令和 6 年度卒業論文

様々な特徴抽出器を用いた動画像の連続フレームからの物体検出の評価

宮崎大学　工学部　工学科　情報通信工学プログラム

学籍番号　60211682

新西　拓斗

指導教員　椋木雅之教授

2025 年 2 月 7 日

目次

1. はじめに

物体検出とは、動画や単一画像の中に含まれる特定の物体の位置と範囲を推定する技術のことである。

近年、深層学習を用いた物体検出手法が提案されている。深層学習では、大量のデータを学習することにより、人手で設計するよりも高性能な特徴量抽出器が構築できるため、高精度の物体検出を可能にしている。例えば、YOLO[1]は DarkNet[2]と呼ばれる深層学習を用いた特徴量抽出器を使用することで高速高精度な物体検出を実現している。

物体検出は、動画像にも適用されている。動画像は、連続する時刻で取得した画像(フレーム)を時間方向に並べたものである。連続するフレームに物体検出を順次適用し、フレーム間で対応付けることで、物体を追跡することができる。このような物体追跡へのアプローチは Tracking by detection(検出による追跡)と呼ばれ、近年、多用されている。その代表例として、SORT[3]と呼ばれる手法がある。SORT では、フレーム間で近い位置に検出された物体同士を対応付けることで追跡を行う。SORT の利点としては、検出ができれば高精度の追跡が行えることや処理速度が速いことが挙げられる。一方、欠点としては、検出に失敗すると追跡も行えないことが挙げられる。連続するフレームでは画像内容の変化は小さい。それにもかかわらず、連続するフレームに物体検出を適用すると、あるフレームでは適切に検出できたのに、ほとんど画像内容に変化のない次のフレームでは検出に失敗することがある。Tracking by detection のアプローチをとると、このような検出失敗により追跡が途切れることが問題となる。

これに対し、田邊[4]は動画像の連続する 3 フレームを入力として与えて物体検出する手法を提案した。前後のフレームにより情報を補うことで、連続して検出が行える。実験結果では、動画像の各フレームに対して 1 枚ずつ独立に物体検出するより、時間方向に安定して物体検出が行えることを示した。しかし、物体検出の精度自体は高くなかった。

物体検出の精度に影響を与える要素の一つに、特徴抽出器の選択が挙げられる。近年の深層学習による物体検出では、特徴抽出器で得られた特徴量を元に、画像のどの範囲に検出対象の物体が存在するか推定する。この際、検出対象固有の特徴量が得られれば、その画像上での範囲を推定することは容易となり、検出精度が向上する。田邊の研究では、ResNet[5]を特徴抽出器として使用していたが、特徴抽出器の影響を十分検討していなかった。特徴抽出器としては、画像分類のタスクで高い性能を示すものを利用するのが良いと考えられる。近年、EfficientNet[6]や SwinTransformer[7]などの手法が画像分類において高い性能を示しており、これらを特徴抽出器として利用することで、物体検出の精度を向上させることができる可能性がある。

そこで本研究では、動画像からの物体検出において、特徴抽出器を変更した際の時間方向での物体検出精度を評価する。そのために、田邊の研究では ResNet によって特徴抽出を行っていた部分を、より高性能な特徴抽出器へと変更し、それらを用いた物体検出手法

を比較・検討する。具体的には、ResNet に加えて EfficientNet、SwinTransformer などの異なる構造の特徴抽出器を試し、それぞれの特徴量を元に物体検出を行う手法を比較して評価する。これにより、特徴抽出器の違いによる検出精度の変化を分析し、最適な構成を探索する。

以下、2 章では、連続フレームを用いた物体検出の従来手法について述べる。3 章では、本研究で検討する複数の特徴抽出器の詳細について述べる。4 章では、特徴抽出器を ResNet、EfficientNet、SwinTransformer などに変更した場合の物体検出結果を比較し、それぞれの手法の精度について評価する。5 章では、本論文の結論と今後の課題について述べる。

2.　深層学習による物体検出

2.1　単一画像からの物体検出

物体検出は、与えられた画像の中に写されている特定の物体の位置や範囲を推定する技術である。通常、物体検出は与えられた単一の入力画像に対して処理を行う。例えば、Viola ら[8]は、与えられた 1 枚の画像の中から、顔の領域を推定する手法を提案している。この手法では、Haar-like 特徴量と呼ばれる簡単な特徴量を使って、比較的性能の低い識別器（弱識別器）を順次大量に適用することで人の顔を検出する。この手法を含め、従来は検出対象となる物体を表現する特徴量を人手で設計し、物体検出処理に与えていた。

これに対して、近年、深層学習を使った画像認識手法が多く提案され、物体検出でも高い性能を示している。深層学習による物体検出では、様々なネットワーク構造が示されているが、それらは主に特徴抽出器（バックボーン）、ネック、物体検出器(ヘッド)の 3 つの主要部分から構成される。

特徴抽出器(バックボーン)では画像中の物体に関する情報を特徴量として抽出する。大量の画像データを学習することで、人手で設計した特徴量よりも物体検出に適した特徴量を抽出できるようになる。特徴抽出器は、特定の物体検出タスク毎に学習することもできるが、多種多量の画像データにより事前に学習したものを利用することもできる。このような学習済みの特徴抽出器として、VGG[9]や ResNet[5]などが広く利用されている。

次に、ネックは、特徴抽出器で得られた特徴量をさらに処理し、物体検出の精度を向上させる役割を担う。一般的な手法として、FPN(Feature Pyramid Network)[10]や PAN(Path Aggregation Network)[11]などが用いられる。特に FPN は異なるスケールの特徴を統合することで、小さな物体の検出精度を向上させる効果がある。

最後に、物体検出器(ヘッド)では、ネックから得られた特徴量を基に、物体の位置（バウンディングボックス）やクラスを予測する。物体検出の代表的な手法である YOLO[2]では、ヘッドにおいて信頼度スコアを使用し、どの領域に対象クラスの物体が存在しているかを判断する。信頼度スコアは、「分割された領域（バウンディングボックス）に物体が入っていて、正確に領域を囲えているかの正確さ」と「各クラスの予測確率」を意味する指標である。この信頼度スコアにより、領域候補の探索とクラスの識別を同時に行うことができるため、リアルタイムに近い処理速度を実現している。

近年 YOLO はさらなる進歩を遂げており、新たな YOLO のモデルが提案されている。YOLO v3[12]では様々なスケールの物体の検出を行うために、特徴マップの大きさに応じて、 3 つの出力層が存在する（図 1)。そのため、特徴量の大きさにあった、特徴抽出器の特徴量と物体検出器の特徴量を結合することで、異なる大きさの物体検出に対応している。例えば、416×416 の画像を入力とする場合、13×13、26×26、52×52 の大きさに合った特徴抽出器の特徴量と物体検出器の特徴量を結合し、それぞれを出力層に与えてい

る。

このように、物体検出結果として物体を囲う矩形の枠を出力する手法に対して、U-Net[13]のような画素単位でより細かく物体の領域を出力するセマンティックセグメンテーション[14]を用いた手法も存在する。U-Net は特定の細胞領域の検出や臓器領域の検出に使用されている。U-Net が適用される画像は、特定分野の類似した画像となるが、特定分野の比較的少数の学習データで高い検出精度が得られる。

