WP-GEB-Estimator-2 ユーザーマニュアル

～ WP-GEB : Weight-Perturbed GEneralization Bounds ～

磯部祥尚
産業技術総合研究所
y-isobe@aist.go.jp

2025 年 2 月 17 日

目次

1　サマリ

ツール名 WP-GEB-Estimator-2 の “WP-GEB” は**重み摂動付加汎化上界**（Weight-Perturbated GEneralization Bounds）を表しており、WP-GEB-Estimator-2 は順伝播型の**ニューラル分類器**[*1]（以下、**分類器**とよぶ）の**重み摂動付加汎化誤差**と**重み摂動付加汎化リスク**の上界を見積もるツールである。重み摂動とは分類器の重みパラメータに付加される摂動であり、重み摂動付加汎化誤差とは（ある分布にしたがって選択された）任意のデータと任意の重み摂動に対する不正解率の期待値である。また、重み摂動付加汎化リスクとは**リスクありデータ**の存在率の期待値であり、リスクありデータとは重み摂動付加に対する不正解率（**敵対的重み摂動**[*2]の存在率）が許容閾値を超えるデータのことである。重み摂動付加汎化誤差は無作為摂動（自然なノイズ）に対する耐性の評価に有効であり、重み摂動付加汎化リスクは最悪摂動（敵対的な攻撃）に対する耐性の評価に有効である。

機械学習の本格的な社会実装に向けて、機械学習を利用したシステムの品質に関する**機械学習品質マネジメント（MLQM）ガイドライン** [1] が策定されいる。このガイドラインには、機械学習要素の内部品質特性の一つとして、**機械学習モデルの安定性**「データセット以外の未見の入力に対しても安定して推論できること」が定義されている。図 1 は MLQM ガイドライン [1] の図 14「各フェーズとレベルに対する安定性の評価・向上技術」である。WP-GEB-Estimator-2 は以下の機能を有しており、評価フェーズでの訓練済み分類器の安定性評価に有用なツールである。

- テストデータセットと無作為重み摂動サンプルに対する分類器の不正解率を計測できる（**ノイズ耐性**）
- テストデータセットの各入力に対する敵対的重み摂動を探索できる（**敵対的攻撃**）
- 任意の入力に対する重み摂動付加汎化リスク上界（確率的信頼度付）を見積もれる（**敵対的検証**）
- 任意の入力に対する重み摂動付加汎化誤差上界（確率的信頼度付）を見積もれる（**汎化誤差**）

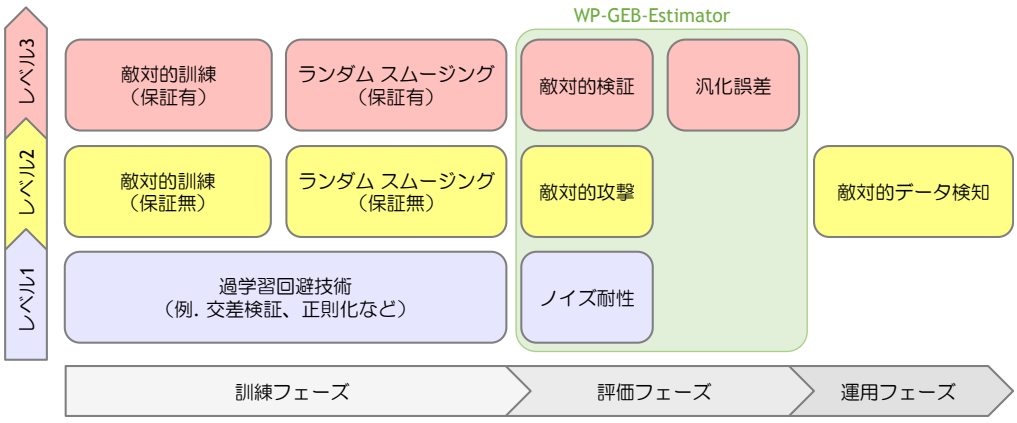

図 1　安定性の評価・向上に関する技術（MLQM ガイドライン [1] の図 14 ）

[*1] ニューラル分類器は分類目的で訓練したニューラルネットワークである。
[*2] 敵対的重み摂動とは付加すると不正解になる摂動である

2 WP-GEB の紹介

本節では、2.1 小節で WP-GEB（重み摂動付加汎化上界）の定義を与え、2.2 小節で WP-GEB の見積法について述べる。

2.1 WP-GEB の定義

重み摂動付加汎化誤差 $\mathbf{E}^{\alpha}(f_w)$ は任意のデータ $(x,y) \sim \mathcal{D}$ に対する分類器 f_w の**重み摂動付加個別誤差** $\mathbf{e}^{\alpha}_{(x,y)}(f_w)$ の期待値である（w は重みパラメータ、$\mathcal{D}$ は入力 x と正解出力 y の組 (x,y) の分布）。

$$\mathbf{E}^{\alpha}(f_w) := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\mathbf{e}^{\alpha}_{(x,y)}(f_w)\right]$$

ここで、重み摂動付加個別誤差 $\mathbf{e}^{\alpha}_{(x,y)}(f_w)$ は、任意の摂動 $u \sim \mathcal{U}_{w,\alpha}$ を重み w に付加したときの、データの一組 (x,y) に対する不正解率（すなわち、敵対的重み摂動の存在率）の期待値であり、重み摂動の分布 $\mathcal{U}_{w,\alpha}$ は重み摂動集合 $U_{w,\alpha}$ から無作為に一つの重み摂動を選択するための多次元一様分布である。

$$\mathbf{e}^{\alpha}_{(x,y)}(f_w) := \mathop{\mathbb{E}}_{u\sim\mathcal{U}_{w,\alpha}}\left[\ell(f_{w+u}(x),y)\right]$$
$$U_{w,\alpha} := \{(u_1,\ldots,u_{|w|}) \mid \forall i.\ |u_i| \le \alpha|w_i|\}$$

