破断面解析におけるディープラーニングの活用事例の紹介[†]

山際 謙太*

Application of Deep-Learning for Classification of Fracture Surface's SEM Image.

by

Kenta YAMAGIWA

Key words:

Fractography, Image analysis, Neural network, Image classification

1 結 言

構造または材料の破壊に起因した事故などの調査を行う時に、フラクトグラフィは重要な役割を果たしている。その中で明らかにしていくことは、破壊機構、破壊の起点および最終破断面である。

破壊機構を推定していくためには、走査型電子顕微鏡(SEM)などを用いてストライエーションなど破壊機構に対応した特徴的な模様を探していく必要がある。しかし、例えば疲労試験機を使って作成した破断面には幅の整ったストライエーションが観察されるが、実機に生じた破断面の場合は、錆または圧縮荷重の作用などによる破断面の潰れなどがあるため、破壊機構の推定が困難である場合もある。また、こうした模様の解釈は観察者の経験に依存しており、客観性に欠く結果となってしまう。一方で、熟練した解析者は年々リタイヤしており、解析技術の伝承も問題となっている。このような状況において、解析の初心者をサポートする仕組みは重要である。

一方で、自動車の自動運転などではカメラの画像から自動車または歩行者などを抽出したりする画像処理または画像認識の技術が開発されている。こうした画像認識には機械学習が用いられており、その中でもディープラーニングが用いられている。

ディープラーニングでは多層型ニューラルネットワークに膨大な量のデータを学習させ、推論用モデルを作成する¹⁾。かつてはコンピュータの計算能力の関係から多層にすることが困難であったが、近年は容易になってきたこともあり、多層型ニューラルネットワークは様々なところで活用されてきている。その一つが画像認識の技術である。ディープラーニングを用いた場合、例えば人が書いた数字の画像を認識し、コンピュータに数値として認識させることが可能である。また、街の風景を撮影した写真から、自動車、歩行者または自転車などを認識し、その領域を示すことも可能となっている。

これまでフラクトグラフィの分野で行われてきた機械学習では、エキスパートシステムを活用した破断面の分類などがある²⁾。機械学習とディープラーニングの違いの一つに、特徴量の決め方がある。これまでの機械学習は技術者が画像を解析し、画像分類に適した特徴量を考えて

分類用モデルなどを構築していた。例えばディンプルであれば、円形状の模様を捉える分類用モデルを構築していた。しかし、現実の破断面は綺麗な円形ではなく、伸長ディンプルもあり、場合によってはゴミもあり、さらには傾斜した破断面であれば画像の部位によってフォーカスが異なることもあることから、綺麗な円形となることはほとんど無い。こうした場合に、従来型の機械学習ではそれぞれの状況に応じた処理を考える必要があることから、ロバストな分類モデルを構築することが困難であった。

一方でディープラーニングは、学習過程で特徴量が決められていく。そのためにはビックデータと呼ばれる膨大なデータとGPU等を積んだ高速な計算機などの計算資源が必要であるが、技術者側で特徴量を決定する必要がなくロバストな分類モデルを構築することが可能である。

こうしたディープラーニングを破断面の解析に応用することで、解析者の経験に依存せず破断面の性状分類が可能となると考えた。もしくは、最終的な判断は観察者が行うものの、その観察者をサポートする仕組みとしても有効であると考えられる。材料強度に関連した研究とディープラーニングの活用という点では、足立ら³⁾は、金属のミクロ組織にディープラーニングを活用し、自動分類に対しての成果を挙げている。

本項では予め破断面模様が同定されている走査型電子顕微鏡(SEM)画像を多層型ニューラルネットワークに学習させた。そして、推定される破壊機構と予め同定されていた破壊機構を比較し、正答率などを検討した。これらについて報告する。

