

Zastosowanie autoenkoderów wariacyjnych do rozpoznawania zmian na obrazach medycznych

()

Tomasz Nanowski

Praca inżynierska

Promotor: dr Jan Chorowski

Uniwersytet Wrocławski
Wydział Matematyki i Informatyki
Instytut Informatyki

31 stycznia 2019

Tomasz Nanowski

.....

.....

(adres zameldowania)

.....

.....

(adres korespondencyjny)

PESEL:

e-mail:

Wydział Matematyki i Informatyki

stacjonarne studia I stopnia

kierunek: informatyka

nr albumu: 279076

Oświadczenie o autorskim wykonaniu pracy dyplomowej

Niniejszym oświadczam, że złożoną do oceny pracę zatytułowaną *Zastosowanie autoenkoderów wariacyjnych do rozpoznawania zmian na obrazach medycznych* wykonałem/am samodzielnie pod kierunkiem promotora, dr. Jana Chorowskiego. Oświadczam, że powyższe dane są zgodne ze stanem faktycznym i znane mi są przepisy ustawy z dn. 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (tekst jednolity: Dz. U. z 2006 r. nr 90, poz. 637, z późniejszymi zmianami) oraz że treść pracy dyplomowej przedstawionej do obrony, zawarta na przekazanym nośniku elektronicznym, jest identyczna z jej wersją drukowaną.

Wrocław, 31 stycznia 2019

(czytelny podpis)

Streszczenie







W tej pracy chciałem sprawdzić skuteczność wykorzystania metody uczenia bez nadzoru w problemie lokalizowania zmian nowotworowych na zdjęciach wykonanych metodą rezonansu magnetycznego. Bazowałem na podstawie obserwacji, że zmiany patologiczne są rzadkie i są pewnym odstępstwem od normy. Zdecydowałem się na użycie autoenkodera wariacyjnego, który pozwala oszacować prawdopodobieństwo wystąpienia danej próbki, co pozwoliłoby mi na ich klasyfikowanie. Pomysł przetestowałem na danych syntetycznych, wykorzystując do tego zbiór MNIST. Otrzymane wyniki okazały się być zadowalające i zachęcały do przeprowadzenia dalszych eksperymentów już na danych medycznych. Niestety w tym przypadku nie można było uznać tego za sukces, a według mojej analizy model nauczył się jedynie zwracać uwagę na prostą własność, jaką jest jasność próbki. Na tym zakończyłem moją pracę i przygotowałem propozycje rozwiązań, które mogłyby według mnie pozytywnie wpłynąć na poprawę rezultatów.

Spis treści

1. Wstęp	9
1.1. Przedstawienie problemu	9
1.2. Sztuczne sieci neuronowe	9
1.3. Uczenie nadzorowane i bez nadzoru	10
1.4. Autoenkoder	10
1.5. Autoenkoder wariacyjny	12
1.5.1. Opis matematyczny	14
1.6. Ocena jakości modelu	14
1.6.1. Krzywa ROC i AUC	14
2. Eksperymenty na danych syntetycznych (MNIST)	17
2.1. Możliwości autoenkodera wariacyjnego	17
2.2. Symulacja docelowego problemu	18
2.3. Wnioski	20
3. Experiments on MRI FLAIR images	21
3.1. Data description & preprocessing	21
3.1.1. Overview	21
3.1.2. Patches	22
3.1.3. Preprocessing	23
3.1.4. Normalizations	24
3.2. Results	24
3.3. Analysis	24

3.3.1. Conclusion	26
3.4. Back to basic models	26
3.4.1. Logistic Regression	26
3.4.2. CNN	26
4. Summary	27

Notes

 Dokończyć matkę	14
Figure: Rekonstrukcje z podziałem na rozmiary reprezentacji	18
Figure: Box plot z rozkładem błędów średniokwadratowych	18
Figure: Generowanie nowych próbek	18
 Uzupełnić podpisy	22
 Może LIME?	24
 Rozważyć normalizacje: histogramowa, tylko środkowy kanał	24
 dodać krótki opis modeli i może dla czego softmax	24
 Dodać z-dim	24
Figure: Zaznaczona na obrazkach oryginalne mazki wraz z tymi zrekonstruowanymi	26
Figure: Dice similarity coefficient	26

Rozdział 1.

Wstęp

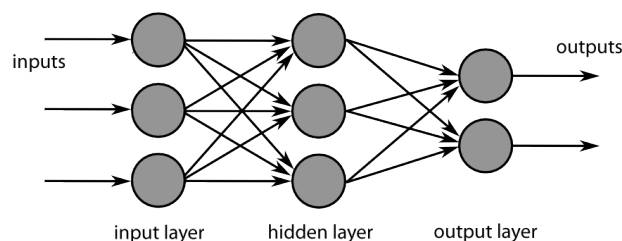
1.1. Przedstawienie problemu

Problem, który zainspirował mnie do napisania tej pracy to lokalizowanie zmian nowotworowych na zdjęciach wykonanych metodą rezonansu magnetycznego (MRI, ang. magnetic resonance imaging). Można podejść do niego od strony nadzorowanej, czyli bazować na danych przeanalizowanych wcześniej przez specjalistów, gdzie każdy przypadek został ręcznie obejrzany i oznaczony. O ile taka metoda ma wiele zalet, jak chociażby korzystanie z rzeczywistej wiedzy eksperckiej, to wykorzystywany zbiór jest bardzo kosztowny w przygotowaniu i dalszym rozwoju, a ponadto może być ograniczony jedynie do pojedynczej partii ciała. Zauważając te wady oraz łącząc je z obserwacją, że zmiany patologiczne tak na prawdę są rzadkie i są pewnym odstępstwem od normy, chciałbym spróbować skorzystać z metody nienadzorowanej, w której to model nauczyłby się oszacowywać prawdopodobieństwo występowania pojedynczej próbki w pewnym kontekście. Przy takim podejściu mógłbym oznaczać obserwacje mało prawdopodobne jako właśnie te nowotworowe. Model, na którego wykorzystanie się zdecydowałem to autoenkoder wariacyjny (VAE, ang. Variational Autoencoder), łączący sztuczne sieci neuronowe z modelowaniem probabilistycznym. Pomysł ten chce przetestować początkowo na danych syntetycznych z wykorzystaniem zbioru MNIST.

