第２回日本眼科AI学会総会

眼科AIコンテスト解析説明

2021年11月21日

南子安眼科　古山　誠

利益相反

該当なし

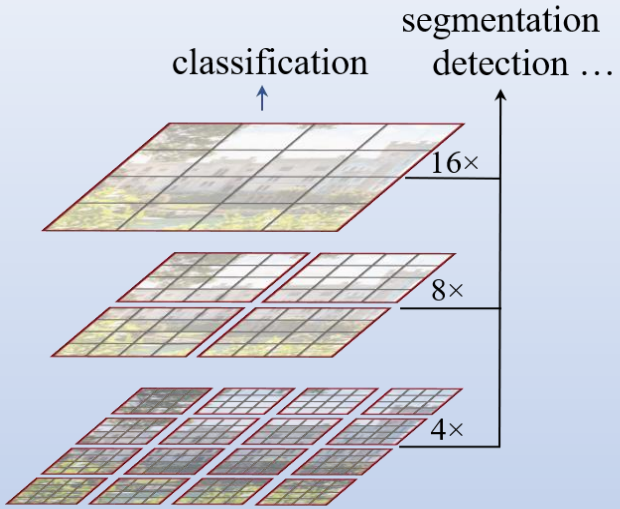
モデルについて

- モデルの構築は、いずれもPyTorchの転移学習を使用。
- timmコマンドを用いる事により、様々な学習済みモデルを容易に構築する事が可能。
- 数多くのモデルを試した結果、以下の２つのモデルを採用。複数回学習し、加重平均を算出。（後述）
- swin_base_patch4_window12_384_in22k

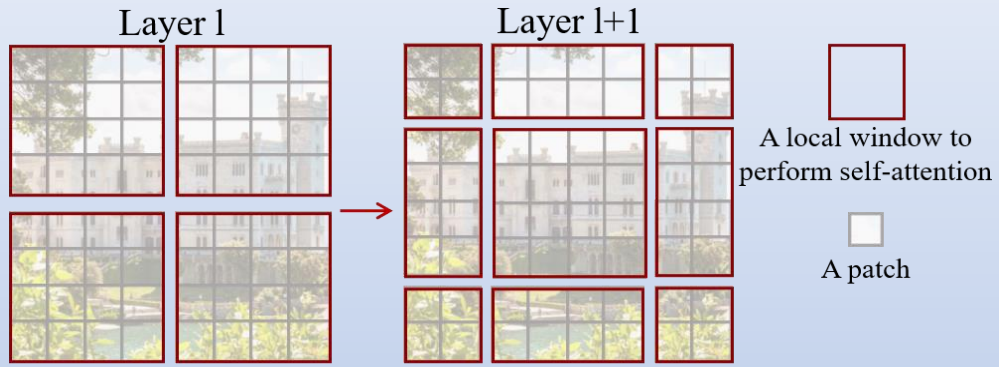
  （ Swin-Transformer、ImageNet22k pre-trained）
- tf_efficientnetb7_ns