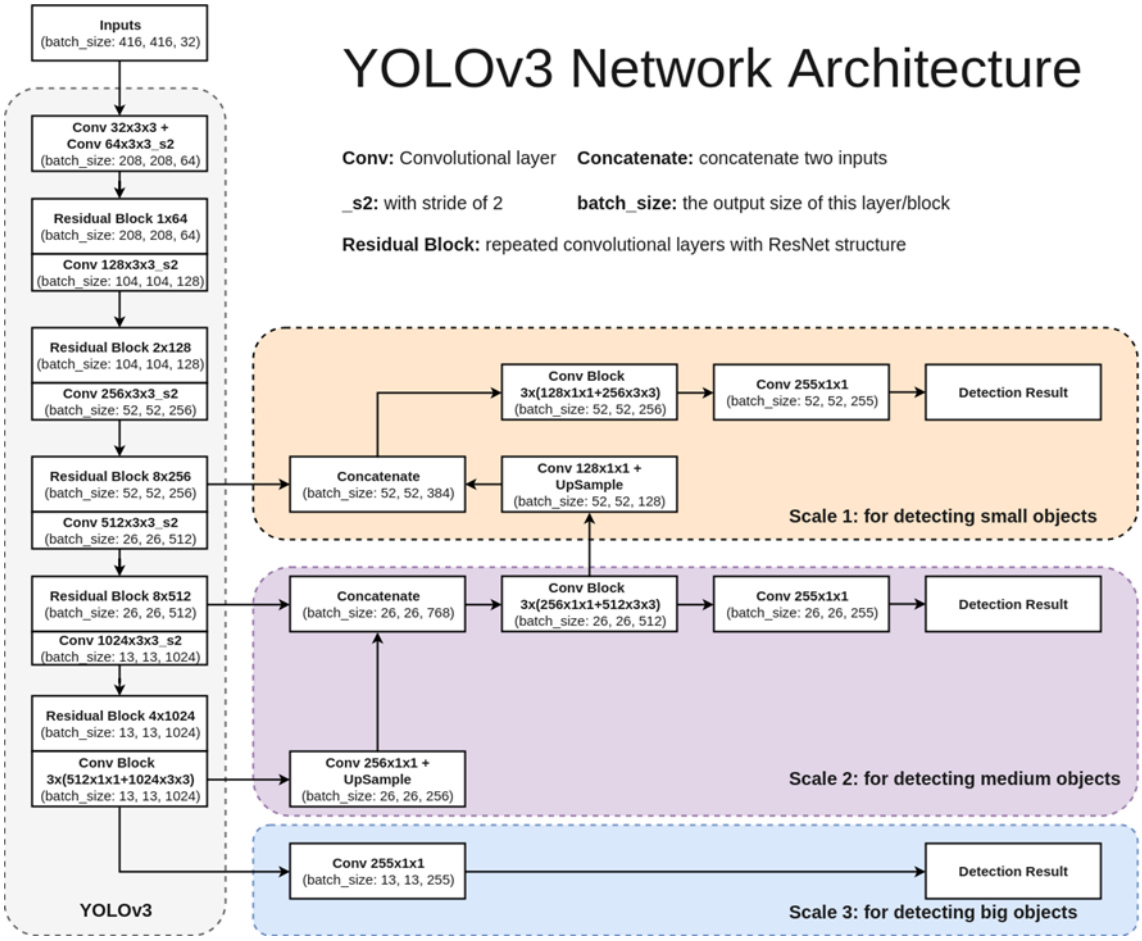

図 1　YOLO v3 のネットワーク構造[15]

2.2　動画像からの物体検出

動画像の連続フレームに物体検出を順次適用し、フレーム間で対応付けることで、物体を追跡することができる。このような物体追跡へのアプローチは Tracking by detection（検出による追跡）と呼ばれ、近年、多用されている。その代表例として、SORT[3]と呼ばれる手法がある。SORT では、フレーム間で近い位置に検出された物体同士を対応付けることで追跡を行う。

Shuai[16]らの手法は、上記の SORT を改良したものであり、YOLO で検出したバウンディングボックス（Bounding Box; BBox）を使用し、フレームの前後で近い大きさと近い動きを持つ BBox を対応づけることで、検出対象に ID をつけて追跡を行う。また、DeepSORT[17]も SORT を改良したモデルであり、外観の類似度を比較する AI モデルを使用することで、対応付けの際に見た目の類似度の情報を利用する。これらの手法は一定時間のフレームの情報を保持するが、物体検出に一定時間続けて失敗すると、そのデータが破棄され、追跡が途切れるという共通した問題点がある。

深層学習による物体検出の性能向上に伴い、SORT のような Tracking by detection のアプローチが実用的になってきている。しかし、検出性能が高くても、動画像の多数のフレームを処理していくと、時々物体検出に失敗することがある。上述の通り、SORT ではこのような検出失敗が生じると追跡処理も失敗してしまう。そのため、深層学習を用いて動画像に特化した物体検出を行う手法も提案されている。

ROLO[18]は、LSTM（Long Short-Term Memory）と呼ばれる時系列データを処理できる深層学習のネットワークを用いた動画像からの物体検出手法である。ROLO では YOLO を用いて物体の位置と範囲（BBox）を抽出し、この BBox を LSTM にフィードバックすることで、各フレームでの物体検出時に前フレームで検出された物体の位置や範囲の情報を利用する。これにより、物体同士の重なりがあった場合などでも、物体検出の位置精度が向上する。しかし、時系列方向で利用しているのは物体の位置や範囲の情報のみであり、前フレームの画像情報は利用していない。そのため、時間方向での連続した物体検出の安定化には十分に寄与しないという問題がある。

U-Net3DT[19]は、U-Net[13]を時間方向に拡張し、3 次元畳み込みを導入することで物体検出の精度向上を目指した動画像からの物体検出手法である。U-Net3DT のネットワーク構造を図 2 に示す。U-Net3DT では、画像を時間方向に束ねた画像群を 3 次元画像とみなして処理を行う。n 枚の束ねた画像を入力として与えると、n 枚の物体検出結果が得られる。物体検出結果は、U-Net と同様に BBox ではなく画素単位で表現される（セマンティックセグメンテーション）。

U-Net3DT では、VGG や ResNet のような学習済みモデルを特徴抽出器（バックボーン）として利用できず、すべてのパラメータをゼロから学習する必要がある。そのため、学習に多くのデータが必要であり、性能があまり良くないといった問題がある。

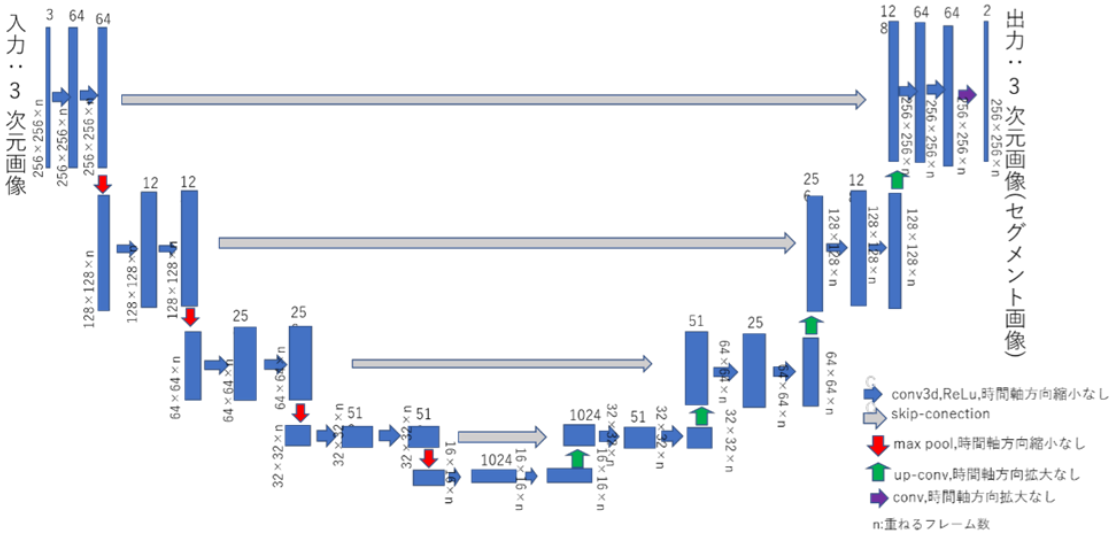

図 2　U-Net3DT のネットワーク構造[19]

