

令和5年度 卒業論文

物体検出アルゴリズムによる Ti-6Al-4V の組織定量化法の開発

指導教員

杉尾 健次郎 准教授

佐々木 元 教授

広島大学工学部第一類

機械材料物理学研究室

B205606 大森祐輝

論文概要

本研究ではディープラーニングを用いたインスタンスセグメンテーションと呼ばれる手法を用いて、Ti-6Al-4V の走査型電子顕微鏡(SEM)画像から構成相の特徴を自動で精度よく抽出するプログラムを作成した。使用する画像データとして様々な熱処理条件下での Ti-6Al-4V の SEM 画像を用いる。例えば画像ファイル 20200518 Ti-64 950STQ 800 30min 500x.jpg は以下の手順により得ることができる。まず、等軸 ($\alpha + \beta$) 微細組織をもつ Ti-6Al-4V 合金に対して熱間圧延を行い、950K で固溶処理を行った後、氷水中で急冷する。次に 800K で時効処理を 30 分間行う。最後に SEM の倍率を 500 倍として撮影する。変数としては固溶処理を行った温度、時効温度、時効時間、SEM の倍率の 4 つであり、これらは画像ファイル名と対応しているものとする。本研究ではこれら 4 つの変数の様々な組み合わせでの SEM 画像を用いた。物体検出アルゴリズムを用いて機械学習を行うためには、画像と各粒子の位置が対応したアノテーションデータの作成が必要である。そこで本研究では、アノテーション作業を行うために Web 上で使用可能な Roboflow を利用した。Roboflow により 78 枚の画像中の粒子 8017 個についてアノテーションを行った。以降の機械学習では 78 枚の内 67 枚を訓練データ、11 枚をテストデータとして用いる。インスタンスセグメンテーションを行える有名な物体検出アルゴリズムとして、Yolov8 と Detectron2 があげられる。本研究ではこれら 2 つの比較も行った。これらの二つのライブラリにおいてハイパーパラメータをできるだけ最適化したのち学習と推論と評価を行った。得られた結果として AP50 と AP90 のテストデータ 11 枚における平均値は Yolov8 においては、それぞれ 0.900, 0.225 となり、Detectron2 においてはそれぞれ 0.858, 0.514 となった。この結果から二つの事が読み取れる。一つ目は Yolov8 においては Detectron2 よりも小さなオブジェクトの見逃しが少ない傾向がある点である。二つ目は Detectron2 では Yolov8 よりも正確なセグメントを作成しやすい傾向がある点である。二つの物体検出アルゴリズムの比較を行った後、Yolov8 と Detectron2 のそれぞれにおいて得られた推論結果と shapely というライブラリを用いることで、検出された粒子に対する面積、図心の座標、幅、高さ、アスペクト比、周囲長、図心からの最大距離の特徴量を csv ファイルに出力するプログラムを作成した。特徴量を抽出した後に、抽出した特徴量をできるだけ真の値に近づけるといふ試みを推論領域の特徴量と正解領域の特徴量の比(Ratio)を計算した上で以下の二つの方法により行った。一つ目の方法は、Ratio の度数分布の最頻値を用いる方法である。二つ目の方法としては目的変数を Ratio、説明変数を推論領域の面積、対象のオブジェクトが属する SEM 画像の倍率として機械学習を行う方法である。二つの方法において評価指標 MSE を計算することにより、機械学習を用いる方法の方が特徴量の補正においてより有用であることが分かった。

目次

1 緒論	5
1.1 研究背景	5
1.2 研究目的と研究内容	5
2 原理	6
2.1 Detectron2(Mask R-CNN)の原理.....	6
2.2 Yolo の原理.....	7
3 データセット作成.....	9
3.1 元画像データ	9
3.2 アノテーション.....	10
3.3 訓練データとテストデータに分割	11
4 ハイパーパラメータ調整	11
4.1 ハイパーパラメータ調整の必要性.....	11
4.2 ハイパーパラメータ調整の大まかな手順.....	11
4.3 AP の計算方法.....	12
4.3.1 IoU(Intersection over Union).....	12
4.3.2 信頼度	13
4.3.3 Precision-Recall 曲線の描き方と AP の計算方法	14
4.4 Yolov8 でのハイパーパラメータ調整	16
4.4.1 事前学習済みモデルの選択.....	16
4.4.2 optimizer の選択	17
4.4.3 lr,weight decay, batch, momentum の調整	18
4.5 Detectron2 でのハイパーパラメータ調整.....	19
4.5.1 事前学習済みモデルの選択.....	19
4.5.2 optimizer の選択	21
4.5.3 lr,weight decay, batch, momentum の調整.....	21
5 学習と推論	23
5.1 学習	23
5.2 推論.....	24
6. 評価	27
6.1 AP の結果	27
6.2 推論領域の可視化手順	29

6.3 推論領域の可視化結果	30
6.4 IoU, Precision, Recall の結果	33
6.5 Yolov8 と Detectron2 の特徴の違い	35
7 特徴量を抽出するプログラムを作成	35
8.特徴量の補正	36
8.1 Ratio の最頻値による面積の補正	36
8.2 機械学習(LightGBM)による面積の補正	37
8.3 Ratio の最頻値を用いる方法と機械学習を用いる方法との MSE の比較	37
9. 結論	40
10. 参考文献	40
11. 謝辞	41

1 緒論

1.1 研究背景

近年、材料開発の領域において、マテリアルズ・インフォマティクス (MI) と呼ばれる分野が注目を集めている。マテリアルズ・インフォマティクス(MI)とは、機械学習などの情報処理技術を積極的に活用し、材料開発を進める手法を指す¹⁾。この手法において、材料開発を行うためには、膨大な数量のデータを収集する必要がある、その際には数量だけでなく、質も重要視される。そのためには短時間で精度の良いデータを収集するための手法を確立する必要がある。

本研究ではディープラーニングを用いたインスタンスセグメンテーションと呼ばれる手法を用いて Ti-6Al-4V の走査型電子顕微鏡(SEM)画像から構成相のデータを自動で抽出することを目指す。インスタンスセグメンテーションを用いて金属材料組織の画像処理を行う取り組みは、"Automated segmentation of martensite-austenite islands in bainitic steel"というタイトルの論文などで発表されている²⁾。この論文ではインスタンスセグメンテーションを用いて熱処理条件の異なるベイナイト鋼の SEM 画像から、特定の領域を自動で特定することに成功している。しかしながらこの論文では、特定の領域をどれくらい精度よく特定できるかということに焦点が当てられており、特定された組織の情報を抽出してデータ化するという点に関しては触れられていない。インスタンスセグメンテーションを用いて金属材料組織の画像処理を行ったという論文はこの論文以外でもいくらか発表されているが、Ti-6Al-4V の SEM 画像に対して行ったものは見つけることはできなかった。そのため、本研究で行うインスタンスセグメンテーションを用いて Ti-6Al-4V の組織を自動で定量化する手法を提案することは、金属材料組織の画像分析という分野において重要であると考えられる。

1.2 研究目的と研究内容

本研究の目的は、1.1 研究背景でも触れたが機械学習を用いて、Ti-6Al-4V の様々な熱処理条件下³⁾での走査型電子顕微鏡 (SEM) 画像から構成相の特徴を自動的に定量化するプログラムを開発することにある。本研究の中心的な取り組みは以下の三つに要約される。第一に、特定の特徴量を抽出するにあたり、粒子の形状を正確に特定する必要がある。これを行うために、本研究では機械学習を利用したインスタンスセグメンテーションと呼ばれる手法を用いることにする。インスタンスセグメンテーションを行える物体検出ライブラリは複数存在するが本研究では Yolov8 と Detectron2 と呼ばれる 2 つを選択した。さらに 2 つのライブラリの内、どちらの方が本研究の目的に適しているのかについても議論する。第二に、インスタンスセグメンテーションにより粒子の形状の情報を得たうえで粒子の数、面積、

アスペクト比などの特徴量を抽出し、表形式データにまとめるためのプログラムを作成した。第三に抽出した特徴量に対して二種類の方法で補正を行い、特徴量をできるだけ真の値に近づけることを試みた。本研究を通じて、機械学習による金属材料組織の画像分析という分野において、より適した手法を提案し、その有効性に関して議論する。

2 原理

2.1 Detectron2(Mask R-CNN)の原理

Detectron2 では Mask-R-CNN と呼ばれる手法を提供している。Mask R-CNN は ICCV 2017 Best Paper に選出された手法で、物体検出やインスタンスセグメンテーションを実現できるセグメンテーションアルゴリズムである。図 2.1 に Mask R-CNN のアーキテクチャを示し、これを参照しながら説明する。なお、以下の説明は Mask R-CNN の技術論文に基づいている⁴⁾⁵⁾。図 2.1 に示す通り、Mask R-CNN のアーキテクチャは大きく Backbone, RPN(Region Proposal Network), HEAD の 3 つに分けられる。第一に Backbone では入力画像中の物体の形状の特徴を抽出する役割がある。具体的には Backbone の浅い層では物体の単純な形状の特徴に反応し、深い層では複雑な物体のパーツに反応するようになっている。Backbone には Mask R-CNN 独自のネットワークを採用しているのではなく、VGG や ResNet と呼ばれる既存のネットワークを採用している。第二に RPN では物体が存在していそうな領域（候補領域）を選定する役割がある。具体的には事前に特徴マップ上に Anchor と呼ばれる点を配置しておき、各 Anchor において様々な大きさとアスペクト比の Anchor Box（領域）を定義しておく。そして RPN では Anchor Box の位置、サイズなどの機械学習を行い、調整することで候補領域とすることが出来る。第三に HEAD ネットワークではそれぞれの候補領域に対して分類、バウンディングボックス作成、セグメンテーション作成を行う。具体的にはまず RPN により選定されたそれぞれの候補領域に対して、RoiAlign を適用し各候補領域に対して特徴量マップを正規化する。次に得られた正規化された特徴量マップを用いて、3 つの並列に配置されたレイヤによりそれぞれの候補領域に対する、分類、バウンディングボックス作成、セグメンテーション作成の出力を得る。以上を要約するとまず Backbone で画像中の物体の特徴を抽出し、次に RPN で候補領域の選定を行い、最後に HEAD で候補領域に対して分類、バウンディングボックス作成、セグメンテーション作成を行う。

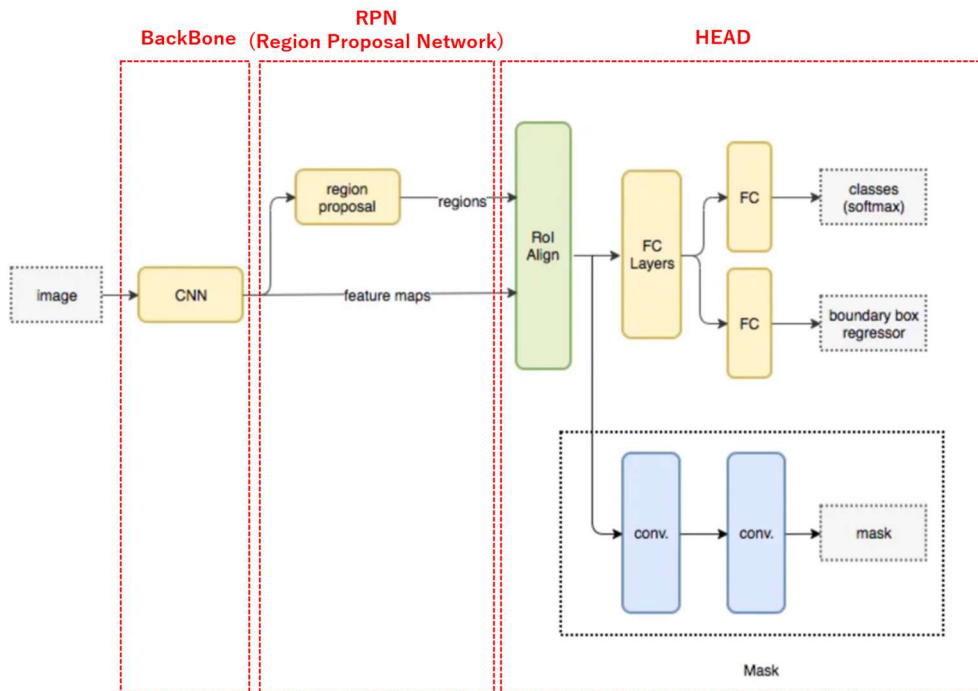


図 2.1 Mask R-CNN のアーキテクチャ

2.2 Yolo の原理

本研究には物体検出シリーズ Yolo の 2023 年に発表された Yolo シリーズの最新バージョン YOLOv8 を用いる。しかし YOLOv8 の技術論文は未だ発表されておらず、技術的な詳細については不明である。そのためここでは代わりに 2016 年に発表された Yolo シリーズの最初のバージョンである Yolo の技術論文から内容を紹介する⁶⁾。なお YOLOv8 ではバウンディングボックス作成とインスタンスセグメンテーションの両方を行えるのに対し、YOLO ではバウンディングボックス作成のみしか行えない点に注意したい。図 2.2 に Yolo のアーキテクチャを示す。YOLO のアーキテクチャは、24 層の CNN と 4 層の Pooling 層、2 層の全結合層から構成されている。まず 24 層の CNN と 4 層の Pooling 層により画像から物体の形状の特徴量を抽出する。その後二つの全結合層により物体のバウンディングボックス作成とクラス分類を行い出力とする。Mask R-CNN では候補領域の選定というタスクを行った後に、分類やバウンディングボックス作成というタスクを行っていた。つまり、Mask R-CNN では画像を二度見ていたのである。しかし YOLO では図からわかるように候補領域の選定を行わず、画像を一度見るだけで物体検出できる。この点により YOLO では Mask R-CNN と比べて推論を高速で行うことが可能であり、リアルタイム検出を可能にしている。ちなみに YOLO (You only look once) という名前は一度画像を見るだけで物体検出が行える点に由来している。ではどのようにして候補領域の選定を行わずして物体の検出を判断しているのかについて以下で述べる。図 2.3 に YOLO のモデル構造を示す。図 2.3 に示すように、まず YOLO では入力画像を $S \times S$ のグリッドセルに分割する。次にバウンディングボックス予測と各グ

グリッドのクラス確率の予測を別々に行う。最後にバウンディングボックス予測と各グリッドのクラス確率の予測結果を組み合わせることで推論結果とする。まずバウンディングボックスの予測について説明する。各グリッドセルは、 B 個のバウンディングボックスの予測を担当する。バウンディングボックスの予測には具体的には以下の 5 つのパラメータを予測する。具体的にはバウンディングボックスの中心座標(x, y)、バウンディングボックスの幅 w 、高さ h 、信頼スコアの 5 つである。結果としてバウンディングボックスの予測では $S \times S \times B$ 個のパラメータの予測を行う。なおオブジェクトが複数のグリッドセルにまたがっている場合は、オブジェクトの重心が存在するグリッドセルが予測を担当することにする。

次にグリッドのクラスの確率の予測について説明する。各グリッドセルにおいてオブジェクトが存在しているとしたとする条件において、それがあある特定のクラスであるという条件付き確率を計算する。これを各グリッドセル事に計算する。さらにこれを C 個のクラスについて全て計算する。結果としてグリッドのクラス確率の予測では $S \times S \times C$ 個のパラメータを予測する。以上をまとめると、バウンディングボックス予測とグリッドのクラスの確率の予測により合計 $S \times S \times (B \times 5 + C)$ 個のパラメータを予測することになる。論文中では $S=7, B=2, C=20$ としているため、図 2.2 の最終出力層は $7 \times 7 \times 30$ テンソルとなっている。

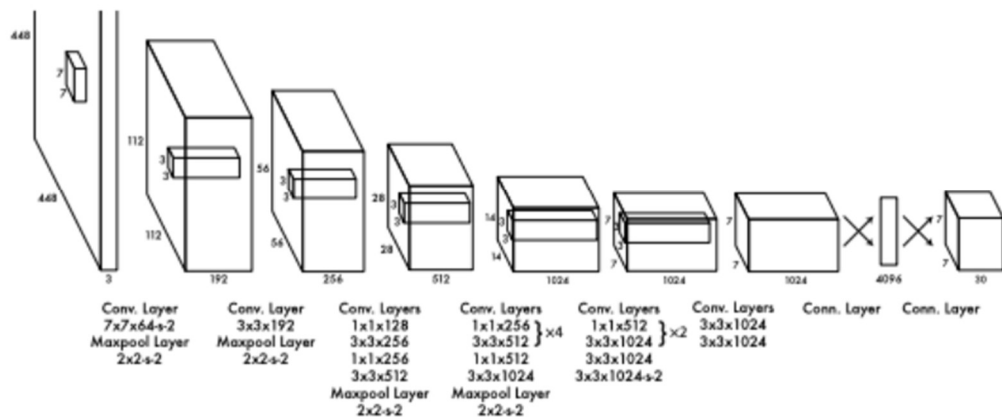


図 2.2 Yolo のアーキテクチャ

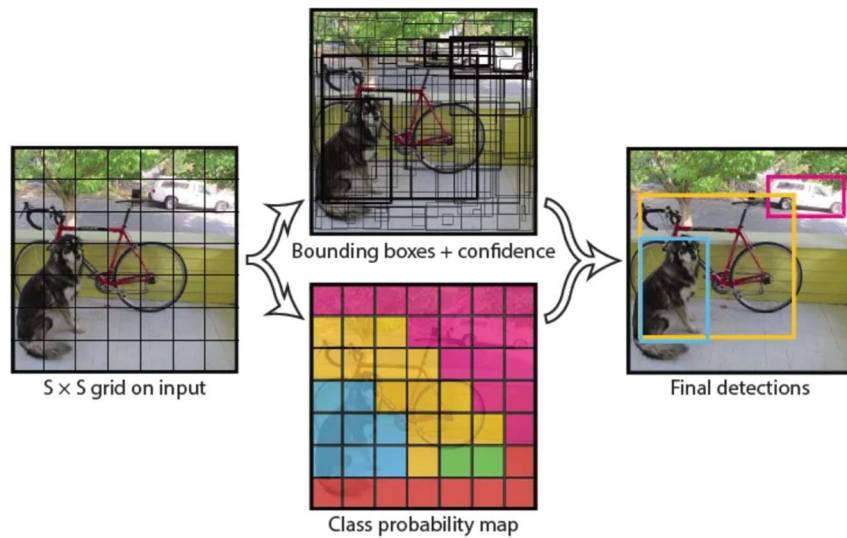


図 2.3 Yolo のモデル構造

3 データセット作成

3.1 元画像データ

図 3.1～図 3.4 に本研究で用いる画像の一部を示す．図 3.1～図 3.4 は全て熱処理後の Ti-6V-4Al 合金の SEM 画像である．熱処理条件と SEM の倍率については画像のファイル名に記されている．例えば図 3.1 の 20200518 Ti-64 950STQ 800 30min 500x.jpg は以下の手順により得られたものである³⁾．まず，等軸 ($\alpha+\beta$) 微細組織をもつ Ti-6Al-4V 合金に対して熱間圧延を行い，950K で固溶処理を行った後，氷水中で急冷する．次に 800K で時効処理を 30 分間行う．最後に SEM の倍率 500 倍として撮影する．変数としては固溶処理を行った温度，時効温度，時効時間，SEM の倍率の 4 つである．これら 4 つの変数の範囲としては固溶処理を行った温度は 920~970K，時効温度については 500~800K，時効時間については 30 分~10 時間，SEM の倍率に関しては 500~10000 倍となっている．本研究では，これら 4 つの変数の様々な組み合わせにより得られた 78 枚の画像を用いる．