  （Efficient Net B7、NoisyStudent pre-trained）

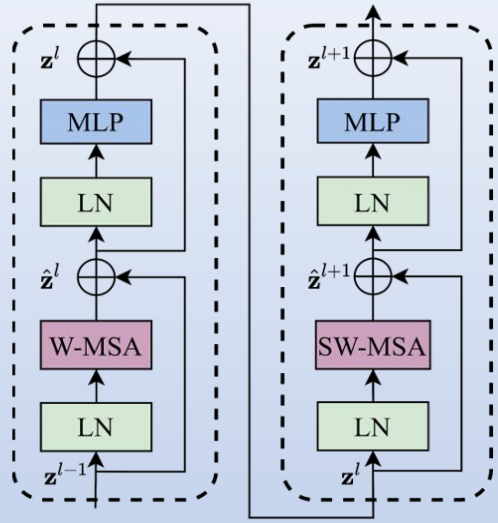
Swin-transformer



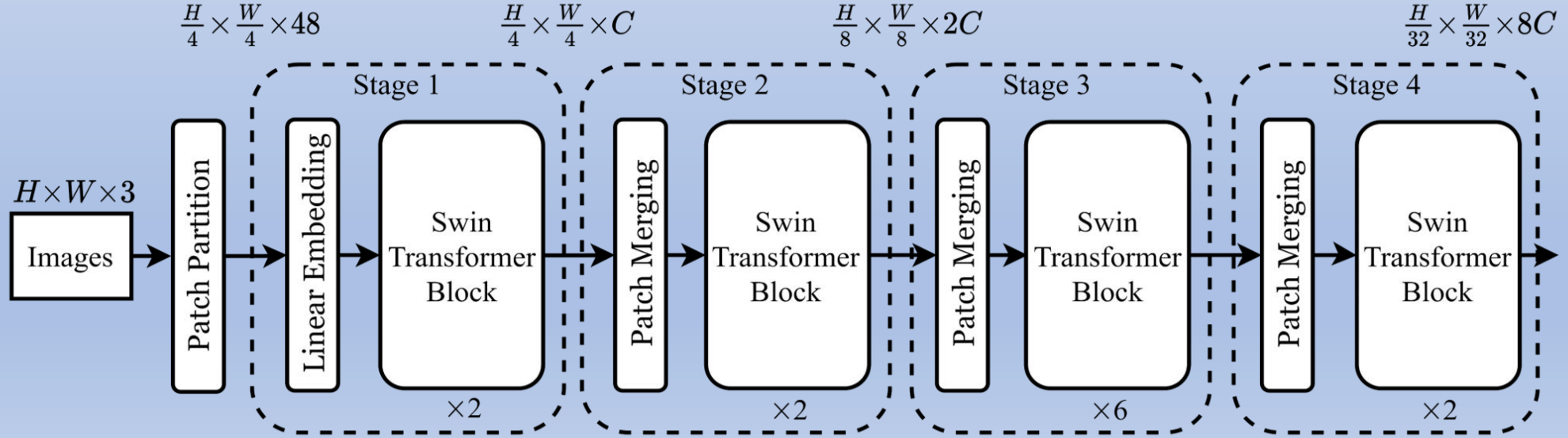
(a) Swin Transformer



(b) Shifted Window



(c) Two Successive Swin Transformer Blocks



(d) Architecture

Dataset, Data Loaderについて

- Dataset, Data LoaderはPyTorchで自分で実装。
- epoch毎にハードディスクやSSDから画像データを読み込んでリサイズをすると非常に時間がかかるため、予め画像データを中間サイズの画像にリサイズしてメモリ上に全て置いておき、そこから読み込みを行う事により高速に処理できた。

画像の処理（random crop , center crop）

- まず予め周囲の黒い部分を取り除き、正方形にし、
- その画像をそのまま使用するのではなく、学習させる画素数よりも大きめの画像からrandom crop(train時)したほうが良い結果が得られた。
- 学習させる画素数の1.15倍(efficientNet B7)～1.2倍(Swin)の画像からのcropが最も良い結果が得られた。
- Validation/test時にはcenter cropを行った。

画像の処理について（crop以外）

- Crop以外にも学習時にはRandom Horizontal Flip / Random Rotate / Random Zoom も施行。
- test時にはHorizontal Flipも併用して平均を採用。
- transformの処理はCPUよりも全てGPUで行った方が高速であった(環境による)。

