

折れ線グラフ（プロット）の全自動数値化プログラム

吉武道子*, 河野敬, 門平卓也

物質・材料研究機構, 〒305-0044 茨城県つくば市並木 1-1

*e-mail: yoshitake.michiko@nims.go.jp

A program for fully automatic conversion of line plots in scientific papers into numerical data has been developed. By the conversion of image data into numerical data, users can treat so-called ‘spectra’ such as X-ray photoelectron spectra and optical absorption spectra in their purpose, plotting them in different ways such as inverse of wave number, subtracting them from users’ data, and so forth. This article reports details of the program consisting of many parts, with several deep-learning models with different functions, elimination of literal characters, color separation, and so forth. Most deep-learning models achieve accuracy higher than 95%. Its usability is demonstrated with some examples.

Keywords: Image data, Numerical data, Automatic conversion, Deep learning, Plot with multiple line

1. はじめに

科学論文や書籍、その他に掲載されている折れ線グラフを数値データに変換するプログラム（“waveform digitizer”）を開発した。それらの数値データがあれば、ユーザーが自身のデータをプロットする際にレファレンスとして用いることができる。例えば、誘電率の温度依存性のプロット画像を数値化することで、横軸を温度、縦軸を（誘電率－1）の逆数としてプロットし直し、プロットの傾き＝キュリー定数を求めることができる。他の例として、他の研究者の論文中のプロットを数値化して、その数値を自身のデータと同じグラフ上にプロットして比較したり、自身の数値データとの間で引き算したりスペクトルを合成したり、デコンボリューションしたりするのに利用できる。

プロット画像を数値化する市販プログラムは有るが、それらはプロット枠を指定するとか折れ線の本数を手入力するなど、ユーザーが何らかの操作をしなければならない。それに比べ、本プログラムは全自動であり、ユーザーがコンピュータの前に座っている必要はない。それは、このプログラムが、NIMS（物質・材料研究機構）の Materials Data Platform [1-3]の TextDataManagement システム内の FigResourceMiner (TDM-FRM) に内包して使用するために開発されたからである。TDM-FRM では、ユーザーは科学論文中の折れ線グラフの数値を CSV [4]ファイルとしてダウンロードすることができる。Fig.1(a)に、TDM-FRM システム全体の模式図を示した。本報のプログラムは図の pre-treatment の一部として用いられるもので、図中“this program”と記してある。TDM-FRM のユーザーが興味ある折れ線グラフを選ぶと Fig. 1(b) のような画面が現れる。画面右上のボタンをクリックするとこの折れ線グラフの数値データが CSV 形式でユーザーのコン

コンピュータにダウンロードされる。TDM-FRM システムでは、科学論文の XML [5] ファイル内の画像ファイルが抽出され png 形式 [6] に変換して特別なフォルダーに格納されている。本プログラムは、この png ファイルを読み込み、数値化対象のファイルを選別し、数値化を行って、png ファイルが格納されているフォルダーに数値を CSV ファイルとして保存する。これらのプロセスはユーザーが折れ線グラフを選ぶ前にすべての画像ファイルに対して行われており、ユーザーの要求に応じて既に保存されている CSV ファイルをダウンロードするようになっている。

本稿の執筆時点では、元となる XML ファイルの使用に関する出版社との契約の関係で、TDM-FRM は NIMS 職員のみ利用可能である。しかし、本プログラム自体は論文が発刊されれば Materials Data Platform の Material Data Repository (MDR) を通じて、外部のユーザーに公開予定であり、ユーザーは各自の png 形式の画像ファイルを数値化することができる。本プログラムは Python [7] で書かれている。本プログラムを TDM-FRM から独立して利用したいユーザーに必要な詳細情報は、プログラムとともに MDR に挙げられる[8]予定である。