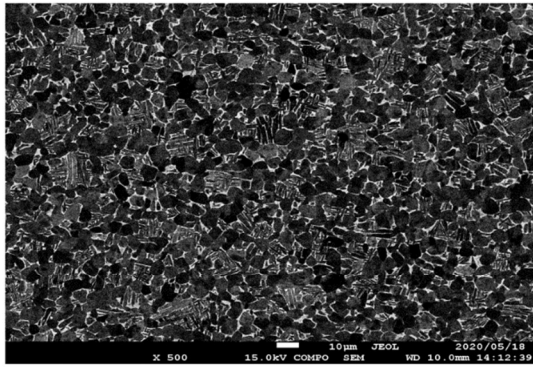


図3.1 20200518 Ti-64 950STQ 800 30min 500x.jpg

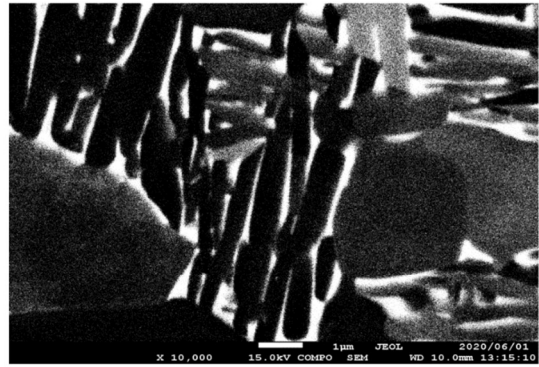


図3.2 20200601 Ti-64 970STQ-700 1hr x10000.jpg

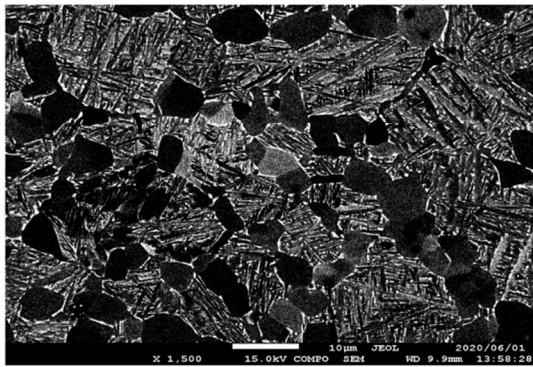


図3.3 20200601 Ti-64 970STQ-700 30min x1500 no1.jpg

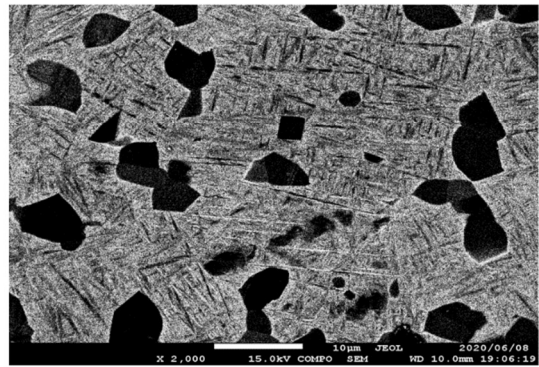


図3.4 20200608 Ti-64 960STQ 2000x.jpg

3.2 アノテーション

Detectron2 や Yolov8 の学習に際しては、画像中の各オブジェクトの位置を示すアノテーションデータの作成が必要である。アノテーションデータは画像内のピクセル座標で表され、Detectron2 では COCO segmentation 形式、Yolov8 では Yolov8 txt 形式のようなあらかじめ決められた形式に保存されなければならない。そこで本研究では、アノテーション作業を行うために Roboflow を利用した⁷⁾。Roboflow は、Web 上で画像データをアップロードして様々な形式でアノテーションデータを出力できるプラットフォームである。Roboflow のアノテーション機能の中で、Polygon tool と Smart Polygon を活用した。図 3.5 に Polygon tool と Smart Polygon を使用している様子について示す。図 3.5 に示すように Polygon tool では、画像上で粒子の輪郭をマウスで左クリックし、対象領域を囲むことで粒子のアノテーションを行うことができる。一方で、Smart Polygon ではオブジェクトに対してカーソルを合わせると領域が赤く表示され、マウスを左クリックすることで自動的に対象物を囲むことができる。Roboflow のこれらの機能の利用により、アノテーション作業を効率的に進めることができた。本研究では、総計 78 枚の画像に対してアノテーションを行い、アノテーションされたオブジェクトの総数は 8017 個であった。

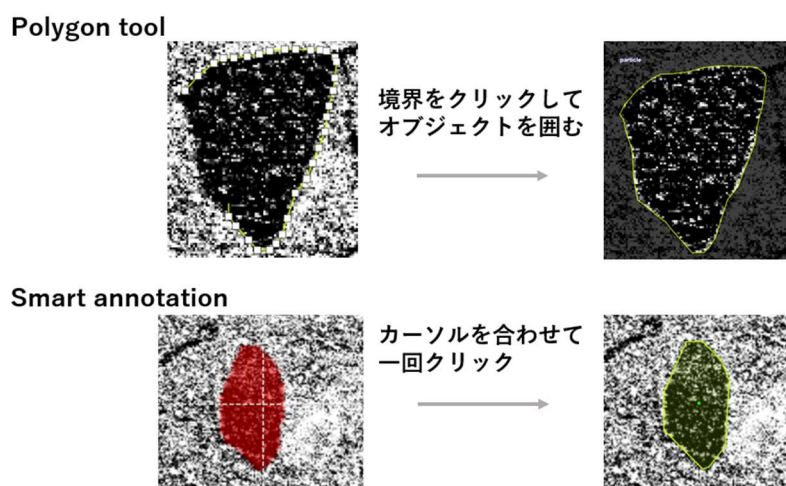


図 3.5 Roboflow の 2 つの機能

3.3 訓練データとテストデータに分割

78 枚の画像の内 67 枚を訓練データ, 11 枚をテストデータに分割しデータセットとした. 訓練データのオブジェクト個数は 6904 個, テストデータ 11 枚に対するオブジェクト個数は 1113 個である.

4 ハイパーパラメータ調整

4.1 ハイパーパラメータ調整の必要性

画像認識系の機械学習では, ハイパーパラメータの値が適切でないと学習が全く進まないということがある⁸⁾. そのためもし, ハイパーパラメータを適切に設定した場合と設定しなかった場合を比較すると, 判断を誤る可能性がある. この問題をできるだけ回避するために, 最初に Detectron2 と YOLOv8 のそれぞれにおいてハイパーパラメータを最適化する必要がある.

4.2 ハイパーパラメータ調整の大まかな手順

YOLOv8 と Detectron2 のハイパーパラメータ調整の手順について大まかに説明する. ハイパーパラメータ最適化の手順として大きく三つ行う. 第一に, 事前学習済みモデルの選択, 第二にオプティマイザの選択, 第三に learning rate, momentum, weight decay, batch の最適化を行う. 本研究では検証データとして, テストデータ 11 枚を用いることにする. またこれらの調整における評価指標としては基本的に AP50 を用いることとする. AP50 の算出関数は各ライブラリに組み込まれており, ハイパーパラメータ設定においてこれを用い

ることとする。評価指標 AP がどのような評価指標であるのかについての説明は 4.3 で行う。

4.3 AP の計算方法

ハイパーパラメータ調整においてはそれぞれのライブラリに組み込まれている関数により計算できる AP を用いると説明した。以下から評価指標 AP がどのような評価指標なのかを説明する。AP を計算するには IoU, 信頼度, Precision-Recall 曲線について理解する必要があるがこれらについても順に説明する⁹⁾。

4.3.1 IoU(Intersection over Union)

IoU (Intersection over Union) の説明を行う。図 4.1 に IoU を説明するための模式図を示す。図 4.1 において、アノテーションにより作成された正解領域は赤い円で示し、推論領域は青い領域で示してある。理想的に推論が行えているとすると、正解領域と推論領域が完全に一致しているべきであるが、実際にはずれが生じる。図 4.1 中の(1)の図では 3 つの中で推論があまり成功していない状態を示しており、反対に(3)の図では 3 つの中で正解領域と推論領域が最も一致していることが見て取れる。IoU は、この正解領域と推論領域がどれくらい一致しているかを定量的に評価するための指標として用いられる。IoU は次の式のように表される。

$$\text{IoU} = \frac{\text{交差部分}}{\text{正解領域} \cup \text{推論領域}} \quad (1)$$

分子には正解領域と推論領域が重なる領域 (交差部分) のピクセル数があり、分母は正解領域または推論領域のピクセル数である。IoU は 0 から 1 の範囲の値を取り、値が大きいほど正解領域と推論領域がよく一致していることを示す。

AP の計算において、IoU はあるオブジェクトにおいて検出が成功したのか、あるいは失敗したのかの判定に利用される。具体的には IoU の閾値を最初に決めておき、閾値を超える場合は検出成功とし、超えない場合は検出失敗とする。例として IoU の閾値を 0.50 に設定した場合には図 4.1 において(1)では検出失敗(False)、(2)では検出失敗(False)、(3)では検出成功(True)とする。

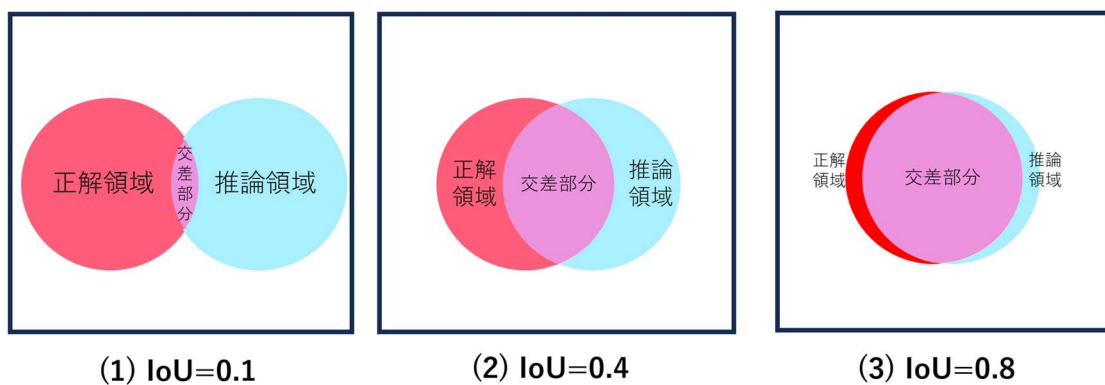


図 4.1 IoU の説明図

4.3.2 信頼度

信頼度の説明を行う。まず図 4.2 に YOLOv8 と Detectron2 における推論画像の一部を拡大して示している。信頼度は推論されたオブジェクトに対して、オブジェクト名と共にバウンディングボックスの上に示されている。この信頼度とは、モデルが出力した推論に対して、その結果に対する自信の程度を示すものである AP の計算には IoU に加えてこの信頼度も考慮する。

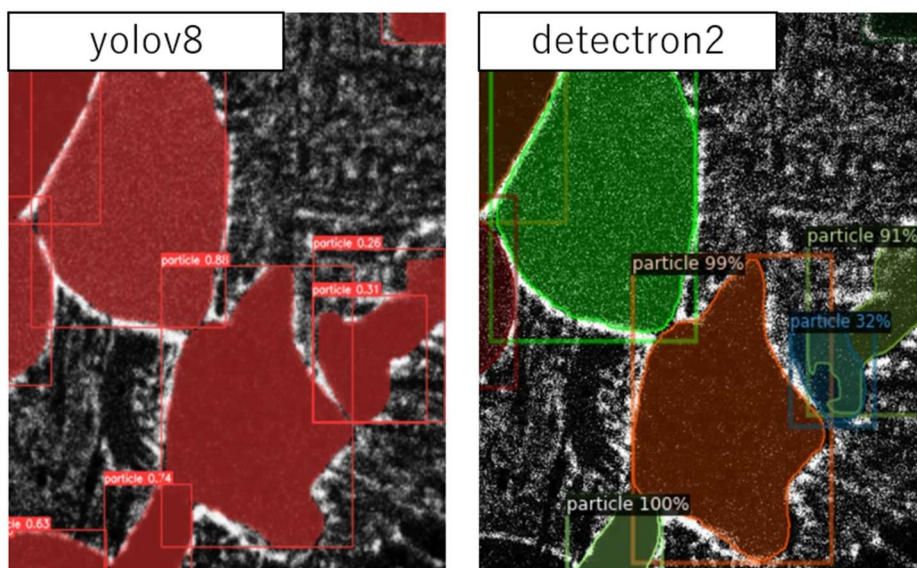


図 4.2 推論画像の例（信頼度の説明）

4.3.3 Precision-Recall 曲線の描き方と AP の計算方法

Precision-Recall 曲線の描き方と AP の計算方法について述べる。図 4.3 に説明のための模式図を示す。これから図 4.3 に示すような推論結果が得られたとして、実際に Precision-Recall 曲線を描き AP 求めることにより説明する。図 4.3 においては赤色の領域が正解領域であるとする。正解領域は 4 つ存在している。また青色の領域は推論領域であるとする。推論領域は 4 つ存在し A,B,C,D と名前を付ける。さらに推論領域 A,B,C,D に対する信頼度と IoU についても図 4.3 中に示してある。また今回は IoU の閾値を 0.50 と決めておくことにする。さらに、Precision, Recall などの計算結果は表 4.1 にまとめることにする。

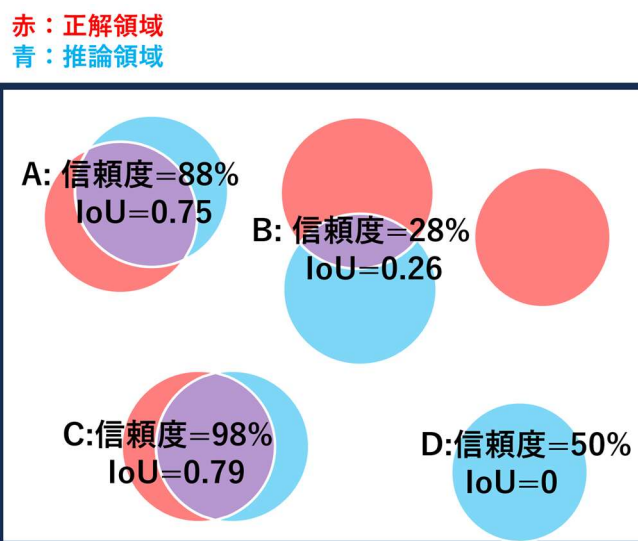


図 4.3 AP の説明図

表 4.1 図 4.3 に対する Precision と Recall の計算結果

index	信頼度	IoU	TorF	Precision	Recall
C	98%	0.79	T	1/1=1	1/4=0.25
A	88%	0.74	T	2/2=1	2/4=0.5
D	50%	0	F	2/3=0.67	2/4=0.5
B	28%	0.26	F	2/4=0.5	2/4=0.5

まず、図 4.3 に示してある推論領域 A,B,C,D の信頼度と IoU を表 4.1 に記入する。また、信頼度を考慮するため、表 4.1 のインデックスの順番については信頼度の高い順に並び変える。次に、TorF の列に値を記入する。今回は IoU の閾値が 0.50 としてあるので、表 4.1 において IoU が 0.50 以上の C,A については TorF の列に検出成功を示す T を記入し、一方 D,B については IoU が 0.50 未満なので検出失敗を示す F を記入する。その後 Precision と Recall の列を計算する。i 行目の Precision と Recall は以下の式に従い計算する。

$$\text{Precision} = \frac{\text{1 行目から } i \text{ 行目までの範囲に存在する T の個数}}{i} \quad (2)$$

$$\text{Recall} = \frac{\text{1 行目から } i \text{ 行目までの範囲に存在する T の個数}}{\text{正解領域の個数}} \quad (3)$$

例として表 4.1 の D に対応する行の Precision と Recall を計算してみる。まず正解領域の個数は図より 4 である。次に D は 3 行目なので $i = 3$ である。また 1 行目から 3 行目までの範囲に存在する T の個数について考える。1 行目は T, 2 行目は T, 3 行目は F であるので 1 行目から 3 行目までの範囲に存在する T の個数は 2 である。これらを上式に代入して計算すると表の D に対応する行に記入された結果になる。ほかの行も同様にして計算すると表 4.1 の結果となる。表 4.1 を得た後、各インデックスに対する、Precision と Recall の組を図 4.4 に示すようにプロットする。さらに $(\text{Precision}, \text{Recall}) = (1, 0)$ の点についてもプロットする。プロットした点を折れ線グラフとしたものを Precision-Recall 曲線と呼ぶ。

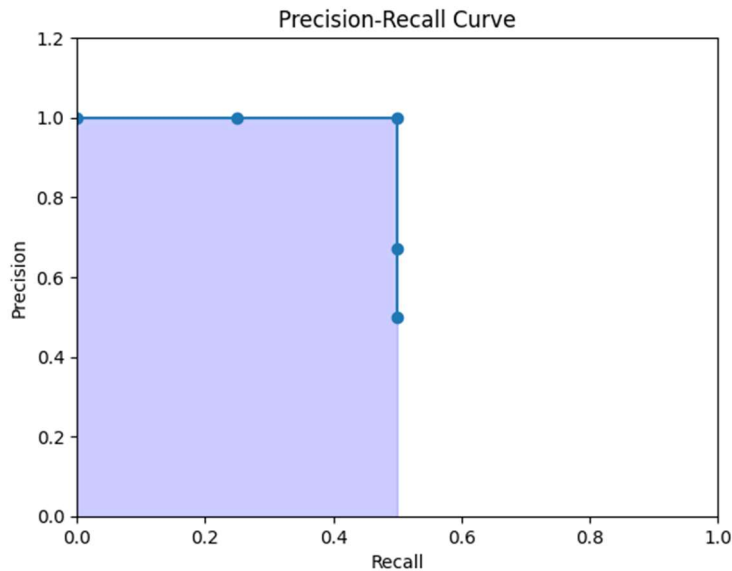


図 4.4 図 4.3 に対する Precision-Recall 曲線

Precision-Recall 曲線を描いた後に、曲線と x 軸との間の面積を計算する。この面積を AP50 とする。また AP50 の 50 は、IoU の閾値を 0.50 としたことに由来する。もし IoU の閾値を 0.25 として同様に計算した場合は AP25、IoU の閾値を 0.75 とした場合は AP75 として表す。

4.4 Yolov8 でのハイパーパラメータ調整

4.4.1 事前学習済みモデルの選択

YOLOv8 では 5 種類の事前学習済みモデルが提供されている。5 種類の事前学習モデルは全て COCO データセットにより学習されている。事前学習済みモデルは全て <https://docs.ultralytics.com/tasks/segment/> によりダウンロードすることができる¹⁰⁾。さらに各学習済みモデルの性能についても同じ URL において以下の図 4.5 のように示されている。

▼ Segmentation (COCO)
See [Segmentation Docs](#) for usage examples with these models trained on [COCO-Seg](#), which include 80 pre-trained classes.