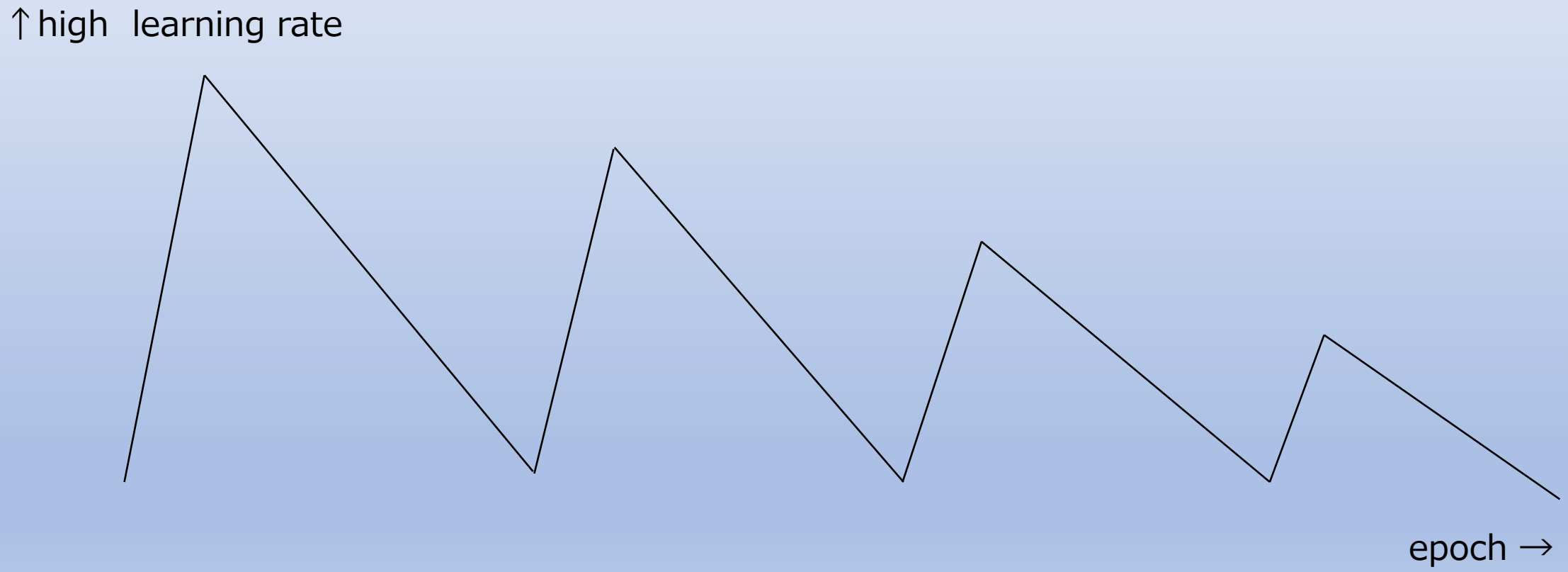
Mixed Precision（16bit混合）

- 学習速度を上げ、メモリの消費量を減らすため、PyTorchのMixed Precisionを使用。
- 16bitの計算で済むので合理的。計算結果にはほとんど影響なし。

Normalization Layerのfreeze

- Layer Normalization及びBatch Normalizationは、file-tuneによって元の学習内容が特に破壊されやすいため、layerの学習をfreeze。PyTorchでは自分で実装する必要あり。

Learning rateのイメージ図



Learning rateについて

- 最初から大きいlearning rateにすると、局所の最適解に陥ってしまい、最終的なscoreが悪くなる恐れがあるため、最初は小さいlearning rateから開始（warm-up）し、徐々に増やし、その後漸減。
- SchedulerにはCyclicLRを使用。

Early Stoppingについて

- Validationのデータをモニターし、best scoreが出た時点でmodelをRAM diskに自動保存。
- Early Stoppingによってpatientceを超えた時点で自動的にbest scoreのデータを読み出し、再度小さなlearning rateから漸増・漸減して学習継続。PyTorchでは自分で実装する必要あり。

Optimizerについて

- 様々なoptimizerを試した結果、今回はAdaBeliefが最も良い最終結果が得られた。
- EfficientNet, Swinともに、今回のモデルの最終結果は
  AdaBelief > AdamW > Adam > SGD（SAM含む）
- 但し、モデルや訓練データによって結果は異なる。
- AdaBeliefではeps=1e-16、 weight_decouple=True、 rectify=True、 weight_decay=1e-4、 betas=(0.9, 0.999)という設定が最も良い結果が出た。

Batch Sizeについて

- 今回のモデルでは、Batch sizeは４が最も良いデータが得られる事がある反面、ばらつきが大きい。
- Batch sizeを８以上にすると安定する反面、良いデータが得られにくい。
- Batch size４を複数回施行し、bestなデータを取得する事で解決。
- Batch sizeも、モデルや訓練データによって最適値は異なる。

Head(最終出力)について

- head（最終出力）部分は、その学習内容に依存している為に取り替えた。
- 単純にモデルからの出力を直接1にするよりも、出力をGELUで256 → 64 → 8 → 1等のように漸減した方が良い結果が得られた。
- 画像の分類問題であれば出力を分類の数に直接固定するのが一般的だが、今回のモデルの場合は複数の出力からの様々な要素を複合したほうが年齢を推定しやすいという事なのではと考えた。

Fine-tune前のheadのみの学習

- Headを取り替えた後、そのまま普通にfine-tuningしてしまうと転移学習の元の学習内容が破壊される恐れがある。
- まず本体の学習を全てfreezeし、取り替えた本体のheadの部分のみを学習させた。
- その後本体のfreezeを解除してfine-tune。

Train / validation / testの振り分け

- 今回の提供して頂いた画像には、同じ患者さんの同一眼のデータが複数あるため、そのままtrain / validationにランダムに振り分けた場合、同じ患者さんの画像がtrainとvalidationに同時に存在して学習効果の判定に不具合が生じてしまう恐れがある。
- そのため、validationに採用した患者さんの同一眼はtrainに採用しないようにした。
- 本来はtrain / validation / testと分けるのが理想だが、今回はサンプル数の関係でvalidationとtestは兼用とした。

