Politechnika Śląska

Optymalizacja architektur sieci głębokiego uczenia dla klasyfikacji obrazów hiperspektralnych

mgr Kamil Książek

promotor:
dr hab. inż. Przemysław Głomb
promotor pomocniczy:
dr Krisztián Búza

Autoreferat i streszczenie rozprawy doktorskiej

Gliwice, październik 2022

Spis treści

1 Wstęp

W obrazowaniu hiperspektralnym (ang. *hyperspectral imaging*, HSI) czujniki rejestrują promieniowanie elektromagnetyczne dla różnych długości fali, gromadząc przy tym dziesiątki lub setki wartości dla pojedynczego piksela [5]. Daje to możliwość uzyskania informacji o stanie obiektu bądź przebiegu danego procesu. Obrazowanie hiperspektralne znajduje zastosowanie w klasyfikacji terenu, rolnictwie, monitorowaniu stanu środowiska czy wojskowości [26].

Sieci głębokiego uczenia są skutecznym narzędziem w odkrywaniu złożonych cech w wielowymiarowych danych. Okazały się one użyteczne w wielu obszarach nauki, zwłaszcza w przetwarzaniu obrazów [20], m.in. w ich klasyfikacji, czyli przypisywania jednej z kilku zdefiniowanych klas do danego obrazu [11] czy detekcji obiektów [2]. Znalazły one również zastosowanie w analizie, w tym klasyfikacji danych hiperspektralnych [23], a także w specyficznych dla tej dziedziny problemach, np. rozmieszaniu widm pikseli [22]. Powyższe tematy są przedmiotem niniejszej rozprawy. Postawiona została w niej następująca teza:

Teza *Optymalizacja architektur sieci głębokiego uczenia i metody reinicjalizacji wag zwiększają wydajność sieci w zastosowaniu do danych hiperspektralnych.*[1]

W badaniach opisanych w rozprawie przeanalizowane zostały różne architektury sieci neuronowych, włączając w to wielowarstwowe perceptrony (ang. *multilayer perceptrons*, MLPs), które są najbardziej podstawowymi sieciami jednokierunkowymi (ang. *feedforward neural networks*), autokodery (ang. *autoencoders*), sieci splotowe (ang. *convolutional neural networks*, CNNs) i rekurencyjne (ang. *recurrent neural networks*, RNNs). W Rozdziale 1, który jest wstępem do pracy, oprócz omówienia tezy rozprawy, przedstawione zostały architektury stosowane w dalszych badaniach, a także podstawowe pojęcia związane z obrazowaniem hiperspektralnym.

W Rozdziale 2 rozprawy przeprowadzona została optymalizacja sieci głębokiego uczenia na przykładzie zbioru danych do klasyfikacji plam krwi. Wykonany został szereg eksperymentów, wliczając w to scenariusze transduktywne i induktywne. Wnioski z powyższych badań doprowadziły do skupienia się na mieszaninach widm pikseli. Między innymi w wyniku tego, że rozlane substancje nie pokrywają w pełni materiałów tła, mogą tworzyć się jednorodne mieszaniny widm, np. krwi z tkaniną, na której została ona rozlana. Wówczas obserwowane widma substancji ulegają zaburzeniu, co utrudnia klasyfikację, zwłaszcza, gdy próbki testowe pochodzą z innego zbioru niż próbki treningowe. W takim przypadku, pomimo zawierania tych samych substancji, zmiana warunki akwizycji obrazu (np. oświetlenia) może negatywnie wpływać na klasyfikator.

Rozdział 3 poświęcony został badaniem liniowych autokoderów do rozmieszania spektralnego, czyli procesu polegającego na odzyskaniu oryginalnych widm, które są mieszaninami różnych substancji oraz proporcji, z jakimi one występują w mieszaninie. Przeprowadzone zostały eksperymenty z użyciem różnych architektur, zbiorów danych, funkcji straty (ang. *loss functions*), metod inicjalizacji wag w sieciach, itd.; zoptymalizowano również hiperparametry sieci. Istotnym elementem prac było badanie stabil-

[1] Oryginalnie postawiona w rozprawie teza brzmi następująco: *Optimization of deep learning network architectures and weight reinitialization methods improve the performance of neural networks for hyperspectral data.*

ności autokoderów. Analiza statystyczna z użyciem testu Kruskala-Wallisa [12] oraz testu post-hoc Conovera-Imana [6] wykazała, że inicjalizacja wag w sieciach neuronowych, tj. zestaw wag początkowych w danym modelu sieci, istotnie przekłada się na błąd rekonstrukcji pikseli po zakończeniu treningu sieci. Ponadto w jeszcze większym stopniu niż w poprzednim rozdziale zaobserwowano niestabilność sieci, tj. wybrane modele były niedotrenowane lub nawet zwracały zdegenerowane wyniki, pomimo stosowania jednakowych zestawów hiperparametrów. Pogłębiona analiza na wybranych modelach, w których stosowana była funkcja aktywacji ReLU, doprowadziła do identyfikacji zjawiska martwych aktywacji i neuronów (ang. *dead activations / neurons*). Polega ono na zaniku istotności wybranych neuronów poprzez zerowy wkład części wag sieci na dalsze obliczenia. Może to utrudnić przepływ gradientu w trakcie treningu sieci, a w skrajnych przypadkach uniemożliwić ich uczenie. Próba rozwiązania tego problemu została zaprezentowana w kolejnym rozdziale.

Rozdział 4 zaczyna się od analizy statystycznej z użyciem współczynnika korelacji rang Spearmana, który miał zbadać zależność między liczbą martwych aktywacji (lub neuronów) w kolejnych warstwach autokodera a końcowym błędem rekonstrukcji pikseli. Uzyskane rezultaty potwierdziły znaczenie martwych aktywacji na końcowy wynik badanej architektury, zwłaszcza w przypadku ostatniej warstwy kodera. Zaprezentowane zostały trzy metody reinicjalizacji wag sieci, które miały zminimalizować negatywny skutek martwych aktywacji na wydajność sieci. W każdej z nich badana była proporcja martwych aktywacji w stosunku do niezerowych aktywacji w danych warstwach lub dla danych neuronów. Zaproponowane rozwiązania to: *metoda pełnej reinicjalizacji sieci*, *metoda reinicjalizacji jednej warstwy sieci* oraz *metoda częściowej reinicjalizacji* (wybranych wag jednej warstwy). Przeprowadzone eksperymenty potwierdziły skuteczność tych rozwiązań w danych hiperspektralnych na przykładzie modeli wytrenowanych w eksperymentach z Rozdziału 3. Dodatkowo zbadano działanie metod reinicjalizacji na zbiorze danych MNIST [15] dla większych architektur oraz różnych zestawów hiperparametrów.

Rozdział 5 stanowi podsumowanie rozprawy i omawia wybrane kierunki przyszłych prac. Punktem wyjścia do badań prezentowanych w rozprawie była optymalizacja architektur sieci głębokiego uczenia w klasyfikacji obrazów hiperspektralnych, na przykładzie zbioru danych z plamami krwi i innych substancji wizualnie do niej podobnych. W trakcie prac zidentyfikowany został problem mieszanin widm, które utrudniały klasyfikację w scenariuszu induktywnym, a także problem niestabilności jednej z sieci splotowych. W dalszej części badań zastosowano autokodery do rozmieszania spektralnego, a także analizowano ich stabilność, w sensie wpływu inicjalizacji wag na błąd rekonstrukcji po zakończeniu treningu sieci. Analiza wybranych modeli autokoderów doprowadziła do wykrycia zjawiska martwych aktywacji, dlatego w kolejnym etapie prac zaproponowano trzy metody reinicjalizacji sieci, które miały złagodzić negatywny charakter tego zjawiska. W kolejnych częściach niniejszego streszczenia zostaną pokrótce zaprezentowane wyniki poszczególnych rozdziałów rozprawy.

1.1 Cele rozprawy

Postawione zostały następujące cele badawcze w celu zweryfikowania tezy sformułowanej w Rozdziale 1:

1. Zestawienie nowoczesnych architektur sieci głębokiego uczenia do klasyfikacji

obrazów hiperspektralnych.

2. Przygotowanie środowiska testowego do optymalizacji sieci głębokiego uczenia na przykładzie zbioru danych hiperspektralnych do klasyfikacji plam krwi.

3. Przeprowadzenie optymalizacji sieci głębokiego uczenia w scenariuszu transduktywnym oraz induktywnym.

4. Przygotowanie architektury autokodera do rozmieszania spektralnego pikseli i badanie jej stabilności.

5. Identyfikacja problemu obniżonej wydajności części modeli autokoderów.

6. Zaproponowanie i zbadanie wydajności metod reinicjalizacji sieci mających złagodzić problem martwych aktywacji.

1.2 Lista publikacji wykorzystanych w rozprawie

Części niniejszej rozprawy są oparte na / zawierają fragmenty następujących publikacji. Rozdział 2 jest oparty na [14], zaś Rozdziały 3 i 4 są oparte na [13] i rozszerzają opisane tam badania.

1. **K. Książek**, P. Głomb, M. Romaszewski, M. Cholewa, B. Grabowski, K. Buza, *Improving Autoencoder Training Performance for Hyperspectral Unmixing with Network Reinitialisation*, ICIAP 2022, w: S. Sclaroff, C. Distante, M. Leo, G.M. Farinella, F. Tombari, Image Analysis and Processing – ICIAP 2022, Lecture Notes in Computer Science, 2022, Vol. 13231, Springer, Cham, ss. 391-403, ISBN: 978-3-031-06426-5, DOI: 10.1007/978-3-031-06427-2_33.

   Mój wkład w [13]: Pomysł zastosowania autokoderów do rozmieszania spektralnego, pomysł i projekt metod reinicjalizacji sieci, przeprowadzenie eksperymentów, współtworzenie kodu źródłowego, wybór i przeprowadzenie testów statystycznych oraz analizy ich wyników, przygotowanie i weryfikacja wyników eksperymentów, udział w dyskusjach, pisanie tekstu artykułu, stworzenie Rysunków 1–2.

2. **K. Książek**, M. Romaszewski, P. Głomb, B. Grabowski, M. Cholewa, *Blood Stain Classification with Hyperspectral Imaging and Deep Neural Networks*, Sensors, 2020, Vol. 20, Issue 22, No. 6666, ss. 1-24, ISSN 1424-8220, DOI: 10.3390/s20226666.

   Mój wkład w [14]: Przygotowanie i przeprowadzenie eksperymentów z użyciem sieci głębokiego uczenia, przygotowanie Rysunków: 1–7, 9, 12–13, współtworzenie kodu źródłowego, opis wybranych architektur sieci głębokiego uczenia, udział w dyskusjach, pisanie tekstu artykułu.