Model	size (pixels)	mAP ^{box} 50-95	mAP ^{mask} 50-95	Speed		params (M)	FLOPs (B)
				CPU GNNX (ms)	A100 TensorRT (ms)		
YOLOv8n-seg	640	36.7	30.5	96.1	1.21	3.4	12.6
YOLOv8s-seg	640	44.6	36.8	155.7	1.47	11.8	42.6
YOLOv8m-seg	640	49.9	40.8	317.0	2.18	27.3	110.2
YOLOv8l-seg	640	52.3	42.6	572.4	2.79	46.0	220.5
YOLOv8x-seg	640	53.4	43.4	712.1	4.02	71.8	344.1

- mAP^{val} values are for single-model single-scale on [COCO val2017](#) dataset. Reproduce by `yolo val segment data=coco-seg.yaml device=0`
- Speed averaged over COCO val images using an [Amazon EC2 P4d](#) instance. Reproduce by `yolo val segment data=coco-seg.yaml batch=1 device=0|cpu`

図 4.5 Yolov8 の学習済みモデルのダウンロード画面

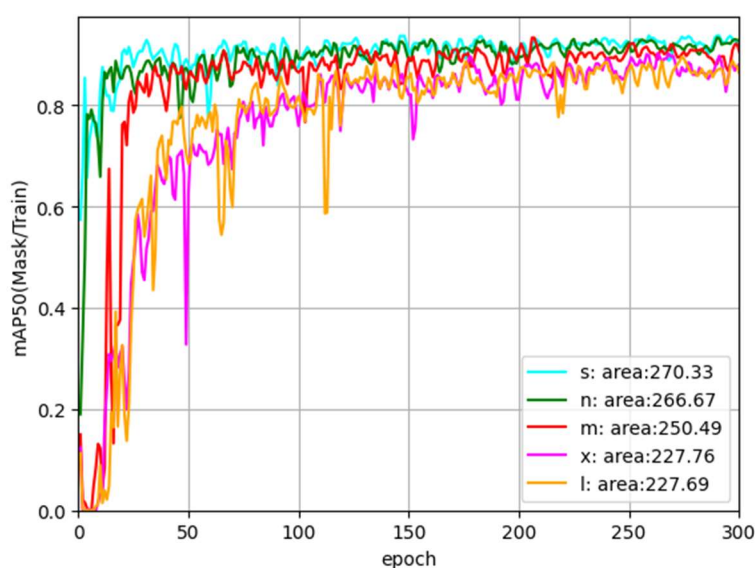


図 4.6 各学習済みモデルにおける学習曲線と面積

本研究では、これらの事前学習済みモデルに対して、自ら作成したデータセットを入力して学習を行う。最初に、5種類の事前学習モデルの内、どのモデルにおいて学習が進みやすいのかを判断するため、すべてのモデルにおいて学習を行い、その学習の履歴の曲線から判断することにする。図4.6に、各学習済みモデルにおいて、自ら作成したデータセットを入力し、学習を行った際の学習曲線を示してある。これらの学習曲線から各学習済みモデルにおいて学習の進み具合をどのように評価するのかについては、各学習曲線とx軸がなす面積について計算を行い、この面積の値を判断基準とすることにした。この面積についても図4.6の凡例中に示している。図4.6に示した通り、面積はsにおいて最大となっているので、YOLOv8s-segにおいて学習が最も進みやすいと判断することにした。そのため以下ではYOLOv8s-segを用いることにする。

4.4.2 optimizer の選択

Yolov8では6種類のoptimizerが用意されており、これをハイパーパラメータとして指定できる。具体的にはSGD, Adam, Adamax, AdamW, NAdam, RAdam, RMSPropの6種類である。図4.7に6種類のoptimizerに対する学習の履歴の曲線を示している。また図4.7には、学習曲線とx軸で囲まれる面積と初期学習率(lr0)についても凡例の中に示してある。なお初期学習率に関して、RMSProp以外はデフォルトの値に設定してある。学習の進み具合の判断については4.4.1で行ったのと同様に面積から判断する。図4.7よりoptimizerに関してはSGDを用いることにする。

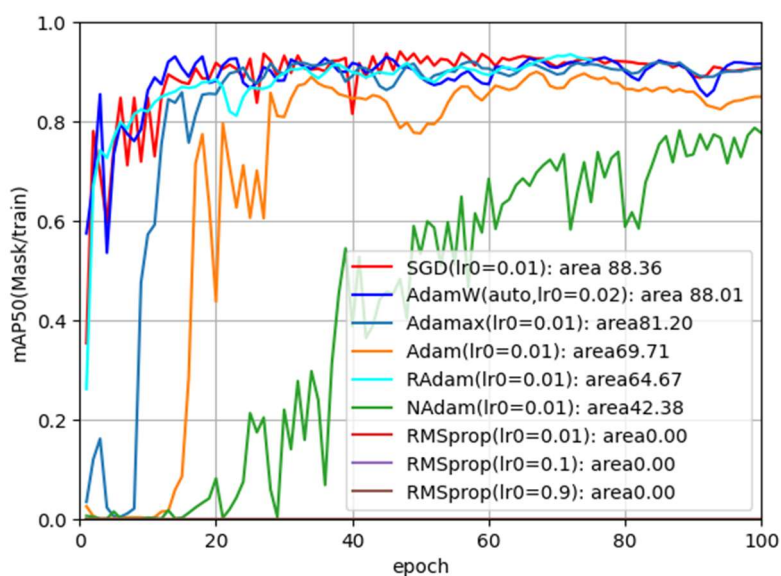


図 4.7 各 optimizer における学習曲線と面積

4.4.3 lr0, weight decay, batch, momentum の調整

事前学習済みモデルと optimizer の選択を行ったあと、次にその他のハイパーパラメータについて調整を行う。ハイパーパラメータの種類に関しては、少なくとも 30 種類以上存在する¹¹⁾。本来であればすべてのハイパーパラメータにおいて調整をするべきである。しかし今回はハイパーパラメータ調整にかかる時間を減らすため、多数存在するハイパーパラメータの内、初期学習率、weight decay, batch, momentum と呼ばれるハイパーパラメータのみを調節することとする。これらのハイパーパラメータを調節するためにハイパーパラメータ自動最適化ライブラリである Optuna を用いる¹²⁾。Optuna はベイズ最適化という手法を用いて、できるだけ少ない試行回数で目的の評価指標を最大化するというものである。これによりグリッドサーチやランダムサーチなどと呼ばれる他のハイパーパラメータ最適化手法と比較して少ない試行回数で最適化ができるとされている。最適化するための評価指標に関しては AP50 を用いる。試行回数は 25 回、Epoch は 40 とする。

探索範囲と調整後のハイパーパラメータの値について図 4.8 に示した。さらに Optuna で行った 25 回の試行におけるそれぞれの AP50 の結果の履歴の曲線を図 4.9 に示す。さらにハイパーパラメータの重要性を示したグラフを図 4.10 に示す。Yolov8 では図 4.8 の探索結果の表に示すハイパーパラメータを用いて学習を行うことにする。

探索範囲			探索結果	
ハイパーパラメータ	探索区間		ハイパーパラメータ	調整後
batch	1~8	Optunaで最適化 →	batch	5
lr0 (初期学習率)	1e-3~0.5		lr0 (初期学習率)	0.05225
momentum	0.50~0.99		momentum	0.55190
Weight decay	1e-5~0.1		Weight decay	1.95e-05

図 4.8 ハイパーパラメータの探索範囲と探索結果

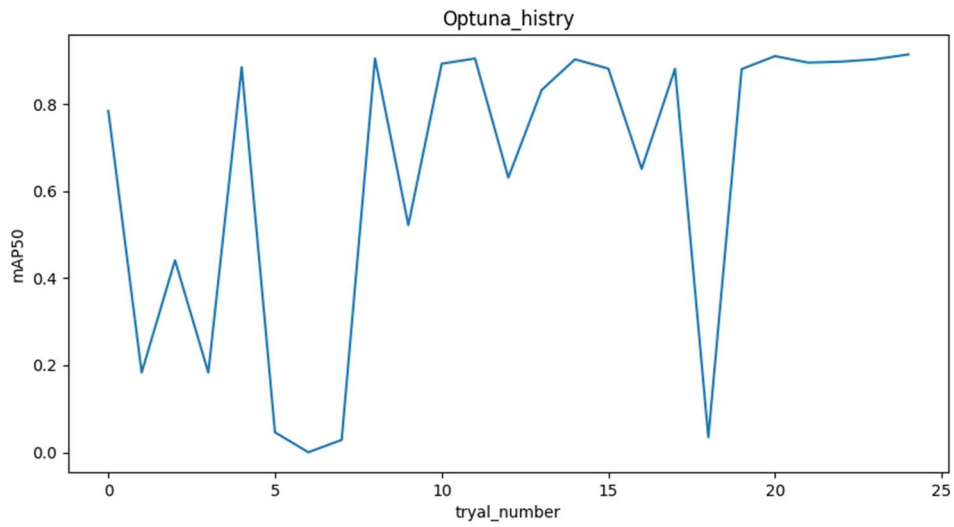


図 4.9 Optuna の履歴曲線

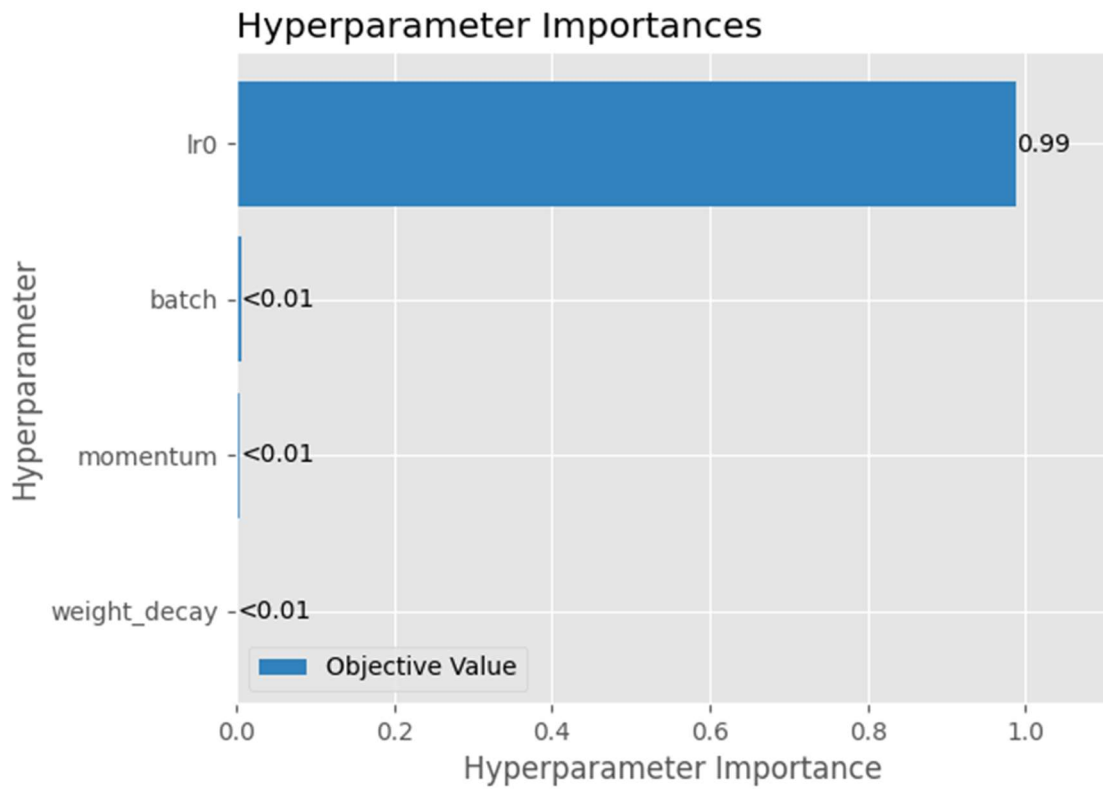


図 4.10 ハイパーパラメータ重要度のグラフ

4.5 Detectron2 でのハイパーパラメータ調整

4.5.1 事前学習済みモデルの選択

Detectron2 では 10 種類の学習済みモデルが提供されている。事前学習済みモデルは

https://github.com/facebookresearch/detectron2/blob/main/MODEL_ZOO.md によりダウンロードすることができる¹³⁾。さらに各学習済みモデルの性能についても同じ URL において以下の図 4.11 のように示されている。Yolov8 の時と同様に Detectron2 においてもこれらの事前学習済みモデルに対してどのモデルが最も学習が進みやすいのかを判断するために 10 種類の事前学習済みモデルの学習の履歴の曲線から判断する。図 4.12 に学習の履歴の曲線の結果を示す。また、学習曲線と x 軸がなす面積についても計算を行い、これをどの学習が進んだかの判断基準とする。図 4.9 より今回は R_101_FPN_3x を用いることとする。

COCO Instance Segmentation Baselines with Mask R-CNN

Name	lr sched	train time (s/iter)	inference time (s/im)	train mem (GB)	box AP	mask AP	model id	download
R50-C4	1x	0.584	0.110	5.2	36.8	32.2	137259246	model metrics
R50-DC5	1x	0.471	0.076	6.5	38.3	34.2	137260150	model metrics
R50-FPN	1x	0.261	0.043	3.4	38.6	35.2	137260431	model metrics
R50-C4	3x	0.575	0.111	5.2	39.8	34.4	137849525	model metrics
R50-DC5	3x	0.470	0.076	6.5	40.0	35.9	137849551	model metrics
R50-FPN	3x	0.261	0.043	3.4	41.0	37.2	137849600	model metrics
R101-C4	3x	0.652	0.145	6.3	42.6	36.7	138363239	model metrics
R101-DC5	3x	0.545	0.092	7.6	41.9	37.3	138363294	model metrics
R101-FPN	3x	0.340	0.056	4.6	42.9	38.6	138205316	model metrics
X101-FPN	3x	0.690	0.103	7.2	44.3	39.5	139653917	model metrics

New baselines using Large-Scale Jitter and Longer Training Schedule

The following baselines of COCO Instance Segmentation with Mask R-CNN are generated using a longer training schedule and large-scale jitter as described in Google's [Simple Copy-Paste Data Augmentation](#) paper. These models are trained from scratch using random initialization. These baselines exceed the previous Mask R-CNN baselines.

