# Fashion Mnist

Mateusz Górski, Jakub Dyczek

## 1 Wstęp

Repozytorium składa się z katologu z plikami .ipynb, w których zaimplementowane są kolejne modele opisywane poniżej, katalogu z danymi, oraz z pliku *mnist_ reader.py*, służącego do ładowania danych, pobranego z repozytorium FashionMnist.

## 2 Fashion Mnist

*FashionMnist* [1] jest stworzonym przez Zalando Research zbiorem $70k$ zdjęć ubrań podzielonych na 10 kategorii. Zbiór przypomina klasyczną bazę MNIST skoładającą się ze zdjęć cyfr zapisanych ręcznie. W obu zbiorach każde zdjęcie jest monochromatyczne wielkości 28x28 pikseli.

Autorzy przekonują, że warto zastąpić szeroko używany zbiór MNIST. Jest on zbyt prosty - konwolucyjne sieci osiągają na nim 99.7%, a podstawowe algorytmy ponad 97%. Co więcej, większość par cyfr można rozróżnić tylko za pomocą jednego piksela.

FashionMnist stał się na tyle popularny, że jest zawarty w wielu bibliotekach służących do uczenia maszynowego zawiera jak TensorFlow lub Torch. Na stronie `http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/` można zobaczyć osiągi podstawowych algorytmów na powyższych zbiorach.

## 3 Proste klasyfikatory

Na początku chcieliśmy sprawdzić, jakie wyniki osiągają podstawowe klasyfikatory, w porównaniu do ich teoretycznych wyników na MNIST. Okazało się że MLP z jedną warstwą ukrytą (100 neuronów) osiąga dokładność równą jedynie 85%, a SVM z jądrem gaussowskim, oraz parametrem $C = 10$, osiąga wynik równy około 90%.

Kolejnym krokiem było sprawdzenie prostej sieci konwolucyjnej. Tym razem osiągnięte wyniki są dużo lepsze, bo na zbiorze testowym osiągnęliśmy wynik 91.45%.

Powyższe wyniki potwierdzają argumenty autorów bazy danych, mówiący o większym niż klasyczny MNIST skomplikowaniu zbioru. W tym przypadku porównywanie znacznie bardziej skomplikowanych modeli wydaje się mieć większy sens.

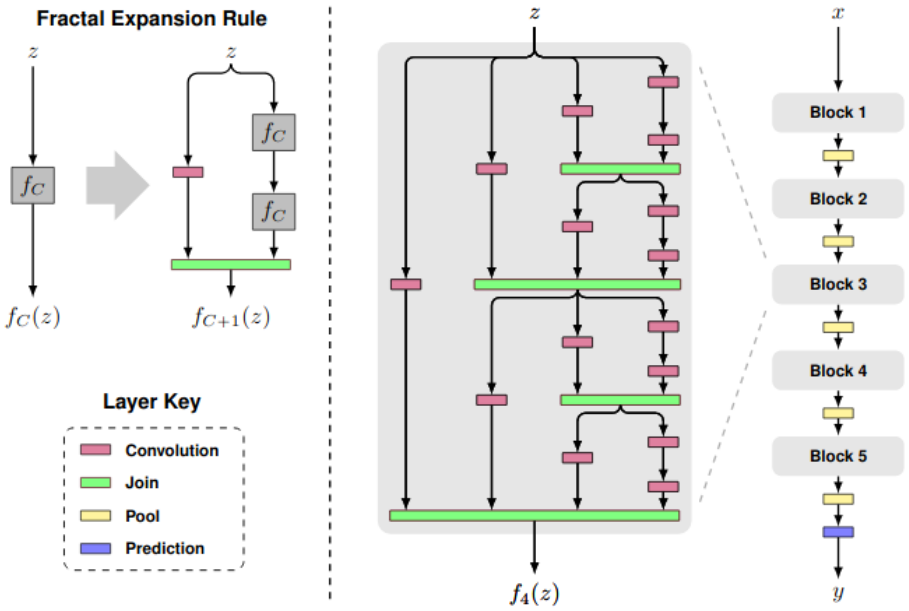



Figure 1: **FractalNet**: po lewej pokazana jest zasada ekspansji, według której tworzone są fraktale. Rozgałęzienia łączone są za pomocą funkcji "join", która w naszym przypadku liczy średnią wejść. Po prawej zaś mamy przedstawiony jeden z fraktali (uzyskany po 4 krokach lewej strony), oraz architekturę całej sieci.

# 4 Fractal Net

Fractal Net [4] jest głęboką siecią neuronową, której struktura oparta jest o powtarzające się kilkukrotnie te same małe elementy. Według autorów projekt sieci jest na tyle prosty, żeby przedstawiło go jedno słowo (fraktal), oraz ilustracja 1. Autorzy uważają, że dzięki takiemu zaprojektowaniu sieci, może ona w trakcie treningu imitować wygląd wielu innych architektur - poprzez aktywowanie i wygaszanie niektórych ścieżek.