ここで、$\ell(y,y')$ は $y=y'$ ならば 0、$y\neq y'$ ならば 1 を返す 0-1 損失関数（i.e. $\ell(y,y') := \mathbb{1}\left[y\neq y'\right]$）である。重み摂動集合 $U_{w,\alpha}$ は、各重み w_i に付加される重み摂動 u_i の絶対値が $\alpha|w_i|$ 以下であることを表している（α は**重み摂動幅対重み比**）。

重み摂動付加汎化リスク $\mathbf{R}^{\alpha}_{\theta}(f_w)$ は、任意のデータに対して重み摂動付加個別誤差 $\mathbf{e}^{\alpha}_{(x,y)}(f_w)$ が閾値以上となる 0/1 真偽値（閾値より大きいならば 1）の期待値である。

$$\mathbf{R}^{\alpha}_{\theta}(f_w) := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\mathbb{1}\left[\mathbf{e}^{\alpha}_{(x,y)}(f_w) > \theta_{(x,y)}\right]\right]$$
$$\Theta := \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}\left[\theta_{(x,y)}\right]$$

ここで、$\theta_{(x,y)}$ はデータ (x,y) ごとに設定可能なリスクの許容閾値であり、Θ はその期待値（**汎化許容閾値**）である。この許容閾値は許容できる敵対的重み摂動の存在率（＝重み摂動付加個別誤差）であり、データ (x,y) の重み摂動付加個別誤差 $\mathbf{e}^{\alpha}_{(x,y)}(f_w)$ が許容閾値 $\theta_{(x,y)}$ を超えるとき、(x,y) を**リスクありデータ**とよぶ。摂動の影響を全く受けないことを保証するための厳しい許容閾値はゼロ（$\Theta=0$）であるが、現実的には計算コスト等の観点から、許容閾値は適切な微小な値に設定することが妥当である。妥当な許容閾値については次の 2.2 小節で説明する。

一般に、データ (x,y) と重み摂動 u は無数に存在するため、重み摂動付加汎化リスクや誤差 G を正確に計算することは難しい。そこで、WP-GEB-Estimator-2 では、任意の確率 $\delta\in(0,1)$ について、信頼度 $(1-\delta)$ 以上で次の不等式が成り立つような、G の上界 B を見積もる。

$$\mathbb{P}[G \le B] \ge 1-\delta$$

2.2 WP-GEB の見積法

重み摂動付加汎化誤差 $\mathbf{E}^{\alpha}(f_w)$ の上界の見積法については多くの研究がある。例えば、Pérez-Ortiz 等 [4] は、有限なデータセットと有限な無作為重み摂動サンプルを用いて分類器をテストし、その不正解率の計測結果から、Maurer の定理 [3] とサンプル収束バウンドの定理 [2] によって、重み摂動付加汎化誤差上界の見積法を与えた。WP-GEB-Estimator-2 は、訓練データセットの代わりにテストデータセットを用いるが、基本的には Pérez-Ortiz 等 [4] の重み摂動付加汎化誤差上界見積法を実装している。

一方、摂動付加汎化リスク $\mathbf{R}^{\alpha}_{\theta}(f_w)$ の上界の見積りについての研究は少ない。無作為摂動サンプルによるテストは確率的に敵対的重み摂動の存在率の上界を保証することができるが、その存在率は極めて小さいことが多く、無作為摂動サンプルでの発見は難しい。また、そのような敵対的重み摂動を容易に発見できる探索法もあるが、その場合はその存在率の上界を保証することはできない。そこで、WP-GEB-Estimator-2 では敵対的重み摂動の探索法と無作為摂動サンプルによるテストを次の手順で組み合わせている。

1. 最初に、敵対的重み摂動の探索法（FGSM, I-FGSM）を適用して、敵対的重み摂動を発見できたデータを許容閾値 0 でリスクありと断定する。
2. 次に、探索法で敵対的重み摂動を発見できなかったデータに対して無作為重み摂動サンプルを用いたテストを行い、この摂動サンプルで敵対的重み摂動を発見できたデータも許容閾値 0 でリスクありと断定する。
3. 最後に、上記の 1 と 2 で敵対的重み摂動を発見できなかったデータに関しては、許容閾値 θ^* でリスクがないことを確率的に保証する。

WP-GEB-Estimator-2 では、許容閾値 θ^* はオプション（`--thr`）によって指定でき、その保証のために必要な摂動サンプルサイズ m は次式により自動的に計算される。

$$m = \left\lceil log_{(1-\theta^*)}\left(\frac{r\delta}{n_0}\right)\right\rceil$$

ここで、n_0 は勾配法による探索で敵対的重み摂動を発見できなかったデータ数、θ^* は許容閾値、δ は不信頼度、r は不信頼度 δ のうち摂動の汎化に使える割合（一般に $r = 0.5$）である。不信頼度 δ は、データサンプルから汎化するときに生じる不信頼度と摂動サンプルから汎化するときに生じる不信頼度の和であり、その割合を r で調整できる。許容閾値は理想的には小さい方がよいが、計算コストを考慮すると摂動サンプルサイズが数千程度になるように指定することが現実的である。例えば、$n_0 = 800$, $\theta^* = 0.02$, $\delta = 0.1$, $r = 0.5$ ならば、$m = \lceil 479.1... \rceil = 480$ である。

WP-GEB-Estimator-2 は、次の許容閾値 $\theta_{(x,y)}$ を用いて重み摂動付加汎化リスク $\mathbf{R}^{\alpha}_{\theta}(f_w)$ の上界（確率的信頼度付）と汎化許容閾値 Θ の上界を計算するツールである。

$$\theta_{(x,y)} := \begin{cases} 0 & \text{if } (x,y) \in T_1 \\ \theta^* & \text{otherwise} \end{cases}$$

