treningowego:
Dla 50% danych treningowych:

- Accuracy: 0.7546
- Macro f1 score: 0.7541
- Weighted f1 score: 0.7550

Dla 10% danych treningowych:

- Accuracy: 0.6355
- Macro f1 score: 0.6314
- Weighted f1 score: 0.6395

Wyniki siamese vanilla network dla problemu klasyfikacji - zmniejszone zbiory treningowe

Analogiczne wyniki dla sieci syjamskiej. W tym przypadku zwiększona została również liczba dodatkowych permutacji na których uczona była sieć (2-krotnie w przypadku 10% oraz 5-krotnie w przypadku 10%) - warto zauważyć że dodatkowe permutacje dostarczane podczas treningu nie miały większego znaczenia dla skuteczności sieci.

Dla 50% danych treningowych:

- Accuracy: 0.7758
- Macro f1 score: 0.7747
- Weighted f1 score: 0.7768

Dla 10% danych treningowych:

- Accuracy: 0.6446
- Macro f1 score: 0.6422
- Weighted f1 score: 0.6469

Wyniki vanilla network dla problemu rozróżniania

Sieć wytrenowana dla problemu klasyfikacji sprawdzaliśmy dla problemu rozróżniania w taki sposób, że dla obydwu podanych obrazów przewidywaliśmy ich klasy i jeśli się zgadzały to zwracaliśmy 1, jeśli nie 0.
Na zbiorze testowym osiągnęliśmy następujące rezultaty:

- Accuracy: 0.9281
- F1 score: 0.6524
- Confusion matrix:
  [[8645 363]
  [336 656]]

Wyniki siamese vanilla network (uczonego pod klasyfikacje) dla problemu rozróżniania

Analogicznie postępowaliśmy z wytrenowaną wcześniej dla klasyfikacji siecią syjamską.
Na zbiorze testowym (gdzie podawane były na wejście duplikaty obrazów) osiągnęła ona rezultaty:

- Accuracy: 0.9347
- Confusion matrix:
  [[8672 336]
  [296 696]]
- F1 score: 0.6877

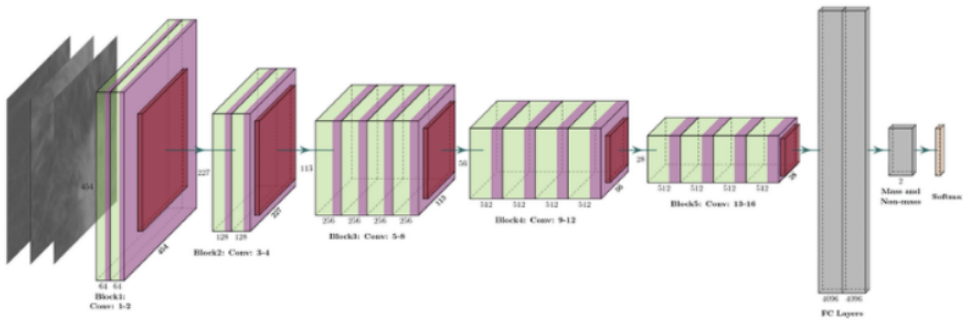
Wyniki siamese vanilla network (uczonego pod rozróżnianie) dla problemu rozróżniania

Sieć tę trenowaliśmy ze stratą binary crossentropy na parach obrazów (oryginalny zbiór i ponad 30 różnych permutacji), gdzie szukaną wartością była informacja czy obrazy pochodzą z tej samej klasy.
Na zbiorze testowym (gdzie podawane były na wejście pary obrazów) osiągnęła ona rezultaty:

- Accuracy: 0.9300
- Confusion matrix:
  [[8647 361]
  [339 653]]
- F1 score: 0.6510

VGG19 opis architektury

VGG19 jest to już przetrenowana sieć neuronowa składająca się z 19 warstw. Została zaproponowana w 2014 przez Visual Geometry Group na Uniwersytecie Oxfordzkim. Pretrenowana wersja tej sieci została wyćwiczona na ponad milionie obrazków z bazy ImageNet.