Poza samą architekturą sieci, w pracy przedstawione jest narzędzie, nazwane "drop-path", które polega na wygaszaniu wybranych fragmentów modelu w trakcie treningu. Dzieli się ona na dwie części: lokalną i globalną, które są wykorzystywane naprzemiennie. W wersji lokalnej, funkcja "join" wyklucza z pewnym prawdopodobieństwem każde z wejść (podobnie jak dropout), ale przy założeniu, że co najmniej jedno z wejść nie zostanie odrzucone. Wersja globalna wybiera dokładnie jedną pionową ścieżkę z każdego z fraktali. Działanie drop-path przedstawione jest na ilustracji 2.

Powyżej opisana technika ma służyć

1. regularyzacji na podobnej zasadzie jak dropout, to znaczy, wykluczanie części sieci z treningu pozwala na ograniczenie przeuczania się sieci;

2. umożliwieniu równoległym ścieżkom we fraktalach, uczenia się niezależnie. Dzięki temu (przynajmniej teoretycznie) każda ze ścieżek prowadzących

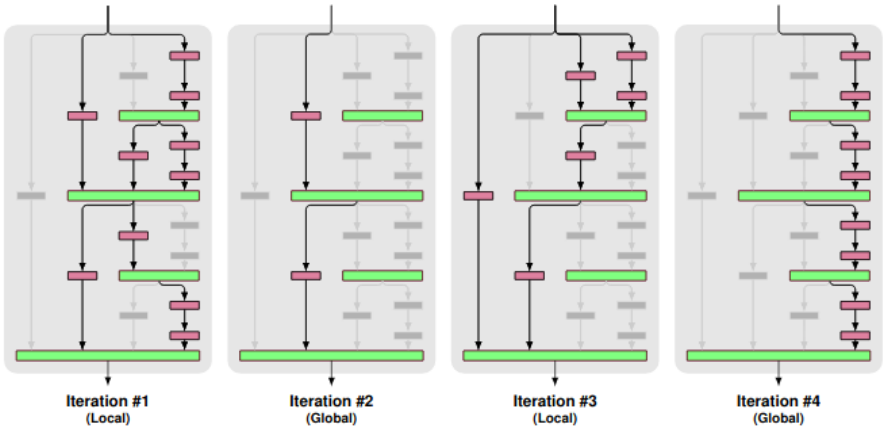



Figure 2: **Drop-path**

od wejścia do wyjścia, powinna być wytrenowana, tak jak cała sieć, oraz dawać rozsądne wyniki.

## 4.1 Rezultaty

Na samym początku, bez dodatkowej regularyzacji, sieć osiągała bardzo wysokie wyniki na zbiorze treningowym ( 95%), oraz średnie wyniki na zbiorze testowym ( 90%). Po dodaniu dropoutu, w kolejnych próbach osiągnięte wyniki oscylowały w okolicach 90-90.5%, co nadal nie było zadowalającym wynikiem.

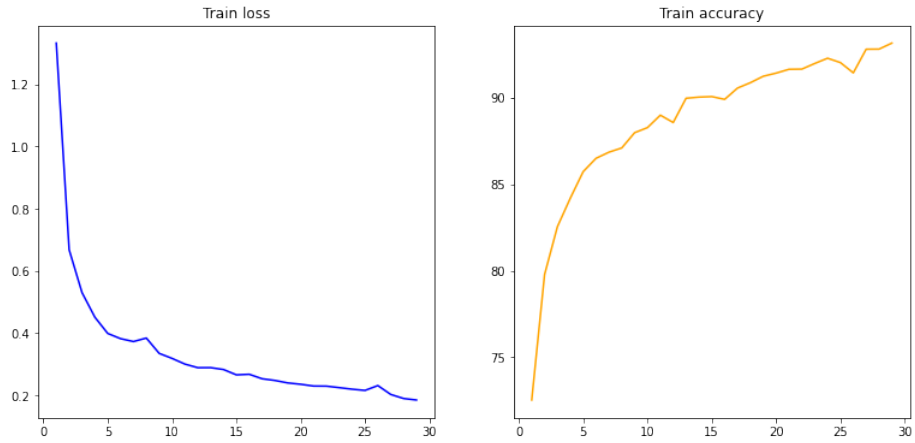



Figure 3: Krzywe uczenia dla najmocniejszego FractalNeta, którego udało się wytrenować

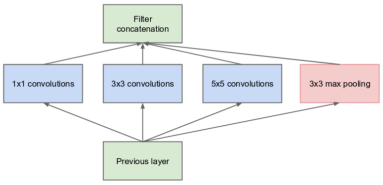
Największe polepszenie wyników przyniosła jedna operacja. Było nią zwiększenie liczby filtrów konwolucji z 8 do 16 w pierwszym fraktalu i jednoczesne zwiększenie dropoutu w ostatnim. Dzięki temu udało się po raz pierwszy uzyskać wynik większy niż 91%. Kolejne drobne poprawki pozwoliły uzyskać największy

wynik równy 91.36%. Krzywe uczenia dla tego modelu przedstawiają wykresy na Ilustracji 3.