2.3　複数フレームを用いた物体検出

動画像からの物体検出において、連続するフレームであっても検出失敗により追跡が途切れる問題がある。そのため、物体検出を適用したいフレームだけでなくその前後を含めた連続した 3 フレームを入力として与えて処理する手法が田邊[4]によって提案された(図 3)。入力された各フレームから特徴抽出器で特徴量を抽出する。得られた 3 フレーム分の特徴量を結合して物体検出器に与えて、物体を検出する。連続フレームを束ねて入力として与えることにより、前後のフレーム情報も物体検出に利用できるようになるため、連続して物体を検出できる。例えば、図 4 のように連続した 3 枚の画像に対して物体検出処理を行ったとする。今までの手法では画像 1 枚に対して独立して物体検出処理を行うため、時刻 t+1 と t-1 の検出に成功しても時刻 t での検出に失敗する場合がある。これに対して、この手法では時刻 t の前後フレームである物体検出に成功している時刻 t-1 と t+1 の特徴量を時刻 t での物体検出処理に利用することができる。これにより、時刻 t での検出結果が向上すると考えられる。

この研究のネットワーク(図 3)は、3 フレームの入力に対し、3 つの特徴抽出器を並べて、それぞれ適用する構造となる。特徴抽出器には、既存の学習済み ResNet152 を利用している。得られた 3 フレーム分の特徴量は結合して物体検出器に与える。物体検出器は、YOLO v3 のネットワーク構造を参考に、結合した特徴量を扱えるようにチャネル数等を変更したものを利用する。

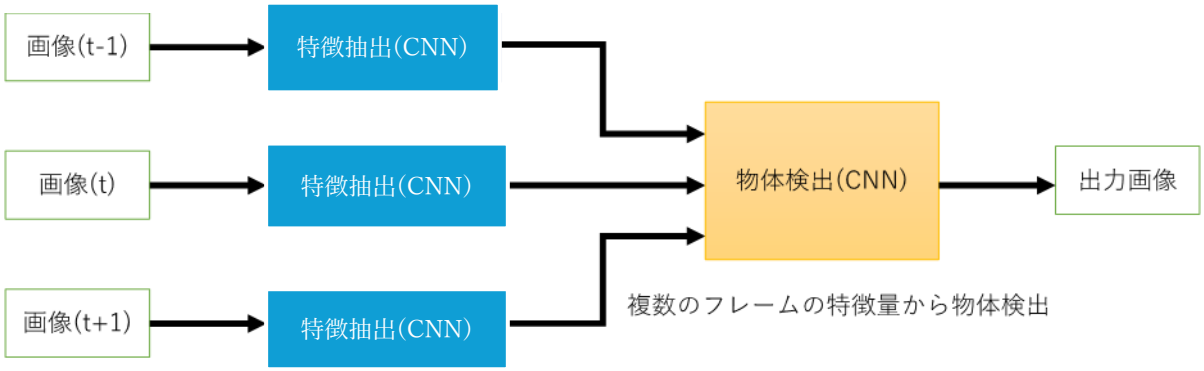

図３　３フレームを用いた手法の物体検出の考え方[4]

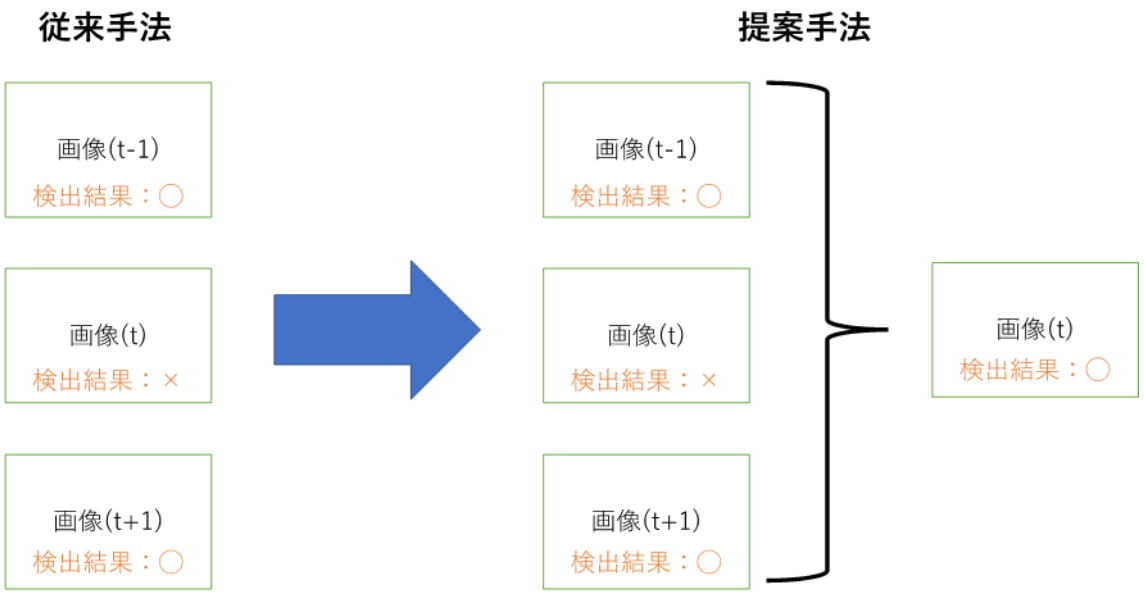

図４　今までの手法と３フレームを用いた手法の違い[4]

2.4　田邊[4]の手法の問題点

田邊の手法では時間方向に安定して物体検出を行うために、動画像の連続する 3 フレームを入力として与えて物体検出する。実験結果では、動画像の各フレームに対して 1 枚ずつ独立に物体検出するより、時間方向に安定して物体検出が行えることを示した。しかし、物体検出の精度自体は高くないという問題がある。

3.　様々な特徴抽出器を用いた物体検出の改良

3.1　様々な特徴抽出器

田邊[4]の手法で物体検出精度が悪い原因の 1 つとして特徴抽出器が考えられる。近年の深層学習による物体検出では、特徴抽出器で得られた特徴量を元に、画像のどの範囲に検出対象の物体が存在するか推定する。この際、検出対象固有の特徴量が得られれば、その画像上での範囲を推定することは容易となり、検出精度が向上する。田邊の手法では、ResNet[5]を特徴抽出器として使用していたが、特徴抽出器の影響を十分検討していなかった。特徴抽出器としては、画像分類のタスクで高い性能を示すものを利用するのが良いと考えられる。近年、EfficientNet[6]や SwinTransformer[7]などの手法が画像分類において高い性能を示しており、これらを特徴抽出器として利用することで、物体検出の精度を向上させることができる可能性がある。

そこで本研究では、田邊[4]の手法の特徴抽出器を様々な特徴抽出器に変更する。特徴抽出器には、ImageNet[20]により学習済みの ResNet18, EfficientNet-B0, EfficientNet-B7, SwinTransforemerB を利用する。ImageNet は、画像認識の分野で最も広く使われている大規模なデータセットの一つである。ImageNet には 1000 種類以上のカテゴリ（クラス）に分類された約 130 万枚のラベル付き画像（ImageNet-1K）が含まれている。本研究では、torchvision[21]、timm[22]が提供している ImageNet で学習済みのネットワークを使用する。

ResNet は、深層学習モデルにおいて層を深くすると学習が困難になる「勾配消失問題」を解決するために、スキップ接続（Skip Connection）を導入したネットワークである。スキップ接続により、入力情報が層をスキップして後の層に直接伝達されることで、勾配が適切に伝播し、より深いネットワークの学習が可能となる。ResNet には 18 層、34 層、50 層、101 層、152 層の 5 種類が提案されており、層が深いほど表現能力が向上するが、計算コストも増大する。ResNet は、畳み込み層と残差ブロック（Residual Block）を基本構造とし、浅い層で学習した特徴を深い層へ伝えながら、高次の特徴を学習する。特に、ResNet50 以降のモデルでは、ボトルネック構造（Bottleneck Block）を採用し、計算コストを抑えつつ高い表現力を持たせている。本研究では、計算コストと精度のバランスを考慮し、18 層の ResNet（ResNet18）を特徴抽出器として使用する。