2 プログラム

2.1 数値化の対象となるプロットの種類

本プログラムは折れ線グラフを数値化するが、画像フォルダーには折れ線グラフではない png ファイルが大量に含まれている。したがって、まず、数値化すべき png ファイルを選別する。現段階では、一つの png ファイルに複数のプロットを含むものは数値化の対象外である。さらに、以下の条件を満たす必要がある。x 軸・y 軸がプロットの下側・左側にくっきりと引かれていること、シンボルの有無を問わず折れ線グラフであること（データ点を通らないフィッティングラインではないこと）。Fig. 2 に本プログラムで数値化できるグラフの例(a)と数値化できないグラフの例(b)を示した。数値化できないのは、棒グラフ (Fig. 2(b)の左上)、フィッティングラインのグラフ (下の真ん中)、一つの横軸 x 値に対し複数の縦軸 y 値を持つ折れ線グラフ (上の真ん中)、記号のプロットで折れ線が無いもの (右上)、一つの png ファイルに複数のプロットがされているもの (右下) である。

2.2 全体のデザイン

本プログラムの全体デザインを Fig. 3 に示した。Png 形式の画像ファイルが入力され、多くのアルゴリズムを用いてコンピュータ処理され、最終的に数値データが CSV 形式で出力される。プログラムは全て Python 3.6 [9] で書かれ、深層学習には Tensorflow [11] をバックエンドとする Keras [10]を用いている。

Png ファイルはまず、数値化の対象となる折れ線グラフかどうかの選別にかけられる（深層学習モデル A を使用）。対象として選ばれた画像のファイル名を一時保存して後に利用する。選ばれた画像から、OCR (Tesseract OCR [12])を用いて数字やアルファベットのような文字を削除する。深層学習 [13]では、画像のピクセルサイズを全ての画像に対して統一（規格化）する必要がある、プログラムの流れの各所でピクセルサイズの規格化がなされている。深層学習を除く主な操作は、画像を各色毎の画像に分解することで、この際何色に分解するかは密度準拠クラスタリング (DBSCAN [14], scikit-learn モジュール[15]の実装を利用) を用いて自動的に決定される。このプロセスにより、一つの png 画像中は色ごとの画像に分解され、それぞれの画像はグレースケールに変換され、以降の操作がされる。

数値化は、以下の 2 つの理由によりピクセルサイズ 1024×1024 で行うことにした。第一の理由は、多くの学術雑誌が著者に 600dpi 以上の解像度の画像を要求しており、600dpi はカラム当たり 1200 ピクセルに相当し、画像中のプロットエリアは 1000 ピクセル程度と考えられるからである。コンピュータでは二進法でデータが処理され、 2^N (N は整数) で表されるピクセルサイズが好ましいので 1000 に近く 2^N で表されるサイズとして 1024 を選んだ。第二の理由は、単純にコンピュータの性能・メモリー制限及び計算時間からくるものである。また、DBSCAN を用いるプロセスでは、実装モジュールにおける制限から 360×360 のサイズを用いている。深層学習の部分でも、以下に記述するようにいくつかの箇所で小さめのサイズの画像を用いている。

深層学習による数値化で得られる数値は、ゼロと 1 の間に規格化されている。そこで、続くプロセスによりこれらの値を、元の画像（入力画像）に OCR (pyocr [16]) をかけて得られた横軸縦軸のスケールの値を用いて再スケール化して出力としている。これに関しては 2.4 で述べる。

2.3 深層学習部分

異なる目的のために5個の深層学習モデルを作成した。深層学習モデルA, B, D及びEについては基本としてResNet [17]を用いた。入力と出力が同じサイズの画像であるモデルCには, U-Net [18]を用いた。Table 1 にリストした学習モデル名のファイル名で supplement にそれぞれの学習モデルを示している。それぞれのモデルの性能は, テストデータを用いて計算した正答率で与えてある。正答率は, (真陽性+真陰性)の数を(真陽性+偽陽性+真陰性+偽陰性)の数で割ったもので定義している。学習用データの生成について本稿内に簡単に述べるが, 自身で学習用データを作成して深層学習モデルを生成したい読者には, プログラムのソースコード[8]を読む方が理解しやすいと思われる。