1.2. Sztuczne sieci neuronowe

Sztuczne sieci neuronowe mają obecnie bardzo mocno ugruntowaną pozycję szczególnie w dziedzinie problemów związanych z analizą i przetwarzaniem obrazów. Pomimo, iż nie jest to nowy pomysł, dopiero ostatni wzrost w wydajności komputerów pozwolił na ich praktyczne zastosowanie. Z matematycznego punktu widzenia są to sparametryzowane nieliniowe funkcje o pewnej ustalonej strukturze. Składają się z prostych elementów zwanych neuronami, a one natomiast są pogrupowane w

warstwy. Połączenia między warstwami definiują przepływ danych. 'Nauka' sieci neuronowych polega na optymalizacji pewnej funkcji straty, czyli wyznaczeniu takich parametrów, żeby osiągnąć minimalny koszt. Do tego celu często korzysta się z metod opartych na Stochastic Gradient Descent, a przy wybranej strukturze można w efektywny sposób zastosować algorytm propagacji wstecznej. W dalszej części pracy będę używał prostszej nazwy - sieci neuronowe. Przykładowa architektura sieci neuronowych jest zaprezentowana na wykresie 1.1.



Rysunek 1.1: Przykładowy model sieci neuronowej

1.3. Uczenie nadzorowane i bez nadzoru

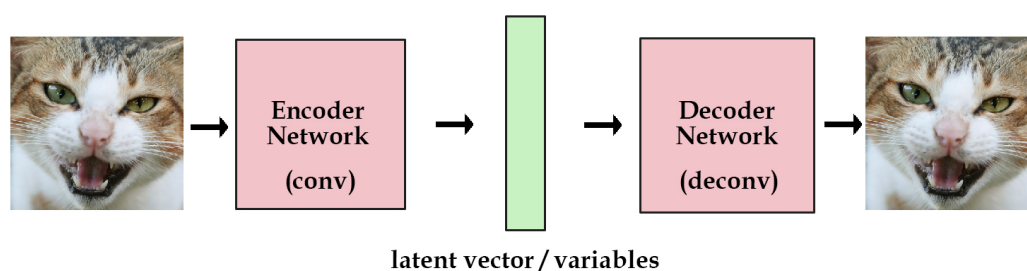
Są to dwa różne rodzaje problemów z dziedziny uczenia maszynowego. W przypadku uczenia nadzorowanego wiemy jaką chcemy uzyskać odpowiedź dla danego wejścia i próbujemy tak dobrać parametry modelu, żeby odpowiadał on z jak największą poprawnością. Przykładowymi zadaniami tego rodzaju są regresja oraz klasyfikacja, która nawiasem mówiąc może zostać uznana za problem regresji, ale na dyskretnym, skończonym zbiorze.

Z grubsza odwrotnie jest w sytuacji bez nadzoru. Tam dla danych wejściowych nie znamy interesującej nas odpowiedzi i staramy się tak zbudować model, żeby był on w stanie sam ją wydobyć. Zazwyczaj interesuje nas rozwiązanie takich problemów jak klasteryzacja, redukcja wymiarowości czy wydajne kodowanie danych. Jednym z modeli wykorzystywanym w tej klasie zadań jest właśnie autoenkoder.

1.4. Autoenkoder

Jest to jeden z rodzajai sieci neuronowych, służący do znajdowania wydajnej reprezentacji danych, co jak wspomniałem wcześniej jest przykładem nauki bez nadzoru. Myślę, że mogę go określić mianem nieliniowej alternatywy dla klasycznej statystycznej metody analizy głównych składowych (PCA, ang. principal component analysis). W autoenkoderach można wyróżnić dwie połączone ze sobą części zwane enkoderem i dekodere. Zadaniem enkodera jest wyprodukowanie właśnie tej reprezentacji, podczas gdy dekodery służy do odtworzenia z niej oryginalnej postaci. Zależy nam na tym, żeby wyjście było w jakimś sensie jak najbardziej podobne

do wejścia przy zachowaniu odpowiednio małej wymiarowości kodowania. Jest to pewnego rodzaju balansowanie pomiędzy ilością informacji, na których przepływać się zgadzamy, a ich jakością. W szczególności można pomyśleć o takiej patologicznej sytuacji jak ustalenie rozmiaru reprezentacji równej rozmiarowi danych wejściowych, co prawdopodobnie będzie skutkować idealnymi rekonstrukcjami, ale przy okazji zerową wiedzą płynącą z takiego modelu. Analogicznie można rozważyć przypadek kiedy obszar kodowania jest za mały. Model w takiej sytuacji skupi się jedynie na przekazaniu wyłącznie trywialnych cech, żeby mimo wszystko jakkolwiek odtworzyć dane. Poglądowy schemat budowy znajduje się na rysunku 1.2.



Rysunek 1.2: Architektura autoenkodera

W przypadku obrazów na funkcję straty często wybierany jest błąd średniokwadratowy (MSE, ang. Mean Squared Error).

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n \left(Y_i - \hat{Y}_i \right)^2 \quad (1.1)$$

gdzie:

Y_i – oryginalne dane

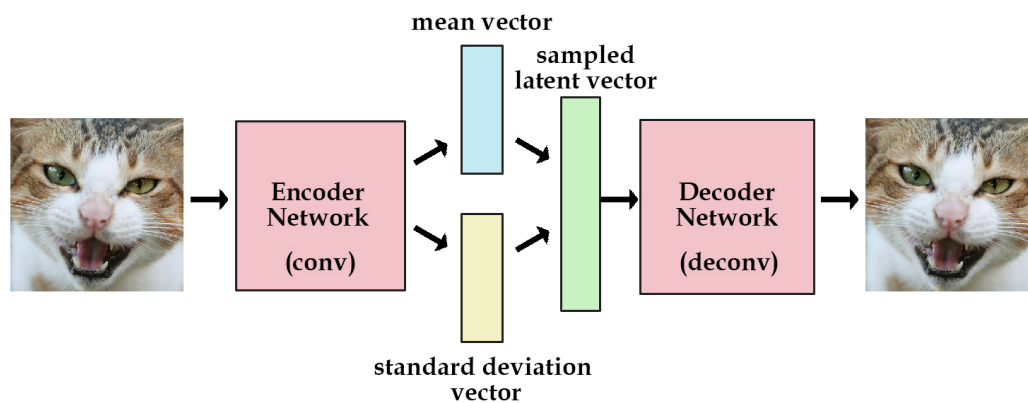
\hat{Y}_i – zrekonstruowane dane

Wymagającym sprostowania może być fakt, iż mimo mówimy jakie oczekujemy wyjście z modelu, w sensie rekonstrukcji, to dalej jest to problem typu nauki bez nadzoru, ponieważ rzeczywistą szukaną przez nas wiedzą jest umiejętność kodowania danych i osobno ich dekodowania. Chciałbym w tym miejscu zastanowić się co dzięki temu zyskujemy. Już teraz zaletą jest oczywiście redukcja wymiarowości, która ułatwia późniejszą analizę danych oraz pozwala na ich kompresję i ewentualne odsumowanie. Dodatkowo wydaje się, że mając wytrenowany dekodery, na konkretnej rodzinie danych, moglibyśmy zaaplikować do niego dowolne kodowanie i w ten sposób może otrzymać wynik pochodzący z oryginalnej dziedziny, czyli wykorzystać go jako generator nowych próbek. Niestety ten pomysł ma taki problem, że podczas nauki nie ma żadnej kontroli nad tym jak dane zostaną rozrzucone w przestrzeni,

