Preferred Networks インターン選考 2018 コーディング課題機械学習・数理分野

変更履歴

- 2018 年 5 月 1 日：初版

回答にあたっての注意

- ソースコードには以下のいずれかの言語を利用してください。
  - C, C++, Python, Ruby, Go, Java, Rust
- 課題 1-3 では、各言語の標準ライブラリの関数のみを使用し、特に NumPy, Eigen など多次元配列ライブラリは使用しないで実装してください。
- 課題 4 ではライブラリを自由に使用して頂いて構いません。
- 課題には自分だけで取り組んでください。この課題を他の応募者を含めた他人と共有・相談することを禁止します。漏洩の証拠が見つかった場合、その応募者は失格となります。ある応募者が別の応募者に回答をコピーさせた場合、双方の応募者が失格となります。
- 想定所要時間は最大 2 日です。全課題が解けていなくても提出は可能ですので、学業に支障の無い範囲で取り組んで下さい。

提出物

以下のものを提出して下さい。ディレクトリ名やファイル名は以下に従って下さい。

- 課題 1-4 のソースコード
  - `src` というディレクトリを作ってそこにソースコードを置いて下さい。
  - ソースコードは課題ごとに別々になっていても、課題 1-4 を通じて 1 つのファイルにまとまっていても構いません。
- ソースコードのビルド方法・実行手順について記したテキスト
  - `README.txt`、`README.md` などのファイル名にしてください。
  - 補足資料として、ソースコードの説明を記述して頂いても構いません。
- 課題 3 で生成した画像ファイル
  - `problem3` というディレクトリを作ってその下に PGM ファイルを置いて下さい。`pgm/1.pgm` に対応するファイルは `problem3/1.pgm`、`pgm/2.pgm` に対応するファイルは `problem3/2.pgm` に置いて下さい。他のファイルについても同様です。
- 課題 2,3,4 のレポート
  - 課題 2 は予測モデルの正解率を、課題 3 は実装した手法とベースライン手法の比較結果を、課題 4 は実験内容と実験結果を記してください。
  - レポートは課題 2,3,4 を通して A4 用紙で 1 枚もしくは 2 枚以内程度でまとめてください。図や表も含めます。

- `report.pdf`、`report.txt`、`report.md`、`report.doc` などのファイル名にしてください。

評価基準

提出物の評価にあたって以下のような要素を考慮します。(必須では無いので、時間がない場合はこれらを満たしていなくても大丈夫です)

- ソースコードが他人が読んで理解しやすいものになっていること。
- ソースコードが正しいものであることを検証するためにある程度の単体テストや検証のためのコードが書かれていること。
- 提出物を見て追試がしやすい形になっていること。
- レポートが要点についてわかりやすくまとまっていること。

提出方法

上記の提出物を単一のパスワード無し zip ファイルにまとめ、こちらの専用フォームより応募してください。締切は日本時間 2018 年 5 月 14 日（月）23:59 です。

問い合わせ

問題文に関して質問がある場合は intern2018@preferred.jp までご連絡ください。問題文に訂正が行われた場合は応募者全員にアナウンスいたします。なお、アプローチや解法に関する問い合わせにはお答えできません。

問題文

本課題では敵対的入力 (Adversarial Examples) を扱います。手書き文字画像のデータセットと、その文字種を識別するための予測モデルが与えられるので、画像に微小な摂動を加えて予測モデルが正しくない結果を返すようなものに改変して、予測モデルを騙すような入力を作ることが目的です。

敵対的入力を考える動機としては、機械学習・深層学習の技術が進歩してきて予測モデルを自動運転や画像認証などの重要な用途に使うことが考えられるようになってきた一方で、悪意のある利用者が予測モデルを悪用しうる可能性を考えることがあります。

課題 1

準備のために以下に挙げるベクトル・行列演算の関数を全て定義して、それぞれについてそれらが正しく動くことを確認する簡潔なコード (単体テスト) を書いて下さい。ベクトル・行列のデータ型や、以下の関数のインターフェースは自由に定義して下さい。ベクトルは全て列ベクトルであるとします。