2 Klasyfikacja plam krwi na podstawie obrazów hiperspektralnych z użyciem sieci głębokiego uczenia

2.1 Metodyka

W niniejszym rozdziale została przeprowadzona optymalizacja architektur sieci głębokiego uczenia w zadaniu klasyfikacji obrazów hiperspektralnych. Zaprezentowano w nim wyniki opublikowane w pracy *Blood Stain Classification with Hyperspectral Imaging and Deep Neural Networks* [14]. Jako środowisko testowe do badań wybrany został scenariusz klasyfikacji plam krwi w oparciu o zbiór danych przygotowany w IITiS PAN [25].

Wśród warstw, które tworzą współczesne architektury sieci neuronowych, można wyróżnić zarówno warstwy liniowe, będące głównym składnikiem wielowarstwowych perceptronów oraz bardziej złożone warstwy splotowe i uśredniające, tworzone przez jedno-, dwu- lub trójwymiarowe filtry, będące składnikami sieci splotowych [1, 7]. Oprócz tego można wymienić m.in. warstwy porzucania (ang. *dropout*) czy normalizacji wsadowej (ang. *batch normalization*). Inną ważną gałęzią sieci neuronowych są sieci rekurencyjne, wsród których istotną rolę odgrywają komórki LSTM (ang. *long short-term memory*) oraz GRU (ang. *gated recurrent unit*).

W prezentowanych badaniach zostało zestawionych 6 zaawansowanych architektur: wielowarstwowy perceptron, jedno-, dwu- i trójwymiarowe sieci splotowe, a także sieć rekurencyjna oparta na komórkach GRU. Wybór ten pozwala na porównanie różnorodnych podejść do klasyfikacji obrazów hiperspektralnych, z których część uwzględnia jedynie zależności widmowe (jak sieć MLP), a część zarówno widmowe jak i przestrzenne (jak dwu- i trójwymiarowe sieci splotowe). Oprócz tego, powyższe architektury zostały zestawione z maszynami wektorów wspierających (ang. *support vector machines*, SVMs). Na potrzeby badań wykorzystano i zmodyfikowano bibliotekę DeepHyperX [3].

Zbiór danych, który został użyty w badaniach, zawiera szereg obrazów hiperspektralnych mających odzwierciedlać wyzwania związane z analizą scen kryminalistycznych. Są w nim obecne różnych rozmiarów plamy krwi oraz substancji wizualnie do niej podobnych: sztuczna krew, farba akrylowa, plakatowa, keczup, koncentrat pomidorowy i sok z buraka. Substancje te zostały rozlane na różnych materiałach, zarówno w odniesieniu do ich koloru jak i rodzaju (np. tkaniny, deski, itd.). Dodatkowo, te same sceny były poddawane akwizycji w różnych odstępach czasowych od rozlania substancji (po kilku godzinach lub dniach). Dzięki temu uwzględniono zmianę charakterystyki widm substancji w zależności od stopnia ich rozkładu. Zbiór danych składa się z dwóch podstawowych grup obrazów. Pierwszą z nich tworzą sceny ramkowe (ang. *frame scenes*) zawierające duże plamy substancji rozlane na białej tkaninie. Mają one odpowiadać scenom przygotowanym w warunkach laboratoryjnych. Druga grupa zawiera trudniejsze w analizie obrazy z mniejszymi plamami substancji umieszczonych na ciemnych materiałach (ang. *comparison scenes*), co ma symulować rzeczywiste miejsca zbrodni.

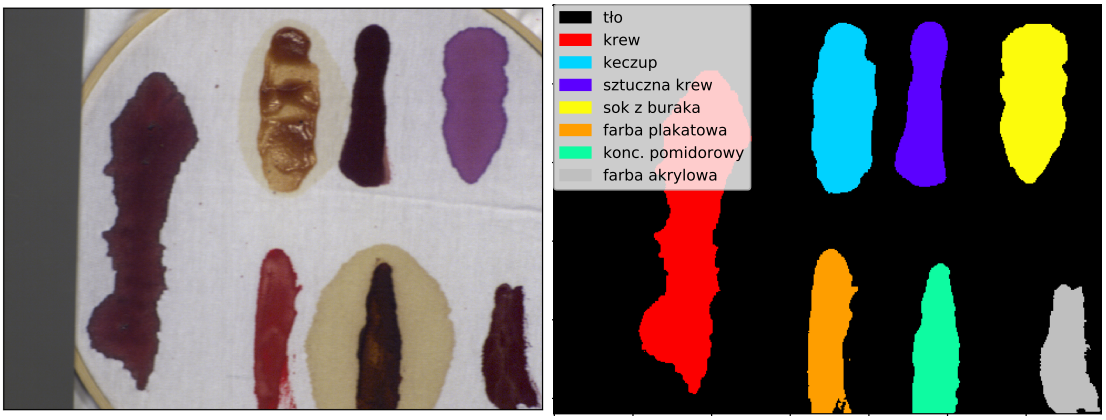
Eksperymenty zostały wykonane w dwóch scenariuszach: transduktywnym (ang. *Hyperspectral Transductive Classification*, HTC) oraz induktywnym (ang. *Hyperspectral Inductive Classification*, HIC). Pierwsze z nich jest typowe w klasyfikacji pikseli w obrazach hiperspektralnych [23], tzn. próbki w zbiorze treningowym i testowym

pochodzą z tego samego obrazu. W scenariuszu induktywnym zbiór testowy pochodzi z innego obrazu (lub obrazów), ale zawiera obiekty tych samych klas. Z powodu potencjalnych różnic w oświetleniu oraz innych materiałach występujących w tle, widma substancji w zbiorze testowym mogą różnić się od widm próbek w zbiorze treningowym. To podejście, choć trudniejsze z punktu widzenia algorytmów uczenia maszynowego, bardziej odpowiada badaniu miejsca zdarzenia. Wówczas klasyfikator mógłby być trenowany w oparciu o makietę przygotowaną w laboratorium, zaś test odbywałby się w rzeczywistym miejscu przestępstwa.

2.2 Eksperymenty

We wszystkich eksperymentach dokonano takiego podziału danych, by zapobiec potencjalnym wyciekom informacji ze zbioru testowego (co jest możliwe w scenariuszu transduktywnym). Procedura ta została szczegółowo opisana w rozprawie.

Do eksperymentów wyznaczono 9 obrazów hiperspektralnych: 6 obrazów ramkowych oraz 3 sceny będące makietami miejsc zbrodni. Obrazy ramkowe przyjmują oznaczenia *F(·)*, zaś 3 pozostałe sceny *E(·)*. Cyfra w nawiasie związana jest z dniem, w którym została wykonana akwizycja obrazu. Oprócz tego *F(1a)* jest sceną, której akwizycja odbyła się 7 godzin po *F(1)*, zaś obraz *F(2k)* powstał przy użyciu innej kamery hiperspektralnej niż wszystkie pozostałe sceny ze zbioru danych. Przykładowy obraz z analizowanego zbioru danych wraz z anotacją klas zaprezentowany jest na Rysunku 1.



(a) Obraz RGB sceny *F(1)*. (b) Etykiety klas dla sceny *F(1)*.

Rysunek 1: Obraz RGB z odpowiadającymi etykietami klas dla sceny ramkowej (ang. *frame scene*) z pierwszego dnia od momentu rozlania substancji.

Oprócz eksperymentów transduktywnych z użyciem 9 scen, przygotowano również 8 eksperymentów induktywnych, które można podzielić na 3 grupy. W pierwszej z nich w zbiorze treningowym znajdują się piksele obrazów ramkowych *F(·)*, a w zbiorze testowym piksele obrazów symulujących sceny zbrodni *E(·)* z tego samego dnia, licząc od momentu rozlania substancji. Druga grupa zawiera obrazy pochodzące z różnych dni, np. scenariusz *F(2)* $\rightarrow$ *E(7)*, gdzie piksele z obrazu *F(2)* znajdują się w zbiorze treningowym, a piksele z *E(7)* w zbiorze testowym. Trzecia grupa zawiera te same sceny, ale wykonane przy użyciu różnych kamer hiperspektralnych, np. *F(2)* $\rightarrow$ *F(2k)*.

Dla każdego obrazu przygotowanych zostało 10 zestawów treningowych. Prezentowane dalej wyniki są uśrednieniem 10 uruchomień metody dla poszczególnych scenariuszy. Jako metryki do oceny wydajności sieci wybrano dokładność klasyfikacji (ang. *overall accuracy*, OA), średnią dokładność klasyfikacji (ang. *average accuracy*, AA) oraz współczynnik κ Cohena.

2.3 Wyniki

Wyniki eksperymentów w scenariuszu transduktywnym zawarte są w Tabeli 3, zaś w scenariuszu induktywnym prezentowane są w Tabeli 4. Obydwie Tabele umieszczone znajdują się w załączniku.

Wydajność wszystkich testowanych metod była wysoka w scenariuszach transduktywnych. Dla każdego zbioru danych dokładność klasyfikacji najlepszej z metod przekraczała 90%. Okazało się, że obrazy ramkowe były łatwiejsze w klasyfikacji niż sceny będące makietami miejsc zbrodni. W przypadku obrazów ramkowych dokładność klasyfikacji dla niemal wszystkich metod sięgała 100%. W 4 spośród 6 z nich najbardziej skuteczne okazały się trójwymiarowe sieci splotowe [4, 17]. W pozostałych sytuacjach zwyciężała maszyna wektorów wspierających, zaś nieznacznie od niej gorsza była sieć MLP. Dla makiet miejsc zbrodni dokładność najlepszych metod wynosiła pomiędzy 90.6% a 94.5%. W przypadku scenariuszów induktywnych, wynik ten wahał się między 57.2% a 99.5%, co w większości przypadków stanowi istotny spadek wydajności. Czterokrotnie najbardziej skuteczna w eksperymentach HIC była sieć rekurencyjna [21]. Podczas gdy w scenariuszach HTC wszystkie metody osiągnęły porównywalne rezultaty, w scenariuszach HIC, za wyjątkiem tych z użyciem różnych kamer, bardziej złożone metody (np. sieć rekurencyjna lub trójwymiarowa sieć splotowa) osiągnęły wyższą dokładność klasyfikacji. Zmiana kamery nie okazała się być dużym wyzwaniem dla większości stosowanych metod. Dokładność klasyfikacji zwycięskiej w tym przypadku maszyny wektorów wspierających przekraczała 98.2%. Wyniki potwierdzają, że optymalizacja architektur sieci głebokiego uczenia zwiększa ich wydajność, co potwierdza pierwszą część tezy rozprawy. Wybór odpowiedniej architektury istotnie przekłada się na rezultaty, np. w eksperymencie *F(1)→E(1)* dokładność klasyfikacji wahała się między 46.4% dla najsłabszej architektury 1D CNN [10] aż do 66.8% dla najbardziej wydajnej sieci RNN [21], co stanowi różnicę przekraczającą 20%.

