内部出力の可視化ツール

1. プログラム名の概要とプログラムを使う順番

—tensorboardのオプションを用いて
main.pyでPredNetを使った計算を行い、runsファイルを出力させる

ext_tensorboard_imgs_tk.py

・ext_tensorboard_imgs.pyを少し編集
・runsファイルから画像出力を行うプログラム

arng_inside.py

・ext_tensorboard_imgs_tk.pyで出力されたファイルから
必要なファイルをコピーし整理するプログラム

dif_img.py

・arng_inside.pyで整理された
ファイルを格納するフォルダを参照して
差分を計算するプログラム
・arng_inside.pyと同じ形式で保存するため
そのままcombine_img.pyで動画化可能

combine_img.py

・arng_inside.pyで整理されたファイルを格納するフォルダを参照して
動画を作成するプログラム

内部出力の可視化ツール

※テキストの入力はターミナル上ではなく、
各プログラムの一番下のブロックにある
if __name__ == '__main__':以下の変数を編集によって行う

2. 各プログラムで用いる変数の説明

ext_tensorboard_imgs_tk.py

root = ‘runs' → runsファイルが入っているフォルダのベースパスを指定
savedir = ‘runsimg’ → 画像の保存先のフォルダを指定

arng_inside.py

path = ‘runsimg' → ext_tensorboard_imgs_tk.pyの出力先のフォルダパスを指定
name = ‘output_test' → 画像の保存先のフォルダパスを指定

combine_img.py

dir='20221028162846' → arng_inside.pyかdif_img.pyの出力先のフォルダを指定
onoff='on' → 出力動画に表示しているフレームの番号を記載するかどうか
fps=10 → 出力動画のフレームレート
nm=3 → モジュール数(通常PredNetの場合, 3)
nl=2 → レイヤー数(PredNetのデフォルトでは4)
testimgpath = None → テストで用いたread_list.txtのパスを指定
predictpath = None → PredNetの出力画像を格納しているフォルダのパスを指定

dif_img.py

path1 = ‘105’ → 差分される側のarng_inside.pyの出力先のフォルダパスを指定
path2 = '115'→ 差分する側のarng_inside.pyの出力先のフォルダパスを指定
rate = 1 → 表示スケールを指定
savedir = datetime.datetime.now().strftime(‘%Y%m%d%H%M%S')
→保存先のパスを指定

内部出力の可視化ツール

3. 各プログラムの補足

ext_tensorboard_imgs_tk.py

- ファイル数がとても多く、20フレームのテストでも数百MBになる
- Temsorboardの画像のため、各チャンネルの画像は各層で規格化されており、0-255の値をもつ

arng_inside.py

- 通常PredNetのデフォルト設定の場合、①のようなフォルダが作成される
  module1.. 深い層へ向かうときの畳み込み計算の出力データを格納
  module2.. 各層の入力値に対する差分計算に用いるためのデータの畳み込み計算の出力データを格納
  module3.. LSTMの最終出力データを格納

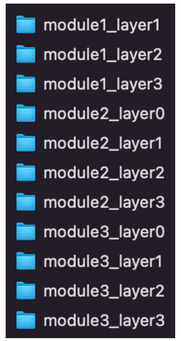
combine_img.py

- 動画内各画像の対応は②のように、左ほど浅いレイヤー、上ほど若いモジュールになっている

dif_img.py

- 保存先のフォルダは①のようになっている
- 計算結果は　(path1 - path2)/255 となっているため、値域は -1から1となっている

①



②