która notabene jest nieskończona (z ograniczeniem do pojemności liczb zmiennoprzecinkowych). W takim razie nie wiemy jaka podprzestrzeń odpowiada tym kodowaniom, które są sensownie dekodowane, co jest potrzebne przy generowaniu, a dodatkowo jeśli bałaby nieciągła, to niemożliwe byłoby interpolowanie pomiędzy próbkami. Znaczącym problemem jest natomiast to, że kodowanie jako wektor liczb jest bardzo precyzyjną informacją, co może doprowadzić do przeuczenia modelu i błędnego działania na danych pochodzących z poza zbioru treningowego.

1.5. Autoenkoder wariacyjny

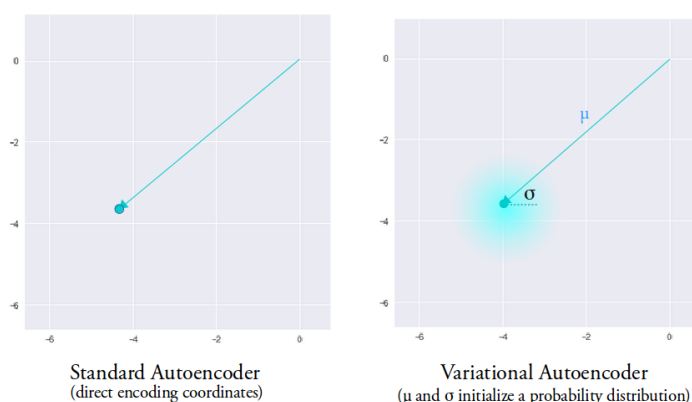
Autoenkoder wariacyjny (VAE, ang. Variational Autoencoder) rozszerza założenia podstawowej architektury o wprowadzenie modelowania rozkładu prawdopodobieństwa dla reprezentacji ukrytej. W odróżnieniu od podstawowej wersji enkoder nie produkuje pojedynczego wektora, ale dwa wektory interpretowane odpowiednio jako średnie μ oraz wariancję σ dla rozkładu Gaussa. Dopiero na podstawie tych parametrów formowane jest kodowanie, gdzie i -ta wartość losowana jest z $\mathcal{N}(\mu_i, \sigma_i^2)$, które aplikowane zostaje do dekodera i rekonstruowana jest na podstawie niego pierwotna próbka. Takie podejście znaczy, że nawet dla tego samego wejścia i podczas gdy średnie oraz odchylenia pozostaną niezmiennie, to i tak reprezentacja będzie się różnić właśnie ze względu na występującą losowość. Zmodyfikowany schemat został przedstawiony na rysunku 1.3.



Rysunek 1.3: Architektura autoenkodera wariacyjnego

Intuicyjnie wektor średnich oznacza miejsce, gdzie zakodowana zmienna powinna być wycelowana, natomiast odchylenie kontroluje obszar, wokół którego kodowanie może się różnić. Sytuację tę zaprezentowałem na wykresie 1.4, przy czym porównałem z podstawową wersją. W jej przypadku informacja wychodząca z enkodera jest bardzo precyzyjna, natomiast w obecnie rozważanym modelu, dzięki dodaniu szumu, zostaje ona rozmyta. Taki zabieg ma za zadanie ograniczyć zjawisko

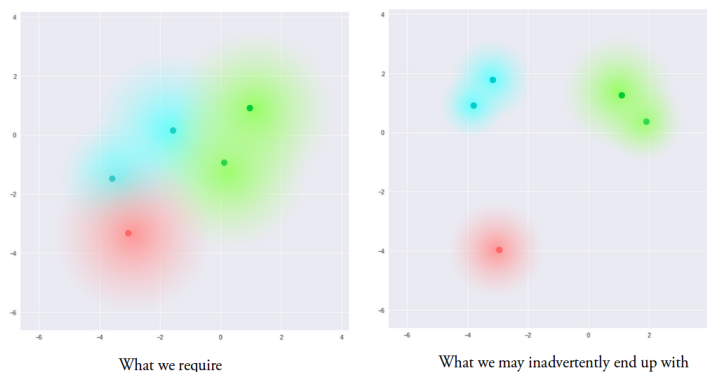
przeuczenia oraz wprowadzić pewną lokalną ciągłość w przestrzeni. Jest to wynikiem tego, że gdy kodowanie losowane jest z wnętrza kuli, notabene o nieskończonym promieniu, ale ustalonej gęstości, to dekodery zmuszony jest nauczyć się, że nie jedynie pojedynczy punkt odnosi się do danej próbki, ale i również jego całe lokalne sąsiedztwo. To pozwala dekodować nie tylko specyficzne reprezentacje (pozostawiając przestrzeń nieciągłą), ale też te trochę różniące się, ponieważ model narażony jest na szereg zmian kodowania tego samego wejścia podczas treningu.



Rysunek 1.4

Model jest teraz wystawiony na pewien stopień lokalnej zmienności przez wprowadzenie szumu do kodowania pojedynczej próbki, co skutkuje ciągłą przestrzenią w reprezentacji ukrytej dla lokalnego sąsiedztwa. Jednak na razie nie ma ograniczeń co do wartości jakie mogą być przyjmowane przez wektory μ i σ , a w rezultacie enkoder może nauczyć się generować bardzo różne μ dla różnych klas i grupować je minimalizując σ , upewniając się przy tym, że samo kodowanie nie różni się zbyt wiele dla tej samej próbki. W szczególności prawdopodobnym jest wystąpienie tak patologicznej sytuacji, gdzie σ zawsze wynosi 0, przez co nowy model nie będzie różnił się w działaniu od jego podstawowej wersji, włącznie z uwzględnieniem wad, które starał się wyeliminować. Omawiany przeze mnie przykład przedstawiony jest na rysunku 1.5. Oczekiwana sytuacja byłoby rozłożenie danych blisko siebie, ale z zachowaniem ich separowalności. Pozwoliłoby to na gładką interpolację pomiędzy próbkami i generowanie nowych danych.