画像選別モデルAは, Fig. 4に示したようなラベル付き学習用データを用いて学習を行った。単なる画像選別には高い解像度は不要で, 計算時間とメモリの節約のために深層学習には 256×256 サイズの画像を用いた。画像の選別後(実際には選別された画像のファイル名がメインプログラムに渡される), 画像は 1024×1024 のサイズで続く処理がなされる。各元画像のサイズ(X^0, Y^0)は様々なので, 画像選別の前処理として画像を 256×256 サイズに規格化している。ラベル付学習用データは, 規格化された画像データと正解ラベルから成っている。数値化対象の画像の正解ラベルは“1”, その他は“0”で, このラベルは人手で付与した。学習用画像データは, 物質・材料研究機構が出版社から購入したXMLデータセット(14TB)内のpngファイルからランダムに抽出して用意した。抽出された画像の内, 正解ラベルが“1”の画像(1784個)は, 正解ラベルが“0”の画像(5238個)に比べ非常に少ないため, 4500個の正解ラベル“1”のデータをコンピュータで生成して用いた。本モデルの正答率は0.95である。

原点・軸長予測モデルBは, 個々の画像の原点位置と横軸縦軸の長さをピクセル単位で求めるものである。 1024×1024 ピクセルの画像全体の内, プロットされている領域(スケールや軸ラベルはプロットの外)は, Fig. 5に示すように様々である。このプロット領域だけを取り出すために, 原点位置(左下, 画像の下半分に位置している)(X_{org}, Y_{org}), x 軸・ y 軸の長さ(軸長, L_x, L_y)を学習させる。学習用データはコンピュータで生成した。原点位置(画像を4分割した左下の領域内)と x 軸・ y 軸の長さをランダムに生成させた。この値を用いて四角い枠を描かせて, 画像データとした。正解ラベルは, 画像データを作成するときに用いた原点位置と x 軸・ y 軸の

長さである。全部で10,000個のデータを作成し, そのうち9000個を訓練データとして使用し, 残りはテストデータとした。このモデルでは, 正答率を上げるために, 原点位置と x 軸・ y 軸の正解データは, ニューラルネットワークの最終段階のみで別々に扱った(逆伝搬を独立させた)。原点位置及び軸長予測の正答率はそれぞれ, 0.966及び0.997である。ここで, 「正答」とは, (予測値-テストデータ正解ラベル)をテストデータ正解ラベルで割ったものが, x 軸・ y 軸方向の両方で0.03以下を示す。

整形モデルCは, それ以前のプロセス(Fig. 3参照)で除去できなかったグリッド線やアノテーションを除去するためのものである。Fig. 6に学習用データがどのようなものかを示す。このモデルでは, 学習用入力画像データはグレースケールである。学習用データは以下のようにしてコンピュータで生成した。まず, 正解ラベル画像データを, 数値化モデルEの項で説明するように, ガウス関数を用いて生成した。学習用入力画像データは, 正解ラベル画像データにグリッド線またはアノテーションを追加して作成した。5,000個の学習用データを作成し, そのうち4000個を学習に用い, 残りをテストデータとして使用した。学習用データの数, コンピュータ性能に制限されている。本モデルCの正答率0.78で, 以下のようにして計算した。各画像について各ピクセルの学習用データと正解ラベルデータの値の差を全ピクセルについて足し, その値を正解ラベルデータの各ピクセル値の和で割ったものをエラーと定義した。各ピクセル値とは, ゼロから255の間の値で, 画素の濃淡を表している。エラーの絶対値が0.1を超えたら不正解として正答率を計算した。

本数判別モデルDは, 一つの画像に何本の折れ線が描かれているかを求めるものである。この目的には高解像度は不要で, 360×360 ピクセルの画像を用いた。ラベル付き学習データはコンピュータで生成した。Fig. 7の左側に示したように折れ線の数既知の画像を生成し, 生成の段階で用いた折れ線の数正解ラベルとして用いた。現段階では, もっとも複雑なプロットでも, 同色の折れ線数は6個で収まっている。そこで, 本モデルの学習は, 6本の折れ線まで数の判別ができるように行った。学習用データは以下のようにして生成した。まず, 折れ線数をランダムに選ぶ。次に, 選ばれた数だけの折れ線グラフを生成する(折れ線グラフの生成は数値化モデルEの項を参照)。全部で6000個の学習用データを作成

