令和４年度修士論文

深層学習による動画像の連続フレームからの物体検出

宮崎大学大学院　工学研究科　工学専攻

機械・情報系コース　情報システム工学分野

学籍番号　T2103288

田邉　英介

指導教員　椋木雅之

概要

本研究では、深層学習を用いた動画像からの物体検出として、時間方向に連続する数フレームの情報を用いた物体検出手法を提案する。

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。物体検出は動画像においても適用されている。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といった問題がある。

そこで、本研究では、動画像からの物体検出において、時間方向での物体検出結果を安定させることを目指す。そのために、従来のように動画像の各フレームに対して１枚ずつ独立に物体検出処理を適用するのではなく、時間方向に連続する数フレームから抽出した特徴量を元に、物体検出を行う手法を提案する。これにより、前後のフレームの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ようになると考えられる。

実験では、提案手法が従来手法と比べて、動画像での物体検出がどの程度安定するかを評価した。実験の結果から、提案手法は物体検出の精度、時間方向での安定性の２つの点で従来手法よりも優れていることを示した。

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。従来は、人手により設計した特徴量に基づき検出対象をモデル化し、検出処理を行なっていた。例えば、Viola ら[1]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高精度に人の顔を検出する手法を提案した。この手法では、弱識別器の構築や選別は、人の顔を撮影した学習データを大量に与えることで自動的に行えるが、使用する特徴量は予め人間が考案したものであった。

これに対し、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[2]は DarkNet[3]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[4]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。

そこで本研究では、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ことを目指す。そのために、従来のように動画像の各フレームに対して 1 枚ずつ独立に物体検出処理を適用するのではなく、時間方向に連続する数フレームから抽出した特徴量を元に、物体検出を行う手法を提案する。これにより、前後のフレームの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ようになると考えられる。具体的には、まず、検出したいフレームの画像とその前後のフレームの画像に対して特徴量抽出を行う。次に、抽出した 3 つの連続するフレームの特徴量を入力として物体検出を行うことで、連続検出の性能向上を目指す。

以下、2 章では、物体検出の従来手法について述べる。3 章では、本研究の提案手法である深層学習を用いた連続フレームからの物体検出手法について述べる。4 章では、提案した連続した 3 フレームを入力とする物体検出手法と 1 枚ずつ独立して処理する物体検出手法

（YOLO）との比較を行い、提案した手法の有効性を評価する。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[1]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、畳み込み層と呼ばれる要素を多段階に並べた畳み込みネットワーク（CNN）と呼ばれるネットワーク構造が主に使われる。物体検出の CNN は特徴量抽出ネットワークと物体検出ネットワークからなる（図 1)。特徴量抽出ネットワークでは畳み込みと呼ばれる画像処理を多段階適用することで、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。特徴量抽出ネットワークは、特定の物体検出のタスク毎に学習することもできるが、多種大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量抽出ネットワークは特徴量抽出器と呼ばれる。現状では、VGG[5]や ResNet[6]など高性能な特徴量抽出器が容易に利用できるようになっている。特徴量抽出ネットワークで抽出した特徴量を入力として、物体検出ネットワークで画像中の物体の位置や範囲、種類を識別して出力する。Faster R-CNN[7]は、深層学習による入力から出力までを直接、単一のモデル（End-to-End）で学習することができる初めての物体検出モデルである。End-to-End の学習により、領域候補の探索とクラスの識別を１つのネットワークで処理することができる。また、SSD[8]は、デフォルトボックスと呼ばれる矩形パターンを配置することでバウンディングボックスを推定する。デフォルトボックスにより、小さい物体の検出にも対応することができる。

深層学習を使った物体検出手法の１つとして YOLO[2]がある。YOLO では、信頼度スコアというものを使用することで、どの領域に対象とするクラスの物体が正確に検出され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。また、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[9]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 2)。そのため、特徴マップの大きさにあった、特徴量抽出ネットワークの特徴量と物

体検出ネットワーク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。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量抽出ネットワークの特徴量と物体検出ネットワークの特徴量を結合し、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[10]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[11]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。