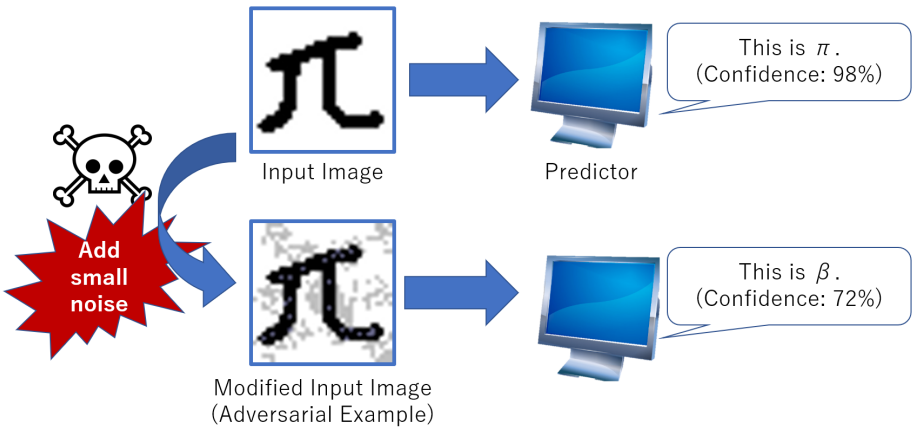

Figure 1: 敵対的入力のイメージ

- ベクトルの加算関数: 2 つのベクトル x と y が与えられるのでその和 $x+y$ を計算して返す。
- 行列ベクトル積関数: 行列 A とベクトル x が与えられるので、その積 Ax を計算して返す。
- 行列転置関数: 行列 A が与えられるのでその転置 $A^\top$ を計算して返す。
- ReLU 関数: ベクトル x が与えられるので、i 番目の要素が $\max(0, x_i)$ であるようなベクトル (要素数は x と同じ) を計算して返す。
- Softmax 関数: ベクトル x が与えられるので、i 番目の要素が $e^{x_i}/\sum_{j=1}^{n} e^{x_j}$ であるようなベクトルを計算して返す。ここで、n は x の要素数を表す。

課題 2

課題ディレクトリに `pgm` というディレクトリがあります。このディレクトリには手書き画像が PGM という形式で複数枚含まれています。画像は全て 32×32 のグレースケールで、各ファイルは以下のようなフォーマットになっています。

```
P2
32 32
255
(ピクセル(1,1)の画素値)(ピクセル(1,2)の画素値) ... (ピクセル(1,32)の画素値)
...
(ピクセル(32,1)の画素値)(ピクセル(32,2)の画素値) ... (ピクセル(32,32)の画素値)
```

ここで、上から 3 行までは固定値です。4 行目以降については、上から i 行目、左から j 列目のピクセルを (i, j) と表しています。画素値は 0 から 255 までの整数値です。

まず、画像ファイルを読み込み、各要素が 0 以上 1 以下の実数値に正規化された 1024 次元の浮動小数点数型のベクトルに変換する関数を実装して下さい。変換後のベクトルを x として、$x_1, ..., x_{32}$ はそれぞれピクセル (1,1), ..., (1,32) に対応し、$x_{33}, ..., x_{64}$ はそれぞれピクセル (2,1), ..., (2,32) に対応しているようにして下さい。他の要素についても同様です。値の正規化は、元の画素値を 255 で割ることで行って下さい。

これらの画像ファイルは手書き文字を表していて、全部で 23 種類のラベルから成り立っています。`labels.txt` に、それぞれの画像ファイルがどのラベルに属するのかを記しています。`labels.txt` の 1 行目には `pgm/1.pgm` のラベルが、2 行目には `pgm/2.pgm` のラベルが、という風に書かれています。ラベルは 1 から 23 の番号が割り振られています。

これらの手書き画像を分類する予測モデルのパラメータを記したファイルが `param.txt` にあります。予測モデルは 1024 (=32×32) 次元ベクトルを受け取り 23 次元の実数ベクトルを返します。具体的には次のようなものになっています。以降、$N = 1024, C = 23$ とおきます。

- 予測モデルはパラメータとして 6 つの変数 $W_1, b_1, W_2, b_2, W_3, b_3$ を持ちます。W_1, W_2, W_3 は行列で、次元はそれぞれ $H \times N, H \times H, C \times H$ です ($H = 256$)。また、b_1, b_2, b_3 はベクトルで、次元はそれぞれ H, H, C です。
- 入力ベクトル $x \in \mathbb{R}^N$ に対して、予測モデルの出力を $f(x) \in \mathbb{R}^C$ と表すことにします。これは以下で計算されます。
  - $a_1 = W_1 x + b_1$
  - $h_1 = \mathrm{ReLU}(a_1)$ (ここで、ReLU は課題 1 で定義した ReLU 関数を指します。)
  - $a_2 = W_2 h_1 + b_2$
  - $h_2 = \mathrm{ReLU}(a_2)$
  - $y = W_3 h_2 + b_3$
  - 各 $i = 1, ..., C$ について、$f(x)_i = e^{y_i} / \sum_{j=1}^{C} e^{y_j}$.
- 定義から分かるように、出力 $f(x)$ は各要素が非負で和が 1 であるような確率ベクトルになっています。出力の i 番目の要素は、x が i 番目のラベルに属している確率値を表すようになっています。確率値が最大となるインデックス $\arg\max_{i=1,...,C} f(x)_i$ を予測モデルの推定するラベルとします。

`param.txt` は以下のようなフォーマットになっています。各行列やベクトルはスペース区切りのテキストになっています。

```
(行列 W_1 をテキストにしたもの、H 行)
(ベクトル b_1 をテキストにしたもの、1 行)
(行列 W_2 をテキストにしたもの、H 行)
(ベクトル b_2 をテキストにしたもの、1 行)
(行列 W_3 をテキストにしたもの、C 行)
(ベクトル b_3 をテキストにしたもの、1 行)
```

パラメータファイル `param.txt` と `pgm` ディレクトリ内の画像ファイルを読み込み、予測ラベル $\arg\max_{i=1,...,C} f(x)_i$ を計算して出力するコードを書いて下さい。また、`labels.txt` と比較して、`pgm` 内の画像データを正しく推定できている正