EfficientNet は、モデルの深さ（Depth）、幅（Width）、解像度（Resolution）を効率的に調整することで、高い精度を維持しながら計算コストを抑えた畳み込みニューラルネットワーク（CNN）である。従来のネットワークは、深さ・幅・解像度を個別に調整することが一般的だったが、EfficientNet ではコンパウンドスケーリング（Compound Scaling）を導入し、これら 3 つの要素をバランスよくスケーリングすることで、少ないパラメータで高精度な特徴抽出を可能としている。EfficientNet には B0 から B7 までの 8 種類のモデ

ルが提案されており、B0 が最も軽量で、B7 に向かうにつれてネットワークが深くなり、高精度な特徴抽出が可能となる。EfficientNet の基本構造には、MBConv（Mobile Inverted Bottleneck Convolution）[23]と呼ばれる特殊な畳み込み層が用いられており、Depthwise Convolution[24]と Pointwise Convolution[25]を組み合わせた構造を採用している。これにより、パラメータ数を削減しながらも、高い表現力を維持することができる。また、活性化関数には Swish 関数が採用されており、ReLU[26]よりも滑らかな非線形性を持つことで、精度向上に寄与している。さらに、SE（Squeeze-and-Excitation）モジュールを導入することで、チャネルごとの特徴を強調し、重要な特徴量をより効果的に学習することができる。本研究では、EfficientNet の B0 および B7 を特徴抽出器として使用する。EfficientNet-B7 はより高精度な特徴抽出が可能であるが、計算コストが高いため、軽量な EfficientNet-B0 との比較を行う。

SwinTransformer は従来の CNN と Transformer の利点を組み合わせたネットワークであり、T(Tiny)、S(Small)、B(Base)、L(Large)の 4 つのモデルがある。まず、入力画像を小さな領域に分割するパッチ分割（Patch Partition）を行い、各パッチを特徴ベクトルとして処理することで、情報を効果的に抽出する。次に、計算コストを抑えながら局所的な特徴を学習するためにウィンドウアテンション（Window Self-Attention）を導入し、ウィンドウ内の情報に基づいた自己注意を適用する。さらに、単なるウィンドウ単位の処理にとどまらず、ウィンドウの位置をずらしながらアテンションを適用するシフトウィンドウアテンション（Shifted Window Attention）を採用することで、異なる領域間の情報を統合し、より広範囲な特徴を学習できるようにしている。また、SwinTransformer は CNN のように 階層的な構造 を持ち、特徴マップの解像度を段階的に縮小しながら局所的な特徴から大域的な特徴へと学習を進める。この構造により、画像全体の文脈を考慮した表現が可能となり、高精度な認識を実現する。最終的に、全結合層を用いたヘッド（Fully Connected Layer）を適用することで、画像分類や物体検出などのタスクに対応できる。本研究では、計算コストと精度のバランスを考慮し、SwinTransformer の B を特徴抽出器として使用する。

3.2　提案手法のネットワーク構造

本研究では、田邊[4]の手法を参考にして、物体検出器を構築した。提案手法のネットワーク構造の EfficientNet-B0 の場合を図 5 に示す。物体検出器では、物体検出畳み込み層において、物体検出の対象に特化した特徴量抽出を行っている。物体検出畳み込み層は「畳み込み+Batch Normalization+ReLu」で構成されている。

YOLO v3 の物体検出ネットワークでは、3 段階の大きさの特徴量を利用する。表 1 に、各特徴抽出器から抽出する層と、入力画像の大きさを 608×608 にした際の各層から得られる特徴量(特徴マップ)のサイズとチャネル数を示す。ただし、SwinTransformer は、入力画像の大きさが 224×224 に固定なので、その時のサイズとチャネル数を示している。表 1 の各特徴抽出器で抽出する 3 つの層の下の方から順に物体検出畳み込み層 1、物体検出畳み込み層 2、物体検出畳み込み層 3 に入力として与える。以下に EfficientNet-B0 の場合の各層の流れを示す。

EfficientNet-B0 の 7 層目からの特徴マップは 1 フレームあたり 192 チャネルで、これを 3 フレーム分結合した 576 チャネルの特徴マップを使用して物体検出畳み込み層 1 で処理を行う。物体検出畳み込み層 1 の出力は、大きさ 38×38、1024 チャネルの特徴マップである。この出力からチャネル数を増やした、大きさ 38×38、2048 チャネルの特徴マップを元に出力層 1 で処理を行う。同時に、この特徴マップを 2 倍の大きさ(76×76)にアップサンプリングし、512 チャネルにした特徴マップを次の物体検出畳み込み層 2 で利用する。物体検出畳み込み層 2 では、5 層目からの特徴マップ(大きさ 76×76、80 チャネル)3 フレーム分と、上記の物体検出畳み込み層 1 から得られる特徴マップ(512 チャネル)を全て結合した 752 チャネルの特徴マップを使用する。同様に、物体検出畳み込み層 3 では、4 層目からの特徴マップ(大きさ 152×152、40 チャネル)3 フレーム分と、物体検出畳み込み層 2 から得られる特徴マップ(256 チャネル)を全て結合した 376 チャネルの特徴マップを使用する。それぞれの物体検出畳み込み層に応じて、3 つの出力層が存在する。出力層は物体検出畳み込み層の結果から形状が(C,H,W)の特徴マップを受け取る。ここで、C はチャネル数、H,W はそれぞれ特徴マップの縦横のグリッド数を表す。出力層では、特徴マップに基づき、各グリッド内での物体の位置、範囲、種類、評価値を算出して出力する。これらの結果を統合して、物体検出結果とする。