2 破断面自動分類用多層型ニューラルネットワーク

2・1 畳み込みニューラルネットワークの活用

画像分類を行うための多層型ニューラルネットワークで最も用いられているのが畳み込みニューラルネットワーク(Convolution Neural Network: CNN)¹⁾である。例えば手書きの0から9の数字の画像を、CNNに入力して、コンピュータ上の数値として0から9に分類するCNNなどがある¹⁾。

破断面も画像であり、破壊機構に対応した模様が観察されることからCNNを活用して破断面の模様の分類を試みた。

[†] 原稿受理 令和2年5月15日 Received May 15, 2020 ©2020 The Society of Materials Science, Japan

* 正 会 員 (独)労働者健康安全機構 労働安全衛生総合研究所 〒204-0024 清瀬市梅園

National Institute for Occupational Safety and Health, Japan, Umezono, Kiyose 204-0024.

CNN は以下の 3 種類の層を組み合わせることで構成されている。

1. 畳み込み層
2. プーリング層
3. 全結合層

これらの層の個々の役割¹⁾については省略するが、これらを多層に組み合わせることで破断面画像の特徴を CNN が認識し、最終的には式(1)に示す Softmax 関数により各ラベルの確率が求められる。ここではラベルがディンプル、擬劈開などの破断面の模様と該当する。

$$y_i = \frac{e^{x_i}}{\sum_{j=1}^{N_{\text{label}}} e^{x_j}} \quad (1)$$

ここで N_{label} は分類に用いたラベル数、 i はラベル用の添字、 x_i は画像から CNN により求めた最終層の出力である。

この確率が最大値となるラベルを解析した破断面画像が該当する破断面のラベルとしている。例えばある画像の破断面模様を推定するとして、対象となる破断面の模様がディンプル、擬劈開そしてストライエーションの 3 分類($N_{\text{label}}=3$)であるとする。そして、CNN の計算途中の結果として、ディンプルである確率が 0.1、擬劈開である確率が 0.3、ストライエーションである確率が 0.6 と算出されるとする。その場合、確率が最大であるストライエーションがその画像の破断面模様として推定される。

機械学習には教師付き学習と教師無し学習と 2 種類に大別されるが、ここでは教師付き学習を選択した。予め破断面の画像と既に推定された破断面模様(ラベル)のセットを用意する。そして、この画像群をトレーニング(training)用と検証(validation)用に分割する。

次に 1 回目の学習を実施する。最初にトレーニング用画像を CNN に入力して分類した結果を得る。これをラベル付けと呼ぶ。すべてのトレーニング用画像に対してラベル付けを行うことをエポック(epoch)と呼ぶ。次に、検証用の画像をネットワークに読み込ませ、ラベル付けを行う。トレーニング用、検証用それぞれにラベル付けの結果と、正解のラベルを比較し、正答率を計算する。そして、トレーニング用の画像に対して不正解だったものが正解になるように誤差逆伝ば法を用いて¹⁾畳み込み層のフィルタの値を決定する。これを用いて再び次のエポックを実施する。これを繰り返して行くことで、フィルタの数字が変化し、正答率(accuracy)が増加する。そして、指定されたエポック数を学習した後にネットワークの構造と各層のフィルタの値をまとめた推論用モデルファイルが作成される。

CNN の性能は、正答率と誤差(loss)によって評価される。また、それぞれがトレーニング用データと検証用データにおいて求める、都合 4 パラメータを考慮しながら CNN の性能を評価していく。

正答率は、正答した画像数(N_{correct})を全画像数(N_{total})で除した値(式(2))を用いた。

$$\text{Accuracy} = \frac{N_{\text{correct}}}{N_{\text{total}}} \quad (2)$$

誤差は、交差エントロピー(式(3))を用いた。

$$\text{loss} = - \sum_{n=1}^{N_{\text{total}}} \sum_{k=1}^{N_{\text{total}}} d_{nk} \log(y_k) \quad (3)$$