2.4 Wnioski

W rozdziale przeprowadzona została optymalizacja sieci głębokiego uczenia do klasyfikacji obrazów hiperspektralnych na przykładzie problemu klasyfikacji plam krwi i innych substancji wizualnie do niej podobnych. Badania te nawiązywały do wyników zawartych w pracy [3], w której autorzy stosowali omawiane tutaj architektury w klasycznych zbiorach hiperspektralnych wykorzystywanych w teledetekcji. Na podstawie opisanych wyników można wywnioskować, że nie zawsze stosowanie skomplikowanych architektur jest opłacalne. W niektórych sytuacjach wielowarstwowy perceptron dorównywał, a nawet przewyższał skutecznością bardziej złożone metody, np. sieci splotowe. Wprawdzie w scenariuszu induktywnym sieć rekurencyjna czterokrotnie uzyskiwała najwyższą skuteczność, to w scenariuszu transduktywnym miała dość niską dokładność klasyfikacji na tle innych metod w przypadku scen *E(·)*. Co wię-

cej, można stwierdzić, że istotnym czynnikiem utrudniającym klasyfikację w scenariuszu induktywnym było zjawisko mieszanin widm pikseli. Różnice między widmami próbek treningowych i testowych doprowadziły do zmniejszenia skuteczności klasyfikatorów. Oprócz tego, dostrzeżone zostały problemy ze stabilnością jednej z sieci splotowych [17], która w niektórych przypadkach dawała znacząco niższe wyniki niż w innych, przy tym samym zestawie hiperparametrów. Wnioski nt. mieszanin i stabilności sieci doprowadziły do zastosowania autokoderów do rozmieszania spektralnego, czyli do procesu odzyskiwania oryginalnych widm na podstawie ich mieszanin. Zbadano również stabilność tych sieci w znaczeniu wpływu inicjalizacji wag na błąd rekonstrukcji po zakończeniu treningu.

3 Badanie metod inicjalizacji wag dla autokoderów

3.1 Metodyka

W wyniku badań, które zostały opisane w poprzednim rozdziale, zauważono, że dokładność klasyfikacji w przypadku scenariuszy induktywnych była znacząco niższa niż dla scenariuszy transduktywnych. Jednym z powodów tego stanu było powstawanie mieszanin pikseli związane m.in. z różnym tłem na badanych scenach. Dlatego zdecydowano się na badanie problemu rozmieszania spektralnego, które polega na uzyskaniu oryginalnych składowych (ang. *endmembers*) oraz współczynników rozmieszania / nasyceń, z jakimi są one obecne w danym pikselu (ang. *fractional abundances*). Ponieważ autokodery są wydajnym narzędziem w zadaniu rozmieszania spektralnego (np. [22, 27]), zostały wybrane do dalszych eksperymentów. W Rozdziale zostały opisane wyniki, na podstawie których opublikowano artykuł *Improving Autoencoder Training Performance for Hyperspectral Unmixing with Network Reinitialisation* [13] opublikowany podczas 21. edycji konferencji ICIAP: *International Conference on Image Analysis and Processing*, która odbyła się w maju 2022 roku.

Dodatkowo, w trakcie prac zidentyfikowany został problem niestabilności sieci polegający na dużej wariancji błędów rekonstrukcji danych wejściowych. Oznacza to, że przy jednakowym zbiorze hiperparametrów sieci, część wytrenowanych modeli osiąga gorszą wydajność niż inne. To odkrycie doprowadziło badań nad wpływem inicjalizacji wag sieci na błąd rekonstrukcji danych wejściowych. Rozważone zostały dwie powszechnie stosowane metody inicjalizacji wag: metoda He [9] oraz Glorota [8]. Przeprowadzony został szereg eksperymentów dla różnych architektur, funkcji straty, zbiorów danych czy metod inicjalizacji wag, w zadaniu rozmieszania spektralnego. Wykonana została analiza statystyczna mająca zbadać istotność zależności między wagami początkowymi sieci a jej końcowym błędem rekonstrukcji.

W pracy wykorzystany został model liniowych mieszanin pikseli. W tym modelu hiperspektralny piksel $\mathbf{x} = [x_1, x_2, ..., x_B]^\top$, mający B pasm spektralnych oraz S składowych, można zapisać następująco:

$$\mathbf{x} = \sum_{i=1}^{S} a_i \cdot \mathbf{e_i} + \mathbf{w}, \tag{1}$$

gdzie wektor $\mathbf{e_i} = [e_1, e_2, ..., e_B]^\top$ wyraża i–tą składową, $i \in \{1, 2, ..., S\}$, zaś wartość a_i jest i–tym współczynnikiem nasycenia, $\mathbf{a} = [a_1, a_2, ..., a_S]^\top$, podczas

gdy $\mathbf{w} = [w_1, w_2, ..., w_B]^\top$ jest wektorem szumu. Współczynniki nasycenia muszą jednak spełniać dwa warunki: każdy z nich musi być niezerowy, zaś ich suma dla każdego piksela ma wynosić 1.

Podczas eksperymentów stosowano dwie architektury autokoderów. Autokodery zawierają część kodującą (ang. *encoder*) oraz dekodującą (ang. *decoder*). Ich konstrukcja sprzyja wyrażeniu kolejnych pikseli obrazu jako iloczynu współczynników rozmieszania oraz widm składowych. Autokodery uczą się wewnętrznej reprezentacji danych i przechowują ją w przestrzeni ukrytej (ang. *latent space*). W prezentowanym w rozprawie przypadku, ostatnia warstwa kodera ma tyle neuronów, ile jest składowych w danym obrazie. Wartości neuronów tej warstwy reprezentują współczynniki rozmieszania, zaś wagi jednowarstwowego dekodera są widmami składowych. W ten sposób wyjście autokodera, czyli rekonstrukcja danych wejściowych, jest iloczynem neuronów ostatniej warstwy kodera oraz wag dekodera. Funkcja straty L ocenia podobieństwo między wejściem i wyjściem z sieci.

Do dalszych prac wybrane zostały dwie architektury: tzw. architektura oryginalna (ang. *original*) oraz uproszczona (ang. *basic*). Architektura oryginalna jest najbardziej wydajną siecią opisaną w publikacji [22]. Jej koder składa się z 4 warstw, warstwy normalizacji wsadowej (ang. *batch normalization*), warstwy miękkiego progowania (ang. *soft thresholding*), porzucania gaussowskiego (ang. Gaussian Dropout) oraz normalizacji do jedynki (ang. *sum-to-one constraint*). Stosowana jest również sigmoidalna funkcja aktywacji. Została również przygotowana uproszczona wersja tej architektury, w której koder składa się z dwóch warstw, a funkcją aktywacji jest ReLU.

3.2 Eksperymenty

Głównym celem eksperymentów była weryfikacja tego, czy istnieje statystycznie istotny wpływ inicjalizacji wag modeli sieci na ich końcową wydajność mierzoną błędem rekonstrukcji danych wejściowych. Przygotowanych zostało 10 schematów eksperymentów, uwzględniając różne metody inicjalizacji wag (He [9] oraz Glorota [8] z rozkładem normalnym i jednostajnym), architektury (oryginalna i uproszczona), funkcje straty (błąd średniokwadratowy, czyli MSE oraz odległość kątowa / spektralna, ang. *Spectral Angle Distance*, SAD), 2 zbiory danych: Samson oraz Jasper Ridge, mający kolejno 3 oraz 4 składowe, a także dwie metody doboru hiperparametrów. Dla 7 scenariuszy eksperymentów zostały one zoptymalizowane przy użyciu narzędzia Ray Tune [18], dla 3 pozostałych z architekturą oryginalną pochodzą bezpośrednio z artykułu [22]. Pełne dane nt. eksperymentów opisane są w Tabeli 1. Dla każdego spośród 10 schematów przygotowano po 50 zestawów wag początkowych (modeli) dla każdego z 4 sposobów inicjalizacji wag. Każdy model był osobno trenowany 50 razy. Łącznie przeprowadzono 100000 indywidualnych sesji treningowych sieci.

Wyznaczone zostały 3 miary jakości rozmieszania spektralnego. Po pierwsze, liczona jest średnia kwadratowa błędów (ang. *Root Mean Square Error*, RMSE) pomiędzy danymi wejściowymi, czyli pikselami obrazu, a rekonstrukcją stworzoną przez autokoder. RMSE jest również stosowany do błędu współczynników rozmieszania pomiędzy rzeczywistymi wartosciami pochodzącymi z etykiet dla zbiorów a tymi wyznaczonymi przez autokoder. Oprócz tego, liczona jest odległość spektralna między rzeczywistymi widmami nasyceń a widmami proponowanymi przez model autokodera.

W celu zweryfikowania hipotezy mówiącej o tym, że inicjalizacja wag istotnie prze-

Tabela 1: Spis wszystkich eksperymentów wraz z hiperparametrami. Kolumna „koder" związana jest z architekturą podstawową i zawiera informacje nt. liczby neuronów w pierwszej warstwie enkodera. L oznacza funkcję straty, S liczbę składowych w danym zbiorze danych, b_s to rozmiar minipaczki (ang. *batch size*), a GD jest skrótem od metody porzucania gaussowskiego.

lp.	architektura	hiperparametry	L	zbiór	koder	b_s	wsp. uczenia	GD
1	oryginalna	Ray Tune	MSE	Samson	–	100	0.01	0.
2	oryginalna	Ray Tune	SAD	Samson	–	100	0.01	0.
3	oryginalna	artykuł [22]	SAD	Samson	–	20	0.01	0.1
4	podstawowa	Ray Tune	MSE	Samson	$10S$	4	0.0001	–
5	podstawowa	Ray Tune	SAD	Samson	$20S$	4	0.0001	–
6	oryginalna	Ray Tune	MSE	Jasper Ridge	–	100	0.01	0.
7	oryginalna	Ray Tune	SAD	Jasper Ridge	–	100	0.01	0.
8	oryginalna	artykuł [22]	MSE	Jasper Ridge	–	5	0.01	0.1
9	oryginalna	artykuł [22]	SAD	Jasper Ridge	–	5	0.01	0.1
10	podstawowa	Ray Tune	MSE	Jasper Ridge	$10S$	20	0.001	–

kłada się na finałowy błąd rekonstrukcji modelu po zakończeniu treningu, przeprowadzona została analiza statystyczna dla serii danych. Nie można jednak było użyć ANOVY z powodu niespełnienia założeń o równości wariancji we wszystkich populacjach modeli, zgodnie z wynikami testu Levene'a. Zamiast tego wybrany został H-test Kruskala-Wallisa [12] dla wielu niezależnych próbek. Miał on za zadanie sprawdzić, czy wartość oczekiwana przynajmniej jednej serii danych (czyli populacji modeli) jest znacząco różna od wartości oczekiwanej co najmniej jednej innej serii danych. Jeśli tak jest, to wciąż nie wiadomo, które serie danych istotnie się różnią. W tym celu wybrano test post-hoc Conovera-Imana [6] do wykonania porównań par.