図 4.11 Detectron2 の事前学習済みモデル

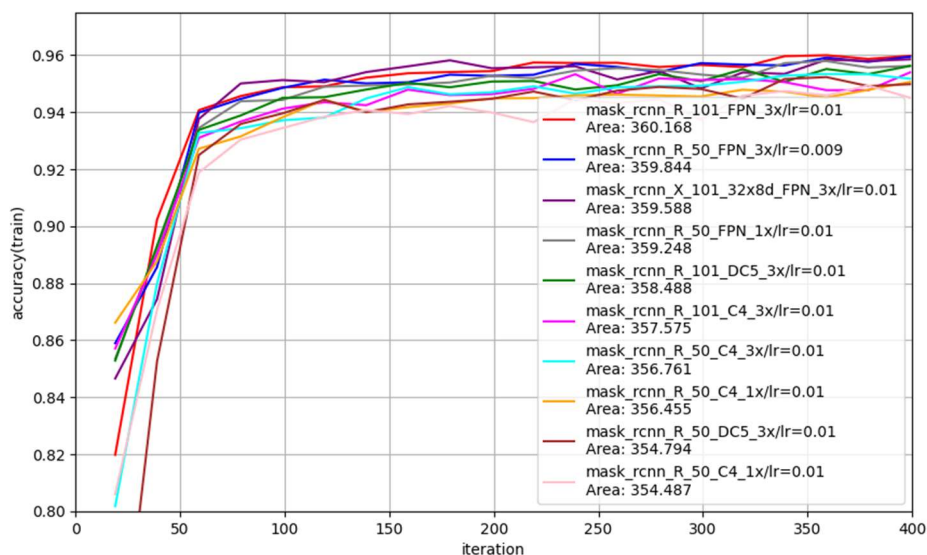


図 4.12 各学習済みモデルにおける学習曲線と面積

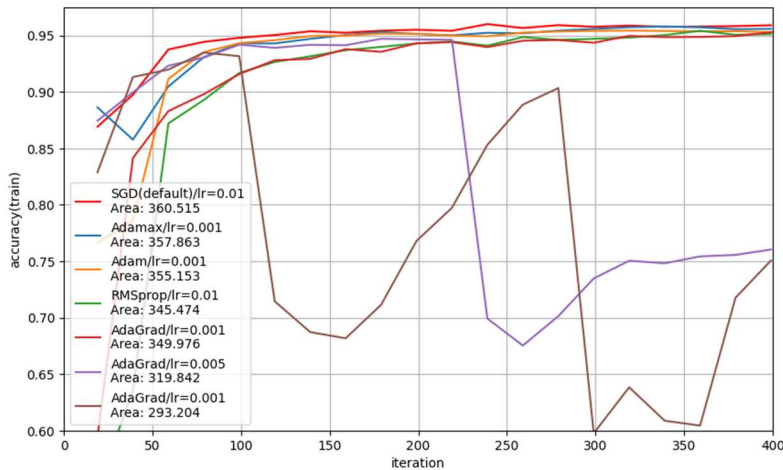


図 4.13 各 optimizer における学習曲線と面積

4.5.2 optimizer の選択

Detectron2 では Pytorch ライブラリを用いることにより、様々な optimizer を使用できる¹⁴⁾。今回は SGD, Adamax, Adam, RMSprop, AdaGrad の中から選択することとする。また同様に学習の履歴と曲線と x 軸で囲まれた面積から判断する。図 4.13 に学習の履歴の曲線の結果を示す。また、その時に設定した学習率および面積についても凡例の中に示してある。図 4.13 より SGD を用いることにする。

4.5.3 lr, weight decay, batch, momentum の調整

モデルの選択, optimizer の選択を行ったあと、その他のハイパーパラメータについても調整を行った¹⁵⁾。Yolov8 の時と同様に今回も Optuna を用いて多数存在するハイパーパラメータの内 lr(学習率), weight decay, batch, momentum のみを調節する。最適化する評価指標としては AP50 を用いた。試行回数は 25 回とした。また各試行における学習時間を短縮するために iteration=200 とした。

探索範囲と調整後のハイパーパラメータの値について図 4.14 に示した。さらに Optuna で行った 25 回の試行におけるそれぞれの AP50 の結果の履歴の曲線を図 4.15 に示す。さらにハイパーパラメータの重要性を示したグラフを図 4.16 に示す。Detectron2 では図 4.14 の探索結果に示すハイパーパラメータの値を用いて、学習と推論を行うことにする。

探索範囲		Optunaで最適化	探索結果	
ハイパーパラメータ	探索区間		ハイパーパラメータ	調整後
batch	1~8	→	batch	8
lr (学習率)	1e-3~0.5		lr (学習率)	0.01648
momentum	0.50~0.99		momentum	0.67563
Weight decay	1e-5~0.1		Weight decay	0.01172

図 4.14 ハイパーパラメータの探索範囲と探索結果

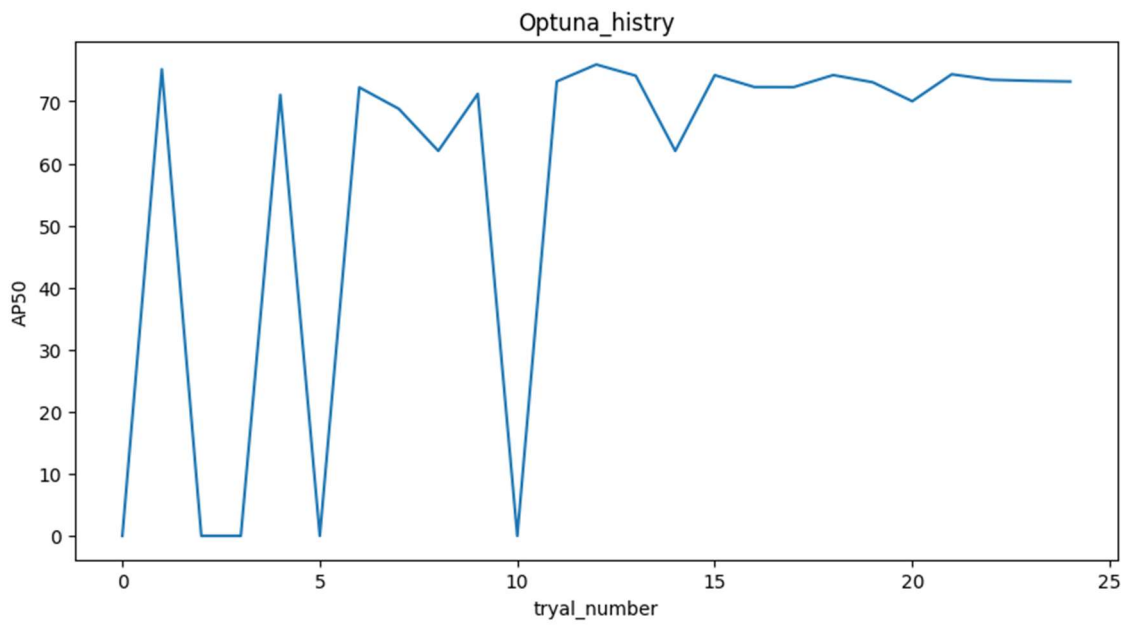


図 4.15 Optuna の履歴曲線

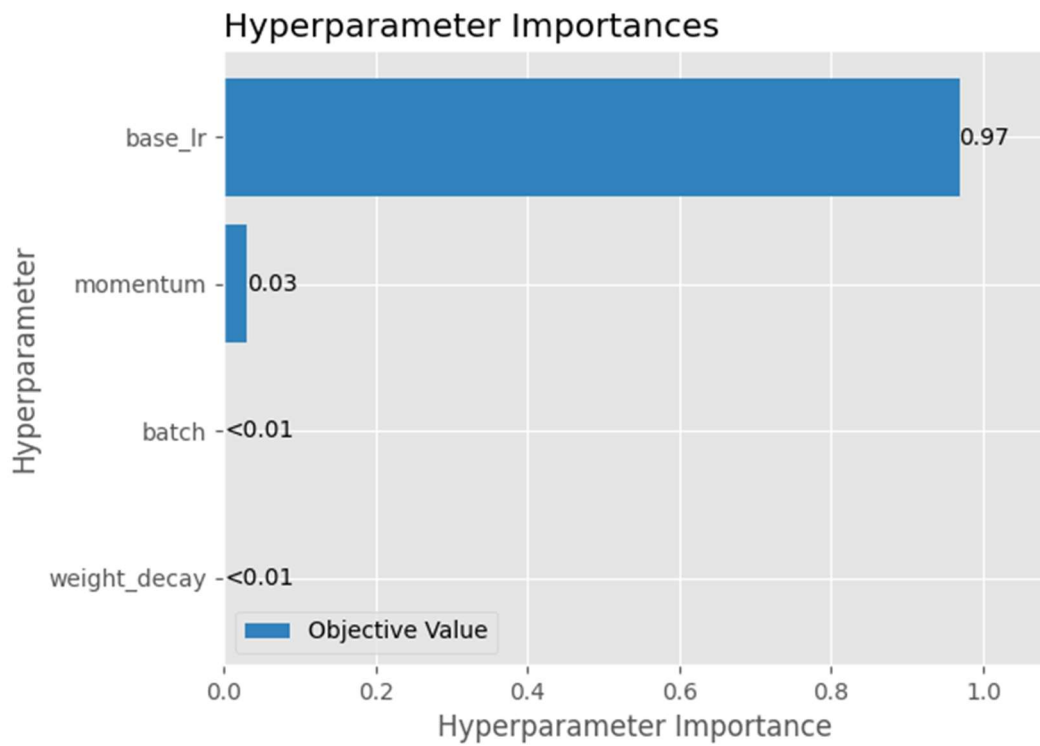


図 4.16 ハイパーパラメータ重要度のグラフ

5 学習と推論

調整後のハイパーパラメータで YOLOv8 と Detectron2 において学習と推論を行った。

5.1 学習

以上により決定した、学習済みモデル、オプティマイザ、その他のハイパーパラメータを用いて学習を行った。図 5.1 に YOLOv8 の学習の履歴の曲線、図 5.2 に Detectron2 の学習の履歴の曲線を示す。学習時間については YOLOv8 では 27 分 47 秒、Detectron2 では 25 分 52 秒であった。

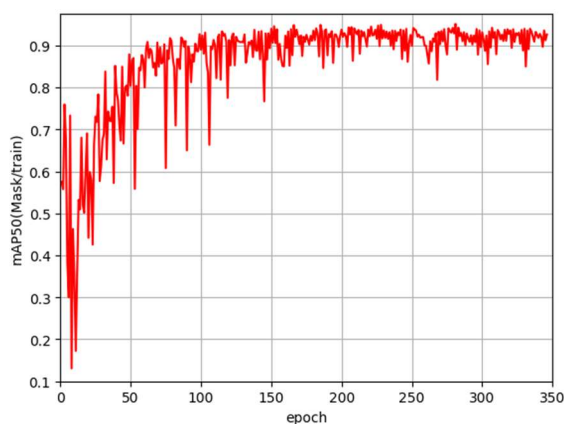


図 5.1 YOLOv8 の学習の履歴の曲線

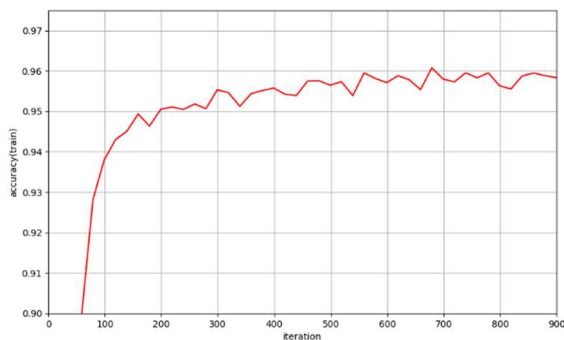


図 5.2 Detectron2 の学習の履歴の曲線

5.2 推論

学習後にテスト画像 11 枚に対して推論を行い、推論画像を出力する。図 5.3~図 5.13 に YOLOv8 での推論画像を示す。図 5.14~図 5.24 に Detectron2 での推論画像を示す。図 5.3~図 5.13 の YOLOv8 では赤色で塗られた領域が推論された領域である。図 5.14~図 5.24 の Detectron2 では様々な色で塗られた領域が推論された領域である。また両方のライブラリにおいて、推論画像と一緒に、図 5.25 に示すような形式で推論領域の座標をテキストファイルとして出力できる。このテキストファイルを後に YOLOv8 と Detectron2 の評価に用いる。