ここで、 y_k は式(1)の Softmax 関数の出力ベクトル、 d_{nk} は正解のラベルを表すベクトルである。例えば前例の $N_{\text{label}}=3$ の事例で 10 番目の画像の正解がディンプル(最初のラベル)であれば、 $d_{10}=(1,0,0)$ となる。

計算結果によっては、正答率が高いが誤差が大きいと言うケースもある。こうした場合は、例えば前述の $N=3$ の例においては、それぞれの模様の確率がディンプルは 0.3、擬劈開は 0.3、ストライエーションは 0.4 といった結果の場合である。式(1)の Softmax 関数を適用した後の結果では、ストライエーションが正解であるため、正答率は良いが、誤差が大きいことになる。正答率が高く、かつ誤差が小さい CNN を開発することが重要であり、そのためには CNN の構造を検討する、もしくは後述するハイパーパラメータの検討などが行われる。

2・2 解析条件

2.2.1 解析に使用した破断面画像 破断面の画像は以下の破断面を観察した。例を Fig. 1 に示す。Fig. 1 はそれぞれ以下の破断面の画像である。

1. ぜい性破壊させたシャルピー衝撃試験後の破断面 (ファセット)
2. 延性破壊させた丸棒の引張試験 (ディンプル)
3. 疲労き裂進展後の CT 試験片 (ストライエーション)

ただし、疲労に関しては、低倍率から高倍率まで倍率を遷移させると、低倍率ではストライエーションが観察されないが、高倍率では観察されると様相が大きく変化する。このことから、観察倍率が 250 倍以下の画像については「低倍率の疲労」、250 倍以上は「中倍率の疲労」、ストライエーションが画像全体に観察されるものは「ストライエーション」とラベルをつけて分類することとした。またこれらに加えて、粒界破壊の破断面を観察し、合計 6 ラベルの分類とした。

さらに同一の SEM で撮影された画像のみの解析であるとして、その SEM のコントラスト等の特徴が解析に影響することを避けるため、Google の画像検索なども活用して、ディンプル等のキーワードを用いて検索した結果から得られる画像も一部画像群に入れた。

ディープラーニングを行うための画像の枚数は基本的には多い方が良いとされているが、一方で破断面の画像の枚数を大量に用意することはスキャン時間等を考慮すると困難である。このような場合、データオーグメンテーション(水増し)と呼ばれる作業を行い、画像の枚数を増やすことがある。具体的には、画像の切り出し、回転、拡大

縮小等を行い、同一の画像から複数の画像を作成することである。

Fig. 1 解析に使用した破断面の走査型電子顕微鏡画像の例 (上からディンプル, ファセット, ストライエーション)

本解説で行った計算では、具体的には 2400×1800 程度の画素数を持った SEM 画像から 598×598 画素で切り出し、その画像を 90, 180 および 270 度回転させた。さらに、その 4 種類の画像を上下に反転させた。したがって、1 枚の切り出した画像からは 8 枚の画像を作成している。これを元の画像からランダムに 64 枚程度抜き出しているため、例えば 1 枚の SEM 画像から 512 枚の画像を作成し

ていることになる。もちろんこれは元の画像の解像度にも依存している。

解析に使用した元の画像ファイルとオーグメンテーション後の画像枚数を Table 1 に示す。検証用の画像がトレーニング用画像の中に入ると、検証の意味が無くなる。したがって、オーグメンテーション前の画像の中で 8 割をトレーニング用とし、残りの 2 割を検証用と分割した上でオーグメンテーションを行なった。

Table 1 ラベルと画像枚数

ラベル名	初期枚数	オーグメンテーション後の枚数(万枚)
ディンプル	416	12.2
擬劈開	400	12.5
疲労(低)	358	10.6
疲労(中)	638	21.3
ストライエーション	701	22.6
粒界割れ	240	6.1