ここで、T_1 は探索によって敵対的重み摂動を発見できたデータの集合（テストデータセット T の部分集合）である。最悪重み摂動付加汎化誤差 $\mathbf{R}^{\alpha}_{\theta}(f_w)$ の上界の詳細な計算式については文献 [7] を参照して欲しい[*3]。

[*3] 基本的には文献 [5] の計算式を実装しているが、無作為摂動付加テストの前に敵対的重み摂動探索を行うなどの変更がされている。

3 WP-GEB-Estimator-2 の紹介

本節では、3.1 小節で WP-GEB-Estimator-2 を構成する 4 つのツールを概説し、3.2 小節と 3.3 小節で WP-GEB-Estimator-2 の実行に必要な入力ファイルと実行時に生成される出力ファイルについて説明する。

3.1 ツール構成

WP-GEB-Estimator-2 は評価用の分類器の訓練から評価までを行う次の 4 つのツールから構成されている。このツールは Python 言語の Tensorflow/Keras を用いて記述されている。

- `train`: 評価用分類器訓練ツール
  - 入力：訓練データセット、ネットワークアーキテクチャ（CSV 形式）等
  - 出力：訓練済み分類器（TensorFlow-SavedModel 形式）
- `search`: 訓練済み分類器の敵対的重み摂動探索ツール
  - 入力：テストデータセット、訓練済み分類器、重み摂動対重み比リスト等
  - 出力：敵対的重み摂動探索結果（入力情報含む、CSV 形式）
- `measure`: 訓練済み分類器の無作為重み摂動付加テスト誤差計測ツール
  - 入力：テストデータセット、訓練済み分類器、敵対的重み摂動探索結果（CSV 形式）
  - 出力：敵対的重み摂動探索結果・無作為重み摂動付加誤差計測結果（CSV 形式）
- `estimate`: 訓練済み分類器の重み摂動付加汎化リスク/誤差上界見積ツール
  - 入力：無作為重み摂動付加誤差計測結果・敵対的重み摂動探索結果（CSV 形式）
  - 出力：重み摂動付加汎化リスク/誤差上界見積結果等（入力情報含む、CSV 形式）

上記の 4 つのツールを順番に実行することによって重み摂動付加汎化リスク/誤差上界 WP-GEB を見積もることができる。各ツールの実行に必要な入力ファイルと実行によって生成される出力ファイルの関係を図 2 に示す。既存の訓練済み分類器を評価する場合は、訓練ツール `train` を実行する必要はない。ただし、探索なしの摂動付加汎化リスクや摂動付加汎化誤差の上界を見積る場合、すなわち、敵対的重み摂動の探索が不要な場合でも、探索ツール `search` を実行する必要はある（`search_out.csv` を生成するため）。この場合はオプション（`--skip_search`）で実際の探索をスキップできる。図 2 の入力ファイルと出力ファイルについては、各々、次の 3.2 小節と 3.3 小節で説明する。

3.2 入力ファイル

評価対象となる訓練済み分類器とテストデータセットは、各々、探索ツール `search` のオプション`--model_dir` と`--dataset_file` で指定することができる。現在サポートされている分類器のフォーマットは TensorFlow-SavedModel 形式、データセットのフォーマットは TF-Record 形式のみである。なお、Keras が提供している分類器（例. Inception v3）やデータセット（例. MNIST）は名前を指定するだけで使用できる。指定方法は 4 節のオプションを参照してほしい。

分類器訓練ツール `train` によってシンプルな分類器を訓練することができる。訓練には分類器のアーキテクチャの情報が必要であり、アーキテクチャ情報を記述した CSV ファイルはツール `train` 実行時に読み込まれる。この CSV ファイルで指定可能な層の種類とパラメータ（CSV ファイルの列）を表 1 に示す。ここで、