3.3 Wyniki

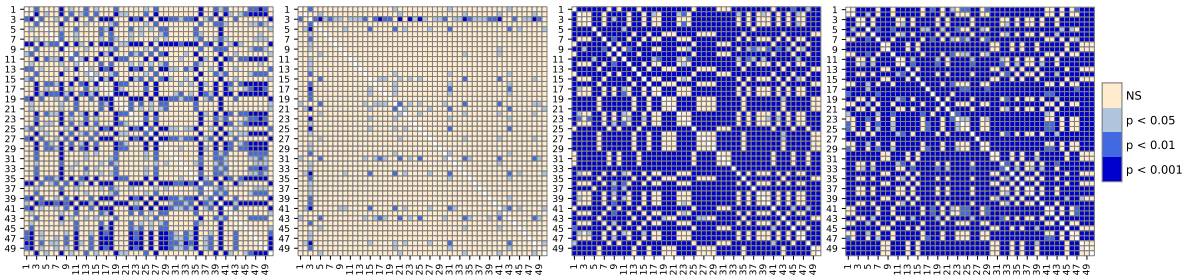
Wyniki testu Kruskala-Wallisa wykazały, że dla 38 spośród 40 przypadków odnotowano statystycznie istotne różnice pomiędzy populacjami modeli. Na tej podstawie można wywnioskować, że inicjalizacja wag w prezentowanych autokoderach istotnie przekłada się na ich błąd rekonstrukcji po zakończeniu procesu uczenia. Dla przypadków, w których test Kruskala-Wallisa potwierdził statystycznie istotne różnice, wykonany został test post-hoc Conovera-Imana. Wybrane wyniki tego testu zaprezentowane są na Rysunku 2. *NS* oznacza, że różnica między populacjami nie jest statystycznie istotna, a odcienie niebieskiego wskazują na to, jak niska była p-wartość dla wybranej pary populacji modeli. Na podstawie tych danych można zauważyć, że w Eksperymencie 1, zwłaszcza dla inicjalizacji Glorota z rozkładem jednostajnym, jedynie wybrane populacje modeli istotnie się różniły, podczas gdy w Eksperymencie 4 różnice te były znacznie częstsze.

3.4 Wnioski

W powyższym rozdziale wykazana została zależność pomiędzy inicjalizacją wag a błędem rekonstrukcji na koniec procesu uczenia w przypadku autokoderów dla rozmieszania spektralnego. Przeprowadzony został szereg eksperymentów dla różnych kon-

Tabela 2: Wyniki eksperymentów opisanych w Tabeli 1. H-test oznacza statystykę testu Kruskala-Wallisa, zaś log p-val jest wartością logarytmu z p-wartości. Test został wykonany na poziomie istotności 0.05. Przypadki, w których p-wartość była większa niż 0.05, zostały pogrubione. Bardzo niskie p-wartości zostały zastąpione przez „− inf". *ph* oznacza stosunek liczby statystycznie istotnie różnych par do łącznej liczby porównywanych par. KHN oznacza metodę He [9] z rozkładem normalnym, a KHU z rozkładem jednostajnym, zaś XGN i XGU są metodą Glorota [8], odpowiednio z rozkładem normalnym i jednostajnym.

init.	KHN			KHU			XGN			XGU		
lp.	H-test	log p-val	*ph*	H-test	log p-val	*ph*	H-test	log p-val	*ph*	H-test	log p-val	*ph*
1	200.9	−45.07	0.33	243.2	−61.76	0.42	78.8	−5.41	0.12	70.4	−3.72	0.10
2	891.2	−355.38	0.70	443.2	−147.76	0.56	625.9	−231.04	0.64	660.3	−246.96	0.66
3	763.8	−295.33	0.67	267.8	−71.82	0.41	239.1	−60.11	0.39	180.9	−37.49	0.33
4	2185.8	− inf	0.88	2025.3	− inf	0.72	1997.6	− inf	0.75	2141.5	− inf	0.76
5	1954.3	− inf	0.88	2093.4	− inf	0.90	1840.8	− inf	0.87	1777.2	− inf	0.85
6	134.0	−20.95	0.24	75.8	−4.79	0.11	76.3	−4.90	0.12	98.8	−10.32	0.16
7	903.8	−361.36	0.71	761.1	−294.07	0.67	953.8	−385.10	0.70	871.0	−345.85	0.69
8	77.3	−5.09	0.12	75.4	−4.71	0.11	78.4	−5.33	0.12	93.3	−8.88	0.16
9	69.2	−3.50	0.10	**66.2**	**−2.98**	–	74.2	−4.46	0.11	**47.9**	**−0.65**	–
10	1767.3	− inf	0.85	2041.9	− inf	0.82	1344.6	−572.45	0.70	1155.1	−481.29	0.73



(a) KHU, Exp. 1 (b) XGU, Exp. 1 (c) KHU, Exp. 4 (d) XGU, Exp. 4

Rysunek 2: Wyniki testu post-hoc Conovera-Imana dla wybranych scenariuszy eksperymentu. KHU oznacza inicjalizację metodą He [9] z rozkładem jednostajnym, zaś XGU metodą Glorota [8] z rozkładem jednostajnym.

figuracji sieci i zbiorów danych. Jedynie w 2 spośród 40 scenariuszy testowych nie odnotowano statystycznie istotnej różnicy pomiędzy populacjami modeli. Pogłębiona analiza wyników wykazała, że dla części modeli architektury podstawowej, które osiągały duże błędy rekonstrukcji, już na początku procesu uczenia występowało zjawisko zanikającego gradientu. Oznaczało to, że taka sieć miała ograniczone zdolności do aktualizacji swoich wag, a w efekcie miala gorsze wyniki od innych modeli. Identyfikacja tego zjawiska stała się przyczynkiem do dalszych badań, które miały ograniczyć jego negatywny wpływ na trening. Ich owocem stały się metody reinicjalizacji sieci i zostały one zaprezentowane w kolejnym rozdziale rozprawy.

4 Metody reinicjalizacji sieci dla poprawy wydajności autokoderów.

4.1 Metodyka

W poprzednim rozdziale wykazano statystycznie wpływ inicjalizacji wag modelu na końcowy wynik błędu rekonstrukcji autokoderów do rozmieszania spektralnego. Zidentyfikowany również został problem zanikających gradientów w wybranych modelach o niskiej wydajności z użyciem architektury podstawowej. Wykonano również pogłębioną analizę tych modeli. Okazało się, że przyczyną zaniku gradientów były zjawiska martwych aktywacji (ang. *dead activations*) i martwych neuronów (ang. *dead neurons*). Są one nierozerwalnie związane z funkcją aktywacji ReLU, która była stosowana w architekturze podstawowej [19, 24]. O martwej aktywacji mówimy wtedy, gdy podczas procesu propagacji w przód (ang. *feedforward propagation*) dla danego zestawu danych wejściowych, wybrany neuron zwraca wartość 0 po nałożeniu funkcji aktywacji. To sprawia, że wagi wychodzące z tego neuronu mają zerowy wkład w dalsze obliczenia, czyniąc je nieistotnymi. Z kolei neuron jest martwy, gdy zwraca 0 dla każdego zestawu danych wejściowych. Należało jednak zbadać w sposób systematyczny wpływ tych zjawisk na wydajność sieci, by móc dzięki temu znaleźć rozwiązanie problemu z treningiem modeli. W tym celu wybrano wszystkie eksperymenty z użyciem architektury podstawowej, tj. 12 spośród 40 scenariuszy i posłużono się współczynnikiem korelacji rang Spearmana. Postawiono kilka hipotez, które zostały szczegółowo omówione w rozprawie. Zbadano wpływ martwych aktywacji i martwych neuronów w różnych warstwach autokoderów na błąd rekonstrukcji. Pod wpływem tych wyników zaproponowane zostały trzy metody reinicjalizacji sieci, który mają za zadanie zapobiec nieudanemu treningowi modelu. Ich wydajność została zweryfikowana dla wybranych modeli wytrenowanych w eksperymentach z Rozdziału 3, a także w ogólniejszym przypadku, tj. na zbiorze danych MNIST [15].

4.2 Eksperymenty

W trakcie badań z użyciem współczynnika korelacji rang Spearmana udało się stwierdzić istotną zależność pomiędzy liczbą martwych aktywacji w wybranym przebiegu wytrenowanego modelu a końcowym błędem rekonstrukcji dla drugiej (końcowej) warstwy kodera w architekturze podstawowej. Dla Eksperymentów 4 i 10 z Rozdziału 3, korelacja ta wynosiła między 0.76 a 0.89, co wskazuje na silną zależność. Warstwa jest kluczowa dla treningu sieci, ponieważ jest ona „wąskim gardłem" tej architektury. Dlatego, że służy ona do rozmieszania spektralnego, liczba neuronów w tej warstwie odpowiada liczbie składowych w zbiorze danych. W przypadku zbioru Samson były to 3 składowe, zaś dla Jasper Ridge występowały 4 składowe. Zwiększa to podatność tej sieci na pojawienie się zjawiska martwych aktywacji i neuronów [19, 24].

Na podstawie tych rezultatów zaproponowane zostały 3 metody reinicjalizacji sieci w oparciu o 2 współczynniki: współczynnik martwych aktywacji dla j–tej warstwy sieci, d^j_{dead},oraz współczynnik martwych aktywacji dla i–tego neuronu j–tej warstwy, $d^{j,i}_{dead}$. Niech u_j będzie liczbą neuronów w j–tej warstwie sieci, zaś P liczbą próbek w minipaczce.

Definicja 1.: Współczynnikiem martwych aktywacji dla j–tej warstwy sieci, d^j_{dead} jest stosunek liczby zerowych aktywacji spośród neuronów tej warstwy, $\mathcal{N}^j_0$, do łącznej liczby aktywacji w danej iteracji treningu, $u_j \cdot P$:

$$d^j_{dead} = \frac{\mathcal{N}^j_0}{u_j \cdot P} \in [0, 1]. \tag{2}$$

$\mathcal{N}^j_0$ jest liczony po wszystkich neuronach j–tej warstwy sieci podczas jednej iteracji treningu.

Definicja 2.: Współczynnikiem martwych aktywacji dla i–tego neuronu j–tej warstwy sieci, $d^{j,i}_{dead}$, gdzie $i \in \{1, ..., u_j\}$, jest stosunek liczby zerowych aktywacji i–tego neuronu podczas danej iteracji do łącznej liczby próbek w danej iteracji treningu:

$$d^{j,i}_{dead} = \frac{\mathcal{N}^{j,i}_0}{P} \in [0, 1]. \tag{3}$$

Niech $\mathcal{U}$ będzie siecią neuronową o n warstwach ukrytych, dla której wektor $\mathbf{u} = [u_0, u_1, ..., u_n, u_{n+1}]$ wyraża liczbę neuronów w kolejnych warstwach. u_0 jest rozmiarem wejścia sieci, zaś u_{n+1} rozmiarem wyjścia. Niech również $t \in (0, 1]$ będzie wartością progową. Zaproponowane zostały następujące metody reinicjalizacji wag sieci:

1. *Metoda pełnej reinicjalizacji sieci:* jeśli istnieje warstwa $i \in \{1, 2, ..., n + 1\}$ taka, że $d^i_{dead} > t$, wtedy wszystkie wagi sieci są generowane losowo zgodnie z danym scenariuszem inicjalizacji (np. He z rozkładem normalnym). Sytuacja ta odpowiada początkowi treningu, kiedy wszystkie wagi w sieci mają losowe wartości.

