Labor 18.11.2019

1) Erstellen Sie eine Generator-Klasse, welche eine **__getitem__** und eine **__len__** Funtkion aufweist. Leiten Sie die Klasse von keras.Sequence ab. Halten Sie sich an das Beispiel von Sequence unter: https://keras.io/utils/. Eine Vorlage für Ihre Klasse könnte wie folgt aussehen:

```
from keras import Sequence
class Generator(Sequence):


    def __init__(self,
                 x_paths,
                 y_paths,
                 batchsize,
                 scale=1/255,
                 ):
        """
            @brief Generator for iterating over dataset
            @param x_paths Paths to the x data
            @param y_paths Path to the y data
            @param batchsize Size of the batch
            @param scaling for data
        """
        self.x_paths = X
        self.y = y
        self.scale = scale
        self.batchsize = batchsize

    def __len__(self):
        # amount of epochs
        return ...

    def __getitem__(self,idx):
        """
            @brief function for returning a batch
            @param idx Batch
            @return Will return the input data, and the target
        """

        X = np.zeros(shape=(...))
        Y = np.zeros(shape=(...))

        try:

            for i in range(0,self.batchsize):

                current_idx = ....

                X[i] = cv2.imread(x_paths[current_idx])
                Y[i] = cv2.imread(y_paths[current_idx])
        except IndexError:
            print("Index error",current_idx)
            pass

        return X,Y
```

2) Vervollständigen Sie das Skript „Unet.py“ um hiermit ein „Unet“ zu trainieren. Dieses Skript liegt ebenfalls schon in Ihrem workspace vor.

3) Trainieren Sie die U-Net Architektur mit Hilfe des Generators aus Aufgabe 1 und dem zur Verfügung gestellten Nuclei Datensatz. Dieser liegt schon in Ihrem workspace vor. Anstelle eines „fit“ Aufrufes müssen Sie nun den „fit_generator“ Aufruf verwenden.

   Instanziieren Sie zwei Objekte der Generatorklasse, einen für die Trainingsdaten, einen für die Validationsdaten.

   Die Paramter können Sie aus der Doku entnehmen:
   https://keras.io/models/sequential/#fit_generator

   Denken Sie daran einen geeigneten Loss zu wählen.

4) Erweitern Sie Ihren Generator um geeignete Augmentationen. Verwenden Sie hierzu das Framework „imgaug“.

   https://imgaug.readthedocs.io/en/latest/source/examples_basics.html

   Verwenden Sie mindestens eine Augmentation, die Maske und Input-Bild verändert.
   In Ihrem Ordner finden Sie ein Skript „imgaugExample.py“ in diesem wird gezeigt wie mit der Hilfe von imgaug Masken und Bilder augmentiert werden können. Dabei werden zwei unterschiedliche Methoden aufgezeigt.

   Hinweis: Viele der Funktionen von imgaug verwenden OpenCV. OpenCV akzeptiert bei den meisten Funktionen keine Float Arrays, sondern nur den uint8-Datentyp.

Zusatz: Machen Sie sich weiter vertraut mit dem imgaug Framework, schauen Sie dabei vor allem auf Beschreibungen wie „OneOf“ oder „sometimes“.