WB DL-1 HW3

Piotr Piątyszek, Adam Frej, Piotr Marciniak

1 kwietnia 2021

1 Zadanie 1

De facto, jeśli chodzi o przekształcanie obrazu takie jak było pokazane na zajęciach (czyli zmiana histogramu) takiego preprocessingu nie zastosowano.

"Two experienced radiologists from INSEL annotated or reannotated ILD typical pathological patterns, as well as healthy tissue in both databases. A lung field segmentation mask was also provided for each case."

Jedyne co można powiedzieć o zdjęciach to, że zostały one opisane przez dwóch radiologów w odpowiedni sposób. Do opisania zdjęć radiolodzy wykorzystali **itk-Snap**. Dodatkowo każde zdjęcie dostało własną maskę, które nie opisywało całej tomografii komputerowej, tylko typowe objawy ILD wzorów. Nie opisane fragmenty płuc były wyłączone zarówno z uczenia jak i późniejszej ewaluacji.

Poniższy fragment można uznać za preprocessing danych.
"In order to artificially increase the volume of training data and avoid overfitting, we transformed the images using flips and rotations, which are considered label-preserving in this domain. The augmentation was performed online i.e. for each training image in each epoch, one operation out of all eight combinations of flip and rotate is randomly selected and applied."
Aby zwiększyć objętość danych i uniknąć przetrenowania sieci (high variance) dla każdego zdjęcia ze zbioru była wykonywana jedna z ośmiu kombinacji obrotu i rotacji. Kod zaprezentowano poniżej

```
def sample_generator(db_path, augment=True):
    db = np.load(db_path)['db'][()]
    X = db['X']
    Y = db['Y']

    numClasses = Y[0][0].shape[-1]
    flip = [(lambda x: x), np.fliplr]
    rotate = [partial(np.rot90, k=0), partial(np.rot90, k=1),
              partial(np.rot90, k=2), partial(np.rot90, k=3)]
    augmentations = list(itertools.product(flip, rotate))

    while 1:
        # suffling samples
        idxs = np.random.permutation(len(X))
        for rs in idxs: # looping over shuffled samples
            if augment:
                # selecting augmentation
                aug = augmentations[np.random.randint(len(augmentations))]

                # flip rotate augmentation
                x = aug[1](aug[0](np.squeeze(X[rs])))
                y = aug[1](aug[0](np.squeeze(Y[rs])))

            else:
                x = np.squeeze(X[rs])
                y = np.squeeze(Y[rs])
```

```
if x.ndim == 2:
    x = x[:,:,None]

yield (x[None,:,:,:], y[None,:,:,:])
```

"Specifically, the 172 scans of the dataset were divided into five non-overlapping sets, with one of them having 36 and the rest 34 scans. Every time a model was tested on a specific set, the rest of the data were used for training. On average over all folds, the number of slices was 2060 for training and 515 for testing." Co więcej skany zostały podzielone na mniejsze kawałki.

"To minimize the number of computations and memory requirements, we discarded part of the data that lack annotations. Hence, we cropped the left and right lung on each slice and used only the ones with relevant annotations as inputs of the networks. This is permitted by the fully-convolutional nature of the tested networks that do not require a fixed input size. For the cropping, we utilized the available lung mask, while a margin of 32 pixels was added on each side to provide context that could be useful to the networks".

W tej części autorzy usunęli dane bez opisów. Później przycięli zdjęcia, tak aby skupić się wyłącznie na fragmentach prezentujących opisane płuca. Dla zachowania kontekstu zostawiono 32 piksele marginesu.

2 Zadanie 2

Do preprocessingu użyliśmy zbioru LIDC, ponieważ dostarczona do niego biblioteka ułatwia operacje na nim. Rozważyliśmy dwie metody preprocessingu wyrównanie histogramu i pocięcie obrazków w zbiorze treningowym tak, żeby były to guzy wraz z marginesem.
Dodatkowo na zbiorze covidowym wykonaliśmy trzeci preprocessing. Zwiększyliśmy kontrast zdjęć.
Analiza wyników znajduje się w notebooku.

3 Zadanie 3

3.1 Treść

Sprawdzić, jaki preprocessing jest często stosowany do klasyfikacji zdjęć histopatologicznych.