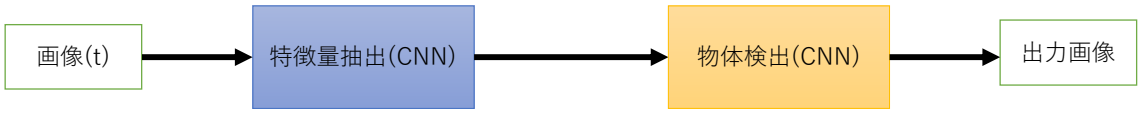

図 1　深層学習による単一画像からの物体検出の考え方

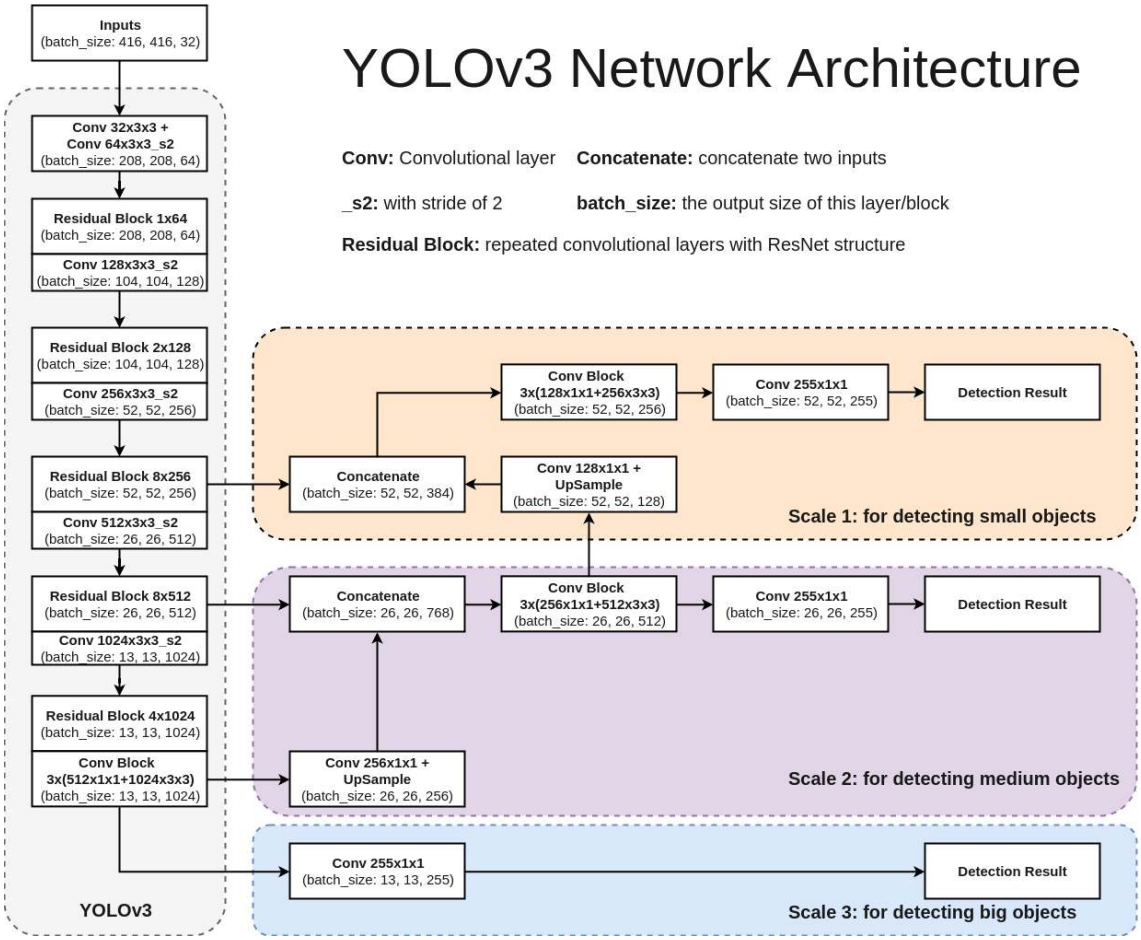

図 2　YOLO v3 のネットワーク構造［12］

２.２　動画像からの物体検出

動画像においても物体検出が適用されている。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[4]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[13]らの手法は、上記の SORT を改良したものである。YOLO で検出したバウンディングボックスを使用して、フレームの前後で近い大きさと近い動きのバウンディングボックスを対応づけることで、検出対象に ID をつけて追跡を行う。DeepSORT[14]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。

動画像からの物体検出では、前節で述べた単一画像からの物体検出を動画像のフレーム毎に適用する方法が一般的である。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[15]は、LSTM と呼ばれる深層学習のネットワークを使った動画像からの物体検出手法である。LSTM は Long Short Term Memory の略で、時系列の情報を活用することができる。ROLO では YOLO を用いて物体の位置と範囲（バウンディングボックス；BBox）を抽出する。この BBox を LSTM を使ってフィードバックすることで、各フレームでの物体検出の際に、前フレームで検出された物体の位置や範囲の情報も利用する。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、前フレームの画像情報は利用しておらず、時間方向での連続した物体検出の安定化には有効ではない。

U-Net3DT[16]は、U-Net[10]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 3 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、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量抽出器として利用できないため、1 から全て学習する必要がある。そのため、性能があまり良くないといった問題がある。