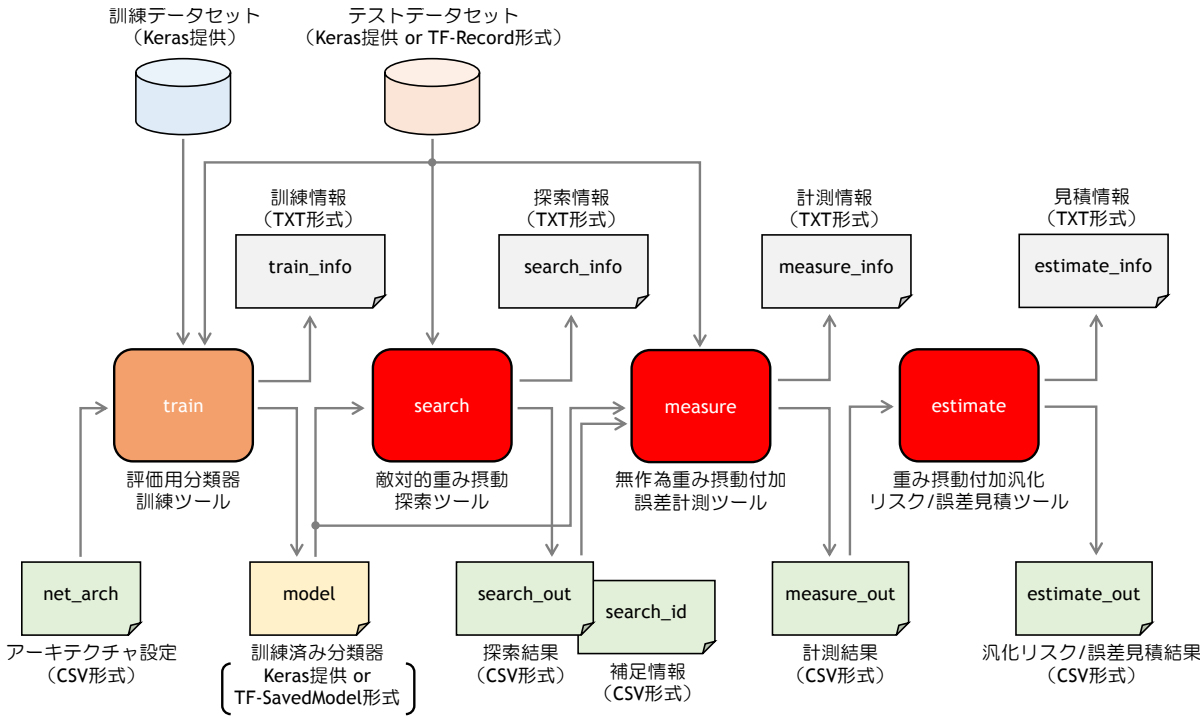

図 2　WP-GEB-Estimator-2 を構成する四つのツールの入出力ファイルの関係（デフォルトのファイル名）

表 1　指定可能な層の種類とパラメータ（$a \in \{\texttt{relu} \mid \texttt{linear} \mid \texttt{softmax}\}$、$n, n_i \in \mathbb{I}$、$r \in \mathbb{R}$）

type	activation	units	filters	int_tuple	regular_l2	rate
Activation	a					
Dense	a	n			r	
Conv2D	a		n	(n_1, n_2)		
MaxPooling2D				(n_1, n_2)		
Dropout						r
BatchNormalization						
Flatten						

$\mathbb{I}$ は 0 以上の整数集合、$\mathbb{R}$ は 0 以上の実数集合である。表 1 に示すように、層の種類に応じてパラメータを指定する必要があるが、L2 正則化係数（regular_l2）とドロップアウト率（rate）は省略可能であり、省略した場合は 4 節で説明するオプションで指定することができる。表 1 の層を入力側から順番に記述することで、簡単な多層パーセプトロンや畳み込みニューラルネットワークを表現することができる。ディレクトリ net_arch にアーキテクチャの CSV ファイルのサンプルが保存されているので参考にしてほしい。

3.3 出力ファイル

各ツール train, search, measure, estimate で使用した入出力情報は、各々、テキストファイル train_info, search_info, measure_info, estimate_info に保存される（ファイル名はオプションで変更可能）。また、重み摂動付加汎化リスク/誤差上界のグラフ化など、見積結果の処理に適した表形式のデータは CSV ファイル estimate_out.csv に保存される（途中の見積結果 search_out.csv, measure_out.csv も含む）。search で複数の重み摂動比を指定した場合、重み摂動比ごとに見積結果がこの CSV ファイルの 1 行に保存され、search → measure → estimate を実行するごとにその見積結果が下の行に追加される。以下、CSV ファイル estimate_out.csv の各列（A〜AK）の意味について説明する。

- searcg の入出力値（search_out.csv の列 A〜O と同じ）
  - A（dataset_name）：テストデータセット名
  - B（dataset_size）：テストデータセットサイズ
  - C（dataset_offset）：テストデータセットの開始インデックス
  - D（dataset_file）：テストデータセットのファイル名
  - E（dataset_fmt）：テストデータセットのファイルのフォーマット
  - F（image_width）：テストデータセット画像の幅
  - G（image_height）：テストデータセット画像の高さ
  - H（model_dir）：分類器の名前/ディレクトリ名
  - I（rnd_seed_search）：ランダムシード
  - J（batch_size_search）：同時に勾配を計算するデータ数
  - K（perturb_bn）：バッチ正規化のパラメータへの摂動付加（0：無、1：有）
  - L（perturb_ratio）：最大重み摂動幅対重み比
  - M（search_mode）：敵対的重み摂動探索モード（0：FGSM、1：I-FGSM）
  - N（max_iteration）：I-FGSM の場合の最大繰返し回数
  - O（err_num_search）：探索で敵対的重み摂動の存在が確認されたデータ数
- measure の入出力値（measure_out.csv の列 P〜Z と同じ）
  - P（rnd_seed_measure）：ランダムシード
  - Q（batch_size_measure）：一度に推論するデータ数
  - R（err_thr）：許容閾値
  - S（err_thr_practical）：実際に使用された実用的な許容閾値
  - T（delta）：不信頼度
  - U（delta0_ratio）：有限摂動数を汎化するために使われる不信頼度の割合
  - V（perturb_sample_size）：無作為重み摂動サンプルサイズ（無作為重み摂動付加推論回数）
  - W（err_num_random）：無作為重み摂動サンプルで敵対的重み摂動の存在が確認されたデータ数
  - X（err_num）：探索または無作為サンプルで敵対的重み摂動の存在が確認されたデータ数
  - Y（test_err_wst）：無作為重み摂動サンプルで敵対的重み摂動の存在が確認されたデータの割合
  - Z（test_err_avr）：無作為重み摂動サンプルとテストデータセットに対する不正解率の平均
- estimate の入出力値

– 最悪重み摂動
AA（gen_risk_ub）：摂動付加汎化リスク上界
AB（test_risk_ub）：摂動付加テストリスク上界
AC（conf_riskt）：摂動付加汎化リスク上界信頼度
AD（conf0_risk）：摂動付加テストリスク上界信頼度
AE（non_det_rate_ub）：探索による不検出率の上界
AF（gen_err_thr_ub）：汎化許容閾値上界
– 無作為重み摂動
AG（gen_err_ub）：摂動付加汎化誤差上界
AH（test_err_ub）：摂動付加テスト誤差上界
AI（test_err）：サンプル摂動付加テスト誤差
AJ（conf_err）：摂動付加汎化誤差上界信頼度
AK（conf0_err）：摂動付加テスト誤差上界信頼度