3.2 Nasza odpowiedź

Jest wiele różnych metod preprocessingu stosowanych do klasyfikacji zdjęć histopatologicznych:

- Takie zdjęcia są ogromne i zawierają wiele szczegółów. Z tego powodu nie można po prostu zmniejszyć ich rozdzielczości, bo grozi to utratą informacji. Dlatego dzieli się je na mniejsze porcje ("patches") o rozmiarze typowo 256 x 256 pikseli. Każdy fragment jest osobno analizowany. Współcześnie dzięki polepszeniu mocy obliczeniowej rozmiar się zwiększa do nawet 960 x 960.
- Dużym problemem jest brak opisów do zdjęć. Ciężko jest znaleźć wystarczająco dużo danych. Z tego powodu próbuje się albo opisywać posiadane dane, albo pozbywać się nieopisanych zdjęć.
- Zdjęcia histopatologiczne przypominają raczej teksturę niż konkretne obiekty. Aby je dostosować stosuje się preprocessing: gray level co-occurrence matrix, local binary pattern, Gabor filter bank, and deep texture representations using a CNN.
- Zdjęcia histopatologiczne są robione pod mikroskopem. Stwarza to wiele problemów. Fotografowane obiekty mogą być zakurzone albo przekrzywione. Niektóre fragmenty zdjęć są oznaczane markerami. Dlatego stosuje się algorytmy, które wykrywają takie zmiany i odpowiednio modifikują zdjęcia.
- Zdjęcia bardzo często mają różne kolory. Z tego powodu zwykle konwertuje się skalę barw. Konwersja do skali szarości grozi utratą informacji, więc raczej się jej nie stosuje. Dlatego często zdjęcia są modyfikowane piksel po pikselu, żeby je ujednolicić do jednej skali. Generalnie stosuje się kombinację color normalization i color augmentation.

Źródło: tutaj

4 Zadanie 4

4.1 Treść

Czy można karmić sieć obrazkami prostokątnymi, a nie kwadratowymi? Co trzeba zrobić, aby to móc zrobić. Jaka była największa rozdzielczość obrazków jakimi karmiona sieć (dla danych niemedycznych i medycznych)?

4.2 Nasza odpowiedź

Oczywiście sieć można karmić obrazkami prostokątnymi, aby to uczynić wystarczy w warstwie input ustawić size na odpowiednio wartości (wymiary prostokątne zdjęcia, można podać także None - dowolny wymiar zdjęcia).

```
keras.Input(shape=(320, 480, 3)) #zdjecie RGB 320x480
```

Kwadratowe obrazki stosuje się tylko dla wygody, jest to kwestia przyzwyczajenia. Większość sieci jest już przystosowana do kwadratowych obrazków. Łatwiej jest zrobić preprocessing, aby zmienić kształt zdjęcia niż przeprojektowywać sieć. Dawniej sieci wymagały, aby wszystkie zdjęcia miały taki sam rozmiar, a to dodatkowo ograniczało zmiany na tym polu. Preprocessing zdjęcia można zrobić w sposób

- Resize - stratna kompresja

```
from PIL import Image

im = Image.open("hopper.jpg")

# Provide the target width and height of the image
(width, height) = (im.width // 2, im.height // 2)
im_resized = im.resize((width, height))
```

- Crop - ucięcia fragmentu zdjęcia, bądź window sliding, czyli przesuwanie się po zdjęciu kwadratową ramą.

```
from PIL import Image

im = Image.open("hopper.jpg")

# The crop method from the Image module takes four coordinates as input.
# The right can also be represented as (left+width)
# and lower can be represented as (upper+height).
(left, upper, right, lower) = (20, 20, 100, 100)

# Here the image "im" is cropped and assigned to new variable im_crop
im_crop = im.crop((left, upper, right, lower))
```

Źródła do kodów

Oczywiście największym bottleneckiem przy stosowaniu dużych zdjęć jest rozmiar pamięci GPU. Największe zdjęcia użyte w zastosowaniu medycznym, jakie udało się na nam znaleźć jest to CT skan o rozdzielczości $512 \times 512 \times 512$, czyli ponad 134217728 pikseli. Do osiągnięcia takich wyników użyli oni kilku GPU/TPU. Według wiedzy autorów na obecnych pojedynczych GPU/TPU nie da się przetwarzać zdjęć powyżej 10^8 pikseli.
Link do pracy

Największym niemedycznym zdjęciem użytym w deep learningu, które udało nam się znaleść to 6000×6000, czyli 36 mln pikseli, są to zdjęcia z lotu ptaku miejscowości Poczdam.
Źródło