W celu osiągnięcia zamierzonych rezultatów, należy wzbogacić funkcję straty o koszt dywergencji Kullbacka-Leiblera, która mierzy różnicę pomiędzy dwoma rozkładami prawdopodobieństwa. Minimalizowanie go oznacza optymalizowanie parametrów rozkładu (μ i σ), tak żeby jak najbardziej przypominał docelowy rozkład, który w przypadku autoenkodera wariacyjnego będzie standardowym rozkładem normalnym ($\mu = 0$, $\sigma = 1$). Przy takim wyborze dane powinny zostać rozrzucone dookoła



Rysunek 1.5

$\vec{0}$ i nie pozostawiać pustych miejsc w przestrzeni ukrytej.

$$\text{KLD} = \sum_{i=1}^n \sigma_i^2 + \mu_i^2 - \log(\sigma_i) - 1 \quad (1.2)$$

1.5.1. Opis matematyczny

To co do tej pory napisałem odnośnie autoenkodera wariacyjnego raczej trzeba uznać za zbiór intuicji i oczekiwań. W tym rozdziale chciałbym pokazać jego funkcjonowanie od strony matematycznej.

Dokończyć matkę

1.6. Ocena jakości modelu

1.6.1. Krzywa ROC i AUC

Krzywa ROC (Receiver Operating Characteristic) jest narzędziem do oceny poprawności klasyfikatora binarnego. Bazuje ona na wyliczaniu charakterystyki jakościowej modelu predykcji w wielu różnych punktach odcięcia. Z praktycznego punktu widzenia działa to na takiej zasadzie, że mamy dane pochodzące z dwóch klas. Model odpowiada z jaką pewnością dane należą do klasy 1. Następnie badamy różne progi i klasyfikujemy obiekty (jeśli przewidziana wartość jest powyżej progu to 1 wpp 0). Dla uzyskanych klasyfikacji liczymy TPR oraz FPR i nanosimy te wartości na wykres. Warto zauważyć, że dla losowego modelu jego wykres to prosta przechodząca przez $(0, 0)$ i $(1, 1)$. Dzieje się tak, ponieważ w przy każdym progu połowa danych będzie nad i połowa pod. Idealny model znajduje się w punkcie $(0, 1)$.

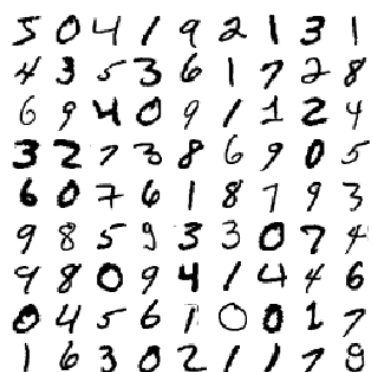
Przydatne jest również obliczenie pola powierzchni pod krzywą AUC (Area Under Curve). Interpretuje się ją jako prawdopodobieństwo, że badany model pre-

dykcyjny oceni wyżej losowy element klasy pozytywnej od losowego elementu klasy negatywnej.

Rozdział 2.

Eksperymenty na danych syntetycznych (MNIST)

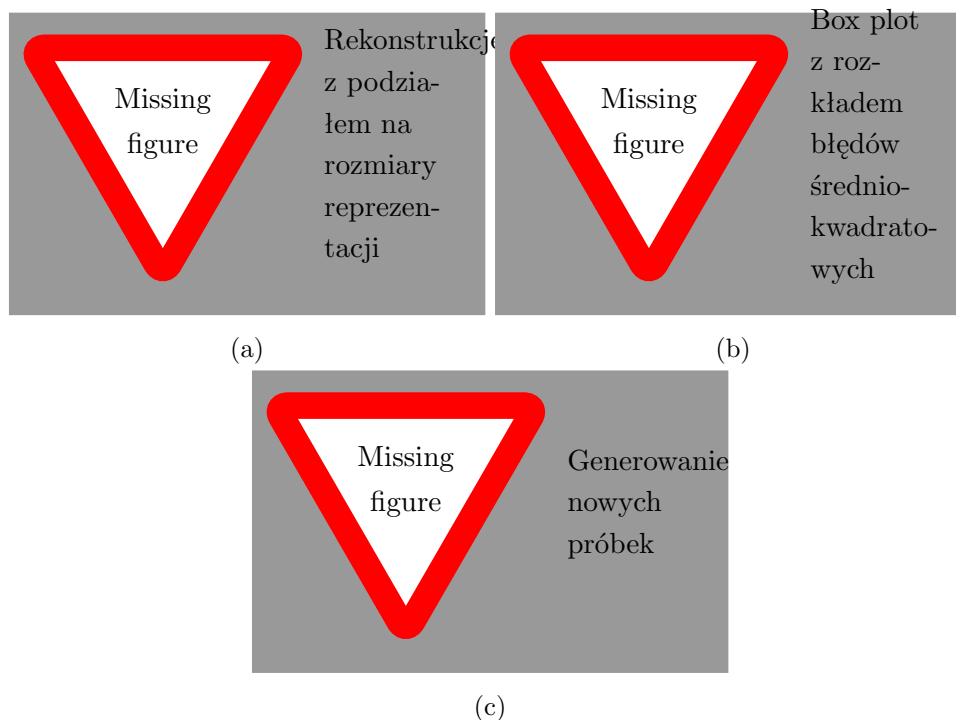
Do przetestowania pomysłu posłuży mi zbiór MNIST. Jest to zestaw odręcznie napisanych cyfr, które następnie zostały znormalizowane względem rozmiaru i wycentrowane. Każdy obrazek ma rozmiar 28x28 pikseli, mających wartości całkowite z przedziału $[0, 255]$. Na zbiór składa się 60 tys. danych treningowych i 10 tys. testowych. Przykładowe obrazki znajdują się na rysunku 2.1.



Rysunek 2.1: Przykładowe dane ze zbioru MNIST

2.1. Możliwości autoenkodera wariacyjnego

Chciałbym zacząć od sprawdzenia podstawowych własności autoenkodera wariacyjnego, przedstawionych w rozdziale 1.5. W tym celu wytrenowałem modele o różnych rozmiarach reprezentacji ukrytej: 2, 5, 7, 20, 50. Następnie sprawdziłem jak radzą sobie z rekonstrukcją danych ze zbioru testowego oraz generowaniem nowych próbek. Wyniki prezentują się na rysunku 2.2.



