中間報告書

2020 年 11 月 19 日

B4 佐々一心

1　実施内容

mnist データセットを用いて Mish と Tanexp の論文に記述されていたモデルと同じハイパーパラメータでモデルを実装しました。活性化関数のみを変化させて各エポック数ごとの訓練データの損失、精度と検証データの損失、精度を比較しました。比較した活性化関数は relu, selu(swish), mish, tanexp の 4 種類の他に relu の簡易的な連続近似の softplus、swish に近く、入力が負の時に出力が 0 に収束しない elu、flatten-T で提案されていた関数に近い relu から定数を引いた関数 (おそらく提案されていない関数) を使用しました。

2　結果

モデルの定義は Mish と Tanexp の論文に記述されていたモデルを使用し、下図の 12 の全結合層から構成される Block 層で使用される活性化関数を変化させて比較しました。

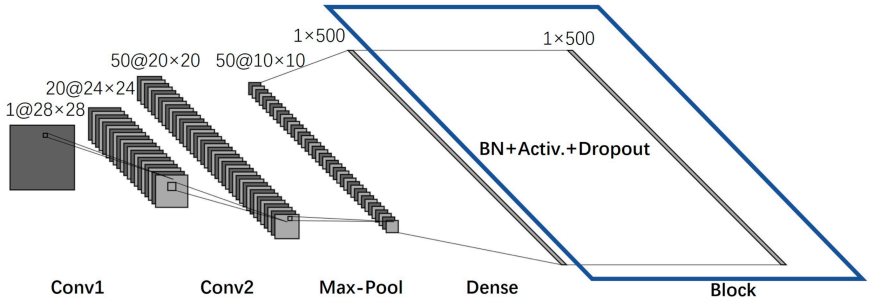

図 1　使用したモデル

各エポックごとの精度と損失を比較した結果、最も収束が早く、精度がよかった活性化関数は selu(swish) でした。さらに次いで elu が selu と同等の収束速度、精度でした。10 エポックまでの最終精度で比較すると mish が最も高い精度であった。

表 1　10 エポック目での最終精度

	relu	selu	mish	tanexp	softplus	elu	myopinion
accuray	0.9509	0.9793	0.9808	0.9786	0.7298	0.9791	0.9718