2. *Metoda reinicjalizacji jednej warstwy sieci:* jeśli istnieje warstwa $i \in \{1, 2, ..., n + 1\}$ taka, że $d^i_{dead} > t$, wtedy wszystkie wagi i–tej warstwy są generowane losowo zgodnie z danym scenariuszem inicjalizacji. Pozostałe wagi sieci nie ulegają zmianom.

3. *Metoda częściowej reinicjalizacji:* jeśli istnieje warstwa $i \in \{1, 2, ..., n + 1\}$ oraz neuron $j \in \{1, ..., u_i\}$ (lub więcej neuronów) taki, że $d^{i,j}_{dead} > t$, wtedy wszystkie wagi j–tego neuronu i–tej warstwy są generowane losowo zgodnie z danym scenariuszem inicjalizacji. Reinicjalizacja ta dotyczy jedynie neuronów wybranej warstwy podczas obecnej iteracji (od jednego do maksymalnie u_i neuronów). Pozostałe wagi sieci nie ulegają zmianom.

Obliczenia współczynników martwych aktywacji odbywają się warstwa po warstwie, zaczynając od pierwszej warstwy ukrytej sieci. Jeśli dla którejś z warstw wartość progowa zostanie przekroczona, dokonywana jest tam reinicjalizacja (lub w przypadku pierwszej metody, reinicjalizacja całej sieci), ale w danej iteracji nie jest już ona wykonywana dla kolejnych warstw, co ma uzasadnienie teoretyczne [19, 24]. Szczegółowy przebieg metod reinicjalizacji sieci prezentuje Algorytm 1 zawarty w Załączniku.

Wykonane zostały eksperymenty weryfikujące wydajność metod reinicjalizacji sieci. Wybrane zostały modele autokoderów z Eksperymentów 4, 5 i 10 z Rozdziału 3, czyli

wszystkich, w których użyta została architektura podstawowa z funkcją aktywacji ReLU. W każdym scenariuszu użyte zostało 200 modeli, jednak tym razem przeprowadzono jedno uruchomienie dla każdego modelu, a nie 50, jak miało to miejsce w poprzednim rozdziale. Dla każdego spośród 200 modeli zostały przeprowadzone cztery niezależne sesje treningowe: z użyciem standardowej metody uczenia (*baseline*), a także zastosowano każdą z metod reinicjalizacji sieci: *pełną reinicjalizację*, *reinicjalizację jednej warstwy* oraz *częściową reinicjalizację*. Metody te zostały jednak nieco zmodyfikowane, aby uwzględnić wnioski z analizy współczynników korelacji rang Spearmana. Obliczenia współczynników martwych aktywacji były prowadzone jedynie dla neuronów drugiej warstwy kodera, czyli tej, która najbardziej korelowała z końcowym błędem rekonstrukcji. Wartość progowa została ustalona na $t = 0.6$, czyli 60% martwych aktywacji było progiem reinicjalizacji. Wykonano również test Wilcoxona dla par obserwacji, aby sprawdzić, czy nastąpiła istotnie statystyczna poprawa po zastosowaniu metod reinicjalizacji.

Zweryfikowano również działanie metod reinicjalizacji dla większych sieci, w scenariuszu wykraczającym poza obrazowanie hiperspektralne. Wybrano zbiór danych MNIST [15] zawierający odręcznie pisane cyfry rozmiaru 28×28 i przeprowadzono ich rekonstrukcję. Wykonano eksperymenty dla różnych zestawów hiperparametrów sieci, tzn. dla różnych architektur autokoderów (m.in. 2 warstwy po 500 neuronów lub 6 warstw po 100 neuronów, zarówno w koderze jak i dekoderze), czterech wartości współczynnika uczenia (od 0.0001 do 0.1) oraz pięciu rozmiarów minipaczek (od 4 do 128). Sprawdzono również kilka wartości progowych dla metod reinicjalizacji. Szczegółowe informacje nt. hiperparametrów eksperymentu znajdują się w rozprawie.

4.3 Wyniki

W Tabeli 5 umieszczonej w Załączniku zaprezentowane są wyniki eksperymentów rozmieszania spektralnego z użyciem metod reinicjalizacji. Zawierają one wartości błędu rekonstrukcji i błędu nasyceń w sensie RMSE oraz błędu składowych w sensie funkcji SAD. Rezultaty te pokazują, że w wielu przypadkach metody reinicjalizacji w sposób statystycznie istotny redukują błąd w porównaniu do standardowego sposobu treningu sieci, zgodnie z wynikami testu Wilcoxona. W przypadku Eksperymentu 4, dwukrotnie najbardziej skuteczna była metoda pełnej reinicjalizacji (dla inicjalizacji wag w sensie Glorota), a dwukrotnie metoda częściowej reinicjalizacji (dla inicjalizacji zgodnie z propozycją He). W Eksperymencie 10 za każdym razem metoda częściowej reinicjalizacji była najbardiej wydajna, w sensie minimalizacji błędu rekonstrukcji. Również błędy nasyceń i składowych zostały zredukowane w wyniku zastosowania metod reinicjalizacji.

Rysunek 3 przedstawia wykresy nachylenia (ang. *slope charts*) z wynikami eksperymentów z użyciem zbioru MNIST, z podziałem na różne metody inicjalizacji. Jeden odcinek reprezentuje jedno uruchomienie modelu, a na każdym z wykresów zebrane są wszystkie kombinacje hiperparametrów stosowane w eksperymencie. Zielone odcinki odnoszą się do sytuacji, w których błąd rekonstrukcji zmalał po zastosowaniu danej metody reinicjalizacji, czerwone to modele, dla których błąd wzrósł, a szare wskazują na modele, których błąd rekonstrukcji w standardowym scenariuszu treningowym był taki sam jak po użyciu metody reinicjalizacji. Można zatem zaobserwować, zwłaszcza w przypadku pełnej reinicjalizacji jak i reinicjalizacji jednej warstwy, że wydajność

dobrych modeli nie uległa pogorszeniu po zastosowaniu proponowanych metod, za to mniej wydajne modele w dużej mierze uległy poprawie. Metoda częściowej reinicjalizacji osiągnęła najlepszy wynik w sensie błędu rekonstrukcji pojedynczego modelu.

4.4 Wnioski

W powyższym rozdziale przedstawione zostały trzy metody reinicjalizacji, które mają za zadanie poprawić wydajność modeli, w których zachodzi zjawisko martwych aktywacji. Wyniki eksperymentów na zbiorach hiperspektralnych jak i na zbiorze MNIST wskazują na potencjał wskazanych rozwiązań. Należy podkreślić, że metody reinicjalizacji mogą naprawić modele mające tendencję do śmierci neuronów przy jednoczesnym zachowaniu wydajności modeli, w których nie zachodzi to zjawisko. Pogłębiona analiza wyników, która została zaprezentowana w rozprawie, wskazała, że śmierć neuronów może być spowodowana m.in. złym doborem hiperparametrów sieci. Dlatego też metody reinicjalizacji mogą posłużyć do przeszukiwania przestrzeni hiperparametrów, podczas którego trening wybranych modeli byłby wcześniej przerywany, aby nie marnować czasu obliczeniowego na ich uczenie.

5 Wnioski

W rozprawie omawiany był temat optymalizacji architektur sieci głębokiego uczenia w klasyfikacji obrazów hiperspektralnych. W Rozdziale 2, w ramach badań z użyciem różnych sieci, scenariuszy eksperymentów oraz reprezentatywnego zbioru danych, pokazano wzrost wydajności związany z wyborem odpowiedniej architektury. Zidentyfikowano również problem związany ze scenariuszem induktywnym, który doprowadził do rozmieszania spektralnego omówionego w kolejnym rozdziale.

W Rozdziale 2 pokazano także problem dotyczący stabilności jednej z sieci. Polegał on na tym, że dla danego zbioru hiperparametrów część sesji treningowych prowadziła do gorszych wyników w porównaniu do pozostałych uruchomień. Zaobserwowano również, iż prostsze architektury wciąż mogą osiągać konkurencyjne wyniki, a są przy tym łatwiejsze do analizy. Dlatego wybrano liniowe autokodery do eksperymentów rozmieszania spektralnego opisanych w Rozdziale 3. Okazało się, że problem stabilności sieci również dotyczy tej architektury. Przeprowadzono zatem analizę statystyczną, z której wynikło, iż inicjalizacja wag w sposób statystycznie istotny przekłada się na błąd rekonstrukcji sieci po zakończeniu procesu uczenia. Przygotowano eksperymenty z użyciem różnych architektur, zbiorów danych, funkcji straty, metod inicjalizacji wag, itd. Wyniki otrzymane w Rozdziałach 2 i 3. potwierdziły pierwszą część tezy rozprawy, tj.: **Optymalizacja architektur sieci głębokiego uczenia zwiększa wydajność sieci w zastosowaniu do danych hiperspektralnych.**

Zdecydowano się zbadać wybrane modele wytrenowane w eksperymentach z Rozdziału 3. Odkryto, że część modeli autokoderów jest podatna na problem martwych aktywacji. Zaproponowane zostały trzy metody reinicjalizacji wag zmniejszające negatywny wpływ tego zjawiska. W wielu przypadkach w sposób statystycznie istotny poprawiały one rezultaty uzyskiwane przez modele uczone w standardowy sposób. Potwierdziło to drugą część tezy rozprawy, tj.: **Metody reinicjalizacji wag zwiększają wydajność sieci w zastosowaniu do danych hiperspektralnych.** Można

zatem stwierdzić, że wyniki prezentowane w pracy w pełni potwierdzają postawioną w niej tezę.

Podziękowanie

Praca została dofinansowana w ramach projektu POWR.03.02.00-00-I029 realizowanego w ramach Programu Operacyjnego Wiedza Edukacja Rozwój, współfinansowanego ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego.

6 Załączniki

Tabela 3: Wyniki dla scenariusza transduktywnego (HTC) pod względem dokładności klasyfikacji (OA), średniej dokładności klasyfikacji (AA) oraz współczynnika κ Cohena. Kolejne wiersze tabeli reprezentują testowane obrazy, a kolumny odnoszą się do średnich wyników (z odchyleniem standardowym) dla różnych architektur.