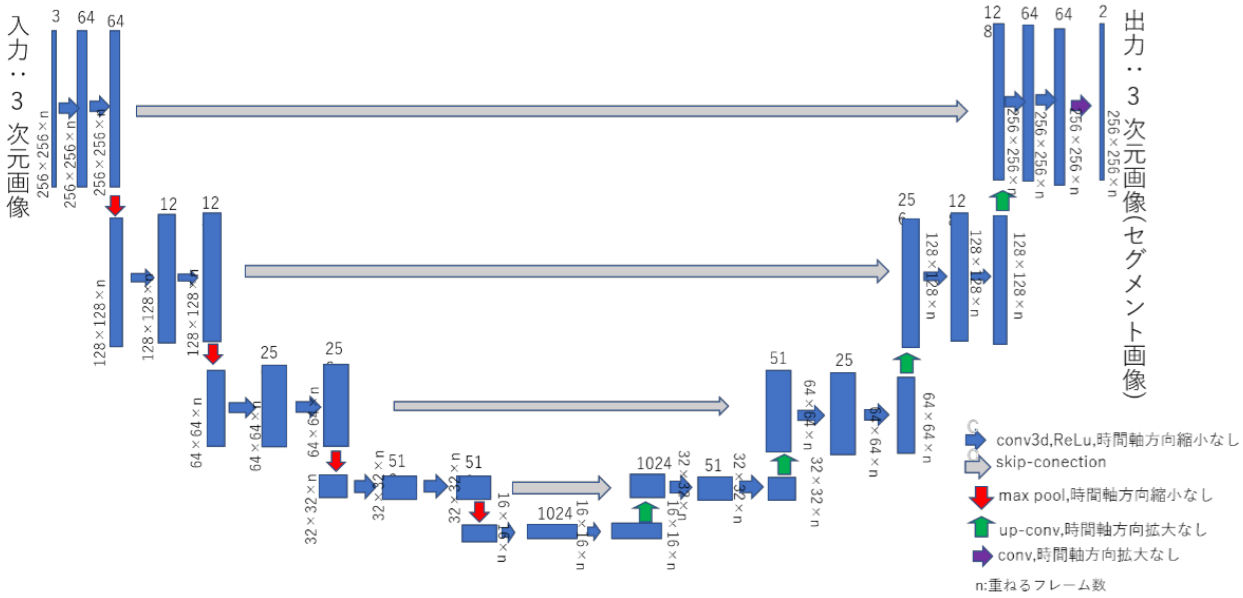

図 3　U-Net3DT のネットワーク構造［16］

２.３　従来手法の問題点

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。

３．動画像の連続フレームを用いた物体検出

３.１　連続フレームからの物体検出

本研究では、動画像からの物体検出において、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を提案する(図 4)。入力された各フレームから特徴量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ネットワークに与えて、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、前後のフレームの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ようになると考えられる。例えば、図 5 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。従来手法では画像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、提案手法では時刻 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。

提案手法のネットワーク（図 4）は、3 フレームの入力に対し、３つの特徴量抽出器を並べて、それぞれ適用する構造となる。特徴量抽出器には、既存の学習済み CNN を利用する。得られた 3 フレーム分の特徴量は結合して物体検出ネットワークに与える。物体検出ネットワークは、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。それぞれのネットワークの詳細について、次節で述べる。