2.2.2 CNN に必要なハイパーパラメータの決定 計算に使用したコンピュータは HPC システムズ製である。CPU は Intel Xeon (E5-2640, clock 2.4GHz, 10 core) で 20 CPU、メモリーは 512G byte、GPU は NVIDIA 製の TITAN V を 10 枚搭載している。

ディープラーニングの計算を行うためには、専用のプログラムライブラリ(Pytorch, Caffee, Tensorflow, Keras など)が存在しているが、ここでは Google が提供している Tensorflow^{4),5)}と Keras を使用した。言語は Python 3.6 を使用した。また、ライブラリには典型的な CNN が予め用意されており、例えば畳み込み層を数層組み合わせたシンプルな CNN から、VGG16, GoogLeNet, ResNet 等の画像分類コンテスト(ILSVRC)で上位に入賞するような CNN まで準備されている。一般的には独自に CNN を開発するよりは、こうした実績のある CNN を活用する方が開発速度も速くなり簡便である。今回は GoogLeNet の中で Inception V3⁶⁾と呼ばれる CNN を活用した。

CNN を実行するためには、技術者側で設定しなければならないパラメータ (ハイパーパラメータ) がある。例えば初期学習率(lr), 学習ごとに kernel は変化するが、その変化のさせ方を決定するオプティマイザ(optimizer), およびオプティマイザ内部で必要とする係数, 1 回の学習で学習させる画像枚数であるバッチサイズ(batch size), その他にも過学習を抑制するための dropout 率, kernel 正則化率等々, 多くのパラメータを設定する必要がある。また、ハイパーパラメータの初期値は技術者の経験に依存することが多い。しかしながら、パラメータの数も多いことと、1 回の計算時間も数時間から長いもので数日とかかることから、最適化パラメータセットを見つけることが非常に困難である。

例えばパラメータを2種類(P1, P2)として、正答率が最大となるための P1, P2 を求めることとする。しかし、正答率と P1, P2 の関数は未知(ブラックボックス関数)であり、最大値となる P1, P2 を求める事は困難である。

Fig. 2 パラメータ探索の概念図

Fig. 2 はパラメータの探索方法の概念図である。P1, P2 のパラメータを最適化すると仮定して、評価する値を円の面積で示している。CNN の場合、この値は正答率もしくは誤差を用いることが多い。特に検証用データの誤差を用いることが、汎化性能のチェックとなるためよく用いられる。こうした問題に対して、Fig. 2 の左上に示すようなグリッドサーチ、または Fig. 2 の右上に示すランダムサーチを行ってきた。

グリッドサーチは、パラメータの空間を一定の間隔で区切り、すべてのパラメータの組み合わせを計算して、その中で評価する値が最大となるパラメータセットを P1, P2 の最適値とする手法である。ランダムサーチは、パラメータの空間をランダムに探索し、その中で得られた評価値の最大値となるパラメータセットを最適値とする。

そして、その中で最も性能が良いパラメータを活用していたが、もちろんこれらの手法で得られたパラメータセットが最も性能が良いという保証は無い。局所解を求めている可能性もある。

そこで、近年はベイズ最適化を活用したパラメータ最適化⁷⁾(Fig. 2 左下)が主流となっている。

ベイズ最適化は最初の数点はランダムに決定して計算を行う。その結果とベイズの定理から確率的に評価値が最大、もしくは最小となると推定されるパラメータセットの近くを重点的に探索する。また探索した結果を元に次の探索するパラメータセットを決定していくことから、効率的なパラメータセットの探索が可能となる。

こうしたベイズ最適化の手法なども Optuna⁸⁾などのライブラリが充実しているので、こちらを積極的に活用することで効率的に CNN を開発していくことが可能である。

3 解析結果

Fig. 3 に計算結果の例を示す。横軸はエポック、縦軸は上段が正答率、下段が誤差である。

Fig. 3 エポックと正答率および誤差の関係

約 30epochs で、トレーニング用のデータについては 90%の正答率となった。最初が 60%程度の正答率なので、学習が進展するにつれ、正答率が向上していることがわかる。一方で、テスト用のデータについては、92%程度の正答率となっている。なお、計算時間は GPU を 2 枚使用して 1epoch が約 1 時間なので、70~80 時間の計算時間である。