4 WP-GEB-Estimator-2 の実行

本節では、WP-GEB-Estimator-2 の各ツールの実行コマンドと実行時オプションについて説明する。以下、引数の集合として、0 以上の整数の集合 $\mathbb{I}$、0 以上の実数の集合 $\mathbb{R}$、長さ 1 以上の文字列の集合 $\mathbb{S}$ の他、次に示す実数のリスト（区切り記号はスペース）の文字列の集合 $\mathbb{SR}^*$ も用いる。

$$\mathbb{SR}^* = \{\texttt{"}r_1 \;\; r_2 \;\; \cdots \;\; r_m\texttt{"} \mid \exists m \in \mathbb{I}.\ \forall i \in \{1,\ldots,m\}.\ r_i \in \mathbb{R}\} \subset \mathbb{S}$$

また、Tensorflow/Keras が提供するデータセットと分類器を利用するため、次の集合も用いる。

$$\begin{aligned}\texttt{KerasDataSets} &= \{\texttt{"mnist"}, \texttt{"fashion\_mnist"}, \texttt{"cifar10"}\} \subset \mathbb{S}\\ \texttt{KerasModels} &= \{\texttt{"inception\_v3"},\ \texttt{"inception\_resnet\_v2"},\ \texttt{"resnet50"},\ \texttt{"xception"},\\ &\quad\ \texttt{"densenet121"}, \texttt{"densenet169"},\ \texttt{"densenet201"},\ \texttt{"vgg16"},\ \texttt{"vgg19"},\\ &\quad\ \texttt{"nasnetlarge"},\ \texttt{"nasnetmobile"}\} \subset \mathbb{S}\end{aligned}$$

4.1 分類器訓練

評価用分類器訓練ツールを実行するには、ディレクトリ src で次のコマンドを入力する。

```
python train_main.py [options]
```

以下、各オプションについて説明する[*4]。

--random_seed n　（$n \in \mathbb{I}$, default $n = 1$）
　ランダムシードを n に設定する（ただし、$n = 0$ ならばランダムシードを設定しない）
--net_arch_file s　（$s \in \mathbb{S}$, default $s =$ "net_arch/cnn_s"）
　分類器のアーキテクチャを読み込むファイル名を s にする
--result_dir s　（$s \in \mathbb{S}$, default $s =$ "result"）

[*4] ファイル名を指定する場合、拡張子（.txt, .csv）は不要

訓練結果を保存するディレクトリ名を s にする

`--model_dir` s （$s \in \mathbb{S}$, default $s =$ `"model"`）

訓練した分類器を保存するディレクトリ名を s にする

`--dataset_name` s （$s \in$ `KerasDataSets`, default $s =$ `"mnist"`）

訓練/テストに用いるデータセット名を s にする

`--train_dataset_size` n （$n \in \mathbb{I}$, default $n =$ `50000`）

訓練データセットのサイズを n にする

`--train_dataset_offset` n （$n \in \mathbb{I}$, default $n =$ `0`）

訓練データセットの開始インデックスを n にする

`--test_dataset_size` n （$n \in \mathbb{I}$, default $n =$ `5000`）

テストデータセットのサイズを n にする

`--test_dataset_offset` n （$n \in \mathbb{I}$, default $n =$ `0`）

テストデータセットの開始インデックスを n にする

`--validation_ratio` r （$r \in [0, 1) \subset \mathbb{R}$, default $r =$ `0.1`）

訓練データセットのうちバリデーションに使用する割合を r にする

`--sigma` r （$r \in \mathbb{R}$, default $r =$ `0.1`）

訓練パラメータ（重みとバイアス）を初期化する正規分布の標準偏差を r にする

`--batch_size` n （$n \in \mathbb{I}$, default $n =$ `100`）

訓練データのバッチサイズを n にする

`--epochs` n （$n \in \mathbb{I}$, default$n =$ `50`）

訓練のエポック数を n にする

`--dropout_rate` r （$r \in [0, 1) \subset \mathbb{R}$, default $r =$ `0.0`）

分類器のアーキテクチャに明記されていない場合、ドロップアウト率を r にする

`--regular_l2` r （$r \in \mathbb{R}$, default $r =$ `0.0`）

分類器のアーキテクチャに明記されていない場合、L2 正則化係数を r にする

`--learning_rate` r （$r \in \mathbb{R}$, default $r =$ `0.01`）

訓練の（初期）学習率を r にする

`--decay_rate` r （$r \in \mathbb{R}$, default $r =$ `1.0`）

学習率の指数関数的減衰率を r にする（$r =$ `1.0` ならば減衰なし）

`--decay_steps` n （$n \in \mathbb{I}$, default $n =$ `0`）

学習率が減衰率倍になるステップ幅を n にする（$n =$ `0` ならば減衰なし）

`--early_stop` b （$b \in \{$`0`, `1`$\}$, default $b =$ `0`）

$b =$ `1` ならば早期終了を有効にする（$b =$ `0` ならば無効にする）

`--early_stop_delta` r （$r \in \mathbb{R}$, default $r =$ `0.0`）

損失値の減少が r 以下の場合に改善なしと判断する

`--early_stop_patience` n （$n \in \mathbb{I}$, default $n =$ `3`）

早期終了が有効な場合、損失値の改善なしが n 回連続したときに早期終了する

`--verbose` b （$b \in \{$`0`, `1`, `2`$\}$, default $b =$ `1`）

$b =$ `1` or `2` ならば訓練中の進捗状況を表示する（$b =$ `0` ならば表示しない）

4.2 敵対的重み摂動探索

敵対的重み摂動探索ツールを実行するには、ディレクトリ src で次のコマンドを入力する。

```
python search_main.py [options]
```

以下、各オプションについて説明する。

--random_seed n （$n \in \mathbb{I}$, default $n = 1$）
ランダムシードを n に設定する（ただし、$n = 0$ ならばランダムシードを設定しない）

--model_dir s （$s \in \mathbb{S}$, default $s =$ "model"）
訓練済み分類器を読込むディレクトリ名を s にする
（$s \in$ KerasModels の場合は Keras が提供する訓練済み分類器 s を読込む）

--dataset_name s （$s \in \mathbb{S}$, default $s =$ "mnist"）
テストデータセット名を s にする（$s \in$ KerasDataSets ならば Keras のデータセット s を読込む）