		SVM	MLP	1D CNN [10]	2D CNN [16]	3D CNN [17]	3D CNN [4]	RNN [21]
F(1)	OA:	99.7 ± 0.1	99.7 ± 0.1	99.2 ± 0.3	98.9 ± 0.8	99.4 ± 0.5	$\mathbf{99.8 \pm 0.2}$	98.6 ± 3.0
	AA:	99.8 ± 0.1	99.7 ± 0.2	99.5 ± 0.1	99.2 ± 0.7	99.3 ± 0.8	$\mathbf{99.9 \pm 0.1}$	97.7 ± 5.5
	κ:	1.00 ± 0.0	1.00 ± 0.0	0.99 ± 0.0	0.98 ± 0.0	0.99 ± 0.0	$\mathbf{1.00 \pm 0.0}$	0.98 ± 0.0
F(1a)	OA:	99.7 ± 0.1	99.6 ± 0.3	99.1 ± 0.2	99.3 ± 0.6	99.8 ± 0.2	$\mathbf{99.9 \pm 0.1}$	99.7 ± 0.1
	AA:	99.7 ± 0.1	99.4 ± 0.5	99.0 ± 0.2	99.4 ± 0.4	99.8 ± 0.4	$\mathbf{99.8 \pm 0.2}$	99.7 ± 0.1
	κ:	1.00 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	1.00 ± 0.0	$\mathbf{1.00 \pm 0.0}$	1.00 ± 0.0
F(2)	OA:	99.8 ± 0.1	99.7 ± 0.2	99.3 ± 0.2	99.7 ± 0.1	99.7 ± 0.3	99.8 ± 0.1	$\mathbf{99.9 \pm 0.1}$
	AA:	99.7 ± 0.1	99.4 ± 0.7	98.8 ± 0.3	99.4 ± 0.4	99.4 ± 0.5	99.6 ± 0.2	$\mathbf{99.8 \pm 0.1}$
	κ:	1.00 ± 0.0	1.00 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	1.00 ± 0.0	1.00 ± 0.0	$\mathbf{1.00 \pm 0.0}$
F(2k)	OA:	$\mathbf{99.9 \pm 0.1}$	99.7 ± 0.1	99.2 ± 0.2	98.9 ± 0.5	99.2 ± 1.7	99.8 ± 0.1	96.8 ± 6.2
	AA:	$\mathbf{99.8 \pm 0.1}$	99.5 ± 0.1	98.8 ± 0.3	98.6 ± 0.8	98.7 ± 2.8	99.7 ± 0.2	95.9 ± 7.5
	κ:	$\mathbf{1.00 \pm 0.0}$	1.00 ± 0.0	0.99 ± 0.0	0.98 ± 0.0	0.99 ± 0.0	1.00 ± 0.0	0.95 ± 0.1
F(7)	OA:	99.8 ± 0.1	99.7 ± 0.1	99.1 ± 0.2	99.4 ± 0.2	$\mathbf{99.9 \pm 0.1}$	99.4 ± 1.4	98.9 ± 1.0
	AA:	99.7 ± 0.1	99.5 ± 0.1	98.6 ± 0.2	99.3 ± 0.3	$\mathbf{99.9 \pm 0.2}$	98.9 ± 2.7	98.1 ± 2.1
	κ:	1.00 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	$\mathbf{1.00 \pm 0.0}$	0.99 ± 0.0	0.98 ± 0.0
F(21)	OA:	99.8 ± 0.0	99.8 ± 0.0	99.4 ± 0.1	99.6 ± 0.3	$\mathbf{99.9 \pm 0.1}$	99.7 ± 0.2	98.7 ± 2.2
	AA:	99.7 ± 0.1	99.6 ± 0.2	99.0 ± 0.2	99.4 ± 0.5	$\mathbf{99.8 \pm 0.3}$	99.5 ± 0.3	96.9 ± 5.8
	κ:	1.00 ± 0.0	1.00 ± 0.0	0.99 ± 0.0	0.99 ± 0.0	$\mathbf{1.00 \pm 0.0}$	1.00 ± 0.0	0.98 ± 0.0
E(1)	OA:	$\mathbf{94.5 \pm 0.7}$	93.5 ± 1.7	89.8 ± 0.9	93.4 ± 0.8	86.4 ± 8.2	90.8 ± 4.5	85.1 ± 8.3
	AA:	$\mathbf{93.7 \pm 0.8}$	93.1 ± 1.3	89.4 ± 0.6	92.7 ± 1.0	84.8 ± 9.3	89.9 ± 3.7	85.8 ± 6.0
	κ:	$\mathbf{0.93 \pm 0.0}$	0.92 ± 0.0	0.87 ± 0.0	0.92 ± 0.0	0.83 ± 0.1	0.89 ± 0.1	0.82 ± 0.1
E(7)	OA:	$\mathbf{90.6 \pm 1.7}$	90.1 ± 1.9	75.2 ± 1.0	85.5 ± 1.9	80.7 ± 11.3	81.2 ± 9.5	85.3 ± 4.4
	AA:	$\mathbf{88.9 \pm 1.6}$	89.1 ± 1.2	74.2 ± 1.0	84.2 ± 2.3	78.2 ± 13.5	79.5 ± 7.7	83.0 ± 6.1
	κ:	$\mathbf{0.88 \pm 0.0}$	0.87 ± 0.0	0.69 ± 0.0	0.81 ± 0.0	0.76 ± 0.1	0.76 ± 0.1	0.81 ± 0.1
E(21)	OA:	87.8 ± 1.7	89.4 ± 1.6	74.0 ± 2.4	83.3 ± 3.5	$\mathbf{94.3 \pm 1.8}$	87.4 ± 3.5	85.0 ± 4.1
	AA:	86.1 ± 1.6	88.1 ± 1.5	72.3 ± 2.3	82.9 ± 2.8	$\mathbf{93.3 \pm 2.4}$	86.9 ± 3.4	83.0 ± 3.7
	κ:	0.85 ± 0.0	0.87 ± 0.0	0.67 ± 0.0	0.79 ± 0.0	$\mathbf{0.93 \pm 0.0}$	0.84 ± 0.0	0.81 ± 0.1

Tabela 4: Wyniki dla scenariusza induktywnego (HIC) pod względem dokładności klasyfikacji (OA), średniej dokładności klasyfikacji (AA) oraz współczynnika κ Cohena. Kolejne wiersze tabeli odnoszą się do testowanych par obrazów, np. *F(1)* $\rightarrow$ *E(1)* oznacza, że piksele sceny *F(1)* znajdują się w zbiorze treningowym, zaś piksele sceny *E(1)* w zbiorze testowym. W kolumnach znajdują się średnie wyniki metryk dokładności (wraz z odchyleniami standardowymi) dla różnych architektur.

		SVM	MLP	1D CNN [10]	2D CNN [16]	3D CNN [17]	3D CNN [4]	RNN [21]
F(1)→E(1)	OA:	55.9 ± 2.9	57.8 ± 3.6	46.4 ± 2.3	48.5 ± 5.4	58.1 ± 2.8	53.8 ± 3.2	$\mathbf{66.8 \pm 3.4}$
	AA:	58.6 ± 3.1	60.3 ± 3.3	48.4 ± 2.4	50.6 ± 5.4	55.2 ± 3.6	54.6 ± 2.9	$\mathbf{68.2 \pm 3.7}$
	κ:	0.48 ± 0.0	0.50 ± 0.0	0.36 ± 0.0	0.39 ± 0.1	0.49 ± 0.0	0.45 ± 0.0	$\mathbf{0.60 \pm 0.0}$
F(1a)→E(1)	OA:	60.7 ± 2.6	64.8 ± 2.0	52.1 ± 1.5	59.0 ± 4.9	59.3 ± 4.4	66.6 ± 2.1	$\mathbf{71.8 \pm 0.9}$
	AA:	62.0 ± 3.1	66.0 ± 1.9	53.1 ± 1.5	60.1 ± 4.7	58.2 ± 4.4	67.6 ± 1.7	$\mathbf{71.6 \pm 1.2}$
	κ:	0.53 ± 0.0	0.58 ± 0.0	0.43 ± 0.0	0.51 ± 0.1	0.51 ± 0.1	0.60 ± 0.0	$\mathbf{0.66 \pm 0.0}$
F(2)→E(7)	OA:	56.5 ± 5.0	58.7 ± 2.0	49.7 ± 1.0	57.0 ± 2.3	55.3 ± 5.0	$\mathbf{63.0 \pm 2.9}$	62.2 ± 1.2
	AA:	58.7 ± 6.0	61.3 ± 2.2	52.1 ± 0.9	59.1 ± 1.9	53.6 ± 6.9	$\mathbf{65.1 \pm 3.2}$	61.7 ± 1.7
	κ:	0.48 ± 0.1	0.50 ± 0.0	0.40 ± 0.0	0.48 ± 0.0	0.45 ± 0.1	$\mathbf{0.55 \pm 0.0}$	0.54 ± 0.0
F(2k)→E(7)	OA:	52.9 ± 2.6	58.8 ± 1.2	45.7 ± 0.9	51.2 ± 7.1	52.4 ± 7.7	57.3 ± 2.7	$\mathbf{59.6 \pm 4.7}$
	AA:	54.9 ± 3.1	60.9 ± 1.3	47.1 ± 1.0	52.4 ± 5.8	52.7 ± 6.9	59.0 ± 3.2	$\mathbf{57.4 \pm 4.5}$
	κ:	0.43 ± 0.0	0.50 ± 0.0	0.35 ± 0.0	0.41 ± 0.1	0.43 ± 0.1	0.48 ± 0.0	$\mathbf{0.51 \pm 0.1}$
F(7)→E(7)	OA:	54.7 ± 2.1	57.2 ± 2.0	47.2 ± 0.8	54.3 ± 2.0	60.3 ± 3.5	59.8 ± 2.6	$\mathbf{63.6 \pm 3.8}$
	AA:	59.4 ± 2.0	60.2 ± 2.2	50.8 ± 0.7	56.1 ± 2.3	57.8 ± 5.0	62.9 ± 2.5	$\mathbf{65.3 \pm 3.7}$
	κ:	0.46 ± 0.0	0.48 ± 0.0	0.37 ± 0.0	0.45 ± 0.0	0.52 ± 0.0	0.52 ± 0.0	$\mathbf{0.56 \pm 0.0}$
F(21)→E(21)	OA:	45.4 ± 1.7	49.7 ± 2.4	44.0 ± 1.4	49.2 ± 1.6	54.4 ± 3.4	$\mathbf{57.2 \pm 1.3}$	56.3 ± 3.1
	AA:	48.3 ± 2.4	51.9 ± 1.9	45.7 ± 1.7	51.0 ± 1.3	51.4 ± 3.6	$\mathbf{59.1 \pm 1.4}$	55.0 ± 2.0
	κ:	0.35 ± 0.0	0.40 ± 0.0	0.33 ± 0.0	0.39 ± 0.0	0.45 ± 0.0	$\mathbf{0.49 \pm 0.0}$	0.47 ± 0.0
F(2)→F(2k)	OA:	$\mathbf{98.2 \pm 0.5}$	97.5 ± 0.6	97.0 ± 0.3	96.9 ± 0.5	80.0 ± 12.1	98.1 ± 0.5	90.4 ± 2.0
	AA:	$\mathbf{97.7 \pm 0.6}$	96.7 ± 0.8	95.8 ± 0.5	95.6 ± 0.8	77.4 ± 12.1	97.2 ± 0.6	87.2 ± 2.5
	κ:	$\mathbf{0.98 \pm 0.0}$	0.97 ± 0.0	0.96 ± 0.0	0.96 ± 0.0	0.75 ± 0.1	0.97 ± 0.0	0.88 ± 0.0
F(2k)→F(2)	OA:	$\mathbf{99.5 \pm 0.2}$	99.2 ± 0.3	98.8 ± 0.3	96.5 ± 2.1	85.9 ± 10.6	99.2 ± 1.0	93.0 ± 6.0
	AA:	$\mathbf{99.4 \pm 0.3}$	98.8 ± 0.5	98.5 ± 0.2	96.6 ± 1.9	82.5 ± 13.9	99.2 ± 0.9	93.2 ± 6.2
	κ:	$\mathbf{0.99 \pm 0.0}$	0.99 ± 0.0	0.98 ± 0.0	0.96 ± 0.0	0.82 ± 0.1	0.99 ± 0.0	0.91 ± 0.1