解率を出力して下さい。正しいコードが書けていれば、8 割以上の正解率が出ます。レポートに、予測モデルの正解率を記して下さい。

課題 3

この課題では Fast Gradient Sign Method (FGSM) と呼ばれる敵対的入力の手法を実装することにします。入力画像 x に微小な摂動 $\varepsilon \in \mathbb{R}^N$ を加えて、元の入力ととても良く似た入力 $\tilde{x} := x + \varepsilon$ を作り、予測モデルが推定を誤る、つまり $f(\tilde{x})$ が正解から程遠いようなものに作り変えるのが目的です。FGSM では問題を簡略化するために予測モデルのパラメータは既知であると仮定します。これは以下のような手法です。

- t を x の正解ラベルとします ($1 \leq t \leq C$)。予測モデルの出力が正解からどれだけ程遠いかを表す指標としてクロスエントロピー誤差と呼ばれる実数値関数 $L(x)$ を次で定義することにします。
  - $L(x) := L := -\log f(x)_t = -y_t + \log \sum_{j=1}^{C} e^{y_j}$
- FGSM では、入力に加える摂動 ε として L の x に関する微分 $\nabla_x L$ からその符号値を取ったものを、適当に定数倍したものを考えます。つまり、$\varepsilon := \varepsilon_0 \mathrm{sign}(\nabla_x L)$ とします。ここで $\varepsilon_0 > 0$ は FGSM のパラメータであり、sign はベクトルの各要素について、もし正なら +1 にして、そうでないなら -1 に置き換えるような関数を指します。ベクトルの微分 $\nabla_x L$ の次元は x と同じになることに注意して下さい。
- $\nabla_x L$ は連鎖律を用いて以下で計算することができます。
  - $\nabla_y L = -\delta_t + f(x)$ (ここで、δ_t は t 番目の要素だけが 1 でそれ以外が 0 であるようなベクトルとします。)
  - $\nabla_{h_2} L = W_3^\top (\nabla_y L)$
  - $\nabla_{a_2} L = \mathrm{Backward}(\nabla_{h_2} L, a_2)$ (ここで、次元が同じ 2 つのベクトル p, q に対し、$\mathrm{Backward}(p, q)$ は i 番目の要素が $q_i > 0$ ならば p_i に等しく、そうでないなら 0 であるようなベクトルを返す関数とします。)
  - $\nabla_{h_1} L = W_2^\top (\nabla_{a_2} L)$
  - $\nabla_{a_1} L = \mathrm{Backward}(\nabla_{h_1} L, a_1)$
  - $\nabla_x L = W_1^\top (\nabla_{a_1} L)$

FGSM を実装して下さい。`pgm` ディレクトリにある各画像 x について、パラメーター ε_0 を 0.1 で実行したときに得られるベクトル $\tilde{x}$ を PGM ファイルとして出力して提出して下さい。$\tilde{x}$ を PGM ファイルにエンコードするにあたって、連続値を 0 以上 255 以下の整数値に変換する必要があります。適切な手段を用いて下さい。

FGSM の性能を比較するために、FGSM で $\mathrm{sign}(\nabla_x L)$ の部分を代わりに各要素がランダムな ± 1 の値にする場合をベースラインとして考えることにします。パラメータ ε_0 を変化させた場合にアルゴリズムの性能が FGSM とベースラインでそれぞれどのように変わるかを比較して、レポートに簡潔に記してください。

課題 4

これまでの課題で実装した FGSM アルゴリズムを以下に挙げるもののどれか 1 つの方向性で発展させ、レポートに記して下さい。この課題ではライブラリ等は自由に使用して頂いて構いません。

- 入力 x に対して FGSM を適用して出来たベクトルを $\tilde{x}_1$、$\tilde{x}_1$ に対して FGSM を適用して出来たベクトルを $\tilde{x}_2$、... という風にして敵対的入力を作ることも考えられます。このようにした場合にアルゴリズムの性能が上がるかどうかを見て下さい。
- 今回の画像はモノクロ画像であるため、予測モデル側で FGSM で生成された敵対的な入力を防御するための方法として、予測モデルが事前に入力をモノクロ化してしまうというのが考えられます。これに対処する攻撃方法を考えて実装し、その性能を報告して下さい。
- `extra` ディレクトリ以下に、予測モデルのパラメータ値のファイルが複数個あります。FGSM による攻撃に対処するために、予測モデルを使う防御側がこれら複数個のパラメータを読み込んでなんらかの多数決を取るような形に予測モデルを改変することを考えます。このようなを防御手法を実装し、さらに必要であればこれに対処するための攻撃手法を考えて実装し、その性能を報告して下さい。

(課題文ここまで)

備考

手書き文字のデータセットとして HASYv2 (ODbL ライセンス) を使用しました。

Thoma, Martin. (2017). HASYv2 - Handwritten Symbol database [Data set]. Zenodo. http://doi.org/10.5281/zenodo.259444

Fast Gradient Sign Method (FGSM) は Goodfellow 達の次の論文に基づきます: https://arxiv.org/pdf/1412.6572.pdf