進捗報告

1 今週やったこと

- FastAutoAugment への SGA の適用
- Augment 周りの論文調査

2 FastAutoAugment への SGA の適用

FastAutoAugment のレポジトリのベイズ最適化の部分を SGA に置き換える実装をしていました．まだデバッグ途中なので，SGA ができ次第，TDGA に移る．

また，使ってる FastAutoAugment が元々公式ではなく個人のレポの中で使いやすそうなのをピックアップしたので，論文中の精度が微妙に出ていないことに気づいた．なのでこのレポではいったん SOTA は気にせずにベイズアルゴリズムの部分を GA に置き換えることの影響を見て，そのあとに GA を使った新しい手法を考える．

3 Augment 周りの論文調査

Augment 研究周りを少し整理．現在主流の画像 Augment 手法としては以下の 4 つに大別できる．

- AutoAugment(2018/05/24)
- Fast AutoAugment(2019/05/01)
- Population Based Augmentation(2019/05/14)
- Rand Augment(2019/09/30)

まだ下の 2 つはちゃんと読めてないが，軽くそれぞれの特徴を紹介．

3.1 AutoAugment

AutoML 分野の Data Augmentation への適用の基盤となる論文．デフォルトの拡張の組み合わせをポリシーと呼び，ポリシーの探索に RNN を用いている．探索されたポリシーにより，多くのデータセットにおいて SOTA を達成した．高速化のために小さいデータセットで学習後，大きなデータセットの学習の際に転移学習をする工夫をしているが，cifer-10 でも 5000GPU 時間と膨大な時間がかかるのが難点．公式レポは TensorFlow 実装．

3.2 Fast AutoAugment

AutoAugment のアイデアを用いて高速化した手法．ポリシーの発想はそのままに，最適化の部分に RNN ではなくベイズアルゴリズムを用い，さらに密度マッチングの考え方を用いてデータセットを分割し，評価にかかる時間を大幅に減らすことにより，精度を保ちながら AutoAugment の 1000 倍以上の高速化に成功．公式レポは PyTorch 実装．

3.3 Population Based Augmentation(PBA)

Population Based Augmentation [1]

PBA も FastAutoAugment と同時期に発表された，AutoAugment の計算コストを削減しようという試みの内容の論文．アイデアの核の部分は Data Augmentation の適用もパラメータの一種と考え、PBT [2] のような進化戦略を用いて良好な結果を出したモデル/Augmentation を残していく形を取っている。モデルのパラメータが持ち越されるため再計算の必要がなく，探索空間自体は AutoAugment よりも広いものの，精度を保ちながら AutoAugment の 1000 倍以上の高速化に成功．公式レポは TensorFlow 実装.

3.4 Rand Augment

Rand Augment [3]．画像系の人はぜひみんな読んでみてほしい.

従来の AutoAugment 系の研究にはデータセットが大きいと膨大な時間がかかるという問題と，評価用の探索空間とテストの探索空間が必ずしも一致するわけではないという問題点があり，Rand Augment はこれら 2 つの問題点を解決した．まさかのパラメータをグリッドサーチで全探索する超単純な手法が有効というのは驚くべき結果．下に RandAugment のソースを示す.

Listing 1: RandAugment

```
transforms = [
"Identity", "AutoContrast", "Equalize",
"Rotate", "Solarize", "Color", "Posterize",
"Contrast", "Brightness", "Sharpness",
"ShearX", "ShearY", "TranslateX", "TranslateY"]
def randaugment(N, M):
    """Generate a set of distortions.
    Args:
    N: Number of augmentation transformations to
    apply sequentially.
    M: Magnitude for all the transformations.
    """
    sampled_ops = np.random.choice(transforms, N)
    return [(op, M) for op in sampled_ops]
```

探索空間は 10^2 オーダーと驚異的に小さい．画像分類や画像検出の様々なタスクで SOTA に匹敵する精度を達成し，速度を大幅に向上させた．公式レポはなし (アルゴリズム自体は超簡単なため).

4 来週のタスク

TDGA + FastAutoAugment 実験. Population Based Augmentation と Rand Augment を読む.

参考文献

[1] Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, and Xi Chen. Population based augmentation: Efficient learning of augmentation policy schedules. *CoRR*, abs/1905.05393, 2019.

[2] Max Jaderberg, Valentin Dalibard, Simon Osindero, Wojciech M. Czarnecki, Jeff Donahue, Ali Razavi, Oriol Vinyals, Tim Green, Iain Dunning, Karen Simonyan, Chrisantha Fernando, and Koray Kavukcuoglu. Population based training of neural networks. *CoRR*, abs/1711.09846, 2017.

[3] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. *arXiv: Computer Vision and Pattern Recognition*, 2019.