VGG19 transfer learning

Pobraliśmy z internetu już wyćwiczone wagi sieci VGG-19.
Następnie metodą transfer learningu cyzelowaliśmy parametry sieci.
Osiągnięte rezultaty:

- Accuracy: 0.9339
- Macro f1 score: 0.9339
- Weighted f1 score: 0.9338

VGG19 - uczenie sieci od zera

Zgodnie z artykułem przepisaliśmy implementację sieci VGG-19 do pythona. Podczas uczenia zastosowaliśmy callback EarlyStopping monitorujący loss na przedziale 30 epok. Z tego powodu ilości epok nie są równe. Uzyskaliśmy następujące rezultaty:
Klasyfikacja:

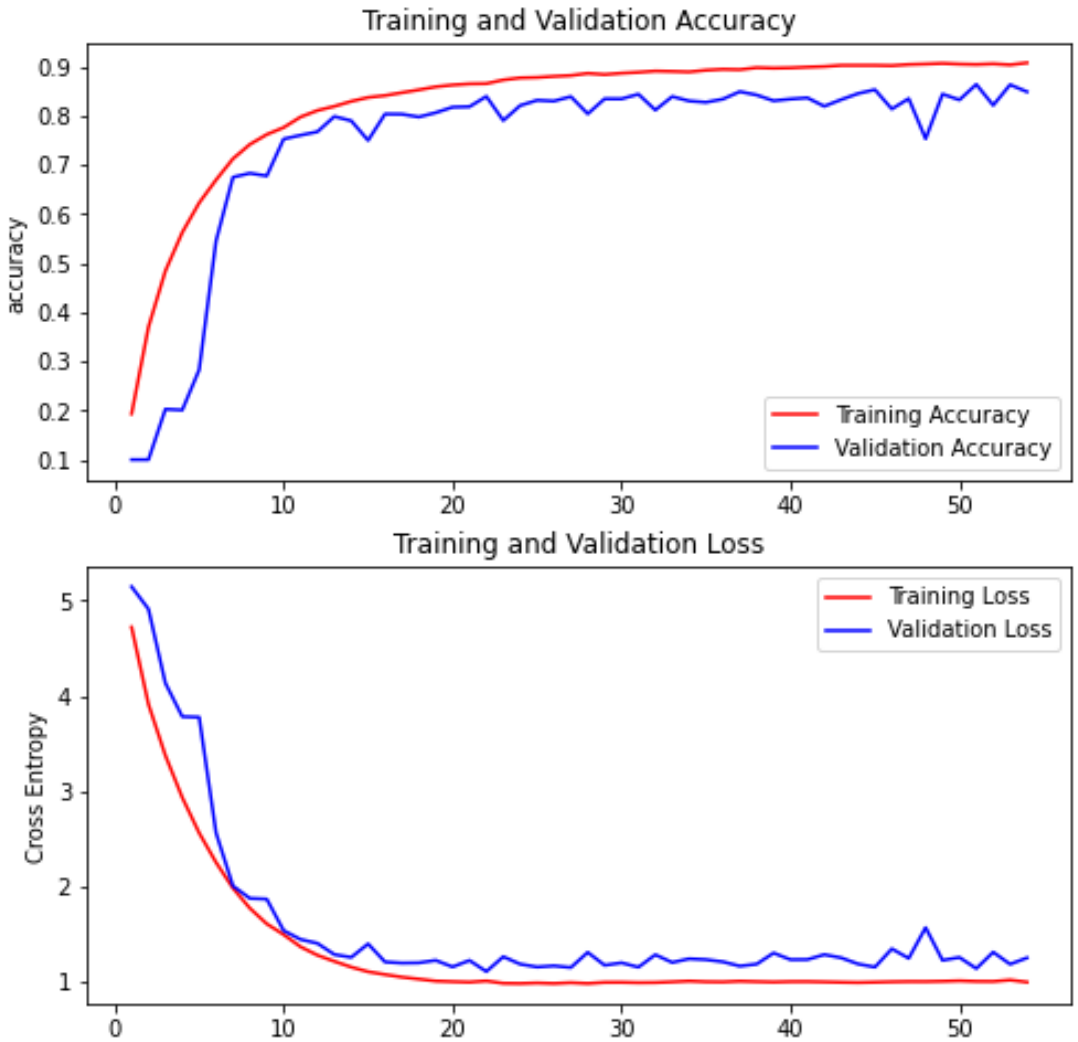
- 54 epoki
- Accuracy: 0.8492
- Macro f1 score: 0.8492
- Weighted f1 score: 0.8491