Algorytm 1: Metody reinicjalizacji sieci

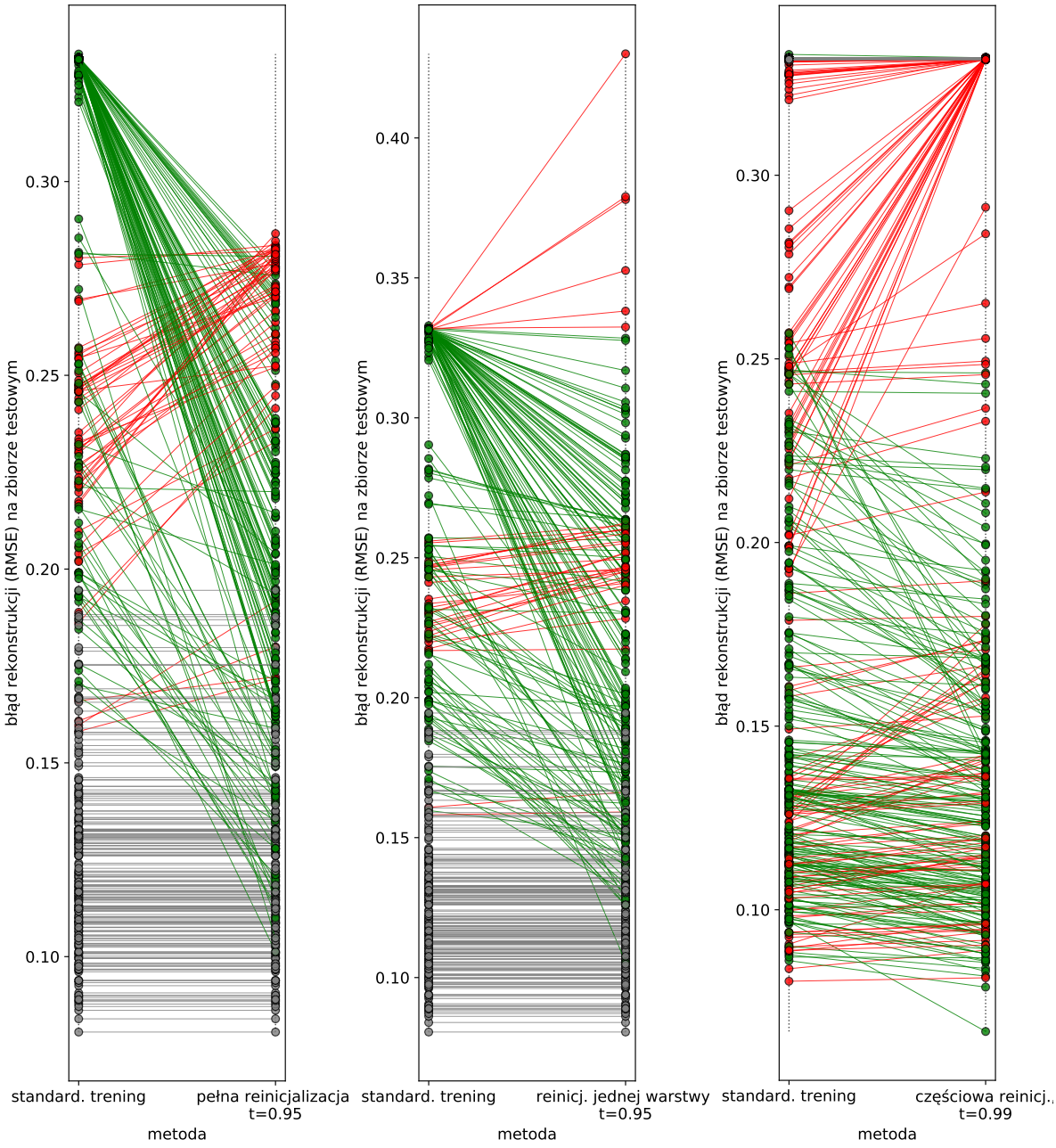
Wejście: sieć neuronowa $\mathfrak{U}$ o n warstwach ukrytych, liczba neuronów w poszczególnych warstwach: $[u_0, u_1, u_2, ..., u_n, u_{n+1}]$, minipaczka z danymi wejściowymi: $\boldsymbol{X}$, $\boldsymbol{X} \in \mathbb{R}^{b_s \times B}$, b_s: liczba próbek w minipaczce, B: wymiar pojedynczej próbki, $t \in (0, 1)$: wartość progowa, *metoda*: wybrana metoda reinicjalizacji (*pełna reinicjalizacja*, *reinicjalizacja jednej warstwy* lub *częściowa reinicjalizacja*)

Wyjście: sieć neuronowa $\mathfrak{U}$ o n warstwach ukrytych

Przeprowadź propagację w przód;
dla $i \leftarrow 1$ **do** $n + 1$ **powtarzaj**
  jeśli *metoda* = pełna reinicjalizacja **lub** reinicjalizacja jednej warstwy **wykonaj**
    Oblicz d^i_{dead};
    jeśli $d^i_{dead} > t$ **wykonaj**
      jeśli *metoda* = pełna reinicjalizacja **wykonaj**
        Dokonaj reinicjalizacji wszystkich wag sieci $\mathfrak{U}$.
      koniec
      w przeciwnym wypadku
        Dokonaj reinicjalizacji i–tej warstwy sieci $\mathfrak{U}$.
      koniec
      Przerwij obliczenia i zwróć sieć $\mathfrak{U}$ z wagami po reinicjalizacji.
    koniec
  koniec
  w przeciwnym wypadku jeśli *metoda* = częściowa reinicjalizacja **wykonaj**
    Oblicz $d^{i,1}_{dead}, d^{i,2}_{dead}, ..., d^{i,u_i}_{dead}$;
    dla $j \leftarrow 1$ **do** u_i **powtarzaj**
      jeśli $d^{i,j}_{dead} > t$ **wykonaj**
        Dokonaj reinicjalizacji wag j–tego neuronu i–tej warstwy sieci $\mathfrak{U}$.
      koniec
    koniec
    /* W trakcie jednej iteracji tylko neurony jednej warstwy mogą zostać zreinicjowane. Jeśli przynajmniej w jednym przypadku wartość progowa zostanie przekroczona, dalsze obliczenia współczynników martwych aktywacji zostają wstrzymane i trening sieci jest kontynuowany. */
    jeśli $\max_{j \in \{1,...,u_i\}} d^{i,j}_{dead} > t$ **wykonaj**
      Przerwij obliczenia i zwróć sieć $\mathfrak{U}$ z wagami po reinicjalizacji.
    koniec
  koniec
koniec

Tabela 5: Średnie wyniki metod reinicjalizacji sieci w porównaniu ze standardowym scenariuszem treningu (wraz z odpowiadającym odchyleniem standardowym) dla Eksperymentów 4 i 10. Czarne koło (•) wskazuje na to, że metoda reinicjalizacji przewyższa standardowy scenariusz treningu w sposób statystycznie istotny, zgodnie z wynikiem testu Wilcoxona dla par obserwacji ($p < 0.05$).

inicjalizacja	metoda reinicjalizacji	błąd rekonstrukcji	błąd nasyceń	błąd składowych
Eksperyment 4 – zbiór danych: Samson, funkcja straty: MSE				
Glorot r. normalny	standardowy trening	0.036 ± 0.04	0.349 ± 0.04	0.744 ± 0.14
	częściowa reinicjalizacja	0.089 ± 0.03	0.413 ± 0.08	**0.451 ± 0.24** •
	reinicjal. jednej warstwy	0.019 ± 0.03 •	0.310 ± 0.06 •	0.777 ± 0.19
	pełna reinicjalizacja	**0.007 ± 0.00** •	**0.276 ± 0.03** •	0.687 ± 0.11 •
Glorot r. jednostajny	standardowy trening	0.052 ± 0.05	0.344 ± 0.05	0.723 ± 0.12
	częściowa reinicjalizacja	0.089 ± 0.03	0.407 ± 0.08	**0.375 ± 0.03** •
	reinicjal. jednej warstwy	0.020 ± 0.04 •	0.307 ± 0.05 •	0.707 ± 0.16
	pełna reinicjalizacja	**0.007 ± 0.00** •	**0.270 ± 0.03** •	0.695 ± 0.11
He r. normalny	standardowy trening	0.047 ± 0.04	0.365 ± 0.03	1.176 ± 0.13
	częściowa reinicjalizacja	**0.024 ± 0.01** •	**0.330 ± 0.03** •	**1.100 ± 0.13** •
	reinicjal. jednej warstwy	0.028 ± 0.01 •	0.355 ± 0.04	1.153 ± 0.11
	pełna reinicjalizacja	0.027 ± 0.01 •	0.355 ± 0.04	1.134 ± 0.13 •
He r. jednostajny	standardowy trening	0.053 ± 0.05	0.357 ± 0.03	0.758 ± 0.19
	częściowa reinicjalizacja	**0.007 ± 0.00** •	**0.308 ± 0.04** •	0.733 ± 0.12
	reinicjal. jednej warstwy	0.014 ± 0.01 •	0.347 ± 0.03	0.732 ± 0.15
	pełna reinicjalizacja	0.008 ± 0.01 •	0.316 ± 0.04 •	**0.663 ± 0.17** •
Eksperyment 10 – zbiór danych: Jasper Ridge, funkcja straty: MSE				
Glorot r. normalny	standardowy trening	0.020 ± 0.03	0.285 ± 0.04	0.894 ± 0.11
	częściowa reinicjalizacja	**0.011 ± 0.00** •	0.289 ± 0.03	0.893 ± 0.08
	reinicjal. jednej warstwy	0.014 ± 0.01	**0.259 ± 0.04** •	**0.714 ± 0.10** •
	pełna reinicjalizacja	0.119 ± 0.10	0.316 ± 0.09	1.016 ± 0.27
Glorot r. jednostajny	standardowy trening	0.019 ± 0.03	0.281 ± 0.04	0.899 ± 0.11
	częściowa reinicjalizacja	**0.011 ± 0.00** •	0.281 ± 0.02	0.863 ± 0.08 •
	reinicjal. jednej warstwy	0.017 ± 0.01	**0.252 ± 0.05** •	**0.685 ± 0.13** •
	pełna reinicjalizacja	0.112 ± 0.10	0.307 ± 0.10	1.017 ± 0.28
He r. normalny	standardowy trening	0.021 ± 0.03	0.300 ± 0.02	1.033 ± 0.11
	częściowa reinicjalizacja	**0.011 ± 0.00** •	0.300 ± 0.02	1.006 ± 0.06
	reinicjal. jednej warstwy	0.015 ± 0.01	0.301 ± 0.02	**0.999 ± 0.10**
	pełna reinicjalizacja	0.038 ± 0.04	**0.298 ± 0.02**	1.098 ± 0.12
He r. jednostajny	standardowy trening	0.037 ± 0.05	0.276 ± 0.04	0.881 ± 0.10
	częściowa reinicjalizacja	**0.010 ± 0.00** •	0.273 ± 0.03	0.827 ± 0.09 •
	reinicjal. jednej warstwy	0.018 ± 0.03 •	**0.261 ± 0.04** •	**0.795 ± 0.14** •
	pełna reinicjalizacja	0.028 ± 0.04	0.265 ± 0.05	0.909 ± 0.14