一方で Optuna によるハイパーパラメータの最適化にはこの計算を繰り返す必要がある。使用している計算機は GPU が 10 枚使用できることから、ハイパーパラメータのセットを 5 セット並列に実行した。さらにこの 5 パターンが終わった結果を考慮した上で次のハイパーパラメータのセットが決定され、次の 5 セットが計算される。この繰り返しを 6 回実施し、合計 30 セットのハイパーパラメータの計算を行った。よって計算時間はおよそ 20 日程度となる。最適化の評価値には、検証用データの誤差を活用し、かつ、誤差が最小になるようにパラメータの探索を行った。Table 2 にその例を示す。

ただし、これらの値は学習用データが変わればまた変化するため、これらの値で計算を行えば必ず良い結果が得られるわけではない。

CNN の計算で注意しなければいけないことに過学習がある。それはトレーニング用のデータに CNN の kernel の値が特化しすぎてしまい、検証用のデータについては正答率が上がらないといった状況が生じる。したがって、過

学習が生じている場合は汎化性能が低いCNNができてい
ると考えられ、他から画像を持ってきて解析を行っても
正答率の低いCNNができてしまう。

Table 2 Optuna を使い得られたハイパーパラメータの
例

パラメータ名	値
学習率	2.68×10^{-6}
Dropout 率	0.59
正則化率	9.54×10^{-8}
Optimizer	SGD
Momentum (SGD 用)	0.464
Decay (SGD 用)	1.67×10^{-6}
Nesterov (SGD 用)	無し

今回の計算ではトレーニング、検証用のどちらの正答
率も同じように上昇し、誤差についても同じように減少
していったことから、過学習になっているとは言えない。
ただし、過学習であるか否かの定量的な指標は無い。

また、学習を行う前に画像をトレーニング用と検証用
に分類したが、分類が異なれば計算結果である推論用モ
デルファイルも異なってくる。したがって、可能であれば
K 交差検証などを行うことでCNNおよびモデルファイル
の性能の評価をすることが望ましい。しかし、これを行う
ためには十分な計算資源が必要であると考えられる。

以上、未だ開発途上のところも多いが、破断面の分類で
6 種類の分類は、CNN を活用することで十分行えること
の例を紹介した。

4 分類用モデルファイルの活用

4.1 推論用ソフトウェアの作成

ディープラーニングの計算を行い分類用のモデルファ
イルを作成した次は、これを活用してトレーニング用また
は検証用のデータに無い画像を実際に分類するための
推論用ソフトウェアを作成する必要がある。

例としてウェブブラウザを介して画像を推論用プログ
ラムに入力し、結果を表示するプログラムの例を Fig. 4 に
示す。こうしたプログラムも Python なら flask⁹⁾等のライ
ブラリを活用することで構築することができる。

4.2 結果の信頼性

近年は学習用データに対する信頼性と、推論結果に対
する説明責任も求められるようになってきた。

例えば本報で使用した画像データと分類結果について
は、事前に筆者が分類していることと、Google 等の画像
検索の結果を入れていることなどから、その分類結果の
信頼性について客観性が高いとは言えない。また、それを
学習した結果であるモデルファイルの性能は、筆者らの
分類能力に影響を受けていると言える。