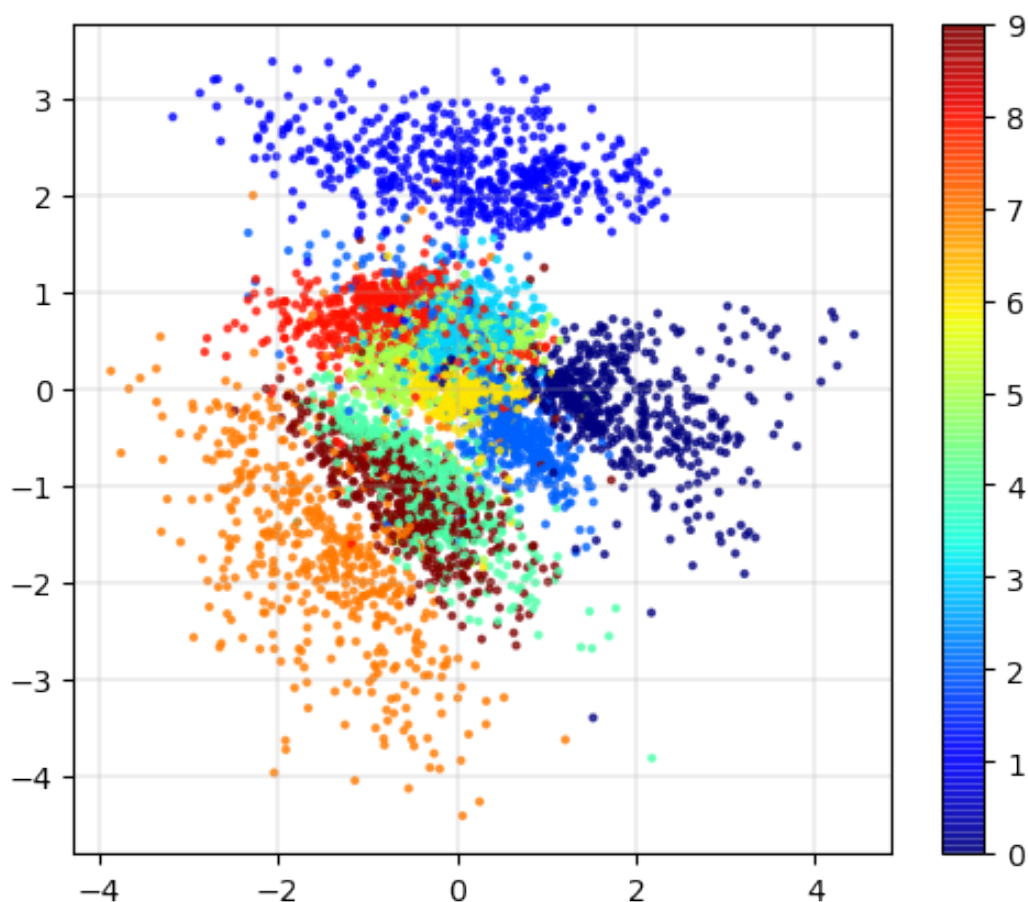
Rysunek 2.2

Ze względu na łatwość wizualizacji, do sprawdzenia jak dane reprezentowane są w przestrzeni ukrytej wybrałem model bazujący na wymiarze rozmiaru 2. Zdaję sobie sprawę, że rekonstrukcje w tym przypadku nie są najlepszej jakości, ze względu na zbyt ograniczoną ilość informacji możliwych do przekazania, ale zależy mi też na przedstawieniu podstawowych założeń. W tym celu na wykresie dla losowych próbek zaznaczyłem odpowiadające im średnie oraz odchylenia. Otrzymany rezultat widać na obrazku 2.3, gdzie można zauważyć opisane wcześniej własności wybranego modelu takie jak skoncentrowanie reprezentacji wokół $\vec{0}$ czy ciągłość przestrzeni.

2.2. Symulacja docelowego problemu

W tym miejscu interesuje mnie odtworzenie oryginalnego problemu na prostszych danych, jakim jest właśnie zbiór MNIST. Pozwoli mi to stwierdzić sensowność skorzystania z wybranego modelu. Zakładam, że jeżeli eksperyment nie powiódłby się na łatwiejszych danych, to raczej nie powiedzie się również na tych bardziej skomplikowanych. Dodatkowo zyskam pewnego rodzaju doświadczenie z wybraną architekturą.

Sytuacją, którą będę chciał poddać testowi jest całkowite niezbalansowanie danych, a wręcz brak danych drugiej kategorii i obserwacja jak zachowuje się w takim przypadku model. Będę chciał to osiągnąć poprzez uczenie modelu jedynie na danych pochodzących z dwóch klas [4, 7], a następnie obserwacja tego co się stanie z mode-



Rysunek 2.3

lem przy zaaplikowania danych z klasy [5]. Ma to odtworzyć sytuację gdzie danych opisujących zmiany nowotworowe jest przytłaczająco mniej od tych zdrowych.

W oryginalnym problemie mamy bardzo duży dysonans pomiędzy ilością przykładów dla każdej z klas. Według statystyk dane z komórkami rakowymi stanowią 1% wszystkich. Ciężko jest więc nawet mówić o jakimś sensownym podejściu supervised. Do zasymulowania tego problemu dla MNIST będę uczył model jedynie na dwóch klasach [4, 7], a następnie testował zachowanie reprezentacji ukrytej dla przykładów z klasy 5.

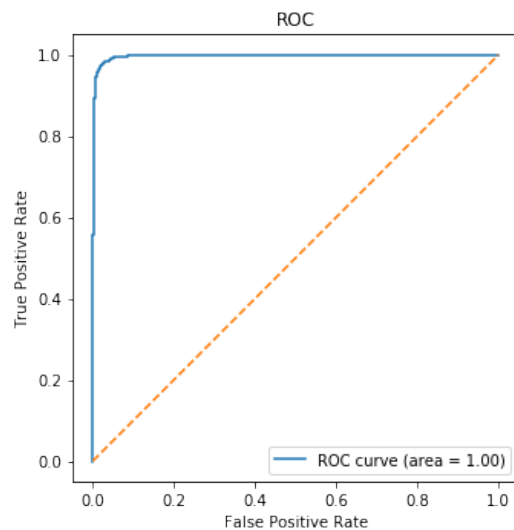
Rozmiar reprezentacji ukrytej wynosi 10. Analizować natomiast będziemy 2 składowe koszty dla modelu VAE: KLD oraz błąd rekonsrukcji MSE. Na rysunku 2.4 znajduje się przedstawienie ich wraz z histogramami. Można zauważyć, że wyłącznie na podstawie samego błędu rekonstrukcji można z bardzo dużą dokładnością określić dane pochodzące z klasy 5, mimo iż model nie widział żadnych ich przykładów podczas uczenia.