Rysunek 3: Porównanie błędów rekonstrukcji w sensie RMSE na zbiorze testowym MNIST pomiędzy standardowym sposobem treningu oraz różnymi metodami reinicjalizacji. Każda linia reprezentuje jedno uruchomienie treningowe modelu. Zielony kolor oznacza, że błąd po zastosowaniu reinicjalizacji spadł, czerwony, że wzrósł, a szary odnosi się do sytuacji, w której wartość błędu nie zmieniła się.

7 Pozostałe publikacje

Następująca lista zawiera pozostałe publikacje stworzone przez autora rozprawy, które nie zostały uwzględnione w tekście dysertacji.

1. M. Żarski, B. Wójcik, **K. Książek**, J. A. Miszczak, *Finicky transfer learning–A method of pruning convolutional neural networks for cracks classification on edge devices*, Computer-Aided Civil and Infrastructure Engineering, 2022, Vol. 37, Issue 4, ss. 500-515, ISSN 1093-9687, DOI: 10.1111/mice.12755.

2. K. Grochla, A. Strzoda, R. Marjasz, P. Głomb, **K. Książek**, Z. Łaskarzewski, *Energy-Aware Algorithm for Assignment of Relays in LP WAN*, ACM Transactions on Sensor Networks, 2022, Association for Computing Machinery, New York, ss. 1-23, ISSN 1550-4859, DOI: 10.1145/3544561.

3. W. M. Kempa, **K. Książek**, R. Marjasz, *On Time-Dependent Queue-Size Distribution in a Model With Finite Buffer Capacity and Deterministic Multiple Vacations With Applications to LTE DRX Mechanism Modeling*, IEEE Access, 2021, Vol. 9, ss. 148374-148383, DOI: 10.1109/ACCESS.2021.3123897.

4. **K. Książek**, K. Grochla, *Flexibility Analysis of Adaptive Data Rate Algorithm in LoRa Networks*, 2021 International Wireless Communications and Mobile Computing (IWCMC), 2021, ss. 1393-1398, DOI: 10.1109/IWCMC51323.2021.9498664.

5. F. Pałka, W. Książek, P. Pławiak, M. Romaszewski, **K. Książek**, *Hyperspectral Classification of Blood-Like Substances Using Machine Learning Methods Combined with Genetic Algorithms in Transductive and Inductive Scenarios*, Sensors, 2021, Vol. 21, Issue 7, No. 2293, ss. 1-18, ISSN 1424-8220, DOI: 10.3390/s21072293.

6. W. M. Kempa, **K. Książek**, *On transient queue-size distribution in a finite-buffer model with threshold waking and early setup policy*, Performance Evaluation, 2020, Vol. 140-141, ss. 102107, ISSN 0166-5316, DOI: 10.1016/j.peva.2020.102107.

7. **K. Książek**, K. Grochla, *Aggregation of GPS, WLAN, and BLE Localization Measurements for Mobile Devices in Simulated Environments*, Sensors, 2019, Vol. 19, Issue 7, No. 1694, ss. 1-15, ISSN 1424-8220, DOI: 10.3390/s19071694.

8. M. Woźniak, **K. Książek**, J. Marciniec, D. Połap, *Heat production optimization using bio-inspired algorithms*, Engineering Applications of Artificial Intelligence, 2018, Vol. 76, ss. 185-201, ISSN 0952-1976, DOI: 10.1016/j.engappai.2018.09.003.

9. **K. Książek**, Z. Marszałek, G. Capizzi, C. Napoli, D. Połap, M. Woźniak, *The Impact of Parallel Programming on Faster Image Filtering*, 2018 Federated Conference on Computer Science and Information Systems (FedCSIS), w: Annals of

Computer Science and Information Systems, 2018, Vol. 15, ss. 545-550, DOI: 10.15439/2018F71.

10. M. Woźniak, **K. Książek**, D. Połap, *Benchmark tests on heuristic methods in the darts game*, International Journal of Electronics and Telecommunications, 2018, Vol. 64, No. 2, ss. 115-121, DOI: 10.24425/119358.

11. D. Połap, K. Kęsik, **K. Książek**, M. Woźniak, *Obstacle Detection as a Safety Alert in Augmented Reality Models by the Use of Deep Learning Techniques*, Sensors, 2017, Vol. 17, Issue 12, No. 2803, ss. 1-16, DOI: 10.3390/s17122803.

12. **K. Książek**, D. Połap, M. Woźniak, R. Damaševičius, *Radiation heat transfer optimization by the use of modified ant lion optimizer*, 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017, ss. 1-7, DOI: 10.1109/SSCI.2017.8280853.

13. **K. Książek**, W. Masarczyk, I. Nowak, *Heuristic approach to the game of darts by using genetic algorithm and ant colony optimization*, Proceedings of the International Conference for Young Researchers in Informatics, Mathematics and Engineering (ICYRIME 2017), w: CEUR, 2017, Vol. 1852, pp. 33-38.

14. **K. Książek**, Z. Marszałek, *Combinatorial Summation Primes: a Discussion of Different Methods to Solve the Problem*, SYSTEM 2016: Symposium for Young Scientists in Technology, Engineering and Mathematics, w: CEUR, 2016, Vol. 1730, pp. 25-34.

Literatura

[1] C. C. Aggarwal. *Neural Networks and Deep Learning.* Springer, 2018.

[2] Y. Amit and P. Felzenszwalb. *Object Detection.* ss. 537–542, Springer US, Boston, MA, 2014.

[3] N. Audebert, B. L. Saux, and S. Lefèvre. Deep learning for classification of hyperspectral data: A comparative review. *IEEE Geoscience and Remote Sensing Magazine*, 7:159–173, 2019.

[4] A. Ben Hamida, A. Benoit, P. Lambert, and C. Ben Amar. 3-D deep learning approach for remote sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(8):4420–4434, 2018.

[5] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot. Hyperspectral remote sensing data analysis and future challenges. *IEEE Geoscience and Remote Sensing Magazine*, 1(2):6–36, 2013.

[6] W. J. Conover and R. L. Iman. *On Multiple-Comparisons Procedures. Informal report*, 1979.

[7] A. Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media, Inc., 2017.

[8] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. ss. 249–256, JMLR Workshop and Conference Proceedings, 2010.

[9] K. He, X. Zhang, S. Ren, and J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*, 2015. arXiv:1502.01852.

[10] W. Hu, Y. Huang, F. Zhang, and H. Li. Deep convolutional neural networks for hyperspectral image classification. *Journal of Sensors*, 2015.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017.

[12] W. H. Kruskal and W. A. Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952.

[13] K. Książek, P. Głomb, M. Romaszewski, M. Cholewa, B. Grabowski, and K. Búza. Improving autoencoder training performance for hyperspectral unmixing with network reinitialisation. In *Image Analysis and Processing – ICIAP 2022: 21st International Conference, Lecce, Italy, May 23–27, 2022, Proceedings, Part I*, ss. 391–403, Berlin, Heidelberg, 2022. Springer-Verlag.

[14] K. Książek, M. Romaszewski, P. Głomb, B. Grabowski, and M. Cholewa. Blood stain classification with hyperspectral imaging and deep neural networks. *Sensors*, 20(22), 2020.

[15] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[16] H. Lee and H. Kwon. Going deeper with contextual cnn for hyperspectral image classification. *IEEE Transactions on Image Processing*, 26(10):4843–4855, 2017.

[17] Y. Li, H. Zhang, and Q. Shen. Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network. *Remote Sensing*, 9(1), 2017.

[18] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica. *Tune: A Research Platform for Distributed Model Selection and Training*, 2018.

[19] L. Lu. Dying ReLU and initialization: Theory and numerical examples. *Communications in Computational Physics*, 28(5):1671–1706, Jun 2020.

[20] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3523–3542, 2022.

[21] L. Mou, P. Ghamisi, and X. X. Zhu. Deep recurrent neural networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):3639–3655, 2017.

[22] B. Palsson, J. Sigurdsson, J. R. Sveinsson, and M. O. Ulfarsson. Hyperspectral unmixing using a neural network autoencoder. *IEEE Access*, 6:25646–25656, 2018.

[23] M. Paoletti, J. Haut, J. Plaza, and A. Plaza. Deep learning classifiers for hyperspectral imaging: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, 158:279–317, 2019.

[24] B. Rister and D. L. Rubin. *Probabilistic bounds on neuron death in deep rectifier networks*, 2021. arXiv:2007.06192.

[25] M. Romaszewski, P. Głomb, A. Sochan, and M. Cholewa. A dataset for evaluating blood detection in hyperspectral images. *Forensic Science International*, 320:110701, 2021.

[26] D. Stein, S. Beaven, L. Hoff, E. Winter, A. Schaum, and A. Stocker. Anomaly detection from hyperspectral imagery. *IEEE Signal Processing Magazine*, 19(1):58–69, 2002.

[27] Y. Su, J. Li, A. Plaza, A. Marinoni, P. Gamba, and S. Chakravortty. DAEN: deep autoencoder networks for hyperspectral unmixing. *IEEE Transactions on Geoscience and Remote Sensing*, 57(7):4309–4321, 2019.