--dataset_file s （$s \in \mathbb{S}$, default $s =$ "~/imagenet/1k-tfrecords/validation-*-of-00128"）
テストデータセットを読込むファイル名を s にする（$s \in$ KerasDataSets の場合は不要）

--dataset_fmt s （$s \in \mathbb{S}$, default $s =$ "tfrecord"）
テストデータセットのファイルのフォーマットを s にする
（現在サポートしているフォーマットは TF-Record（"tfrecord"）のみ）

--image_width n （$n \in \mathbb{I}$, default $n = 0$）
テストデータ画像の幅を n にする（Keras 提供のモデルやデータセットで指定不要の場合は 0）

--image_height n （$n \in \mathbb{I}$, default $n = 0$）
テストデータ画像の高さを n にする（Keras 提供のモデルやデータセットで指定不要の場合は 0）

--dataset_size n （$n \in \mathbb{I}$, default $n = 5000$）
テストデータセットのサイズを n にする

--dataset_offset n （$n \in \mathbb{I}$, default $n = 0$）
テストデータセットの開始インデックスを n にする

--result_dir s （$s \in \mathbb{S}$, default $s =$ "result"）
訓練結果の読込みや探索結果を保存をするディレクトリ名を s にする

--search_file s （$s \in \mathbb{S}$, default $s =$ "search"）
敵対的重み摂動探索結果を保存するファイル名を s にする

--perturb_ratios s （$s \in \mathbb{S}\mathbb{R}^*$, default $s =$ "0.01 0.1 1"）
重み摂動幅対重み比（各重みの大きさに対するその重み摂動最大幅の比率）リストを s にする
（リストの先頭の重み摂動幅から順番に重み摂動付加汎化誤差上界を見積もる）

--perturb_bn b （$b \in \{0, 1\}$, default $b = 0$）
$b = 1$ ならば、バッチ正規化の訓練パラメータ（スケール、シフト）にも摂動を付加する
（$b = 0$ ならば、バッチ正規化の訓練パラメータには摂動を付加しない）

--skip_serach b （$b \in \{0, 1\}$, default $b = 0$）
$b = 1$ ならば敵対的探索をスキップする（$b = 0$ ならば探索する）

（無作為重み摂動付加汎化誤差上界のみ見積もる場合は探索をスキップできる）

`--search_mode` n　（$n \in \{0, 1\}$, default $n = 0$）

敵対的な重み摂動の探索法を mode-n にする

- mode-0：FGSM（勾配で損失が最大になる方向に最大の摂動を重みに付加する）
- mode-1：I-FGSM（不正解になるか損失低下しなくなるまで FGSM を繰り返す）

`--batch_size` n　（$n \in \mathbb{I}$, default $n = 10$）

FGSM（mode-0）の場合、並列に勾配計算するデータセットサイズを n にする

`--max_iteration` n　（$n \in \mathbb{I}$, default $n = 20$）

I-FGSM（mode-1）の場合の探索の最大繰返し回数（探索打切り回数）を n にする

`--verbose_search` b　（$b \in \{0, 1\}$, default $b = 1$）

$b = 1$ ならば敵対的重み摂動探索中の進捗状況を表示する（$b = 0$ ならば表示しない）

4.3 無作為重み摂動付加誤差計測

無作為重み摂動付加誤差計測ツールを実行するには、ディレクトリ `src` で次のコマンドを入力する。

```
python measure_main.py [options]
```

以下、各オプションについて説明する。

`--random_seed` n　（$n \in \mathbb{I}$, default $n = 1$）

ランダムシードを n に設定する（ただし、$n = 0$ ならばランダムシードを設定しない）

`--result_dir` s　（$s \in \mathbb{S}$, default $s =$ `"result"`）

探索結果の読込みや計測結果を保存をするディレクトリ名を s にする

`--search_file` s　（$s \in \mathbb{S}$, default $s =$ `"search"`）

敵対的重み摂動探索結果を読み込むファイル名を s にする

`--measure_file` s　（$s \in \mathbb{S}$, default $s =$ `"measure"`）

無作為重み摂動付加誤差計測結果を書き込むファイル名を s にする

`--batch_size` n　（$n \in \mathbb{I}$, default $n = 0$）

一度に誤差を計測するデータセットサイズを n にする（$n = 0$ ならばデータセットサイズと同じ）

`--err_thr` r　（$r \in (0, 1) \subset \mathbb{R}$, default $r = 0.01$）

許容閾値 θ^* を r にする

`--perturb_sample_size` n　（$n \in \mathbb{I}$, default $n = 0$）

無作為に選択する重み摂動サンプルサイズを n にする（最悪摂動の場合は $n = 0$）

（$n = 0$ ならば許容閾値に適したサンプルサイズに自動的に設定される）

`--delta` δ　（$\delta \in (0, 1) \subset \mathbb{R}$, default $\delta = 0.1$）

重み摂動付加汎化誤差がその上界の見積結果を超える確率を δ（2.2 小節の δ）まで許容する

（i.e. 上界の信頼度は $1 - \delta$ になる）

`--delta0_ratio` r　（$r \in (0, 1) \subset \mathbb{R}$, default $r = 0.5$）

重み摂動付加テスト誤差がその上界見積結果を超える確率を $r\delta$（2.2 小節の $r\delta$）まで許容する

（i.e. 上界の信頼度は $1 - r\delta$ になる）

`--verbose_measure` b　（$b \in \{0, 1\}$, default $b = 1$）
　$b = 1$ ならば無作為重み摂動付加誤差計測進捗状況を表示する（$b = 0$ ならば表示しない）

4.4 重み摂動付加汎化リスク/誤差上界（WP-GEB）の見積り

重み摂動付加汎化リスク/誤差上界見積ツールを実行するには、ディレクトリ `src` で次のコマンドを入力する。

```
python estimate_main.py [options]
```