Do określenia jak rzeczywiście dobra jest ta separacja można wykorzystać krzywą ROC. Traktujemy to jako problem binarnej klasyfikacji, gdzie dane z klasy 5 będą



Rysunek 2.4

oznaczone jako 1, a z $[4, 7]$ jako 0. Wartość confidence to suma kosztów KLD i MSE. Jak widać na rysunku 2.5 takie podejście osiąga prawie 100% skuteczność. Podobne podejście będę chciał zastosować do danych medycznych.



Rysunek 2.5

2.3. Wnioski

Na podstawie otrzymanych wyników mogę stwierdzić, że eksperyment się powiódł. W przypadku danych MNIST udało się sklasyfikować dane, których wcześniej nie widzieliśmy. Jest sens w takim razie w użycie tego modelu dla już prawdziwych danych.

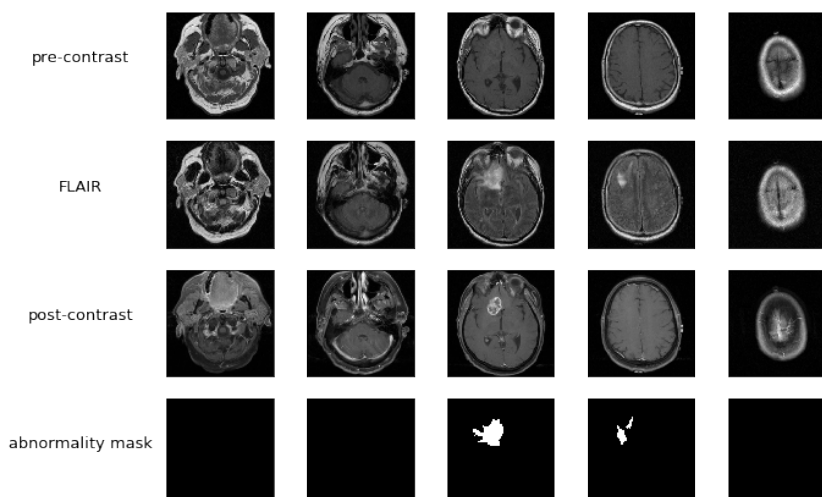
Rozdział 3.

Experiments on MRI FLAIR images

3.1. Data description & preprocessing

3.1.1. Overview

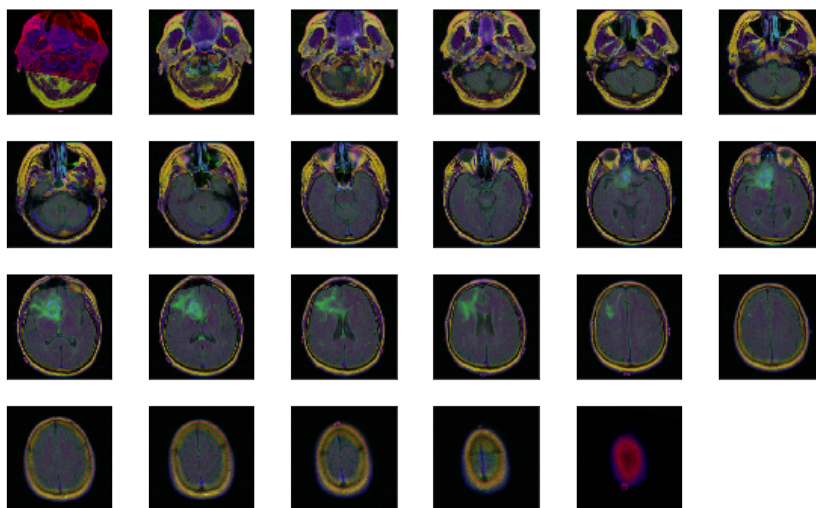
Dane pochodzą z Duke University. Są to zdjęcia z rezonansu magnetycznego głowy wraz z zaznaczonymi obszarami zmian nowotworowych. Obrazy są rozmiaru 256x256x3. Pierwszy kanał jest zdjęciem wykonanym przed wprowadzeniem kontrastu, drugi w trakcie, a trzeci po. Maski mają rozmiar 256x256, a każdy piksel ma wartość 0 albo 255 (255 oznacza komórkę nowotworową). Na rysunku 3.1 pokazane są osobne kanały oraz maska.



Rysunek 3.1

Dane pogrupowane są dla 110 pacjentów. Zestaw dla jednego z nich znajduje się na rysunku 3.2. Łącznie znajduje się 3929 par obrazów, z czego w 1373 jest przynaj-

mniej jedna komórka nowotworowa ($\sim 34.95\%$). Natomiast w całym zbiorze piksele oznaczające zmiany nowotworowe stanowią jedynie $\sim 1.02988\%$ wszystkich pikseli. Przy tak znaczącej dysproporcji ciężko liczyć na rzetelne podejście supervised, dlatego będę chciał spróbować podejścia unsupervised. Założeniem wykorzystania VAE będzie patrzenie na koszty. Hipotezą jest to, że obrazki z rakiem, czyli takie mało prawdopodobne będą zauważalne w rozkładzie i rekonstrukcji.



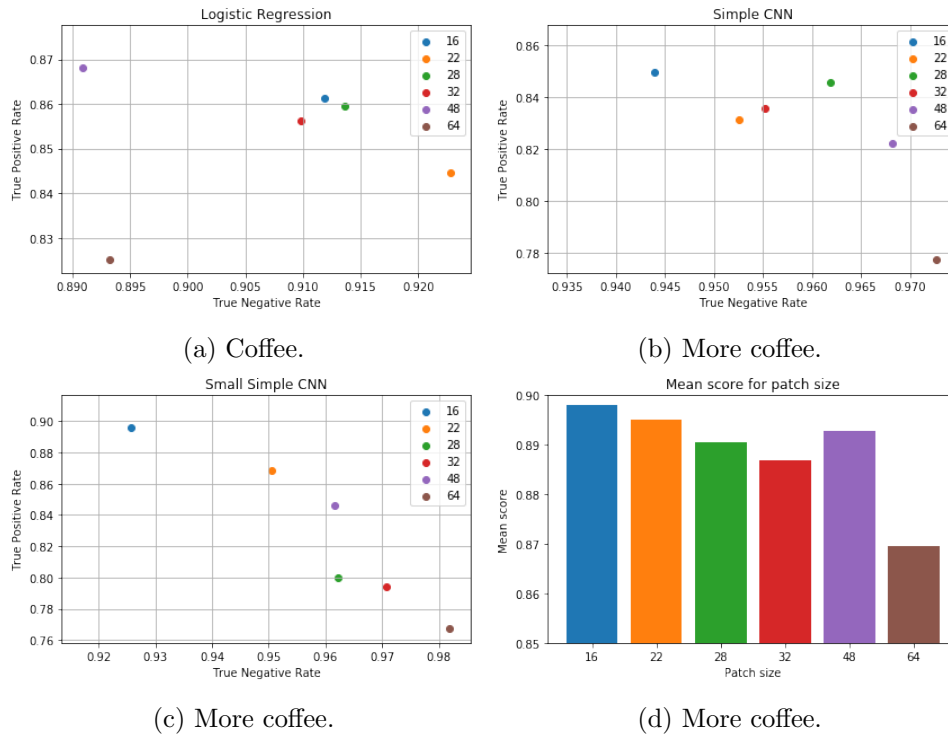
Rysunek 3.2

Pacjenci zostali podzieleni na 2 zbiory: treningowy (70%), testowy (30%). Obrazki oznaczające komórki rakowe będę nazywał pozytywnymi, a te bez negatywnymi.