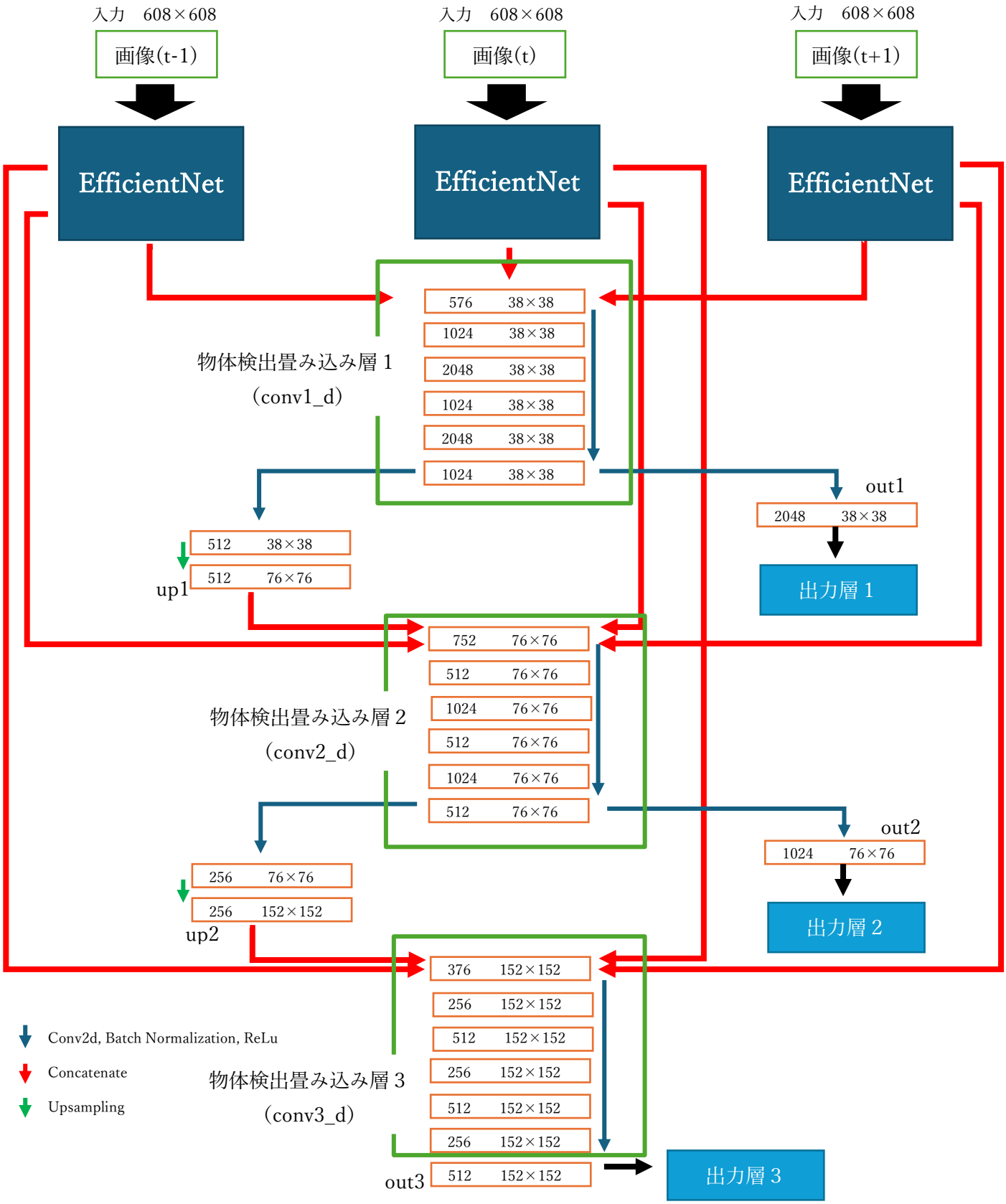

図 5　物体検出のネットワーク構造

表 1　各特徴抽出器からの出力

	層	サイズ	チャネル
ResNet152	Conv3_x	76×76	512
	Conv4_x	38×38	1024
	Conv5_x	19×19	2048
ResNet18	Conv3_x	76×76	128
	Conv4_x	38×38	256
	Conv5_x	19×19	512
EfficientNet-B0	4 層目	152×152	40
	5 層目	76×76	80
	7 層目	38×38	192
EfficientNet-B7	4 層目	38×38	80
	5 層目	76×76	160
	7 層目	152×152	384
SwinTransformerB	Stage2	28×28	256
	Stage3	14×14	512
	Stage4	7×7	1024

3.3　提案手法の学習と結果の出力

提案手法の学習では、教師データとして、まず、動画像の各フレームについて、フレーム内の物体の種類を表す識別子とその物体のフレーム内での位置と範囲とを人手で与える。あるフレームとその前後フレームの 3 フレームを束ねたものを入力、そのフレームの教師データを出力とした組を学習データとして提案手法に与えて学習することで、物体検出モデルを作成する。

物体検出結果を得る際は、検出したい動画像を入力として与える。与えられた動画像から連続したフレームを 3 枚束ねた画像群を動画像のフレームの数だけ作成する。与えられた画像群を学習済みの物体検出モデルに入力として与えることで、物体検出を行う。結果の出力として、矩形の位置情報が与えられる。この位置情報から信頼度スコアを求める。信頼度スコアは、「分割された領域(バウンディングボックス)に物体が入っていて、正確に領域を囲えているかの正確さ」と「各クラスの予測確率」を意味する指標である。信頼度スコアが閾値よりも高いバウンディングボックスを物体検出結果として出力する。

4. 実験

4.1 実験設定

本実験では、各特徴量抽出器の精度を田邊[4]が実験で用いていた ResNet152 も含めて比較し、精度向上しているかを調査する。

検出する対象物体は馬の 1 クラスとし、実験データとして田邊の研究でも用いられていた 35 本の馬の動画像[27]を用いる。実験データに用いる馬の画像例を図 6 に示す。動画像によって動画像内の馬の数や画像サイズが異なる。図 6 の左上の画像から右の画像に向かって対応する動画像を動画 1、動画 2・・・動画 35 として、動画像ごとの実験データの情報を表 2 に示す。動画像をフレームに分解し 10 枚の連続フレームの画像群を作成する。この画像群を本研究ではクリップと呼ぶ。動画像により、フレーム数の違いや動画像内での馬の動きの変化が異なるため、動画像に応じて 1 つの動画像から 3〜8 個のクリップを作成する。また、同じ動画像であってもクリップによっては、存在する馬の数が異なる。例えば、動画 8 の場合、あるクリップでは 1 頭の馬がクリップ内に存在するが、別のクリップでは 2 頭の馬がクリップ内に存在する。合計で 1400 枚(140 クリップ)の画像を実験データとして用いる。また、実験データから馬が存在する正解の位置と範囲(BBox)を示すデータも同様に、田邊の研究のものを用いた。

本研究では、クロスバリデーションと呼ばれる手法を用いて評価を行う(図 7)。クロスバリデーションは実験データを K 個に分割して、そのうち一つを検証データ、残りの K-1 個を学習データに使用することで、K 個のパターンで精度評価を行う手法である。実験データが少ない場合、学習や検証に使うデータによって結果が大きく異なってしまう恐れがある。クロスバリデーションでは実験データを分割することで、学習結果の偏りがなく精度評価を行える。本研究では 35 個の動画像を 5 つに分割することで、5 パターンの実験から精度評価を行う。

特徴抽出に用いる EfficientNet、ResNet、SwinTransformer は学習済みのモデルを使用する。学習の際は入力画像の大きさを 608×608 に設定し、検証の際は入力画像の大きさを 416×416 に設定する。ただし、SwinTransformer には 224×224 の画像しか入力できないため、学習と検証の際はどちらも画像の大きさを 224×224 に設定する。学習時に高解像度(608×608)の画像を用いる理由は、より多くの詳細な特徴を学習し、物体の細部を捉えるためである。高解像度の画像を入力とすることで、小さな物体や微細な特徴を含めた情報を効率的に学習できる。また、CNN ベースの特徴抽出器（EfficientNet、ResNet）は、入力サイズが大きいほど高次の特徴を学習しやすくなるため、モデルの表現能力を最大限活かすことができる。さらに、データ拡張(スケーリングや回転)を適用する際、高解像度の画像を使用することで、拡張後も十分な解像度を維持し、効果的な学習を行うことが可能になる。

一方、検証時には入力画像のサイズを 416×416 に設定する。これは、推論時の計算コストを削減し、リアルタイム処理を考慮するためである。高解像度の画像を使用すると、メモリ消費量が増え、推論速度が低下するため、検証時には画像サイズを小さくして、より効率的に処理を行う。また、学習時に高解像度で特徴を十分に学習しているため、推論時に縮小しても物体検出精度を維持できることが期待される。

学習回数は 10000 回とする。このように設定する理由は、十分な学習を確保しつつ、過学習を防ぐためである。学習回数が少なすぎると適切な特徴抽出が行えず、逆に多すぎると過学習のリスクが高まる。10000 回という設定は、学習データに対して適切なバランスを保ちつつ、モデルの性能を最大化するための選択である。また、学習回数を統一することで、公平な比較を行い、モデル間の性能差を正しく評価することが可能となる。

図 6　実験データに用いる馬の画像例[4]

表 2　実験データの情報[4]

	大きさ	フレームレート（フレーム／秒）	フレーム数	クリップ数	馬の数
動画1	1920×1080	60.00	687	6	1
動画2	3840×2160	30.00	382	4	1
動画3	1920×1080	30.00	408	5	1
動画4	1920×1080	25.00	151	3	2
動画5	1920×1080	50.00	328	4	1
動画6	3840×2160	60.00	1828	4	3
動画7	3840×2160	25.00	255	3	3
動画8	1920×1080	29.97	713	8	1~2
動画9	3840×2160	29.97	976	4	1
動画10	1920×1080	30.00	356	3	1
動画11	1920×1080	24.00	369	4	1
動画12	3240×2160	25.00	538	4	4
動画13	1920×1080	30.00	678	3	1
動画14	3840×2160	30.00	258	3	1
動画15	3840×2160	25.00	180	3	1
動画16	1920×1080	30.00	401	4	1
動画17	1920×1080	30.00	588	6	1
動画18	3840×2160	30.00	235	3	1
動画19	3840×2160	23.98	314	3	3
動画20	1920×1080	30.00	461	4	2
動画21	1920×1080	25.00	535	4	2
動画22	3840×2160	23.98	312	4	1
動画23	1920×1080	23.98	415	4	2
動画24	4096×2160	29.97	277	3	2~4
動画25	4096×2160	25.00	347	3	1
動画26	3840×2160	29.97	198	3	2
動画27	4096×2160	29.97	1057	6	3~6
動画28	1920×1080	30.00	262	3	4~6
動画29	4096×2160	29.97	905	5	1
動画30	3840×2160	25.00	406	3	2
動画31	3840×2160	50.00	1099	7	2
動画32	1920×1080	59.94	567	4	1
動画33	3840×2160	29.97	1110	4	2
動画34	3840×2160	25.00	202	3	3
動画35	1920×1080	25.00	368	3	3~4