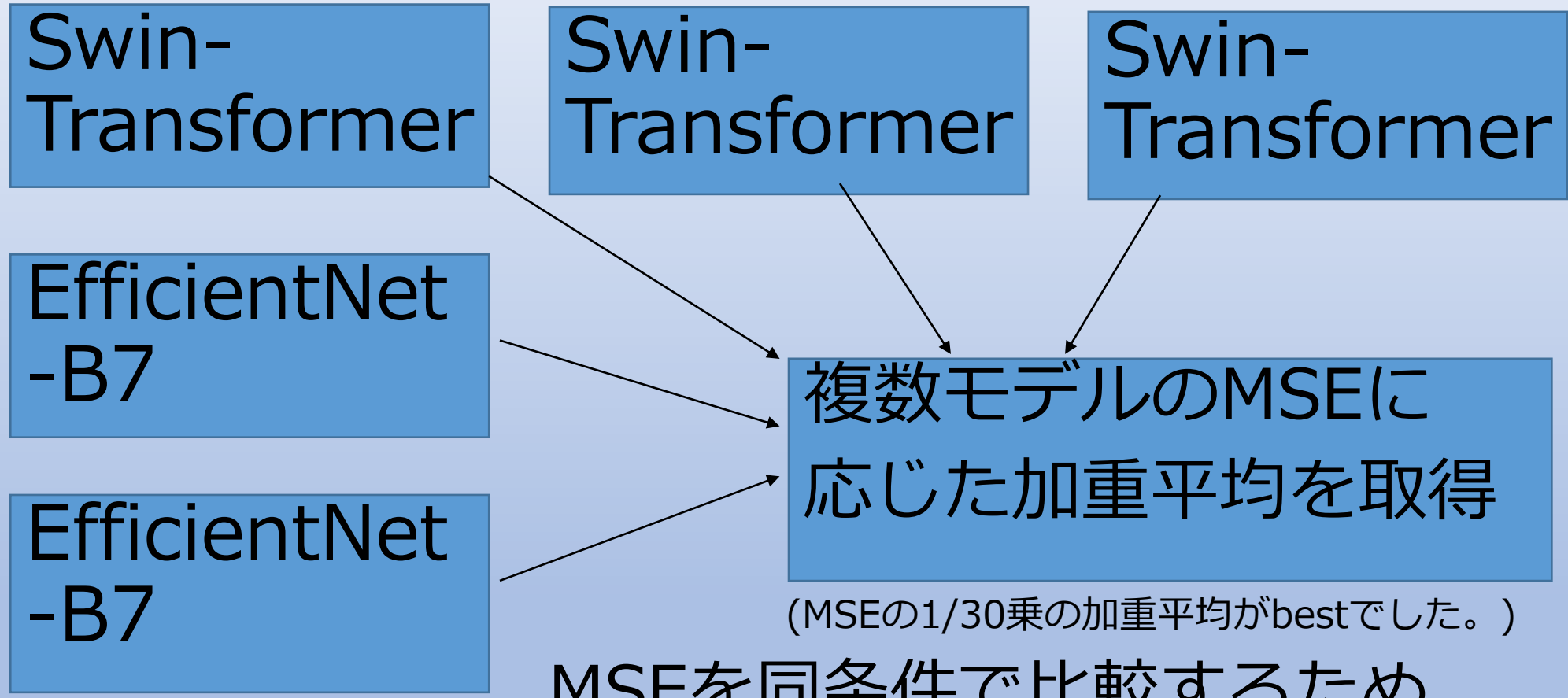
Horizontal Flipを行い平均値を算出

test/result時、左右を反転して、
得られたageの平均値を算出。

同一IDの平均値を算出

test/result時、同一のIDの写真が複数枚あった場合、得られたデータの平均値を算出。

複数のモデルの加重平均を算出



(MSEの1/30乗の加重平均がbestでした。)

MSEを同条件で比較するため、
Train / validationは固定して学習。

train / validationを再度割当

複数モデルのMSE
に応じた加重平均

複数モデルのMSE
に応じた加重平均

平均値

最終出力

最後に

- このような学習の機会を与えて頂きました、日本眼科AI眼科学会及び関係者の皆様に深く感謝を致します。
- まだ機械学習の勉強を始めたばかりですので、今後Kaggleへの参加や、数学の勉強、OCTの画像等からのオリジナルの学習等、新たなチャレンジを続けて行きたいと思います。