3.1.2. Patches

Rekonstrukcje nie chcemy robić od razu z całego obrazku, a jedynie z niewielkich wycinków. Rozmiar wycinków ustale na podstawie wyników osiąganych przez modele supervised, tj. regresja liniowa, małe cnn (mnist), cnn (cifar10). Będę testował rozmiary 16, 22, 28, 32, 48, 64.

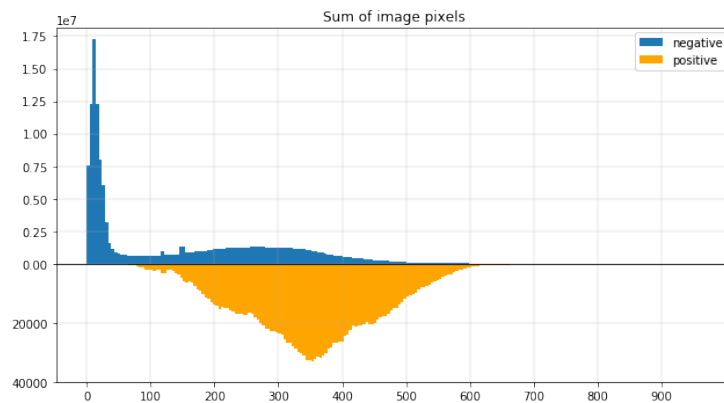
Na obrazku 3.3 widać wyniki dla poszczególnych modeli i rozmiarów. Patrzymy na 2 wartości true positive rate i true negative rate. Najlepszy przypadek jest w takim razie w punkcie (1, 1). Obrazek (d) prezentuje średnie wyniki dla każdego z rozmiarów. Przy takim liczeniu najlepsze wyniki wyszły dla rozmiarów odpowiednio 16 i 22. W dalszych eksperymentach będę korzystał z pacy o rozmiarze 22 ze względu na trochę większe pole obserwacji.



Rysunek 3.3: The same cup of coffee. Two times.

3.1.3. Preprocessing

Na obrazku 3.4 przedstawiony jest histogram zsumowanych wartości pikseli dla każdego patchu rozmiaru 22 z podziałem na klasy. Minimalna wartość sumy dla klasy pozytywnej wynosi 56.5. Oznacza to, że ze zbioru treningowego można usunąć obrazki o mniejszej sumie, gdyż automatycznie są to negatywne obrazki. Usunięcie obrazków o wartości mniejszej niż 42 zmniejszy dane o 45.5% i usunięcie trywialnych przypadków czyli prawie całych czarnych obrazków.



Rysunek 3.4

3.1.4. Normalizations

IME?

Rozważyć normalizacje: histogramowa, tylko środkowy kanał

3.2. Results

Przetestuje 4 kombinacje modeli, ze zmienioną funkcją kosztu

crótki opis modeli i
la czego softmax

W tabeli 3.1 zostały przedstawione wyniki dla czterech kombinacji modeli oraz parametrów. Wartości liczbowe opisują najlepsze AUC ROC podczas uczenia wraz z odpowiadającą epoką. Jak widać w tym zestawieniu najlepiej wypadły podstawowe modele VAE.

Model	Best ROC AUC
VAE	0.68 (1)
VAE + Softmax	0.68 (1)
C-VAE	0.59 (1)
C-VAE + Softmax	0.67 (1)