し、そのうち 5400 個を学習に用い、残りはテストデータとして利用した。モデル D の正答率は 0.995 である。

数値化モデル E は基本的に 1 つのモデルであるが、折れ線グラフの数に応じてモデルの最終段階の逆伝番における正解ラベルの数が異なるので、その部分のみ折れ線グラフの数に応じたモデルになっている。ラベル付き学習用データはコンピュータで生成した。折れ線グラフの波形として次式で表現されるガウス関数を用いた。

$$y = offset + \exp \left\{ -\frac{(x - f_c)^2}{2 \times p_k^2} \right\}$$

この式で、the offset, fc, 及び pk の値は下記の値の範囲内でランダムに選ばれた。 $0.01 < offset < 0.1$, $-0.3 < fc < 1.3$, and $0.02 < pk < 0.3$ 。これらの値を用いて 1 本の波形 (= 折れ線の数値) を生成 (一本目なので、それらの値を offset(1), fc(1) 及び pk(1) とする) したのち、続く波形を Table 2 に示すパラメータを用いて作成し、2 本目以降を波形として用いた。

折れ線グラフの数だけ波形を生成したのち、波形の最大値が 1 になるように (生成したすべての数値が 1 以下になるように)、数値を規格化する。ラベル付き正解データは、0 と 1 の値が 1024 個並んだ列が、折れ線グラフの数だけ並んだものである。この数値から画像化した画像がラベル付き学習データとなる。1 本の折れ線グラフのプロットでは、ラベル付き正解データは 1024 個の数字から成る 1 列のデータ、2 本の折れ線グラフのプロットでは、ラベル付き正解データは 1024 個の数字から成る列が 2 列、すなわち 1024×2 個の数値から成るデータ、というようになる。現時点では、3 本の折れ線グラフまで学習させている。それ以上の本数の学習は、技術的には容易であるが非常に多くの時間がかかる。(一つの数値化対象画像に、一つの色で 4 本以上の折れ線を含む画像はほとんどないので、4 本以上の学習は中断している。Fig. 8 に、本数によってラベル付き学習データがどのように異なるのかを模式的に示してある。10000 個のラベル付き学習データを作成し、そのうち 9000 個を学習に用い、残りをテストデータとして用いた。モデル E の正答率は、1 本の折れ線グラフに対しては 0.96, 2 本に対しては 0.82, 3 本に対しては 0.60 である。なお、「正答」とは、一つの画像内の全部の折れ線について、折れ線の (予測データーテストデータ) をテストデータの値で割った値の絶対値が 0.03 以下である場合を指す。

2.4 再スケール

既に述べたように、数値化モデル E により得られる数値は規格化されており、0 から 1 の間の数値である。したがって、これらの数値を、元のプロット画像と合うように再スケールする必要がある。そのために、OCR (pyocr [16]) を用いて元の画像の x 軸・y 軸の最小値・最大値を下記のように求めた。x 軸線の下にある数字をスケールとみなし、その数字のピクセル位置を出す。見つけた数字の内、最も小さい値 (x_l) の x 方向のピクセル位置 X_l 、最も大きい値 (x_2) とその x 方向のピクセル位置 X_2 を用いて、下記の式により元画像における最小値 (x_{min}) と最大値 (x_{max}) を計算した。

$$x_{min} = x_1 - \frac{X_1 - X_0}{X_2 - X_1} \times (x_2 - x_1), \quad X_0 = \frac{X^0}{1024} \times X_{org}$$

$$x_{max} = x_2 + \frac{X_0 + X_L - X_2}{X_2 - X_1} \times (x_2 - x_1), \quad X_L = \frac{X^0}{1024} \times L_x$$

y 軸に対する同様な式により、元画像における最小値 (y_{min}) と最大値 (y_{max}) が計算される。これらの値を用いて、数値化モデル E により得られた数値が計算した最大値と最小値の間に入るように再スケールする。

3 数値化の例

Fig. 9 に、色が異なる 3 つの折れ線から成るプロットを数値化した例を示す。Fig. 9(a) が元の png ファイルの折れ線グラフである。この折れ線グラフに対しては、プログラム内で 3 つの画像、青、赤、黒に対応する画像が生成され、それぞれの画像はグレースケールに変換される。一連のプロセスの中で、数値化モデル E においては、1 本の折れ線グラフ用の深層学習モデル E を用いて数値化が行われる。これは、各色ごとの画像には折れ線グラフは 1 本だからである。青、赤、黒に対応するそれぞれの 1 本の折れ線グラフに対して得られた数値列を合わせて、一つの CSV ファイルに格納される。CSV ファイルに格納された数値データから再現した折れ線グラフを Fig. 9(b) に示した。x 軸・y 軸ともに再現性は悪くない。学習の際に波形にノイズを入れていないので、再現した折れ線グラフはいつも滑らかな曲線になる。Fig. 10 に示したのは、元の画像の y 軸のスケールが任意で、数値が与えられていない場合 (Fig. 10(a)) の例である。このようにスケールが任意の場合、本プログラムは、Fig. 10(b) のように、軸を 0 から 1 の間の数値にスケールする。