# 5 IRCNN

Inception Recurrent Convolutional Neural Network [2] jest głęboką siecią neuronową wykorzystującą warstwy rekurencyjne. Inspiracją były badania z dziedziny neuronauki dotyczącej rozpoznawania obrazów przez płat skroniowy [3].

Sieć IRCNN wykorzystuje dwie znane już wcześniej idee - inception nets oraz rekurencyjnych sieci konwolucyjnych. Ideą sieci incepcyjnych jest stosowanie kerneli różnych wielkości na "tym samym poziomie", zamiast dokładania ich na końcu sieci co powoduje jej pogłębianie. Dzięki temu można uniknąć wielu wad zbyt głębokich sieci, m.in. dużej złożoności obliczeniowej lub tendencji do overfitting. W bloku incepcyjnym (Fig. 4) input jest przekazywany do kilku różnych kerneli. Ich output jest zestawiany, a następnie przekazywany do kolejnego modułu. Cała sieć składa się z kilku takich bloków połączonych warstwami przejściowymi.



(a) Inception module, naïve version

Figure 4: Inception blok

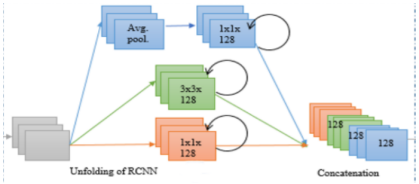



Figure 5: IRCNN blok

Blok IRCNN (Fig. 5) ma podobną architekturę do bloku incepcyjnego. Zamiast prostych warstw konwolucyjnych są rekurencyjne warstwy konwolucyjne - ich output jest tego samego wymiaru co input. Pozwala to na wielokrotne przekazanie outputu danej warstwy no niej samej jako input. Dzięki takiemu rozwiązaniu wagi cech mających większy wpływ są wzmocnione.

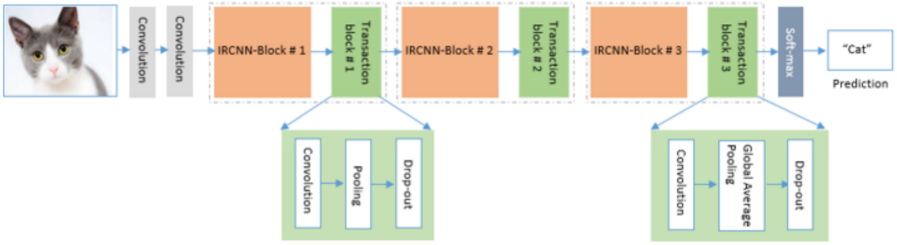



Figure 6: Sieć IRCNN

W sieci IRCNN (Fig. 6) bloki są połączone modułami przejściowymi które składają się z warstwy konwolucyjnej, poolingu oraz dropoutu.

## 5.1 Rezultaty

Różne warianty sieci osiągają na zbiorze testowym między 89% do 92%. Liczba bloków IRCNN zawsze wynosi 3, jak w cytowanej pracy. Wersje sieci różnią się wyborem algorytmów optymalizacyjnych oraz ich parametrów. Niestety we wszystkich wersjach występuje overfitting - na zbiorze treningowym wyniki dochodzą nawet ponad 99% podczas gdy na testowym zatrzymują się w okolicy 91%.

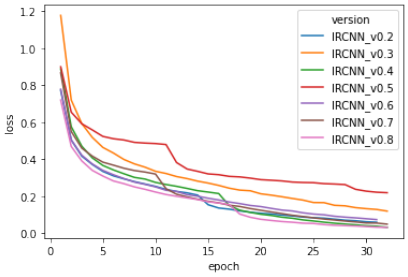



Figure 7: Loss

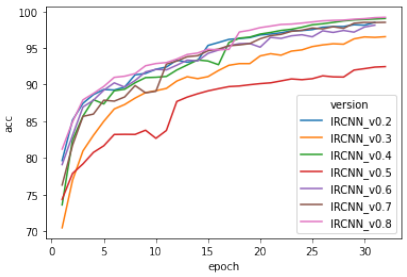



Figure 8: Dokładność

# References


[1] Zalando Research, FashionMnist
`https://github.com/zalandoresearch/fashion-mnist`

[2] Alom, M. Z., Hasan, M., Yakopcic, C., Taha, T. M. *Inception recurrent convolutional neural network for object recognition.* (2017) arXiv preprint arXiv:1704.07709.

[3] DiCarlo, James J., Davide Zoccolan, and Nicole C. Rust. *How does the brain solve visual object recognition?.* Neuron 73.3 (2012): 415-434.

[4] Larsson, Gustav and Maire, Michael and Shakhnarovich, Gregory *Fractalnet: Ultra-deep neural networks without residuals* (2016) arXiv preprint arXiv:1605.07648