Klasyfikacja na zmniejszonym (50%) zbiorze treningowym

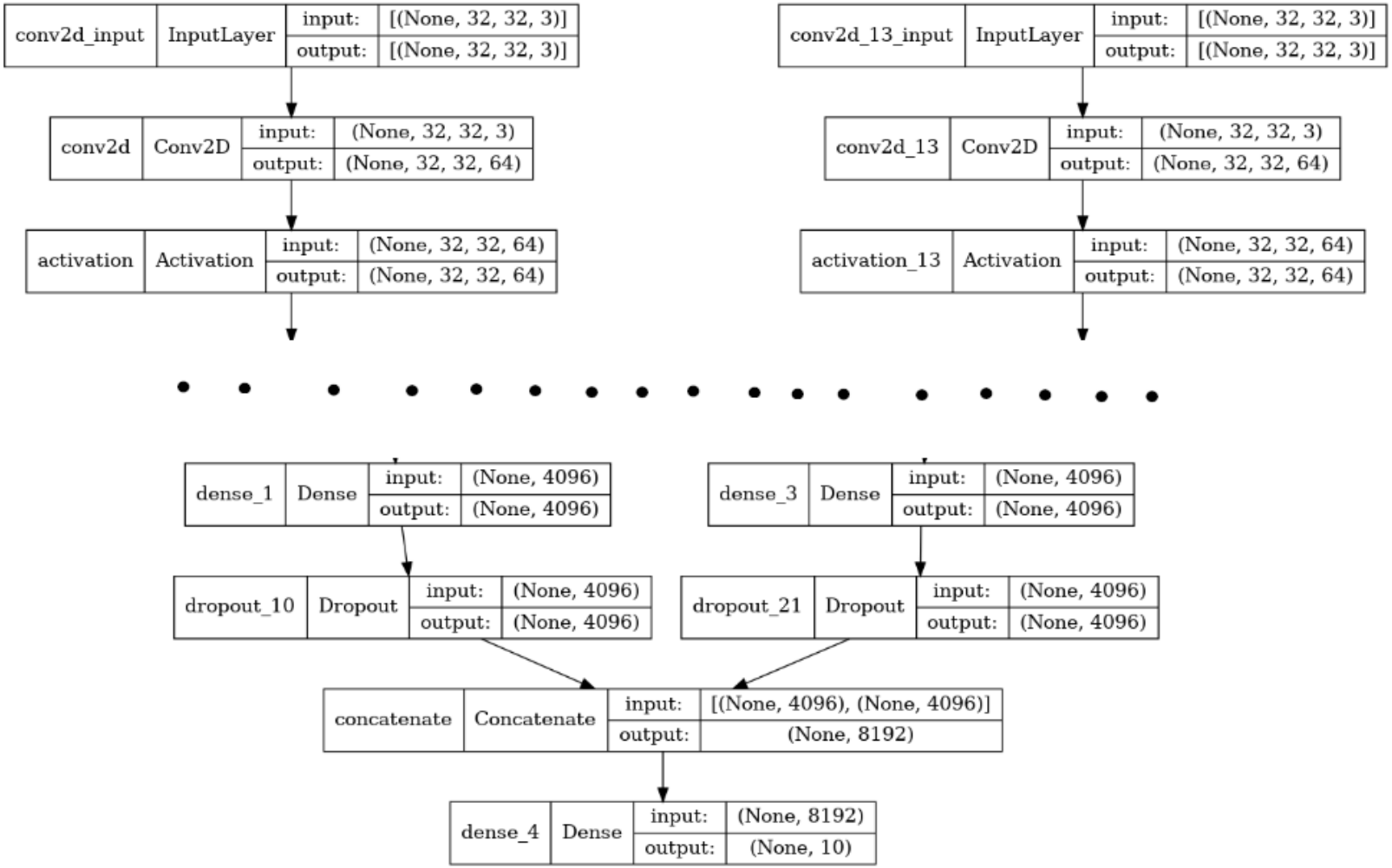
- 73 epoki
- Accuracy: 0.7998
- Macro f1 score: 0.7980
- Weighted f1 score: 0.8012