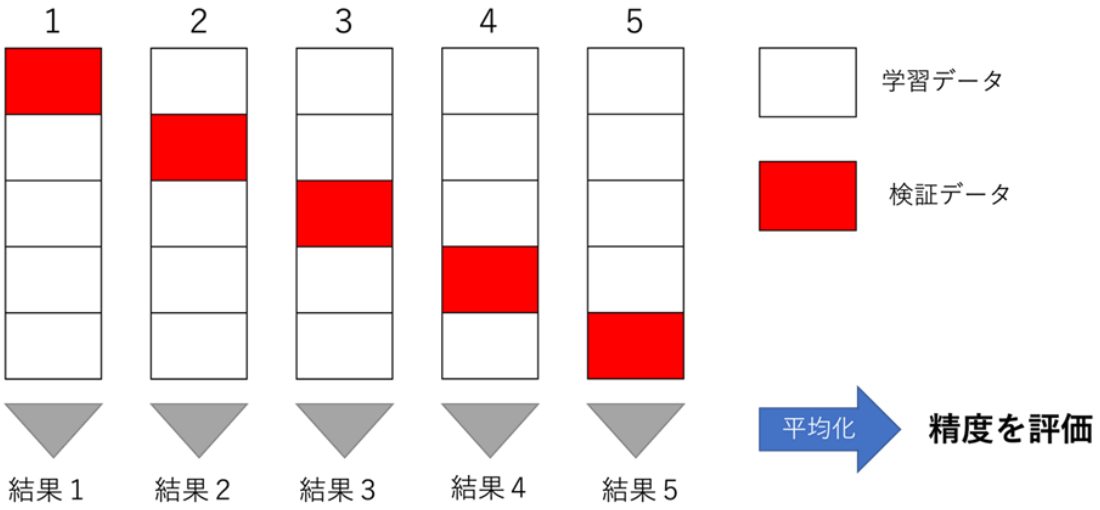

図 7　クロスバリデーションの例[4]

4.2　実験 1

実験 1 では、各特徴抽出器の性能を平均適合率（AP）で比較する。AP は m 個の正解ラベルのうち、どのくらいのラベルを検出できているかを平均的に表したものである。AP の計算には、Intersection over Union(IoU)（図 8）を用いる。IoU は正解データの領域（正解領域）と検出結果の領域（検出領域）の一致度を示すものである。IoU が閾値以上である場合、検出した矩形(BBox)を正解とする。正解した BBox を True Positive(TP)、正解でない BBox を False Positive(FP)、どの検出した BBox とも紐付いていない正解の矩形を False Negative(FN)とする。この値から Precision と Recall を計算する。Precision と Recall の式は以下のようになる。

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

この Precision と Recall から AP を求める。物体検出では、BBox の信頼度スコアがある閾値以上のものを検出結果とする。この閾値を変えると、Precision、Recall の値が変化する。Recall が r の時、Presicion の値を P(r)とする。AP は、Recall のとり得る範囲[0,1]での P(r)の平均として定義され、式は以下のようになる。

$$AP = \int_0^1 P(r)dr$$

AP の最大値は 1 となり、値が大きいほど検出精度が高いことを示す。今回の実験では、馬のクラスのみであるため、馬のラベルがどのくらい検出できているかを比較する。

本実験ではクロスバリデーションの手法により５つのパターンに分解して検証を行う。５つのパターンごとに 10000 回の学習を行う。それぞれの特徴量抽出器での実験結果を表 3 に示す。

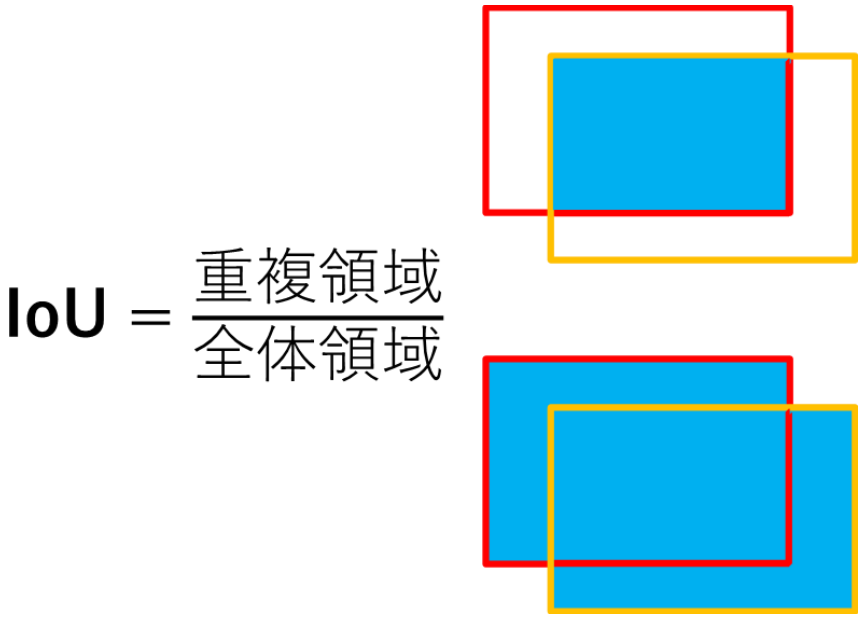

図 8　IoU の概念（赤枠：正解領域、橙枠：検出領域）[4]

実験 1 の実験結果の考察

表 3 の結果から、まず ResNet152 が最も高い性能を示していることがわかった。従来手法として使用されている ResNet152 の平均スコアは 0.6102 であり、他のモデルと比較して最も優れた物体検出性能を発揮している。しかし、再実験の ResNet152 のスコアは 0.4937 と低下しており、最初の実験との差が約 0.12 ある。この低下の原因として、学習条件の違いやデータセットが少ないため、データのランダム性が影響した可能性が考えられる。

次に、ResNet18 のスコアが低く、ネットワークの深さが物体検出の精度に影響を与えていることが確認できる。ResNet18 の平均スコアは 0.3474 であり、ResNet152（0.6102）と比べて大幅に低下している。ResNet の特徴であるスキップ接続により、深いネットワークでも学習が可能になっているが、層の深さが 18 層と 152 層では大きく異なり、その違いが物体検出の精度に影響を与えていると考えられる。

EfficientNet については、B7 の方が B0 よりも高いスコアを示しており、深いネットワークの方が物体検出に適していることがわかる。EfficientNet-B7 の平均スコアは 0.5138 であり、ResNet152 に次いで高い性能を示している。一方、EfficientNet-B0 のスコアは 0.3247 と低く、ネットワークの深さやパラメータ数の違いが精度に影響を与えていると考えられる。EfficientNet はコンパウンドスケーリングを採用しており、B7 の方がより高精度な特徴抽出が可能なため、この結果は妥当であるといえる。

SwinTransformer の結果については、期待された性能が発揮されていない可能性がある。結果を見ると、ResNet18（0.3474）や EfficientNet-B0（0.3247）と同程度、もしくは若干上回るレベルであり、ResNet152 や EfficientNet-B7 と比べると低い数値となっている。SwinTransformer は、自己注意（Self-Attention）を用いたモデルであり、高い特徴抽出能力を持つが、本実験では SwinTransformer のみ入力サイズが 224×224 であったため、期待した性能を発揮できなかった可能性が高い。一般に、Transformer 系のモデルは入力解像度がモデルの性能に影響を与えるため、他のモデルと異なる入力サイズが結果に影響を及ぼしたと考えられる。