以下、各オプションについて説明する。

`--result_dir` s　（$s \in \mathbb{S}$, default $s =$ `"result"`）
　探索結果（計測結果含む）の読込みや見積結果を保存するディレクトリ名を s にする
`--measure_file` s　（$s \in \mathbb{S}$, default $s =$ `"measure"`）
　無作為摂動付加誤差計測結果（敵対的重み摂動探索結果含む）を読み込むファイル名を s にする
`--estimate_file` s　（$s \in \mathbb{S}$, default $s =$ `"estmate"`）
　重み摂動付加汎化リスク/誤差上界見積結果を保存するファイル名を s にする
`--max_nm` n　（$n \in \mathbb{I}$, default $r =$ `10`）
　汎化の見積りに用いるニュートン法の最大繰返し回数を n にする
`--eps_nm` r　（$r \in \mathbb{R}$, default $r =$ `0.0001`）
　汎化の見積りに用いるニュートン法の許容誤差を r にする

5 WP-GEB-Estimator-2 の実行例

本節では、WP-GEB-Estimator-2 の実行コマンド（実行スクリプト）の例と、その実行結果の例を紹介する。なお、WP-GEB-Estimator-2 の実行には TensorFlow と NumPy ライブラリをインストールした Python 環境が必要である。WP-GEB-Estimator-2 開発時に使用したソフトウェアのバージョンは Python 3.10.16, TensorFlow 2.16.2, Keras 3.6.0, NumPy 1.26.4 である。詳細は添付の `requirements.txt` を参照して欲しい。

5.1 分類器訓練〜汎化リスク/誤差見積り

デフォルトパラメータを用いた手書き数字（MNIST）による 3 層パーセプトロンの訓練、敵対的重み摂動の探索、無作為重み摂動サンプルによる誤差の計測、汎化リスク/誤差上界の見積りは、図 3 に示すスクリプト `run.sh` （ディレクトリ `examples` に保存されている）により実行でき、その実行結果はディレクトリ `results/result` に保存されている。その実行時間については、MacBook Pro（CPU: Apple M2 Max, Mem: 96GB）で実行した場合、訓練時間は約 90 秒、一つの摂動幅対重み比あたりの敵対的重み摂動探索時間は約 70 秒（データセットサイズ 5000）、無作為重み摂動付加誤差計測時間は約 4 分（探索による検出データ数に依存、許容閾値 1%（重み摂動サンプルサイズ 1146）、汎化リスク/誤差見積時間は 0.1 秒以下であった。

図 4 に、実行スクリプト `run.sh` の実行結果（`estimate_info.txt` の一部）を示す。重み摂動幅対重み比

```
python train_main.py  --net_arch_file "net_arch/mlp_s_bn"

# with search of adversarial perturbations by FGSM
python search_main.py  --skip_search 0
python measure_main.py
python estimate_main.py

# without search
python search_main.py  --skip_search 1
python measure_main.py
python estimate_main.py
```

図 3　実行スクリプト run.sh（簡単な分類器の訓練と評価）

```
---with search---
Perturbation ratio = 0.01
  Random perturbation sample size: 1117
  Risk (with search):
    Perturbed generalization risk bound: 26.74% (Conf: 90.00%)
    Perturbed test risk bound: 25.22% (Conf: 95.00%)
    Generalization acceptable threshold bound: 0.7608% (Conf: 90.00%)
---without search---
Perturbation ratio = 1.0
  Random perturbation sample size: 1146
  Risk (without search):
    Perturbed generalization risk bound: 100.00% (Conf: 100.00%)
    Perturbed test risk bound: 100.00% (Conf: 100.00%)
    Generalization acceptable threshold bound: 0.0000% (Conf: 100.00%)
  Error:
    Perturbed generalization error bound: 32.80% (Conf: 90.00%)
    Perturbed test error bound: 30.17% (Conf: 95.00%)
```

図 4　実行スクリプト run.sh の実行結果（estimate_info.txt の一部）

が 0.01 と 0.1 の探索ありの重み摂動付加汎化リスク上界、0.1 と 1 の探索なし重み摂動付加汎化リスクと摂動付加汎化誤差の上界が出力されている。例えば、重み摂動幅対重み比 0.01 の重み摂動付加汎化リスク（汎化許容閾値 0.76% 以下））は 27% 以下（信頼度 90% 以上）、重み摂動幅対重み比 1 の重み摂動付加汎化誤差は 33% 以下（信頼度 90% 以上）である。

```
ImageNet_DIR="$HOME/datasets/imagenet/1k-tfrecords/validation"

# with search of adversarial perturbations by FGSM
python search_main.py \
    --skip_search 0 \
    --result_dir "result_imagenet" --dataset_name "imagenet" \
    --dataset_file "$ImageNet_DIR/validation-*-of-00128" \
    --model_dir "inception_v3"  --dataset_size 1000 \
    --perturb_ratios "0.0001"
python measure_main.py \
    --result_dir "result_imagenet" --err_thr 0.02
python estimate_main.py --result_dir "result_imagenet"

# without search
python search_main.py \
    --skip_search 1 \
    ...
```

図 5　実行スクリプト `run_imagenet.sh`（訓練済み分類器 Inception-v3 の評価）

5.2　訓練済み分類器（Inception-v3）の評価

図 5 は、TensorFlow の Keras で提供されている訓練済み分類器 Inception-v3 の汎化リスク/誤差を見積るためのスクリプトの例（`examples/run_imagenet.sh` に保存されている）であり、その実行結果はディレクトリ `results/result_imagenet` に保存されている。なお、この実行スクリプトでは、実行前にデータセット ImageNet のファイル（TF-Record 形式）がディレクトリ `ImageNet_DIR` に保存されていることを仮定している。TF-Record 形式の ImageNet のファイル（`1k-tfrecords-0.zip`, `1k-tfrecords-1.zip`）は下記の Kaggle のサイトからダウンロードできる。