VGG19 - uczenie sieci od zera

Wykresy dla uczenia sieci VGG-19 na pełnym zbiorze cifar10



Siamese VGG19 wizualizacja



VGG19 - uczenie syjamskich

Utworzyliśmy sieć syjamską składającą się z dwóch nienauczonych sieci vgg-19. Uczenie przebiegało na środowisku Prometeusz.
Klasyfikacja:

- Epoki 400
- Accuracy: 0.8824
- Macro f1 score: 0.8834
- Weighted f1 score: 0.8813

Klasyfikacja na zmniejszonym (50%) zbiorze treningowym

- Epoki 400
- Accuracy: 0.8343
- Macro f1 score: 0.8377
- Weighted f1 score: 0.8308

Podsumowanie problemu klasyfikacji

W przypadku sieci typu vanilla network architektury typu siamese powstałe z sieci bazowych osiągały dla problemu klasyfikacji na zbiorze Cifar-10 wyniki o około 3 p.p lepsze od sieci bazowych. Dla zmniejszających się zbiorów treningowych powyższa kilkuprocentowa przewaga sieci syjamskich była utrzymywana, jednak pomimo naszych oczekiwań nie rosła ona nawet jeśli dokładaliśmy do treningu sieci syjamskich większą liczbę permutacji zbioru treningowego.

Podsumowanie problemu klasyfikacji cd.

W przypadku sieci VGG19 sytuacja wyglądała bardzo podobnie, to znaczy w przypadku kiedy trenowaliśmy sieci od zera sieci syjamskie osiągały wyniki lepsze o kilka punktów procentowych od sieci bazowej.
Nawet im nie udało się natomiast przebić wyników osiągalnych przez dostępne w internecie sieci VGG19 z gotowymi wagami, wytrenowanymi przy użyciu metod transfer learningu.

Podsumowanie problemu rozróżniania

Dużym rozczarowaniem były dla nas wyniki osiągane przez sieci tworzone z myślą o rozróżnianiu obrazów (decydowaniu czy obrazy pochodzą z jednej klasy).
Dopiero po wielu eksperymentach z architekturą i metodami uczenia, a ponadto długim czasie nauki (spowodowanym przekazywaniem wielu permutacji zbioru treningowego) wyniki tej architektury przebiły wyniki zwykłej sieci.
Ponadto sieć syjamska wytrenowana z myślą o klasyfikacji wciąż poradziła sobie z tym zadaniem lepiej.

Literatura

Główne źródła informacji:

- "Siamese Neural Networks for One-Shot Image Recognition" - G. Koch
- "Siamese Neural Networks: An Overview" - D. Chicco
- "Deep learning with Keras" - A. Gulli, S. Pal
- "A friendly introduction to Siamese Networks" - S. Benhur

Dziękujemy za uwagę

Dziękujemy za uwagę

Kod źródłowy:
`https://github.com/sgrzegorz/siamese-neural-networks`