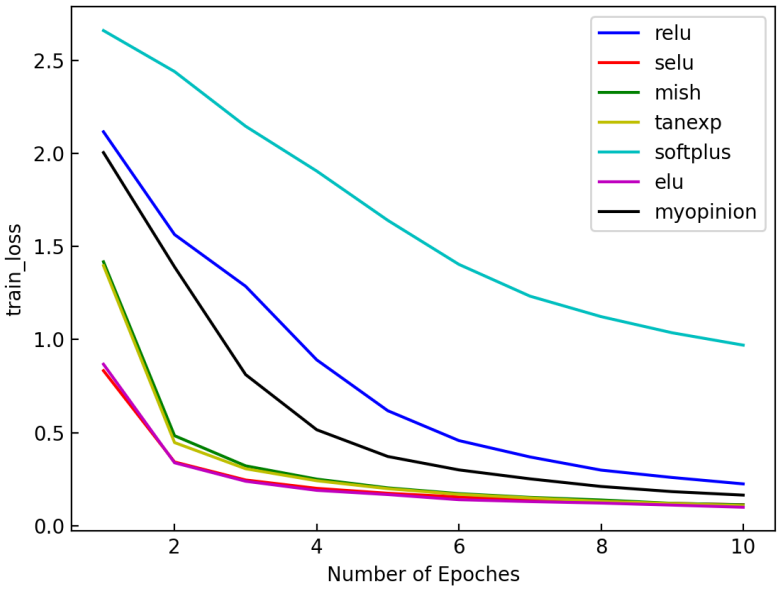

図 2　訓練データでの損失

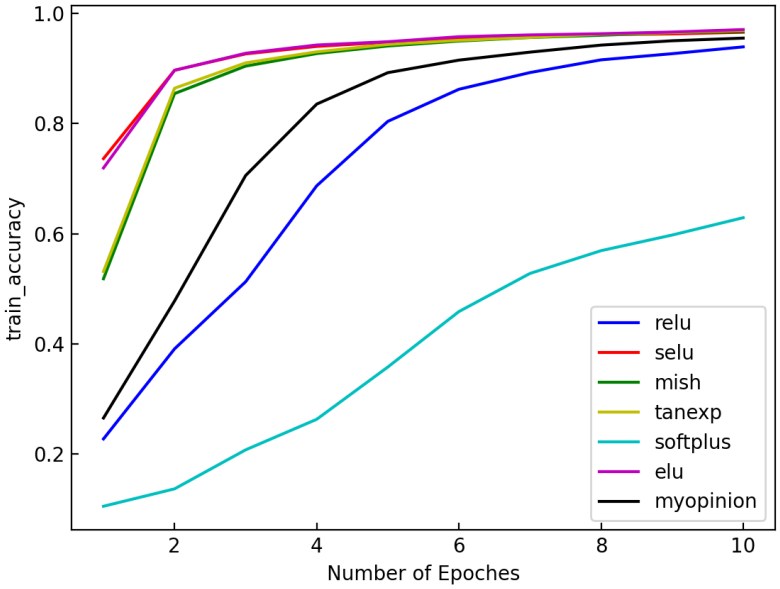

図 3　訓練データでの精度

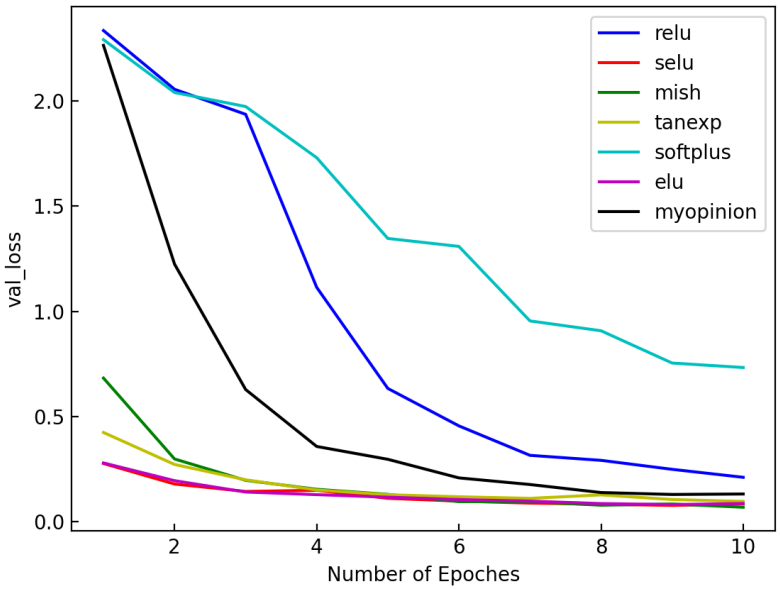

図 4　検証データでの損失

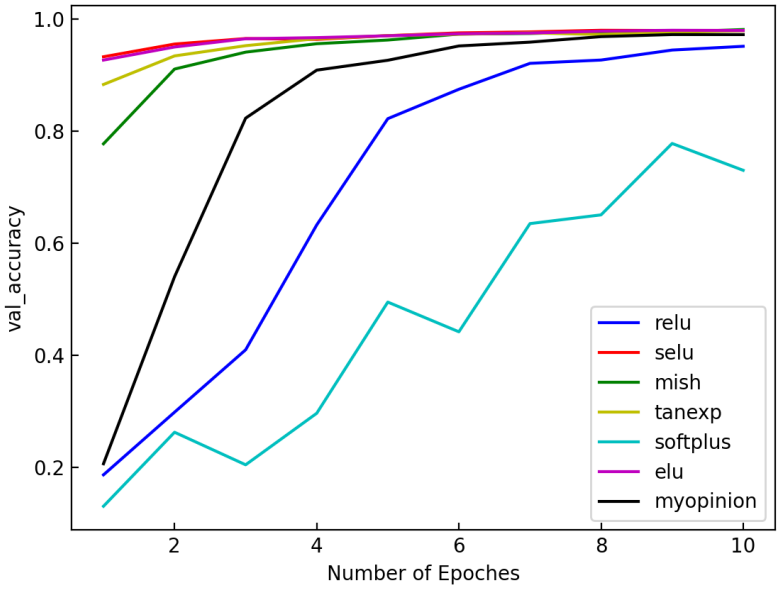

図 5　検証データでの精度

この結果から selu は入力が大きな負の値である時に 0 に収束しているが、0 に収束していない elu が同等の精度を出していることから入力が負の時に 0 に収束することが性能をよくしている訳ではないことが明確になりました。そして、relu から定数を引いた自作関数は relu と同等の計算量にも関わらず、relu よりも早い収束を見せている結果となった。活性化関数の入力が 0 の時に出力も 0 でなければいけないという暗黙のルールに反しているが、精度が上がったため、このルールに関して触れている論文がないか調べようと考えています。