元の画像において多くの異なる色の折れ線が重なっている場合、重なった部分の画像ピクセルの色が折れ線の色とは違う色として記録されており、そのようなピクセル

は色分解後の画像からは抜け落ちる。したがって、色別に分けられた画像の折れ線が切れ切れになる。また、元の画像の折れ線が点線や破線の場合も折れ線は連続ではない。そのような不連続部分を補って連続的に数値化する機能も数値化モデル E に内包されている。

4 まとめ

科学論文中の折れ線グラフを、全自動で数値に変換するプログラムを開発した。変換に当たって、あらかじめ折れ線グラフは論文の XML ファイルから png ファイル形式で抽出しておく必要がある。変換された数値は CSV 形式で供給される。本プログラムは、異なる機能を持つ以下のような多くのパーツから成る：主なパーツは、文字の消去、色分解、5つの深層学習モデル（多くの画像 png ファイルから数値化の対象となる折れ線グラフを選別する画像選別モデル、画像内のプロットされている領域を取り出すための原点・軸長予測モデル、グリッド線やアノテーションを除去するための整形モデル、一つの画像に何本の折れ線が描かれているかを求める本数判別モデル、折れ線グラフを数値に変換する数値化モデル）。

ほとんどの深層学習モデルは、テストデータにおいて正答率 95%以上を達成した。正答率は、より多くの学習を行うことで向上させることが可能である。本稿中に示した数値化の例では、現状でも満足できる数値化のレベルであることを示している。

References

- [1] Tanifuji M, Matsuda A, Yoshikawa H, Proceedings of Advanced Applied Informatics, IIAI International Conference, July 2019
- [2] <https://www.nims.go.jp/eng/research/materials-data-pf/index.html>
- [3] Kadohira T, Kikuchi S, Sakamoto K, Naito H, Tanifuji M, Conference on a Fair Data Infrastructure for Materials Genomics, 3 - 5 June, 2020, virtual meeting.
- [4] CSV (comma separated values) is a data format where each value or item is separated by comma. More explanations at <https://www.ietf.org/rfc/rfc4180.txt>
- [5] XML (extensible markup language) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. World Wide Web Consortium defines the rules. More explanations at <https://www.w3.org/TR/xml11/#sec-xml11>

[6] png (portable network graphics) format is one of file formats for handling bitmap images such as GIF, TIF, and JPEG. Png is license-free format.

[7] python is a programming language that has many modules for machine learning, and free of use.

[8] <https://mdr.nims.go.jp/concern/publications/df65v866p>

[9] <https://www.python.org/downloads/release/python-368/>

[10] Keras is a module for describing simplified deep learning models. More explanations at <https://keras.io/ja/>

[11] Tensorflow is an open source software library for machine learning and can be downloaded from <https://github.com/tensorflow/tensorflow>

[12] OCR (optical character recognition) is a software to convert images of characters into character codes. Tesseract OCR is one of free software downloadable from <https://github.com/tesseract-ocr/>

[13] Deep learning is one of machine learning techniques using multiple layer neural network.

[14] DBSCAN (density-based spatial clustering of applications with noise) is one of algorithms for data clustering. More explanations at <https://en.wikipedia.org/wiki/DBSCAN>

[15] scikit-learn is a free software machine learning library for the Python programming language, which includes a module for DBSCAN. More explanations at <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html>

[16] pyocr is a free OCR software. More explanations at <https://pypi.org/project/pyocr/>

[17] ResNet (residual networks) is a neural network architecture used for image recognition. More explanations at <http://torch.ch/blog/2016/02/04/resnets.html>

[18] U-Net is convolutional network architecture for fast and precise segmentation of images. More explanations at <https://arxiv.org/abs/1505.045>