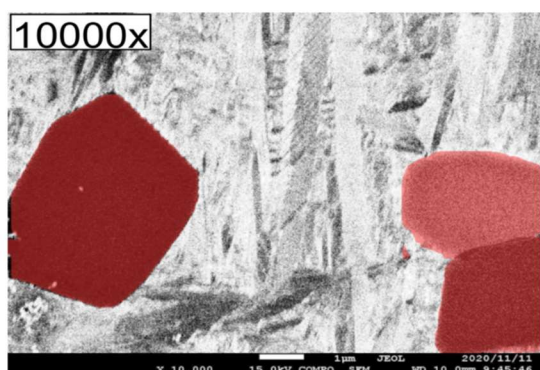


図5.3 YOLOv8による10000xの推論結果

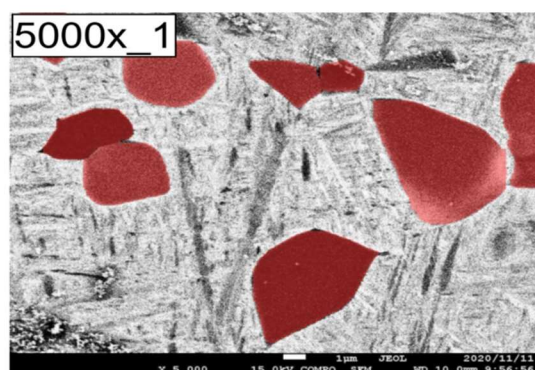


図5.4 YOLOv8による5000x_1の推論結果

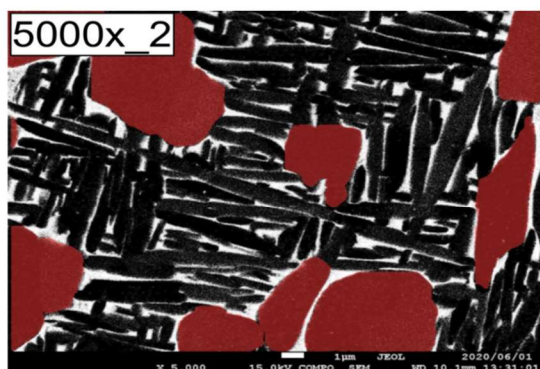


図5.5 YOLOv8による5000x_2の推論結果

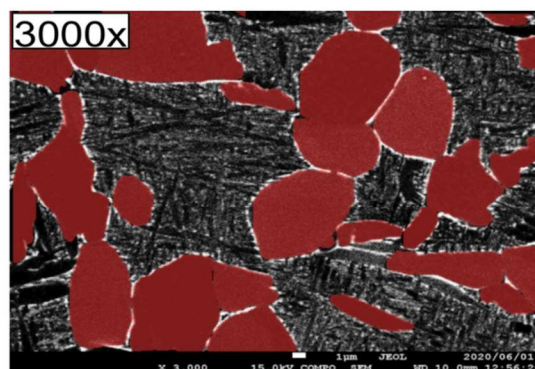


図5.6 YOLOv8による3000xの推論結果

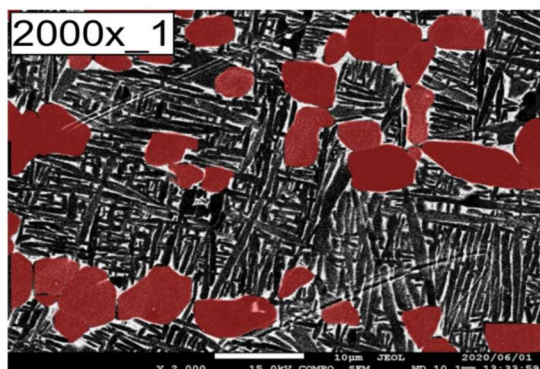


図5.7 YOLOv8による2000x_1の推論結果

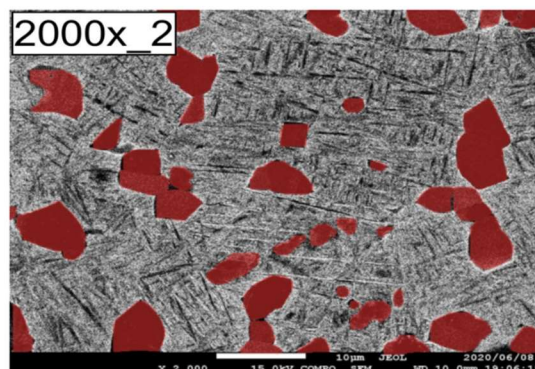


図5.8 YOLOv8による2000x_2の推論結果

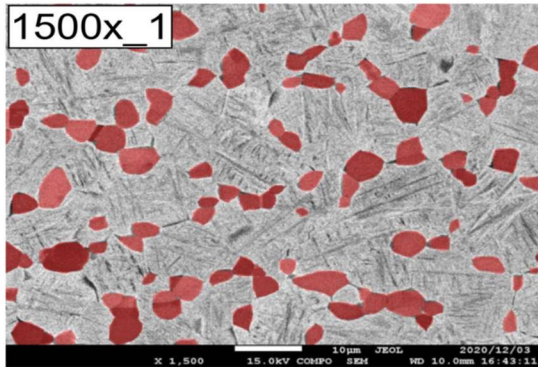


図5.9 YOLOv8による1500x_1の推論結果

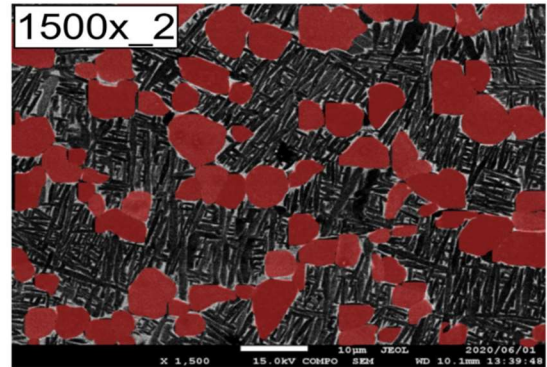


図5.10 YOLOv8による1500x_2の推論結果

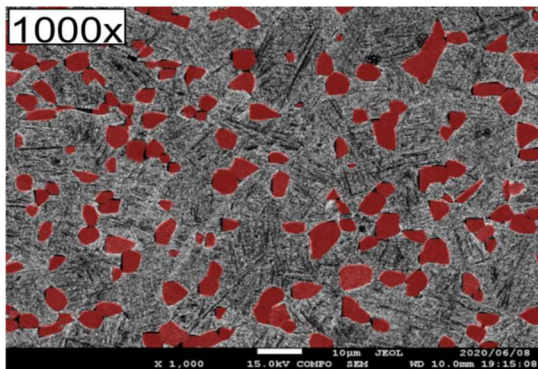


図5.11 YOLOv8による1000xの推論結果

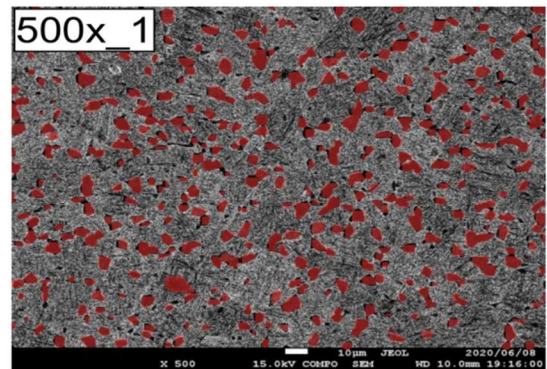


図5.12 YOLOv8による500x_1の推論結果

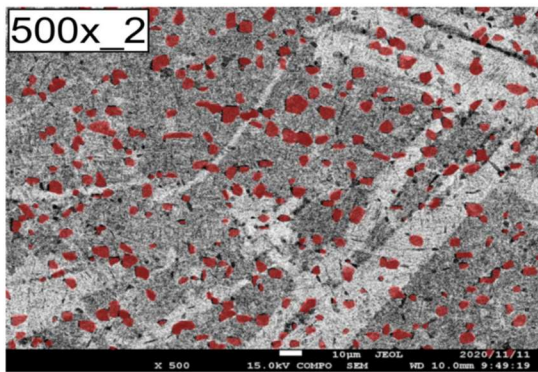


図5.13 YOLOv8による500x_2の推論結果

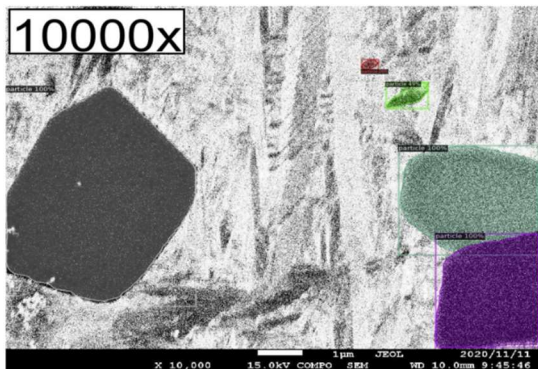


図5.14 Detectron2による10000xの推論結果

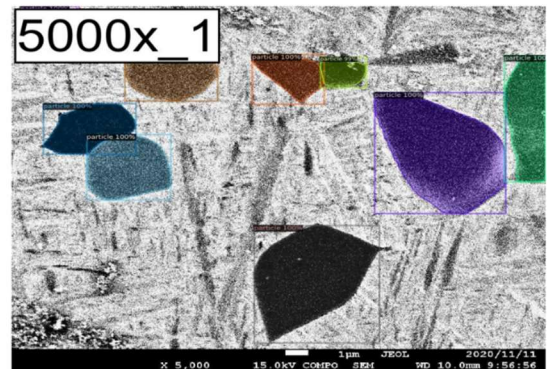


図5.15 Detectron2による5000x_1の推論結果

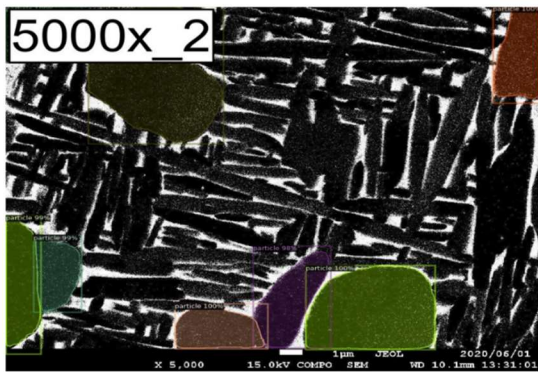


図5.16 Detectron2による5000x_2の推論結果

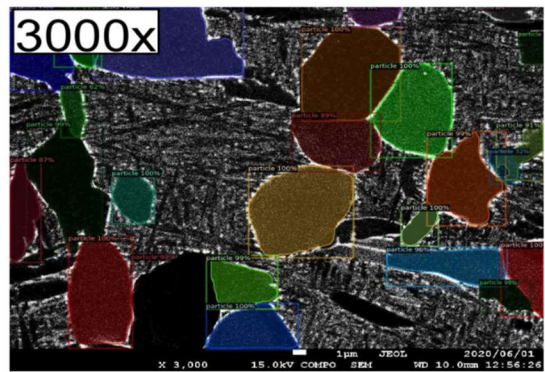


図5.17 Detectron2による3000xの推論結果

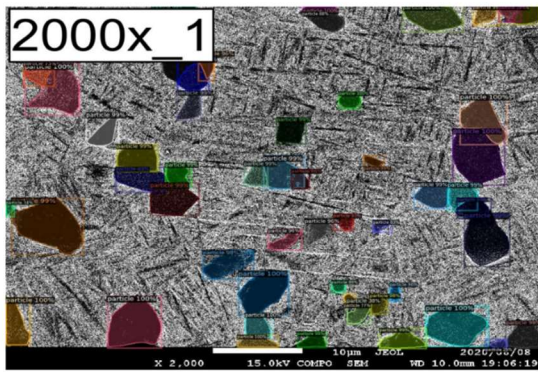


図5.18 Detectron2による2000x_1の推論結果

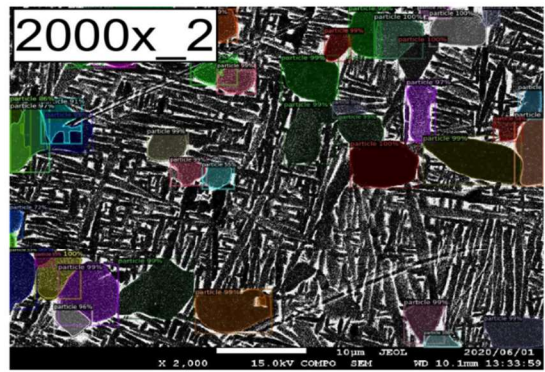


図5.19 Detectron2による2000x_2の推論結果

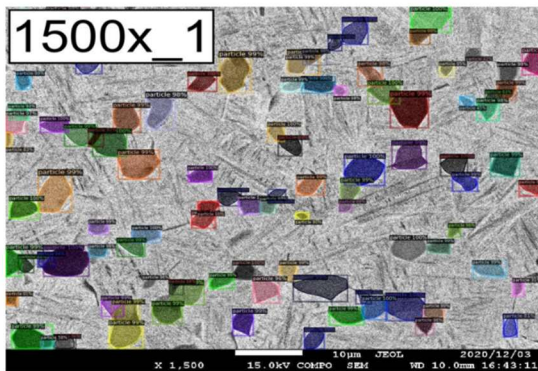


図5.20 Detectron2による1500x_1の推論結果

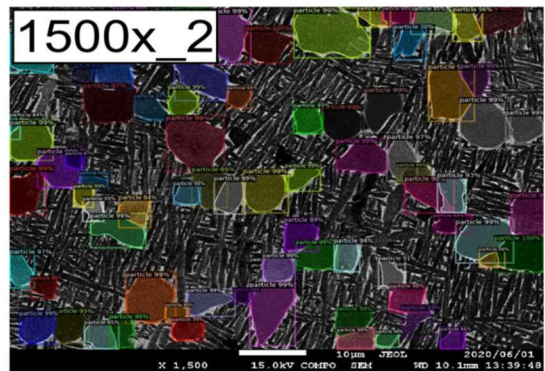


図5.21 Detectron2による1500x_2の推論結果

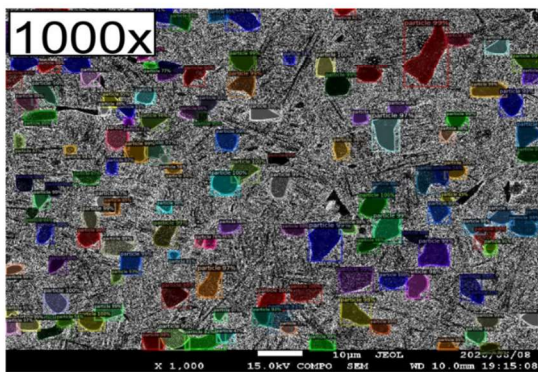


図5.22 Detectron2による1000xの推論結果

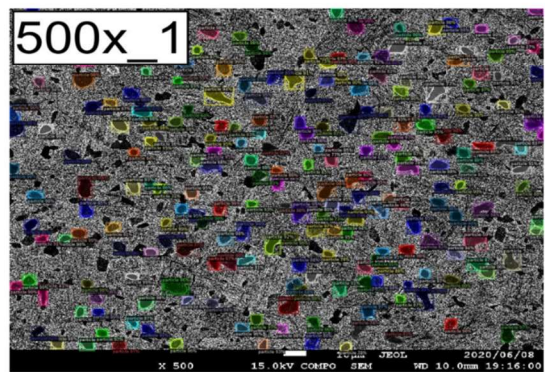


図5.23 Detectron2による500x_1の推論結果

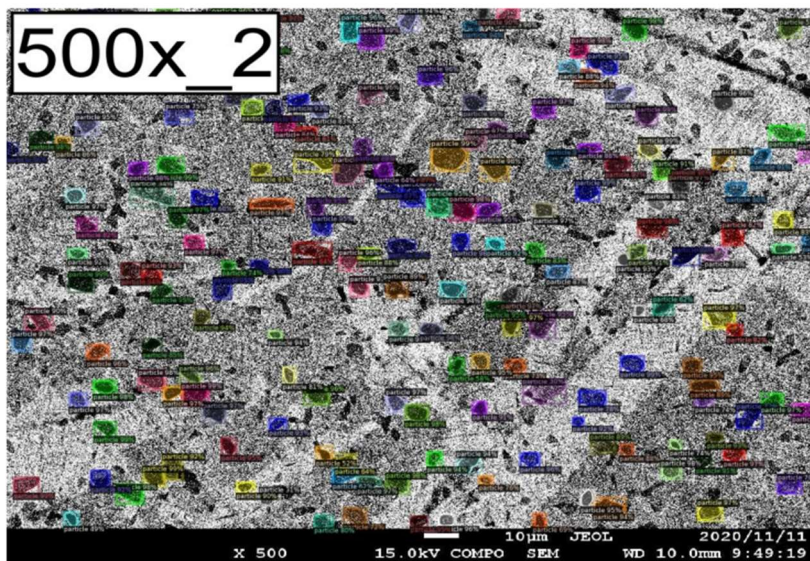


図5.24 Detectron2による500x_2の推論結果

セグメントの座標	X1	Y1	X2	Y2	X3	Y3	X4	Y4	X5	
オブジェクト1	0	0.926562	0.875	0.925	0.876953	0.925	0.910156	0.926562	0.912109	0.926562
オブジェクト2	0	0.701563	0.28125	0.7	0.283203	0.7	0.298828	0.703125	0.302734	0.709375
オブジェクト3	0	0.639063	0.03125	0.6375	0.0332031	0.6375	0.0585938	0.639063	0.0605469	0
オブジェクト4	0	0.139062	0.75	0.1375	0.751953	0.1375	0.761719	0.139062	0.763672	0.14218
オブジェクト5	0	0.523438	0.796875	0.521875	0.798828	0.520312	0.798828	0.51875	0.800781	

図 5.25 推論領域の座標が記述されたテキストファイル

6. 評価

推論時に得られたテキストファイルを用いて YOLOv8 と Detectron2 の評価を行った後、結果を比較する。今回は評価方法として大きく 2 つの方法を導入する。第一にインスタンスセグメンテーションで一般的に用いられる AP を計算し、これを用いる。第二に推論領域を分かりやすく可視化したうえで、セマンティックセグメンテーションなどで用いられる IoU, Recall, Precision を計算し、これを用いる。これらの二つの評価方法を用いることで YOLOv8 と Detectron2 の特徴の違いについて考察する。

6.1 AP の結果

ハイパーパラメータ調整時には YOLOv8, Detectron2 のそれぞれに評価指標 AP を算出するためのプログラムが用意されており、これを用いた。しかしながら両者の AP を算出するプログラムは厳密には異なる理屈でプログラムされている可能性があるため、統一的な評価ができない可能性があると考えられる。そこで今回は AP を計算するプログラムを 4.3 で説明した理屈において自分で作成し、これを用いる。今回は AP25, AP50, AP75, AP90 について計算を行う。YOLOv8 と Detectron2 に対する、AP の結果について図 6.1 に示す。

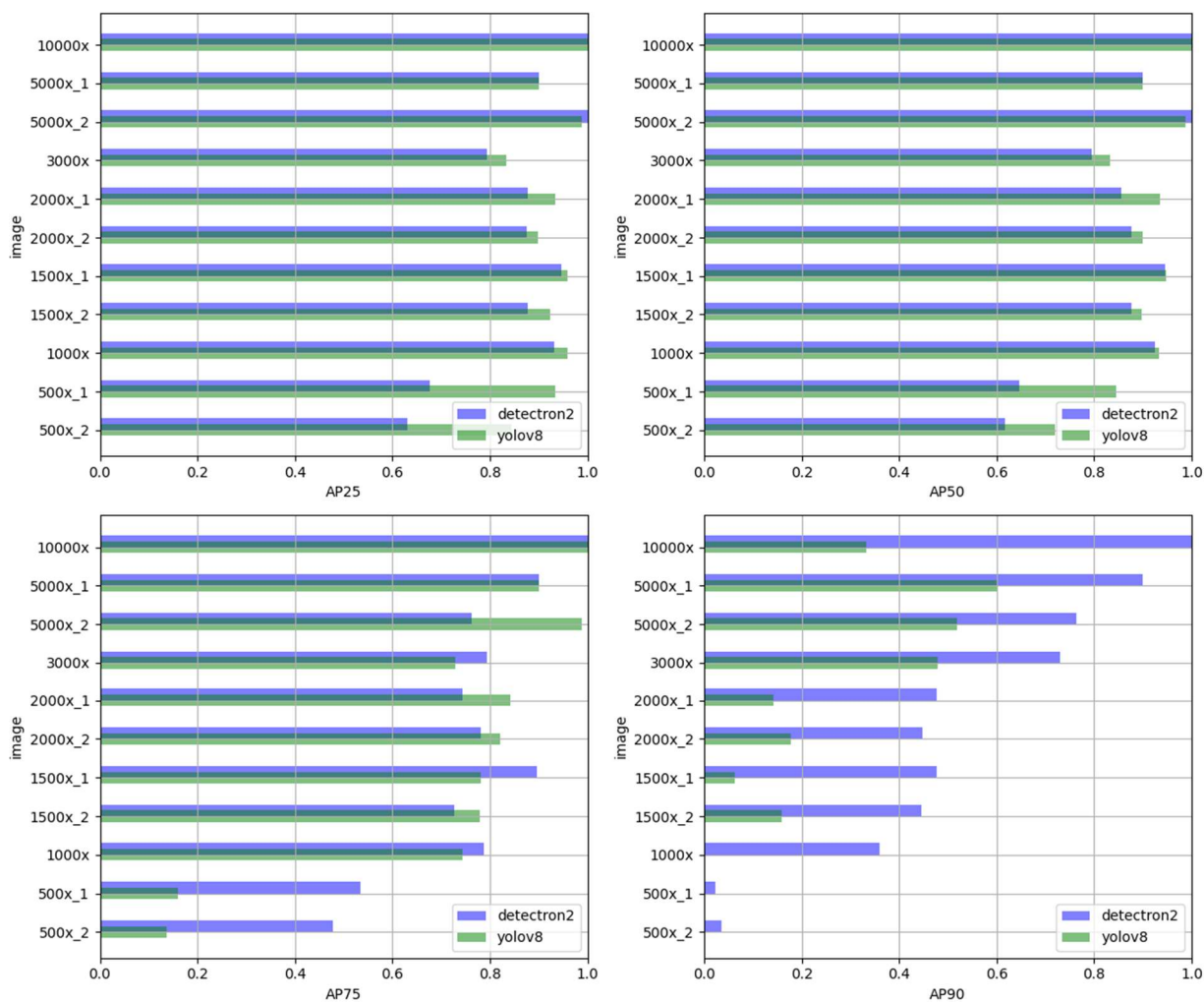


図 6.1 YOLOv8 と Detectron2 における AP25, AP50, AP75, AP90

図 6.1 から読み取れることについて述べる。第一に SEM の倍率が小さくなるにつれて AP20, AP50, AP75, AP90 において値が小さくなる傾向がある。つまり 10000x や 5000x_1 などの高倍率の SEM 画像に比較して 500x_1 や 500x_2 などの画像では各 AP の値が小さくなる傾向がある。このことは YOLOv8 と Detectron2 の両方のライブラリにおいて、低倍率の画像に多く見られるような小さなオブジェクトについてはセグメンテーションが困難であることを示している。第二に各 IoU の閾値における AP について、IoU の閾値が小さいほど YOLOv8 の値がおおきくなり、IoU の閾値が大きくなるほど Detectron2 の値が大きくなっている。例えば 500x_1 に注目してみると AP25 と AP50 においては YOLOv8 の方の値が大きくなっており、AP75 と AP90 においては Detectron2 の方の値が大きくなっている。この IoU の閾値の違いにより、YOLOv8 と Detectron2 の間の AP の大小関係が入れ替わることから両者においてどのような特徴の違いがあるのかについて詳しく読み取りたい。そのために AP とは別の評価方法を導入したうえで両者の特徴の違いについて議論することにする。