また、tanexp の論文での結果の図と比較すると relu が 5 エポック目から傾きが緩やかになっている点や上から三番目の活性化関数が 2 エポック目から傾きが緩やかになっている点など、グラフ自体は極めて近い形をしていますが、それぞれの活性化関数がグラフと対応していないように見られます。よって tanexp の論文は信頼性に劣る論文であることが検証によってわかりました。

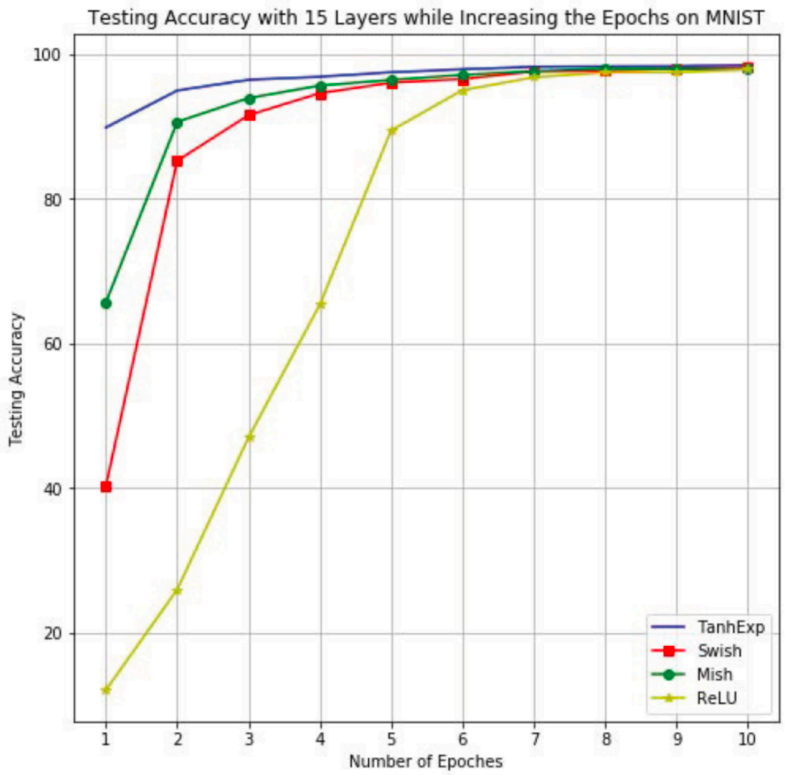

図 6　tanexp 論文での精度の比較結果

3　今後の方針

まず、tanexp の論文でも比較されている問題で、ニューラルネットワークは層を増やしすぎてしまうと精度が落ちてしまう問題があり、活性化関数によって層をどこまで増やしても精度を保つことができるか比較実験を行う予定です。ただし、今回の訓練時間の約 10 倍の時間がかかるため、湯川先生に勧められた論文や実装に時間を取られてしまい、まだ読めていないリプシッツ定数と活性化関数に関連する論文を読みすすめて行ければと思っています。