一方で、こうした技術者の分類能力を極力排除した学
習データを作成しようと考えたとき、破壊試験により既

Fig. 4 ウェブブラウザを使用した破断面模様推定ソフ
トウェアの例

Fig. 5 CNN の結果のヒートマップ表示

知の破壊機構による破壊を行い、その破断面のみを観察
し、学習用データとする方法が考えられる。こうした方法
により、破壊試験と破断面画像のセットに技術者の主観
を入れない、言わば客観性の高いデータを作ることは可
能である。しかし、こうした言わば教科書のような破断面
に対して、実機の破断面は潰れ、汚れまたは錆などが多い。
従って、実験のみで作成した学習用データと、実機の破断
面とは少なからず乖離がある。そのため、こうした破壊
試験のみのデータで学習することは、学習用データとし
て客観性は高いが、十分な汎化性能があるモデルファイ
ルが構築できる可能性については議論として残されてい
ると考えられる。

また、推論結果に対しての説明責任については、Fig. 5 に示したヒートマップなどが提案されている¹⁰⁾。これは推論する画像の中で判断に至った領域を赤く示している。こうした画像を提示することで、推論に至った根拠を示すことができる。また、これ以外では Inference function¹¹⁾ と呼ばれる手法もあり、学習用画像の中から最も推論用画像に近いと判断される画像を提示する手法である。こうした様々な手法が現在も推論結果の説明のために提案されている現状である。

5 結言

本稿ではディープラーニングを活用して破断面に特徴的な模様を自動分類する試みと、その結果について報告した。破断面は今回の6ラベル(ディンプル・ファセット・粒界・疲労(低倍率・中倍率・ストライエーション))の分類については、Inception V3 等の CNN を用いることで、SEM 画像を十分に分類することができると考えられる。

一方で、破断面も材料によっても例えばストライエーションの見え方などは大きく異なることから、より多くのデータを収集し、継続的に学習させてモデルファイルを改善していくことが今後も重要である。また、実機の破断面においては、潰れ、錆なども多いことから、こうした破断面では無い模様についても分類できるようになると、より精度が高まっていくと考えられる。

今回の学習用データには含めなかったが、例えば鋳鉄またはアルミダイカストの様に破断面が繰り返し荷重で破断したのか、もしくは過荷重による破断であるかを判断するのは非常に困難である。こうした、これまで判断が困難であった破断面に対しても破壊試験の破断面 SEM 像の蓄積と学習により分類が可能となり、判断が可能になっていくことが期待される。

こうしたディープラーニングの仕組みを作るためには破断面の量とディープラーニング等の機械学習に詳しい人間が必要である。したがって、研究者個人で例えば破断面を多く集めることは非常に困難であることから、横浜国立大学に”フラクトグラフィとディープラーニングの融合研究コンソーシアム(FraD)”という研究コンソーシアムを2019年4月より立ち上げて、破断面の収集などの活動を行なっている。本解説記事も FraD 関係者の協力があっての解説であることを付記し、謝辞といたします。

参 考 文 献

- 1) T. Okatani, “Shinsou Gakusyu”, Koudansha(2015).
- 2) K. Komai, K. Minoshima and M. Koyama: "Development of Diagnostic Expert System for Environmentally Induced Cracking and Importance Evaluation of Knowledge" Trans. JSME, Series A. Vol.57. 188-194 (1991).
- 3) Y. Adachi, et. al., “Microstructure Recognition by Deep Learning”, Tetsu-to-Hagane, Vol.12 (2016), No.12, pp. 62-69.
- 4) Tensorflow, <https://www.tensorflow.org/>
- 5) K. Ariyama, “Tensorflow hajimemashita”, NextPublishing.
- 6) Christian Szegedy et al., “Going Deeper with Convolutions”, arXiv (2014), <https://arxiv.org/abs/1409.4842>
- 7) C.M. Bishop, “Pattern Recognition and Machine Learning”, Springer, 2006.
- 8) Optuna, <https://preferred.jp/en/projects/optuna/>
- 9) Flask, <https://flask.palletsprojects.com/>
- 10) R. R. Seleveraju et al., arXiv (2017), <https://arxiv.org/abs/1610.02391>
- 11) Pang Wei Koh and Percy Liang, “Understanding Black-box Predictions via Influence Functions”, arXiv (2017), <https://arxiv.org/abs/1703.04730>