6.2 推論領域の可視化手順

別の評価指標を導入するため、まずは正解領域に対する推論領域の可視化を行う。その手順について説明する。可視化方法の手順を図 6.2 に示す。まずアノテーションデータから二値化画像を作成する。二値化画像においては、オブジェクトの領域を白い領域、それ以外の領域を黒い領域として表現する。次に推論時に出力された txt ファイルから推論データの二値化画像を作成する。同様オブジェクトの領域を白い領域、それ以外の領域を黒い領域として表現する。そしてこれら二つの二値化画像を重ね合わせる。重ね合わせた時に正解データの二値化画像の白い領域と推論データの二値化画像の白い領域が重なった部分を正解エリア(TP)として赤色で表現する。正解データの二値化画像では黒い領域であるのに、推論データの二値化画像では白い領域である部分を誤検出エリア(TN)として黄色で表現する。正解データの二値化画像では白い領域であるのに、推論データの二値化画像では黒い領域である部分を誤検出エリア(FP)として青色で表現する。それら以外の残った部分(FN)は黒色で表現する。このような手順で YOLOv8 と Detectron2 のそれぞれで推論されたテスト画像 11 枚について可視化を行う。

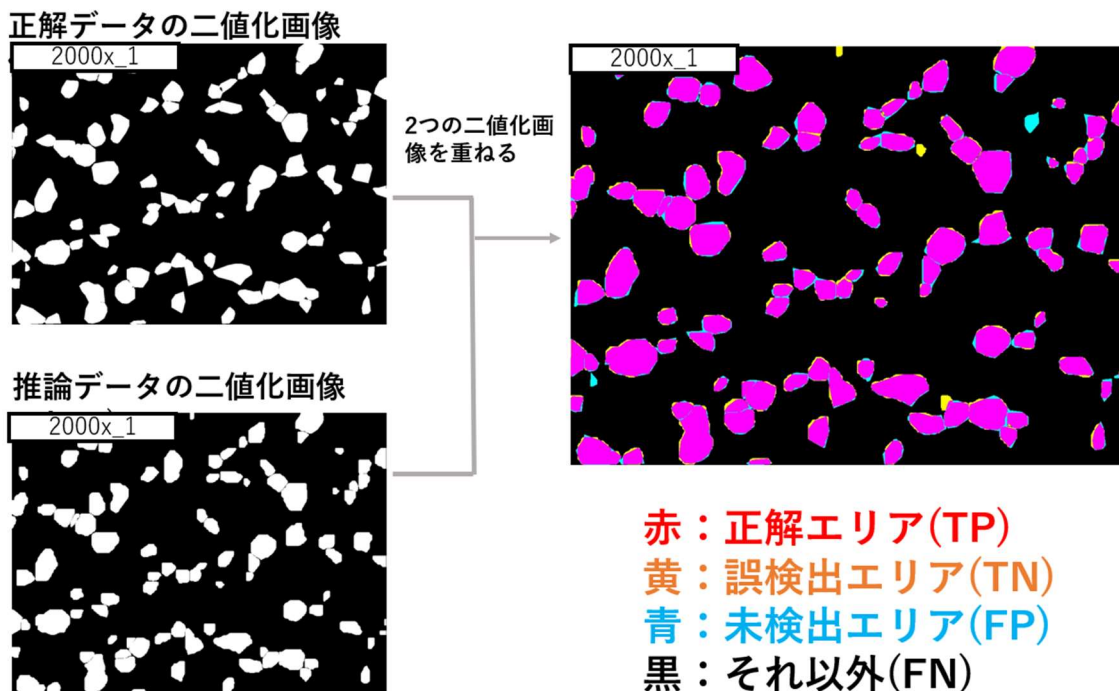


図 6.2 推論領域の可視化方法

6.3 推論領域の可視化結果

Yolov8 での可視化結果を図 6.3~6.13 に, Detectron2 での可視化結果を図 6.14~6.24 に示す. 図 6.3~6.13 と図 6.14~6.24 の比較において読み取れることについて述べる. 第一に Detectron2 では Yolov8 と比較して, 全体が青色で表示されている粒子が多数存在することが分かる(図 6.13 と図 6.24 の比較が分かりやすい). これが意味することとして Detectron2 では Yolov8 と比べて粒子自体の見逃しが多いと考えられる. 第二に, Yolov8 で検出された粒子では正解領域(赤色)の周りに誤検出領域(黄色)や未検出領域(青色)の領域が目立つ. 一方 Detectron2 で検出された粒子では正解領域(赤色)の周りに誤検出領域(黄色)や未検出領域(青色)の領域は Yolov8 より少ないように見える(同様に図 6.13 と図 6.24 の比較が分かりやすい). これが意味することとして Yolov8 よりも Detectron2 の方が検出できた粒子においては正確なセグメントを作成できるということが分かる.

以上をまとめると可視化画像から, 粒子の見逃しの少なさは Yolov8 の方が優れており, 正確なセグメントの作成は Detectron2 の方が優れているという傾向があると考えられる. この傾向は 6.1 で既に述べた IoU の閾値の変化により, Yolov8 と Detectron2 の間の AP の大小関係が入れ替わることに対応していると考ええる.

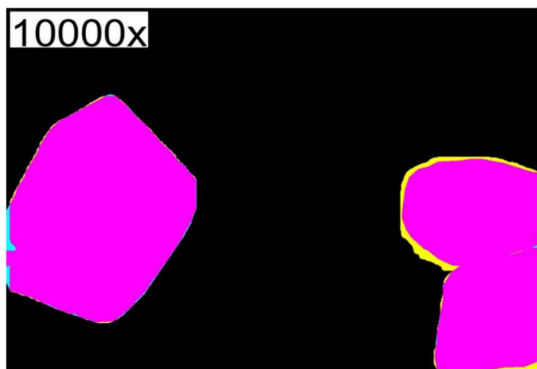


図6.3 Yolov8の推論画像における10000xの可視化結果

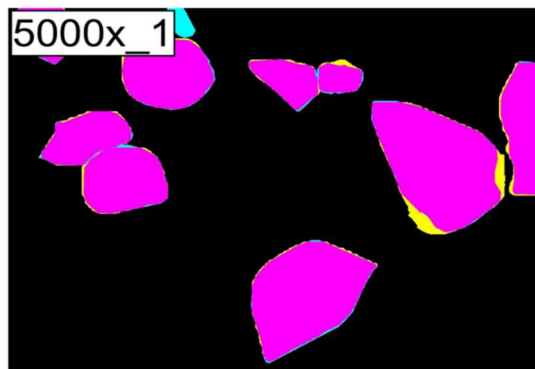


図6.4 Yolov8の推論画像における5000x_1の可視化結果



図6.5 Yolov8の推論画像における5000x_2の可視化結果

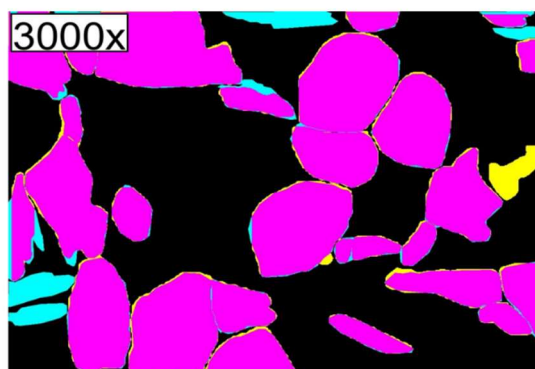


図6.6 Yolov8の推論画像における3000xの可視化結果

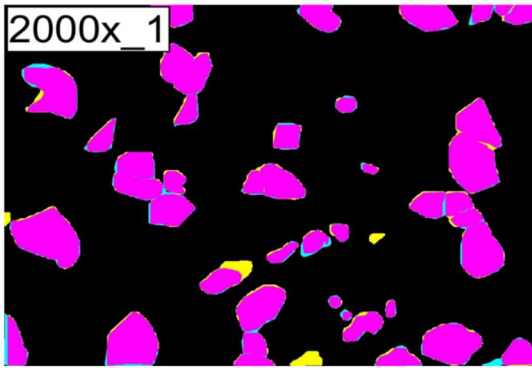


図6.7 Yolov8の推論画像における2000x_1の可視化結果

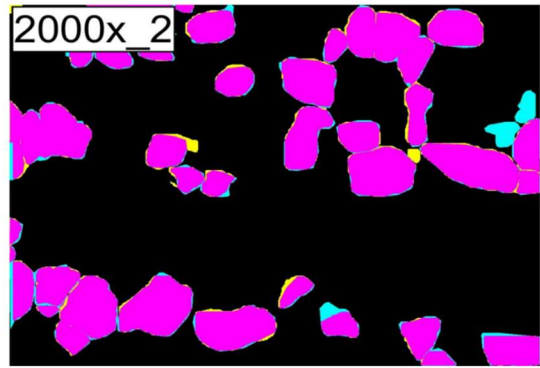


図6.8 Yolov8の推論画像における2000x_2の可視化結果

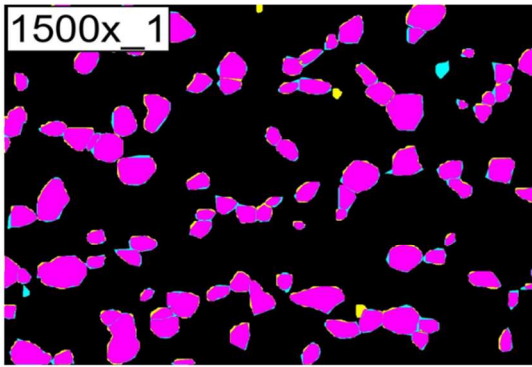


図6.9 Yolov8の推論画像における1500x_1の可視化結果

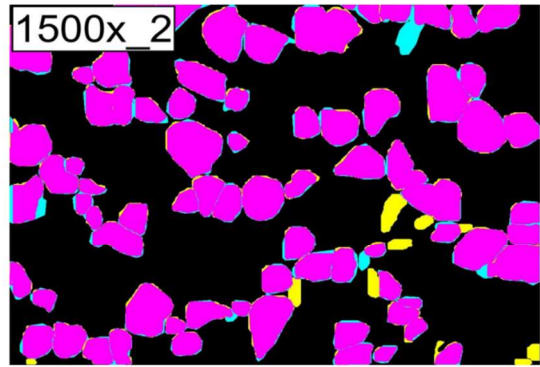


図6.10 Yolov8の推論画像における1500x_2の可視化結果

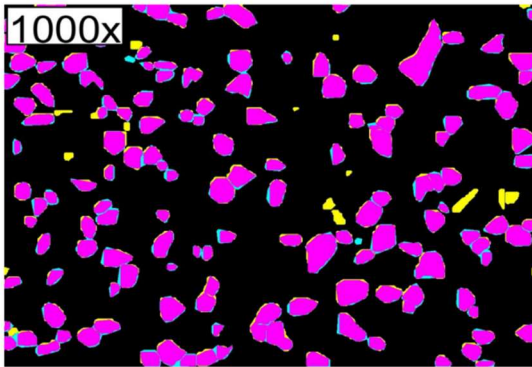


図6.11 Yolov8の推論画像における1000xの可視化結果

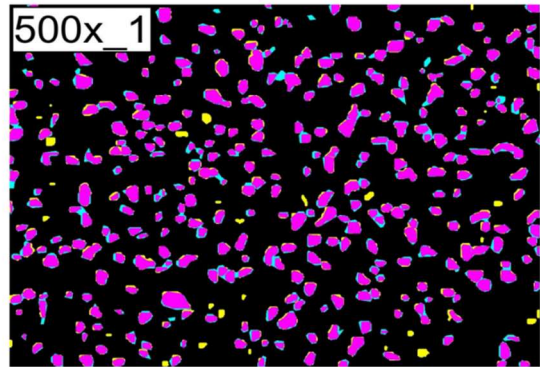


図6.12 Yolov8の推論画像における500x_1の可視化結果

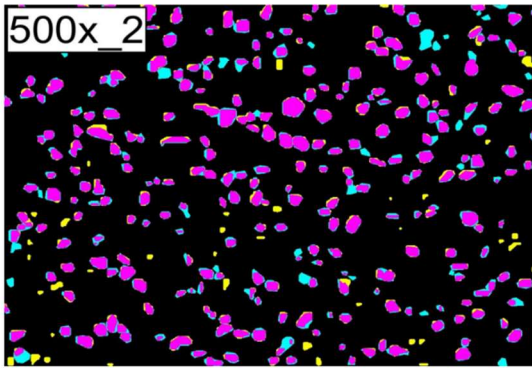


図6.13 Yolov8の推論画像における500x_2の可視化結果

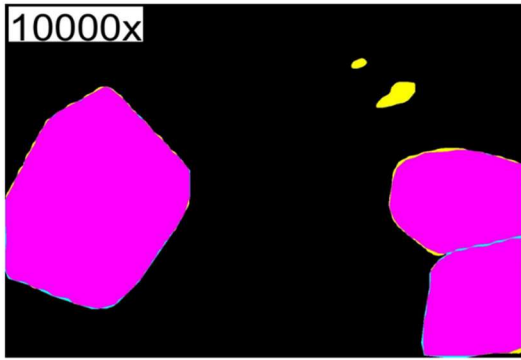


図6.14 Detectron2の推論画像における10000xの可視化結果

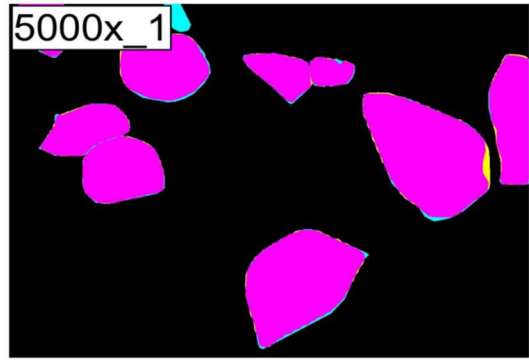


図6.15 Detectron2の推論画像における5000x_1の可視化結果

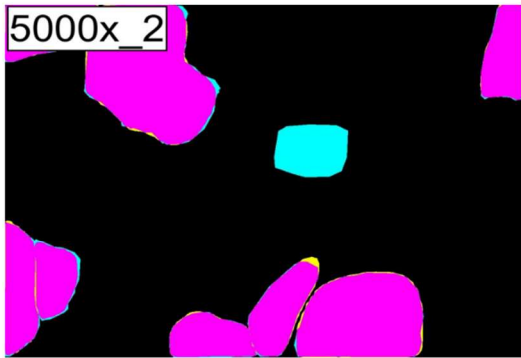


図6.16 Detectron2の推論画像における5000x_2の可視化結果

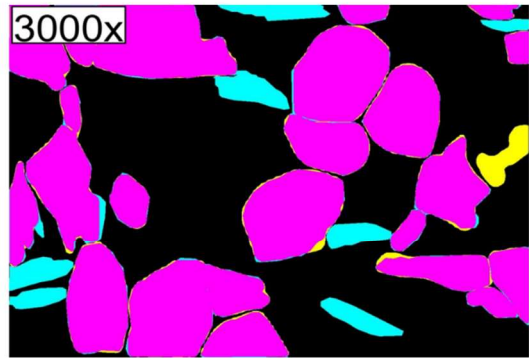


図6.17 Detectron2の推論画像における3000xの可視化結果

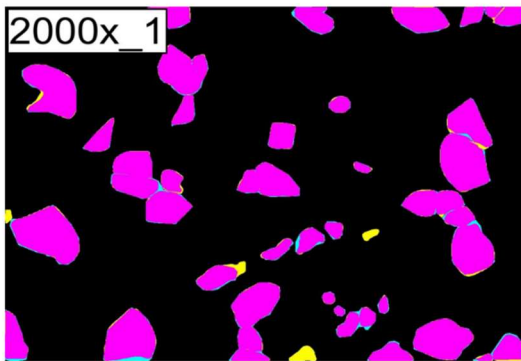


図6.18 Detectron2の推論画像における2000x_1の可視化結果

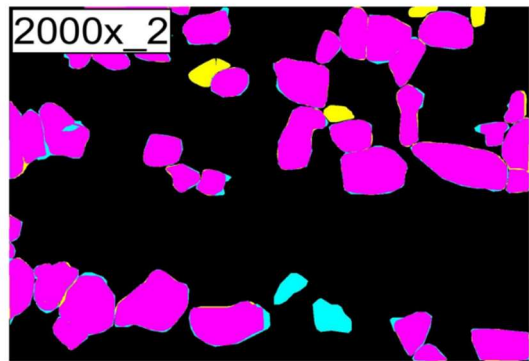


図6.19 Detectron2の推論画像における2000x_2の可視化結果

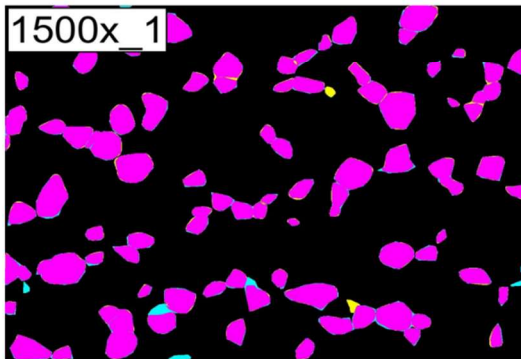


図6.20 Detectron2の推論画像における1500x_1の可視化結果

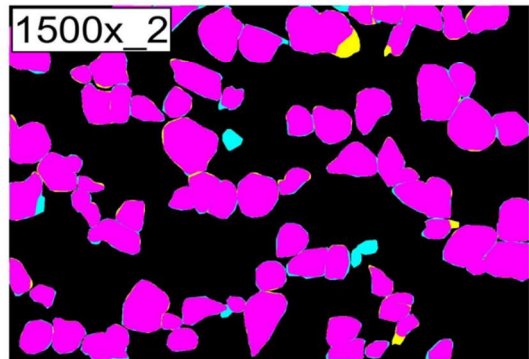


図6.21 Detectron2の推論画像における1500x_2の可視化結果

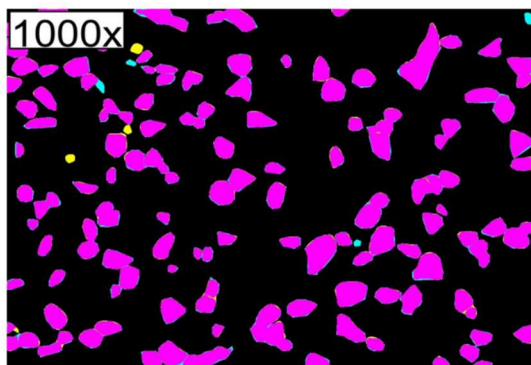


図6.22 Detectron2の推論画像における1000xの可視化結果

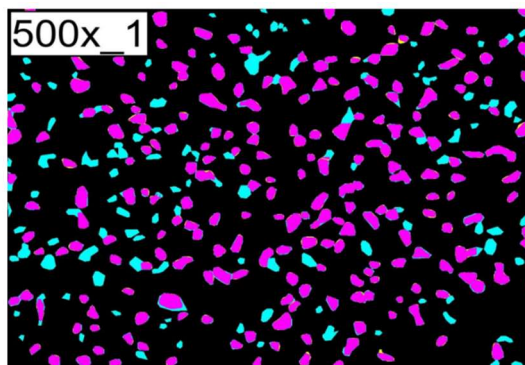


図6.23 Detectron2の推論画像における500x_1の可視化結果

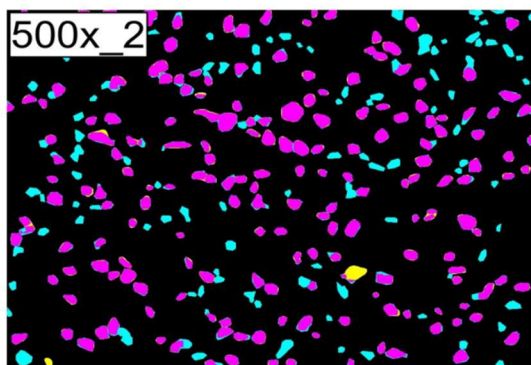


図6.24 Detectron2の推論画像における500x_2の可視化結果

6.4 IoU, Precision, Recall の結果

AP の計算においては個々の粒子が検出できたか検出できなかったかという判断基準を用いて計算を行っていた。つまり個々の粒子に着目して計算していたのである。これ以降では個々のオブジェクトに対して着目するのではなく、画像全体に対して精度がどのようなかという視点から評価するため、新たな評価指標 IoU, Precision, Recall を導入する¹⁷⁾。まずはこれらの計算方法について述べる。計算には先ほど作成した検出領域を可視化した画像を用意する。次に正解エリア (TP), 誤検出エリア (FP), 未検出エリア (FN) に属するピクセル数を算出する。そして以下の式に従いそれぞれ IoU, Precision, Recall を計算する。