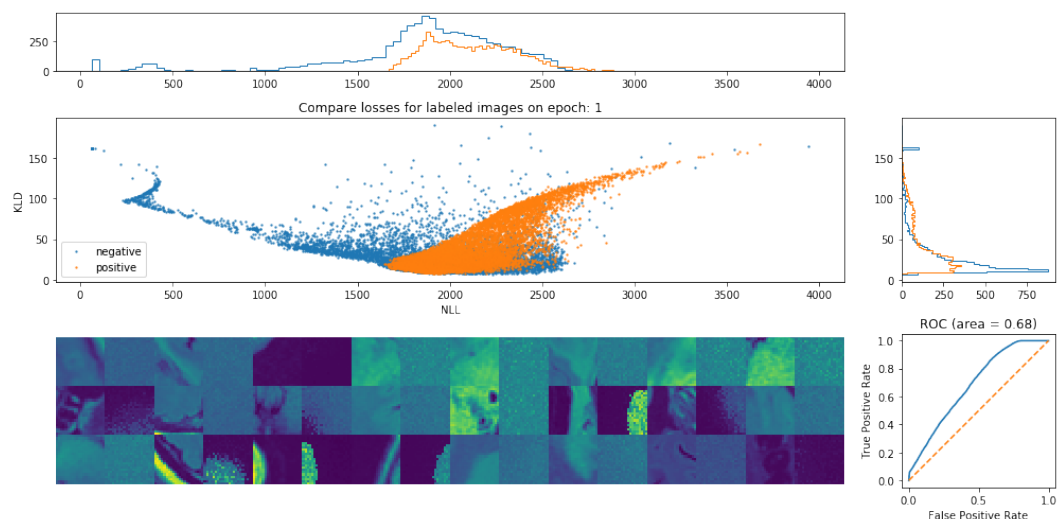
Tablica 3.1: My table

z-dim

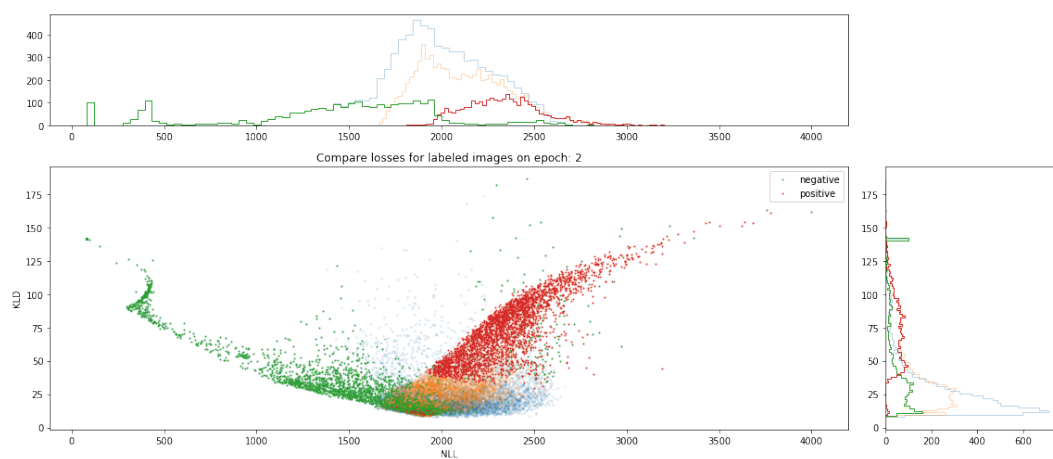
3.3. Analysis

Zajmę się analizą modelu w wersji VAE + Softmax. Na wykresie 3.5 przedstawione są koszty, ich rozkłady, przykłady rekonstrukcji oraz krzywa ROC. Ciekawie wyglądają te dwa zauważalne ogony na wykresie. Postaram się lepiej przyjrzeć ich specyfice.

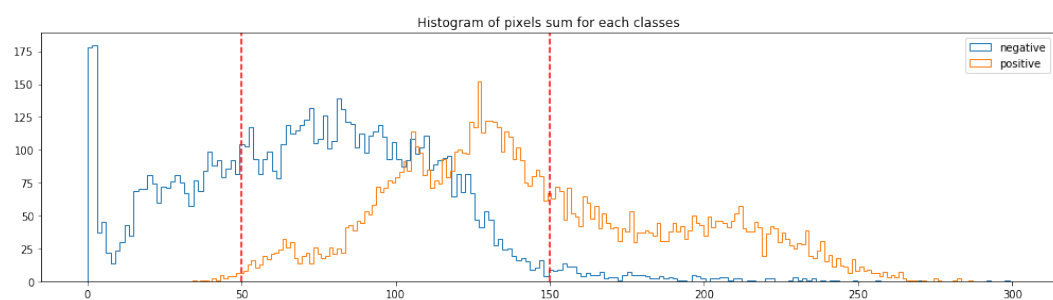
Pierwszym pomysłem jest, żeby sprawdzić rozkład obrazków w zależności od ich jasności, czyli łącznej sumy pikseli. Ciemne obrazki definiuje jako te o niskiej sumie, a jasne o wysokiej. Na rysunku 3.6 zaznaczyłem negatywne obrazki, dla których suma $\mu = 50$ oraz pozytywne obrazki z sumą co najmniej 150. Zaznaczone są one odpowiednio zielonym i czerwonym kolorem na wykresie. Jak widać po rozkładach składają się one na dokładnie te 2 separowalne ogony. Wniosek z tego jest następujący. Na podstawie tego modelu możemy odesparować jedynie najciemniejsze i najjaśniejsze obrazki z obydwu klas. W takim razie model nie robi nic nadzwyczajnego. Wystarczy przyjrzeć się histogramowi na rysunku 3.7 z zaznaczonymi progami. One już bardzo dobrze separują te klasy.



Rysunek 3.5

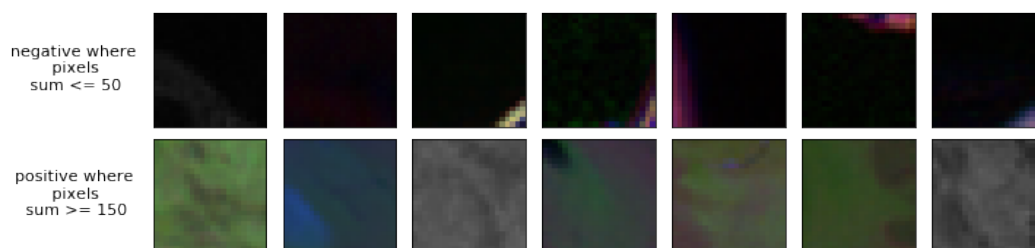


Rysunek 3.6



Rysunek 3.7

Na rysunku 3.8 pokazane są przykłady obrazków z zadanymi poziomami sumy pikseli.



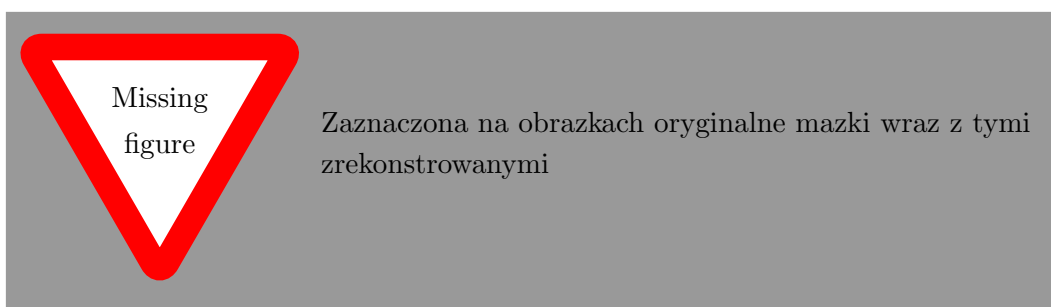
Rysunek 3.8

3.3.1. Conclusion

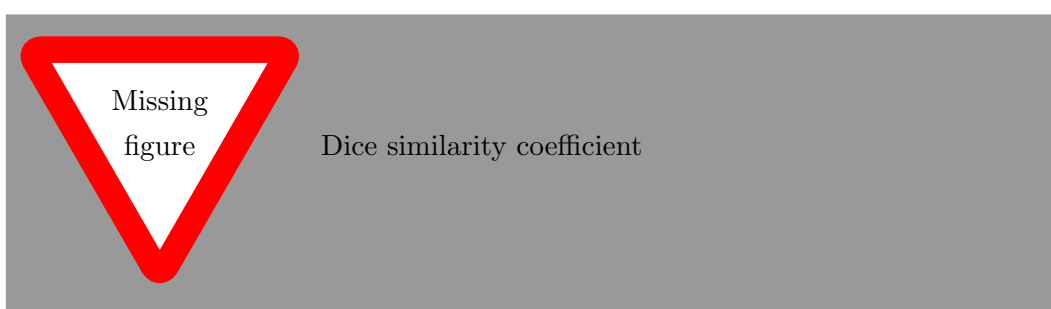
Niestety podejście z użyciem VAE nie zadziałało.

3.4. Back to basic models

3.4.1. Logistic Regression



Rysunek 3.9



Rysunek 3.10

3.4.2. CNN

Rozdział 4.

Summary

Koniec