パターンごとにスコアのばらつきが大きい点も重要なポイントである。特に、各モデルのスコアが一貫せず、パターンによって大きく変動していることが確認できる。例えば、ResNet152 はパターン 1 で 0.7050 と高いスコアを記録しているが、パターン 5 では 0.4325 まで低下しており、大きな差が生じている。同様に、EfficientNet-B7 もパターン 1 では 0.4860 であるのに対し、パターン 5 では 0.5599 と変動が見られる。

また、全体的にパターン 1 と 2 では比較的スコアが高い傾向がある一方で、パターン 3 以降ではスコアが低下するケースが多く、特にパターン 5 では全モデルでスコアのばらつきが顕著になっている。このことから、データセットの分割によって学習や評価の安定性に影響が生じており、一部のパターンでは十分な学習が行われなかった可能性があると考えられる。

これらの結果を踏まえると、まず ResNet152 は依然として強力な特徴抽出器であり、高い物体検出性能を発揮することが確認された。

今回の実験結果からは、データセットの量が限られていたことが、スコアのばらつきに影響を与えたと考えられる。パターンごとのスコアの違いもデータセットの少なさが一因となり、特定のパターンでは学習が不十分だったことが示唆される。実験 2 では、より多くのデータを用いた場合の性能変化や、ネットワークの規模を調整することでどのように結果が変化するかを分析する。

表 3　実験 1 の結果(AP)

	パターン 1	パターン 2	パターン 3	パターン 4	パターン 5	平均
Resnet152	**0.7050**	**0.7240**	**0.6362**	**0.5533**	0.4325	**0.6102**
(再実験)ResNet152	0.6089	0.5955	0.6012	0.3821	0.2809	0.4937
ResNet18	0.4147	0.3816	0.4194	0.3365	0.1847	0.3474
EfficientNet-B0	0.2788	0.4643	0.3777	0.2273	0.2752	0.3247
EfficientNet-B7	0.4860	0.6069	0.5179	0.3982	**0.5599**	0.5138
SwinTransformerB	0.3819	0.3367	0.5007	0.3156	0.3649	0.3800

4.3　実験 2

実験 2 では、追加実験として、物体検出器(ヘッド)の部分に注目し、3 つの実験を行った。実験には EfficientNet-B0 を用いる。

1 つ目は物体検出器の再構築である。物体検出の実験において、使用したネットワークの層の多さに対して、学習に用いたデータセットの量が少なかったことが、期待した性能を発揮できなかった要因の一つであると考えられる。一般的に、深いネットワークは大量のデータを必要とし、データが不足している場合、過学習を引き起こしやすくなる。そのため、余分な層を減らすことで、よりシンプルで効率的なモデルを構築し、適切な物体検出が可能になるのではないかと考えた。そこで、本実験では ResBlock の一部を削減することで、パラメータ数を抑えつつ、学習の安定性を保つことを試みた。層を減らすことでパラメータ数が削減され、学習が安定し、小規模なデータセットでも十分な精度を発揮できる可能性がある。このようにモデルの構造を適切に調整することで、より良い結果を得られると期待できる。実際に層を減らしたネットワークを図 9 に示す。各物体検出畳み込み層には 5 つの層があったが、2 つの層を削除し、3 層に減らした。削除した基準としては、繰り返し同じような処理を行っている部分を削除した。

2 つ目はデータセットの増加である。データセットを増やす理由として、特にデータセットの量が極端に少ないことが挙げられる。データが不足していると、モデルが十分な学習を行えず、汎化性能が低下し、新しいデータに対して正しく認識できない可能性が高まる。また、データの多様性が不足すると、モデルが特定のパターンに過度に適応し、過学習を引き起こすリスクもある。そのため、データセットを増やすことで、モデルの学習をより安定させ、精度向上や過学習の防止につながると考えられる。具体的にはもともとの 35 個の実験データに加えて学習用に 16 個の学習データを LaSOT Dataset[28]から用意し、それぞれの動画像を元のデータセットと同じように 1 フレームずつ分解し複数のクリップを作成した。追加した学習データの情報を表 4 に示す。

3 つ目には追加として、物体検出器の再構築とデータセットの増加を同時に行って評価する実験を行った。

評価としては実験 1 と同様にクロスバリデーションの手法を用いて平均適合率(AP)を用いて評価する。特徴抽出に用いる EfficientNet-B0 は学習済みのモデルを使用する。学習回数はパターンごとに 10000 回行う。学習の際は入力画像の大きさを 608×608 に設定し、検証の際は入力画像の大きさを 416×416 に設定する。

元の EfficientNet-B0 と比較した物体検出結果を表 5 に示す。

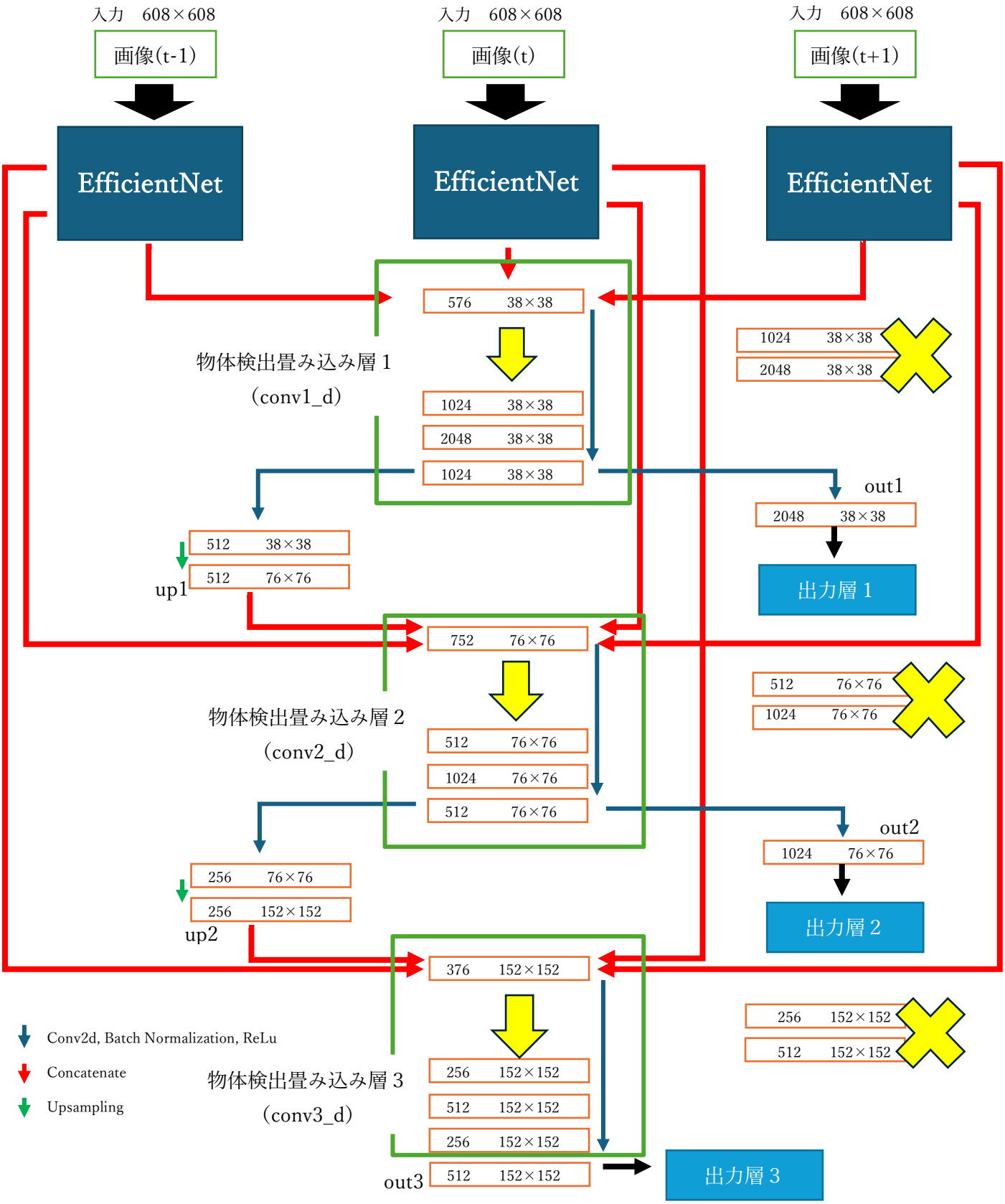

図 9　再構築後のネットワーク

表 4　追加学習データの情報

	大きさ	フレーム数	クリップ数	馬の数
動画 1	1280×720	1313	12	1
動画 2	1280×720	3813	38	1
動画 3	1280×720	1451	14	1
動画 4	540×360	2435	24	1
動画 5	320×240	2073	20	1
動画 6	1280×720	4596	45	1
動画 7	270×360	3666	36	1
動画 8	1280×720	2811	28	1
動画 9	1280×720	2385	23	1
動画 10	1280×720	2194	21	1
動画 11	480×360	2254	22	1
動画 12	1280×720	2377	23	1
動画 13	1280×720	3137	31	1
動画 14	480×320	3974	39	1
動画 15	1280×720	2972	29	1
動画 16	1280×720	5797	57	1