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

$$Recall = \frac{TP}{TP + FN} \quad (5)$$

$$IoU = \frac{TP}{TP + FN + FP} \quad (6)$$

これらの評価指標の意味について説明する。Precision は適合率とも呼ばれ黄色の誤検出エリアの少なさを示す。Recall は再現率とも呼ばれ、青色の未検出領域の少なさを示す。IoU, に関しては Precision, Recall を総合的に評価したものであり全体的な評価を示す。

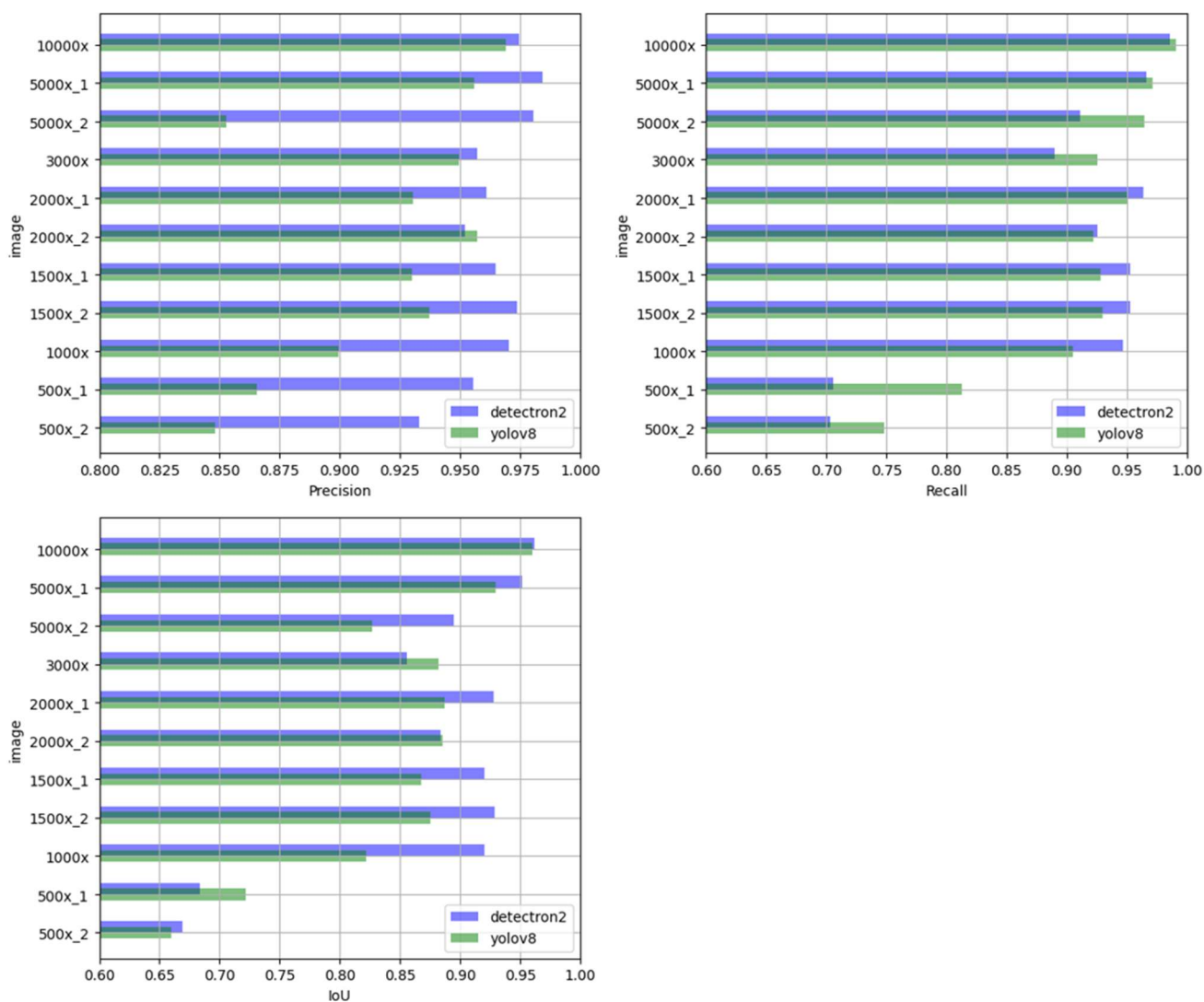


図 6.25 IoU, Precision, Recall の結果

Precision, Recall, IoU は全て 0 以上 1 以下の範囲において値を取り、値が大きいほど精度がよいと判断する。図 6.25 に Yolov8 と Detectron2 における推論画像において IoU, Precision, Recall を計算した結果を示す。

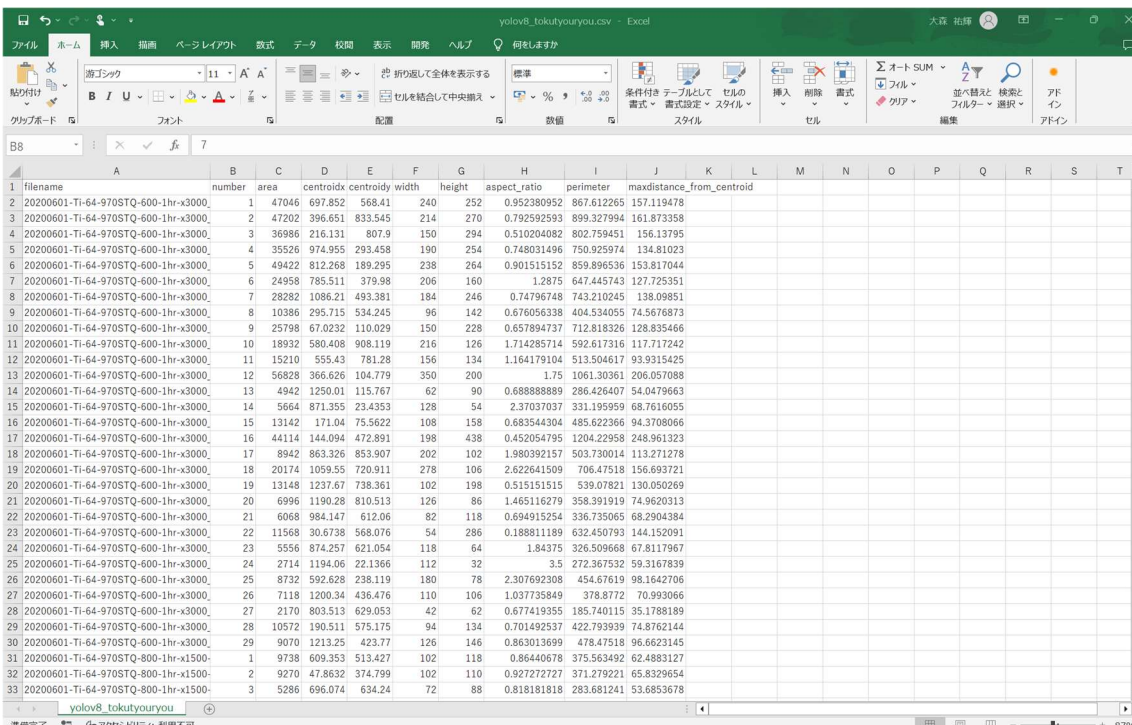
図 6.25 から読み取れる傾向について述べる。第一に、Precision については Detectron2 の方が優れている。これは誤検出エリアが少ないことを示しており、Detectron2 が推論できた粒子においては精度が良いということに起因すると考える。Recall については画像によるといえる。しかしながら 500x_1 や 500x_2 においては Yolov8 の方が明らかに大きくなっている。これは Yolov8 の方が低倍率の SEM 画像に映るような小さなオブジェクトの見逃しが少ないことが原因であると考えられる。Precision と Recall を総合的に評価した IoU に関しては 500x_1 と 3000x を除いた画像において Detectron2 の方が高くなっていることが理解できる。

6.5 YOLOv8 と Detectron2 の特徴の違い

以上で Detectron2 と YOLOv8 の推論結果の評価を行った。以上により Detectron2 と YOLOv8 の特徴の違いは以下の二点に集約される。一つ目の特徴は YOLOv8 においては Detectron2 よりも小さなオブジェクトの見逃しが少ない傾向があるという点である。二つ目の特徴は Detectron2 においては YOLOv8 よりも正確なセグメントを作成しやすい傾向があるという点である。以上の特徴により、目的に応じて YOLOv8 と Detectron2 を使い分ける必要があると考える。

7 特徴量を抽出するプログラムを作成

YOLOv8 と Detectron2 で得られた推論結果と共に出力された図 5.4 の形式のテキストファイルから、検出されたそれぞれの粒子に対する面積、図心の座標、幅、高さ、アスペクト比、周囲長、図心からの最大距離などの特徴量を csv ファイルに出力するプログラムを作成した。特徴量抽出には shapely というライブラリを用いた¹⁷⁾。YOLOv8 による出力結果の様子を図 7.1 に、Detectron2 による出力結果の様子を図 7.2 に示す。



1	filename	number	area	centroidx	centroidy	width	height	aspect_ratio	perimeter	maxdistance_from_centroid
2	20200601-Ti-64-970STQ-600-1hr-x3000	1	47046	697.852	568.41	240	252	0.952380952	867.612265	157.119478
3	20200601-Ti-64-970STQ-600-1hr-x3000	2	47202	396.651	833.545	214	270	0.792592593	899.327994	161.873358
4	20200601-Ti-64-970STQ-600-1hr-x3000	3	36986	216.131	807.9	150	294	0.510204082	802.759451	156.13795
5	20200601-Ti-64-970STQ-600-1hr-x3000	4	35526	974.955	293.458	190	254	0.748031496	750.925974	134.81023
6	20200601-Ti-64-970STQ-600-1hr-x3000	5	49422	812.268	189.295	238	264	0.901515152	859.896536	153.817044
7	20200601-Ti-64-970STQ-600-1hr-x3000	6	24958	785.511	379.98	206	160	1.2875	647.445743	127.725351
8	20200601-Ti-64-970STQ-600-1hr-x3000	7	28282	1086.21	493.381	184	246	0.74796748	743.210245	138.09851
9	20200601-Ti-64-970STQ-600-1hr-x3000	8	10386	295.715	534.245	96	142	0.676056338	404.530455	74.5676873
10	20200601-Ti-64-970STQ-600-1hr-x3000	9	25798	67.0232	110.029	150	228	0.657894737	712.818326	128.835466
11	20200601-Ti-64-970STQ-600-1hr-x3000	10	18932	580.408	908.119	216	126	1.714285714	592.617316	117.717242
12	20200601-Ti-64-970STQ-600-1hr-x3000	11	15210	555.43	781.28	156	134	1.164179104	513.504617	93.9315425
13	20200601-Ti-64-970STQ-600-1hr-x3000	12	56828	366.626	104.779	350	200	1.75	1061.30361	206.057088
14	20200601-Ti-64-970STQ-600-1hr-x3000	13	4942	1250.01	115.767	62	90	0.688888889	286.426407	54.0479663
15	20200601-Ti-64-970STQ-600-1hr-x3000	14	5664	871.355	23.4353	128	54	2.37037037	331.195959	68.7616055
16	20200601-Ti-64-970STQ-600-1hr-x3000	15	13142	171.04	75.5622	108	158	0.683544304	485.622366	94.3708066
17	20200601-Ti-64-970STQ-600-1hr-x3000	16	44114	144.094	472.891	198	438	0.452054795	1204.22958	248.961323
18	20200601-Ti-64-970STQ-600-1hr-x3000	17	8942	863.326	853.907	202	102	1.980392157	503.730014	113.271278
19	20200601-Ti-64-970STQ-600-1hr-x3000	18	20174	1059.55	720.911	278	106	2.622641509	706.47518	156.693721
20	20200601-Ti-64-970STQ-600-1hr-x3000	19	13148	1237.67	738.361	102	198	0.515151515	539.07821	130.050269
21	20200601-Ti-64-970STQ-600-1hr-x3000	20	6996	1190.28	810.513	126	86	1.465116279	358.391919	74.9620313
22	20200601-Ti-64-970STQ-600-1hr-x3000	21	6068	984.147	612.06	82	118	0.694915254	336.735065	68.2904384
23	20200601-Ti-64-970STQ-600-1hr-x3000	22	11568	30.6738	568.076	54	286	0.18881189	632.450793	144.152091
24	20200601-Ti-64-970STQ-600-1hr-x3000	23	5556	874.257	621.054	118	64	1.84375	326.509668	67.8117967
25	20200601-Ti-64-970STQ-600-1hr-x3000	24	2714	1194.06	22.1366	112	32	3.5	272.367532	59.3167839
26	20200601-Ti-64-970STQ-600-1hr-x3000	25	8732	592.628	238.119	180	78	2.307692308	454.67619	98.1642706
27	20200601-Ti-64-970STQ-600-1hr-x3000	26	7118	1200.34	436.476	110	106	1.037735849	378.8772	70.993066
28	20200601-Ti-64-970STQ-600-1hr-x3000	27	2170	803.513	629.053	42	62	0.677419355	185.740115	35.1788189
29	20200601-Ti-64-970STQ-600-1hr-x3000	28	10572	190.511	575.175	94	134	0.701492537	422.793939	74.8762144
30	20200601-Ti-64-970STQ-600-1hr-x3000	29	9070	1213.25	423.77	126	146	0.863013699	478.47518	96.6623145
31	20200601-Ti-64-970STQ-800-1hr-x1500	1	9738	609.353	513.427	102	118	0.86440678	375.563492	62.4883127
32	20200601-Ti-64-970STQ-800-1hr-x1500	2	9270	47.8632	374.799	102	110	0.927272727	371.279221	65.8329654
33	20200601-Ti-64-970STQ-800-1hr-x1500	3	5286	696.074	634.24	72	88	0.818181818	283.681241	53.6853678