`ImageNet 1K TFrecords ILSVRC2012 - part 0`
`https://www.kaggle.com/datasets/hmendonca/imagenet-1k-tfrecords-ilsvrc2012-part-0`

このスクリプト `run_imagenet.sh` の実行時間については、MacBook Pro（CPU: Apple M2 Max, Mem: 96GB）で実行した場合、敵対的重み摂動探索時間が約 10 分（データセットサイズ 1000)、無作為重み摂動付加誤差計測時間が約 20 分〜30 分（許容閾値 2%（重み摂動サンプルサイズ 472〜491)、汎化誤差計算時間が 0.1 秒以下であった。見積結果は、重み摂動幅対重み比 0.0001 の重み摂動付加汎化リスク（汎化許容閾値 1.42% 以下）が 36% 以下（信頼度 90% 以上)、重み摂動付加汎化誤差が 33% 以下（信頼度 90% 以上）であった。ImageNet の他の分類器の重み摂動付加汎化リスク/誤差上界の見積結果についてはポスター発表 [6] も参照してほしい。

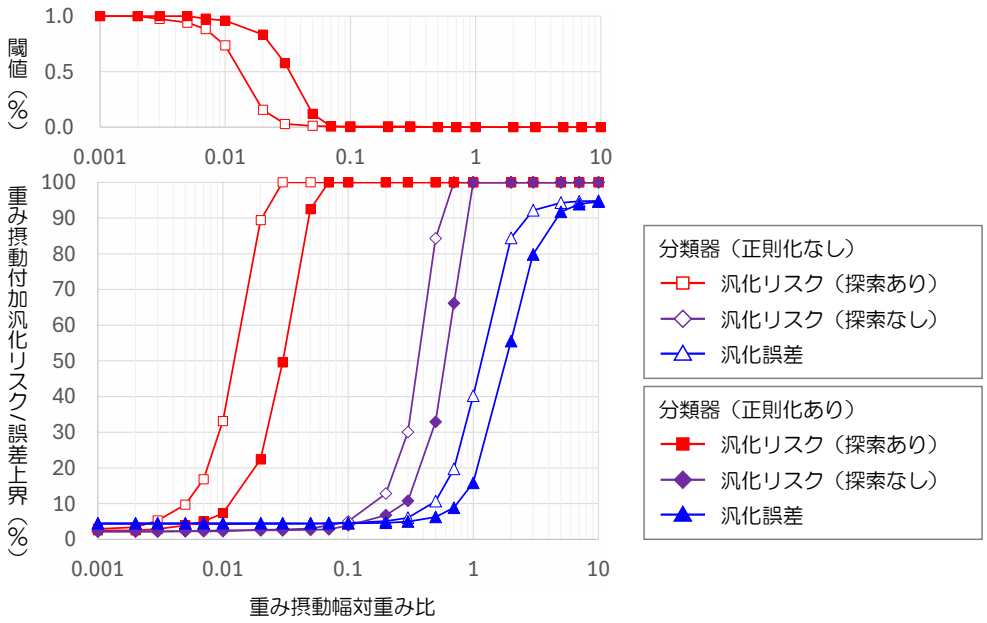

図 6　重み摂動付加汎化リスク/誤差上界と汎化許容閾値上界の見積結果（信頼度 90%）

5.3　分類器の WP-GEB の比較

ディレクトリ `examples` に保存されているファイル `run_main.sh` は、正則化無しと有りの二つの分類器を訓練し、重み摂動付加汎化リスク/誤差上界を比較するための実行スクリプトである。アーキテクチャファイルの正則化係数が省略されている場合はオプションで指定できるため、様々な正則化係数で簡単に試すことができる。正則化無しで訓練した分類器と正則化有り（L2 係数 0.001）で訓練した分類器の重み摂動付加汎化リスク/誤差上界（WP-GEB）と汎化許容閾値上界の見積結果（信頼度 90%）を図 6 に示す。このグラフは `estimate_out.csv` を表計算ソフト（Microsoft Excel）で読込み、グラフ描画機能で作成した。図 6 の横軸は重み摂動幅対重み比であり、重み摂動を付加しない場合の二つの分類器のテスト誤差はほぼ同じであるが、重み摂動を付加することによって、正則化有りで訓練した分類器の方が重み摂動に対する耐性が高いことが明確になっている。また、汎化リスク（探索あり）の場合は汎化誤差よりも 2 桁程度小さい重み摂動幅で性能の低下（誤差の増加）が見られる。

参考文献

[1] 大岩 寛他, 機械学習品質マネジメントガイドライン 第 4 版, Digiarc-TR-2023-03, CPSEC-TR-2023003, 2023. `https://www.digiarc.aist.go.jp/publication/aiqm/`

[2] J. Langford and R. Caruana, (Not) Bounding the True Error, 14th International Conference on Neural Information Processing Systems (NIPS 2001), pp.809–816, 2001.

[3] A. Maurer, A Note on the PAC Bayesian Theorem, arXiv:cs/0411099, 2004.

[4] M. Pérez-Ortiz, O. Rivasplata, J. Shawe-Taylor, and C. Szepesvári, Tighter risk certificates for

neural networks, Journal of Machine Learning Research (JMLR), 2021.
[5] 磯部, 最悪重み摂動付加ニューラル分類器の汎化誤差上界の見積法, 第 38 回人工知能学会全国大会（JSAI2024）, 2024.
[6] 磯部, 敵対的摂動に対するニューラル分類器の確率的安全性保証, 産総研サイバーフィジカルセキュリティ研究シンポジウム, 2025.
[7] Y. Isobe, A Proof Note of Weight-Perturbed Generalization Error Bounds, 2025 (to be published in the web-site of WP-GEB-Estimator-2)

謝辞

本ツール WP-GEB-Estimator-2 は、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務（JPNP20006）にて開発されたものです.