実験 2 の実験結果の考察

表 5 から、パターン 1 やパターン 3、パターン 5 では元の EfficientNet-B0 より良い結果は出ているものの平均では劣る結果となっている。

物体検出器を再構築した場合、平均スコアは 0.2833 となり、基本モデルより若干低下している。この結果から、単にネットワークの構造を変更するだけでは、必ずしも性能向上につながらないことが分かる。特にパターン 4 では大幅にスコアが低下しており、再構築したネットワークがデータの一部に対して適応できていない可能性がある。

データセットを増加させた場合、平均スコアは 0.2979 となり、EfficientNet-B0 とはほぼ同様の結果となった。パターン 1 やパターン 3 ではスコアが向上したが、パターン 2 やパターン 4 で低下していることから、データの増加が必ずしもすべてのケースで有効に働くとは限らないことが分かる。追加したデータの質が十分でなかった可能性も考えられる。

物体検出器の再構築とデータの増加を組み合わせた場合、平均スコアは 0.2380 となり、最も低い結果となった。特に、パターン 1 やパターン 4 ではスコアが大幅に低下している。クロスバリデーションを用いているため、データの分割の影響が結果に反映されやすく、特定のデータ分割において性能が低下した可能性も考えられる。

表 5　実験 2 の結果(AP)

	パターン 1	パターン 2	パターン 3	パターン 4	パターン 5	平均
EfficientNet-B0	0.2788	**0.4643**	0.3777	**0.2273**	0.2572	**0.3247**
ネットワークの再構築	0.2880	0.3866	0.2946	0.1427	**0.3045**	0.2833
データセットの増加	**0.3921**	0.2229	**0.4598**	0.1699	0.2450	0.2979
ネットワークの再構築＋データセットの増加	0.1629	0.3002	0.3006	0.1801	0.2464	0.2380

5. おわりに

本研究では、動画像からの物体検出において時間方向での物体検出結果を安定させるために提案された、複数フレームからの特徴量を元に物体検出を行う手法の更なる精度向上のために特徴抽出器(バックボーン)部分を様々なモデルに変更して評価を行った。

従来研究では 3 フレームからの特徴抽出器として ResNet152 を用いていたが、その部分を EfficientNet-B0、B7、ResNet18、SwinTransformerB に変更した。平均適合率(AP)から、田邊[4]の手法でも採用されている ResNet152 の結果が一番良いことが分かった。また、それぞれの特徴抽出器の結果からデータセットの少なさが影響しているのではないかと考えた。

そこでデータセットの少なさがどのくらい影響するかを追加で調査した。3 つの工夫を考え、調査した。まず一つ目は、物体検出器を再構築することを行った。二つ目には、データセットを増やすことを行った。三つ目には、一つ目と二つ目を合わせたもので行った。結果としては、元の EfficientNet-B0 より良い結果を残すところもあったが、平均では元のモデル(Efficientnet-B0)より良い結果を残すことはできなかった。

本実験の実験結果からは、ネットワークの深さが物体検出性能に大きく影響すること、EfficientNet-B7 が有望なモデルであること、SwinTransformer についてはさらなる調整が求められること、パターンごとのデータ特性が結果に影響を与えていることが示唆された。特に、SwinTransformer の調整には、事前学習の際に大きなサイズの画像で学習させることが重要であると考えられる。今後の研究では、これらの点を踏まえて、最適なモデル選択や学習手法の改善を行う必要がある。

謝辞

本研究を進めるにあたり、ご指導いただいた指導教員である椋木教授に深く感謝いたします。研究を進める中で発生した問題に対して、事細かくアドバイスいただき、研究活動をよりスムーズに行うことが出来ました。論文着手以降も、論文の添削や論文構成のアドバイスなどをしていただき深く感謝いたします。また、研究室内での相談や助言をくださった椋木研究室の皆様にも深く感謝いたします。

参考文献

[1] Joseph Redmon , Santosh Divvala, Ross Girshick , Ali Farhadi “You Only Look Once: Unified, Real-Time Object Detection”, CVPR (2016)
[2] DarkNet：https://pjreddie.com/darknet/
[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft, “Simple Online and Realtime Tracking”, ICIP (2016)
[4]田邊英介, ”深層学習による動画像の連続フレームからの物体検出”, 宮崎大学工学院工学研究科令和 4 年度修士論文 (2022)
[5] Kaiming He , Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, CVPR (2016)
[6] Tan, Mingxing, and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks.” ICML (2019)
[7] Ze Liu(Microsoft Research Asia) et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, ICCV (2021)
[8] Viola, P., Jones, M.J.,“Robust Real-Time Face Detection.”, International Journal of Computer Vision 57, 137–154 (2004)
[9] Karen Simonyan, Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, ICLR (2015)
[10] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, Serge Belongie, “Feature Pyramid Networks for Object Detection”, CVPR (2017)
[11] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, Jiaya Jia, “Path Aggregation Network for Instance Segmentation”, CVPR (2018)
[12] Joseph Redmon, Ali Farhadi, “YOLOv3: An Incremental Improvement”, arXiv18.04.02767[cs.CV] (2018)
[13] Olaf Ronneberger, Philipp Fischer, Thomas Brox,“U-Net: Convolutional Networks for Biomedical Image Segmentation”, MICCAI (2015)
[14] Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.39, pp.2481-2495 (2015)
[15] YOLO v3：https://www.nature.com/articles/s41598-021-81216-5
[16] Bing Shuai, Andrew Berneshawi, Xinyu Li, Davide Modolo, Joseph Tighe, “SiamMOT: Siamese multi-object tracking”, CVPR (2021)
[17] Nicolai Wojke, Alex Bewley, Dietrich Paulus, “Simple Online and Realtime Tracking with a Deep Association Metric”, ICIP (2017)
[18] Guanghn Ning, Zhi Zhang, Chen Huang, Zhihai He, Xiaobo Ren, Haohong Wang,

“Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking”, ISCAS (IEEE International Symposium on Circuits and Systems) (2017)
[19] 中山隼人, “動画像からの物体検出のための U-Net3D の改良”, 宮崎大学工学部情報システム工学科令和 3 年度卒業論文 (2021)
[20] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Li Fei-Fei, “ImageNet: A large-scale hierarchical image database”, CVPR (2009)
[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, Soumith Chintala, “PyTorch: An Imperative Style, High-Performance Deep Learning Library”, NeurIPS (2019)
[22] Victor Sanh, Lysandre Debut, Julien Chaumond, Thomas Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, NeurIPS (2019)
[23] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen, “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, CVPR (2018)
[24] François Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions”, CVPR (2017)
[25] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, “Going deeper with convolutions”, CVPR (2015)
[26] Geoffrey E. Hinton, Simon Osindero, Yee-Whye The, “A Fast Learning Algorithm for Deep Belief Nets”, Neural Computation, vol.18, pp.1527-1554 (2006)
[27] 動画ダウンロードサイト：https://www.pexels.com/ja-jp/search/videos/
[28] 動画ダウンロードサイト：http://vision.cs.stonybrook.edu/~lasot/