図 7.1 YOLOv8 による特徴量を抽出した結果

filename	number	area	centroidx	centroidy	width	height	aspect_ratio	perimeter	maxdistance_from_centroid
20200601-Ti-64-970STQ-600-1hr-x3000-ji	1	10287	296.873	534.583	97	146	0.664383562	403.9899	76.33086
20200601-Ti-64-970STQ-600-1hr-x3000-ji	2	13484	1239.85	742.227	98	205	0.47804878	540.3919	129.631
20200601-Ti-64-970STQ-600-1hr-x3000-ji	3	47661.5	696.742	569.008	250	252	0.992063492	859.3107	153.2159
20200601-Ti-64-970STQ-600-1hr-x3000-ji	4	13476	171.018	73.2294	111	166	0.668674699	500.1076	98.97521
20200601-Ti-64-970STQ-600-1hr-x3000-ji	5	18634.5	580.464	908.156	216	124	1.741935484	591.2031	116.8335
20200601-Ti-64-970STQ-600-1hr-x3000-ji	6	57122	364.65	104.746	355	200	1.775	1045.99	208.9629
20200601-Ti-64-970STQ-600-1hr-x3000-ji	7	4705	1252.03	116.834	60	90	0.666666667	279.7401	51.32035
20200601-Ti-64-970STQ-600-1hr-x3000-ji	8	5720.5	982.85	615.041	86	113	0.761061947	321.262	63.17236
20200601-Ti-64-970STQ-600-1hr-x3000-ji	9	37409.5	216.813	807.777	151	302	0.5	809.4874	162.8477
20200601-Ti-64-970STQ-600-1hr-x3000-ji	10	35489.5	975.588	293.969	188	256	0.734375	745.6539	134.4537
20200601-Ti-64-970STQ-600-1hr-x3000-ji	11	48693	814.365	190.125	235	262	0.896946565	846.3818	152.9383
20200601-Ti-64-970STQ-600-1hr-x3000-ji	12	26679	68.393	111.586	149	234	0.636752137	689.1615	133.1291
20200601-Ti-64-970STQ-600-1hr-x3000-ji	13	46048.5	396.926	835.961	208	263	0.790874525	855.1442	158.5204
20200601-Ti-64-970STQ-600-1hr-x3000-ji	14	5455.5	873.099	23.4623	125	54	2.314814815	323.4386	66.14543
20200601-Ti-64-970STQ-600-1hr-x3000-ji	15	15419.5	553.308	781.919	158	140	1.128571429	518.0904	99.91597
20200601-Ti-64-970STQ-600-1hr-x3000-ji	16	24692	784.861	379.686	210	164	1.280487805	642.2153	126.5197
20200601-Ti-64-970STQ-600-1hr-x3000-ji	17	28283.5	1086.8	494.509	182	259	0.702702703	741.796	143.9321
20200601-Ti-64-970STQ-600-1hr-x3000-ji	18	36384	139.778	496.467	180	351	0.512820513	915.9209	181.8797
20200601-Ti-64-970STQ-600-1hr-x3000-ji	19	6368.5	1189.23	809.569	121	84	1.44047619	347.463	70.51571
20200601-Ti-64-970STQ-600-1hr-x3000-ji	20	20201.5	1061.48	721.85	286	104	2.75	710.5757	158.6034
20200601-Ti-64-970STQ-600-1hr-x3000-ji	21	7790.5	1230	404.093	112	159	0.704402516	517.7473	89.81503
20200601-Ti-64-970STQ-600-1hr-x3000-ji	22	3827	1194.53	17.8697	127	42	3.023809524	315.3381	69.36647
20200601-Ti-64-970STQ-600-1hr-x3000-ji	23	13677.5	29.6187	575.396	71	280	0.24567474	652.2914	150.4913
20200601-Ti-64-970STQ-600-1hr-x3000-ji	24	5470	152.703	287.073	54	146	0.369863014	350.7939	78.89114
20200601-Ti-64-970STQ-600-1hr-x3000-ji	25	4375.5	1186.52	447.933	71	91	0.7802197	270.6834	51.31891
20200601-Ti-64-970STQ-800-1hr-x1500-r	1	7720.5	612.69	105.391	107	104	1.028846154	348.7767	60.9069
20200601-Ti-64-970STQ-800-1hr-x1500-r	2	9605.5	1229.03	691.176	111	103	1.077669903	384.066	67.75261
20200601-Ti-64-970STQ-800-1hr-x1500-r	3	8575	248.587	166.667	96	135	0.711111111	389.0193	77.42699
20200601-Ti-64-970STQ-800-1hr-x1500-r	4	6146	292.173	33.8946	105	82	1.280487805	335.3381	68.27424
20200601-Ti-64-970STQ-800-1hr-x1500-r	5	14994.5	442.971	378.358	137	153	0.895424837	477.4874	80.22752
20200601-Ti-64-970STQ-800-1hr-x1500-r	6	11681.5	624.166	844.883	105	175	0.6	471.5462	105.9105
20200601-Ti-64-970STQ-800-1hr-x1500-r	7	9775	46.4099	126.156	101	126	0.801587302	408.3087	81.65155

図 7.2 Detectron2 による特徴量を抽出した結果

8.特徴量の補正

Yolov8 と Detectron2 において推論領域に対する特徴量を抽出した。以降では、この推論領域に対する特徴量に対して補正を行い、正解領域に対する特徴量にできるだけ近づけるということを行いたい。今回は例として推論領域の面積に対して補正することを考える。面積の補正方法としては二つの方法を試す。一つ目の方法は確率密度関数の最頻値により補正する方法である。二つ目の方法は機械学習により補正する方法である。以降で順に説明する。

8.1 Ratio の最頻値による面積の補正

一つ目の方法について説明する。まず IoU の閾値を 0.25 とした際に検出成功とされた粒子に対して正解領域の面積と推論領域の面積を計算する。次に以下の式に従い Ratio を計算する。

$$Ratio = \frac{\text{正解領域の面積}}{\text{推論領域の面積}} \quad (7)$$

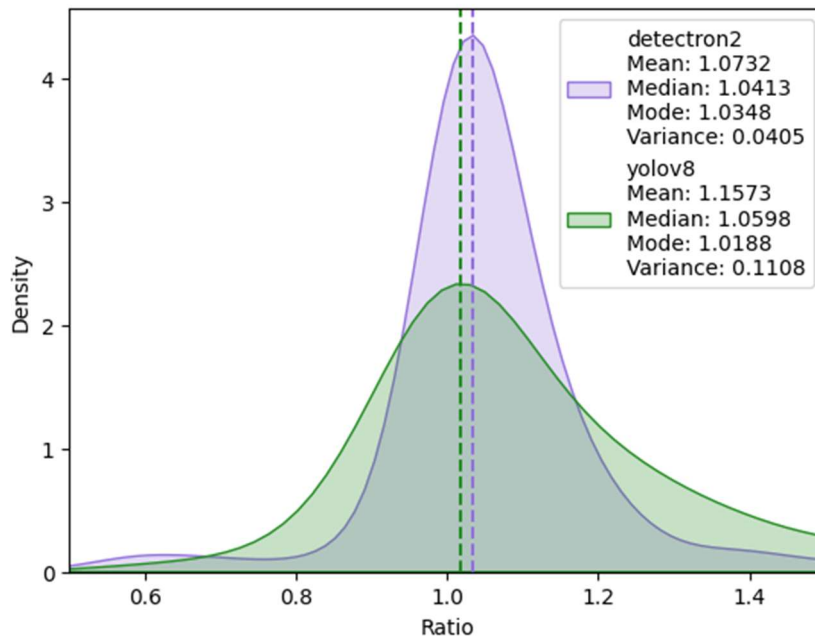


図 8.1 Ratio の確率密度関数と最頻値

次に計算した Ratio の確率密度関数を計算し、確率密度関数が最大値をとるときの Ratio、つまり最頻値(Mode)を計算する。図 8.1 に Yolov8 と Detectron2 における Ratio の確率密度関数と、その凡例中に最頻値(Mode)の値を示す。また参考として他の統計量である平均値(Mean)、中央値(Median)、分散(Variance)についても凡例中に示す。

面積の補正を行う方法としては、推論領域の面積と最頻値(Mode)の積を考えればよい。この方法の長所としては、全ての特微量に対して同じ値を掛け算するので、補正前におけるすべての特微量の間の大小関係は補正後においても維持される点である。欠点としては全てに同じ値を掛けるので、一部の特微量においては補正方法として適切でないという点がある。

8.2 機械学習(LightGBM)による面積の補正

8.1 とは別の補正方法を考える。その手順について説明する。まず 8.1 と同様の方法で Ratio を計算する。次に目的変数を Ratio、説明変数を推論領域の面積、対象のオブジェクトが属する SEM 画像の倍率として機械学習を行う。なお学習においては検出成功とした粒子の内、8割を用いて学習を行った。機械学習モデルとして今回は LightGBM を用いた¹⁸⁾。学習後には学習済みモデルに対して未知のデータを入力し Ratio を予測する。最後に推論領域の面積と予測された Ratio との積を補正後の面積とする。

8.3 Ratio の最頻値を用いる方法と機械学習を用いる方法との MSE の比較

8.2 においては検出成功とした粒子の内、8割を用いて学習を行った。以下の評価では残りの 2割のみを考慮する。8.1 により求めた最頻値の Ratio の結果と 8.2 により予測した

Ratio の結果を比較するために両方の結果に対して、MSE を計算する。MSE は以下の式で定義される¹⁹⁾。

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (8)$$

ここで n はデータ数である。 y_i は実際の Ratio である。 \hat{y}_i については 8.1 を評価する際は最頻値の Ratio を用い、8.2 を評価する際は学習済みモデルで予測した Ratio を用いる。また MSE は小さいほど実際の値と予測した値の差が小さく精度がよいことを示す。表 8.1 に MSE を計算した結果を示す。表 8.1 より YOLOv8, Detectron2 両方において機械学習を用いた方が MSE の値が小さくなっており、精度が良いことが分かる。この点が機械学習を用いた場合の長所であると考えられる。一方、短所としては補正前におけるすべての特徴量の間の大小関係は補正後においても維持されるとは限らないという点である。最後に参考として図 8.2 に YOLOv8 において 8.1 により求めた Ratio と実際の Ratio、図 8.3 に YOLOv8 において 8.2 により求めた Ratio と実際の Ratio、図 8.4 に Detectron2 において 8.1 により求めた Ratio と実際の Ratio、8.5 に Detectron2 において 8.2 により求めた Ratio と実際の Ratio を示す。

表 8.1 2つの補正方法における MSE の結果

	YOLOv8	Detectron2
8.1(最頻値)によるMSE	0.1531	0.0371
8.2(機械学習)によるMSE	0.0639	0.0202

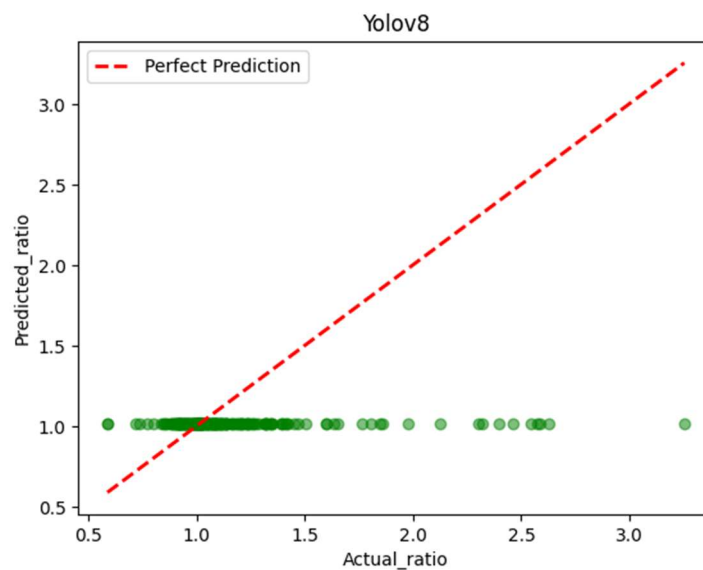


図 8.2 YOLOv8 において 8.1 により求めた Ratio と実際の Ratio

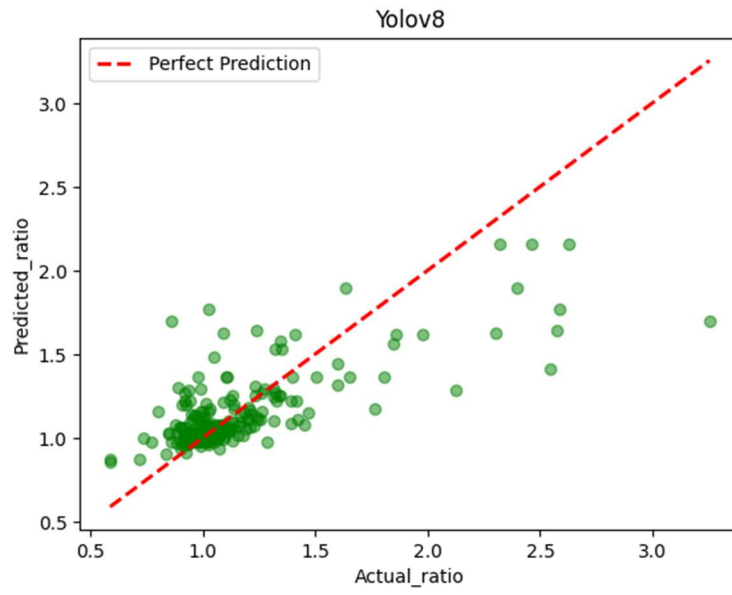


図 8.3 Yolov8 において 8.2 により求めた Ratio と実際の Ratio

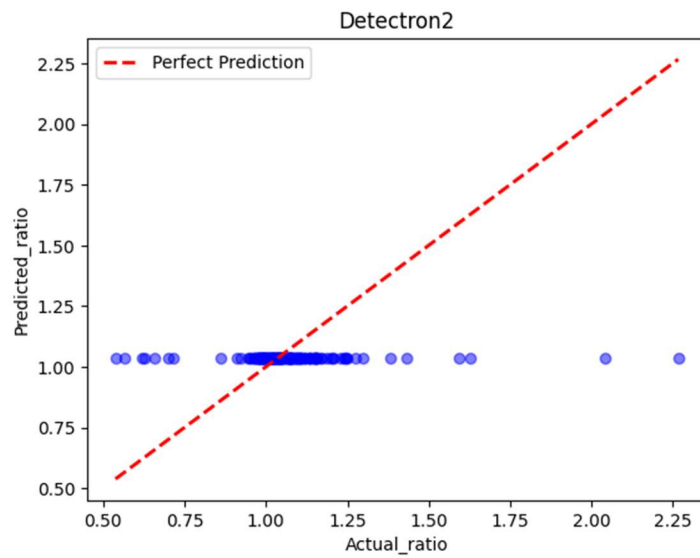


図 8.4 Detectron2 において 8.1 により求めた Ratio と実際の Ratio

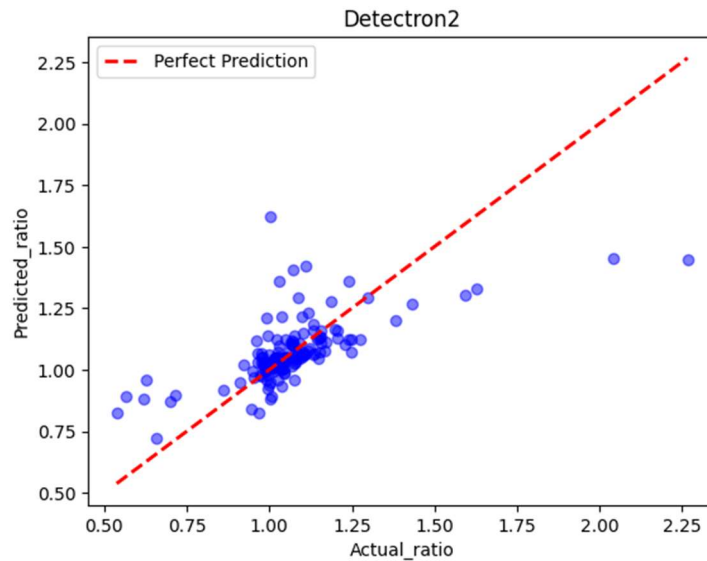


図 8.5 Detectron2 において 8.2 により求めた Ratio と実際の Ratio

9. 結論

本研究ではまず、二つのセグメンテーションライブラリ YOLOv8 と Detectron2 を用いて、Ti-6Al-4V の様々な熱処理条件下での走査型電子顕微鏡 (SEM) 画像の構成相に対してインスタンスセグメンテーションを行い、両ライブラリの比較を行った。その結果として YOLOv8 は Detectron2 よりも小さなオブジェクトの見逃しが少ないことが分かった。一方、Detectron2 は YOLOv8 よりも正確なセグメントを作成できることが分かった。次にインスタンスセグメンテーションにより得た粒子のセグメントの座標から特徴量を抽出するプログラムを作成し、特徴量を抽出できた。最後に、抽出した特徴量を真の値に近づけるために、抽出した特徴量を Ratio の最頻値を用いる方法と機械学習を用いる方法の二つにより補正した。MSE の評価指標においては機械学習を用いる方法の方が特徴量の補正において有用であることが分かった。

10. 参考文献

- [1] <https://www.libcon.co.jp/column/materials-informatics/>
- [2] Marc Ackermann, Deniz Iren, Sebastian Wesselmecking, Deekshith Shetty, Ulrich Krupp. Automated segmentation of martensite-austenite islands in bainitic steel. *Materials Characterization* 191(2022)112091
- [3] Hiroaki Matsumoto, Hiroshi Yoneda, Kazuhisa Sato, Shingo Kurosu, Eric Maire, Damien Fabregue, Toyohiko J. Konno, Akihiko Chiba. Room-temperature ductility of Ti-6Al-4V alloy with α' martensite microstructure. *Materials Science and Engineering A*. 528(2011) 1512-1520.
- [4] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. Mask R-CNN. Cornell

- University. (arXiv:1703.06870v3 [cs.CV] 24 Jan 2018)
- [5] <https://blog.negativemind.com/2019/04/27/general-object-detection-and-instance-segmentation-mask-r-cnn/>
- [6] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. University of Washington, Allen Institute for AI, Facebook AI Reserch. (arXiv:1506.02640v5 [cs.CV] 9 May 2016)
- [7] <https://roboflow.com/>
- [8] 斎藤康毅. ゼロから作る Deep Learning Python で学ぶディープラーニングの理論と実装. 2016.
- [9] https://qiita.com/cv_carnavi/items/08e11426e2fac8433fed
- [10] <https://docs.ultralytics.com/tasks/segment/>
- [11] <https://docs.ultralytics.com/usage/cfg/>
- [12] <https://www.preferred.jp/ja/projects/optuna/>
- [13] https://github.com/facebookresearch/detectron2/blob/main/MODEL_ZOO.md
- [14] <https://pytorch.org/docs/stable/optim.html>
- [15] <https://detectron2.readthedocs.io/en/latest/>
- [16] <https://tech-blog.optim.co.jp/entry/2019/03/18/173000>
- [17] <https://shapely.readthedocs.io/en/stable/manual.html>
- [18] <https://lightgbm.readthedocs.io/en/stable/>
- [19] <https://www.scsk.jp/sp/mwai/blog/cat/rmse.html>

11. 謝辞

本研究を行うにあたり、様々なご指導をいただきました指導教員である杉尾健次准教授、並びに佐々木元教授には、常日頃より親切丁寧にご指導していただきましたことを深く感謝申し上げます。さらに、公私にわたり様々な助言をいただきました材料物理学研究室の皆様、誠にありがとうございました。