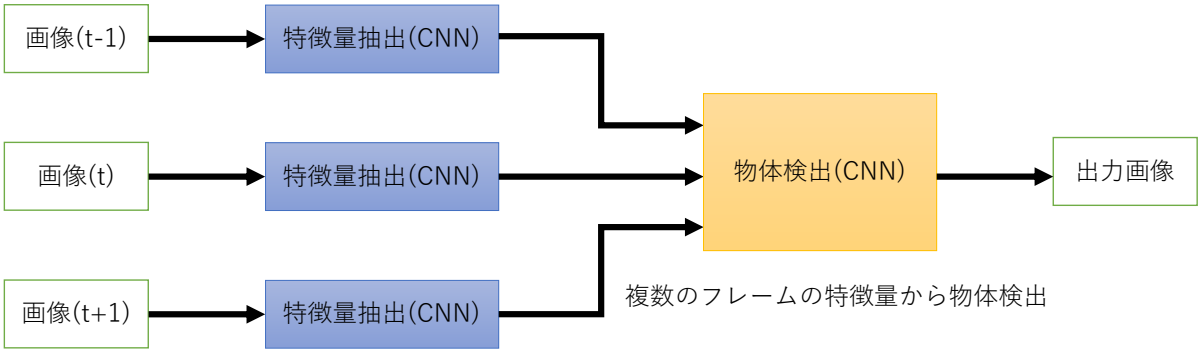

図 4　提案手法の物体検出の考え方

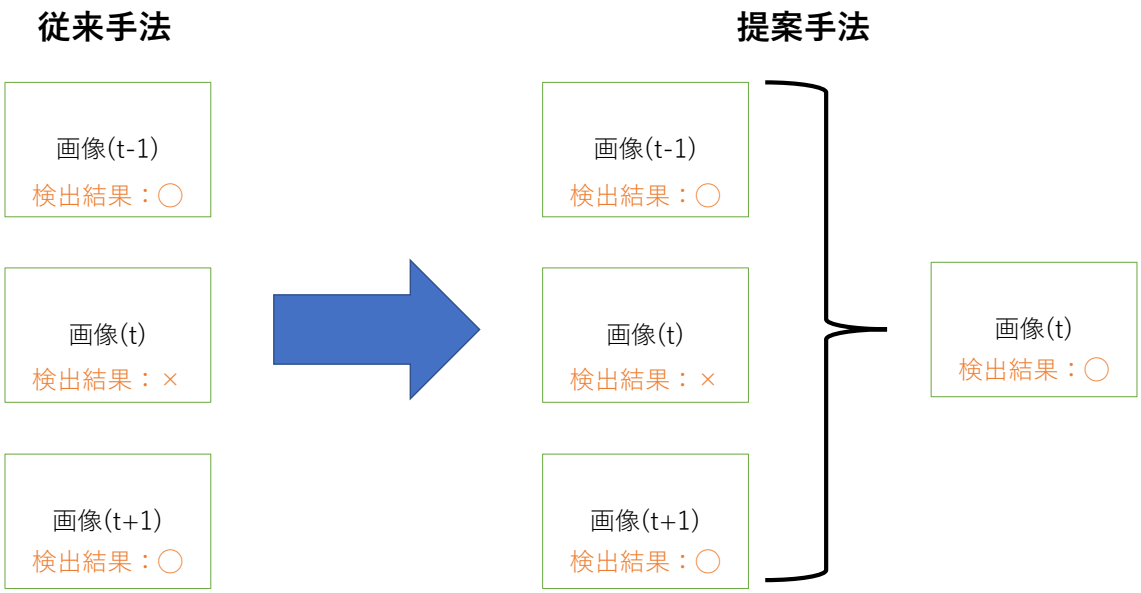

図 5　提案手法と従来手法の違い

3.2　提案手法のネットワーク構造

3.2.1　特徴量抽出ネットワーク

本研究では、特徴量抽出器として学習済みの ResNet[6]を使用する。ResNet は、深層学習を用いた特徴量抽出器であり、従来の深層学習を用いた手法よりも圧倒的な層の数を実現した手法である。深層学習において層を深くすることは、より高度で複雑な特徴を抽出するために重要である。しかし、層を深くすると勾配消失問題や劣化問題などにより、学習が進まない問題が生じる。ResNet では、Residual block と呼ばれる手法を使うことで、そのような問題を解決している。Residual block の考え方は、ある層で求める最適な出力を学習するのではなく、層の入力を参照した残差関数を学習するというものであり、これにより最適化しやすくしている（図 6)。例えば、$F(x) = x$と恒等写像を学習するのが最適である場合を考える。図 6 の左の図では、非線形関数F のパラメータを調整し、恒等写像を学習する必要があるが、この方法では層を増やすと劣化問題などの問題が起こることが考えられる。ResNet では、右の図のように、Shortcut Connection という迂回路を追加し、$F(x) + x$を出力するように変更している。これにより、恒等写像を学習するには$F(x) = 0$、つまりパラメータが 0 になるよう学習すればよいため、左の図に比べ学習が簡単になる。図 6 の右の図の 2 つの畳み込み層と 1 つの Relu 層、そして Shortcut Connection から成るブロックを Residual block と呼ぶ。この Residual block を複数重ねたネットワークが ResNet である。

ResNet は 18 層、34 層、50 層、101 層、152 層の 5 種類が提案されている。表 1 は、5 種類のネットワークに対して、入力画像が 224×224 のサイズの場合での畳み込み層ごとの特徴量のサイズやチャネル数などを示している。本研究では、152 層の ResNet を特徴量抽出器として使用する。YOLO v3 では精度と処理速度のバランスを考慮して、DarkNet-53 を特徴量抽出器として採用している。本研究では、特徴量抽出の性能を重視して ResNet-152 を採用した。表 2 に、YOLO v3[9]で行われた特徴量抽出器の性能評価結果を引用して示す。本論文で採用した ResNet-152 の方が、YOLO v3 で採用されている DarkNet-53 より処理速度は遅くなるが、高い精度での物体検出が行えることがわかる。CNN の畳み込み層の出力を特徴マップと呼ぶ。YOLO v3 の物体検出ネットワークでは、様々な大きさの物体に対応するために、 3 段階の大きさの異なる特徴マップを使用する。本研究では、表 1 の conv3_x、conv4_x、conv5_x の出力をそれぞれ 3 フレーム分結合したものを物体検出ネットワークの入力として使用する。

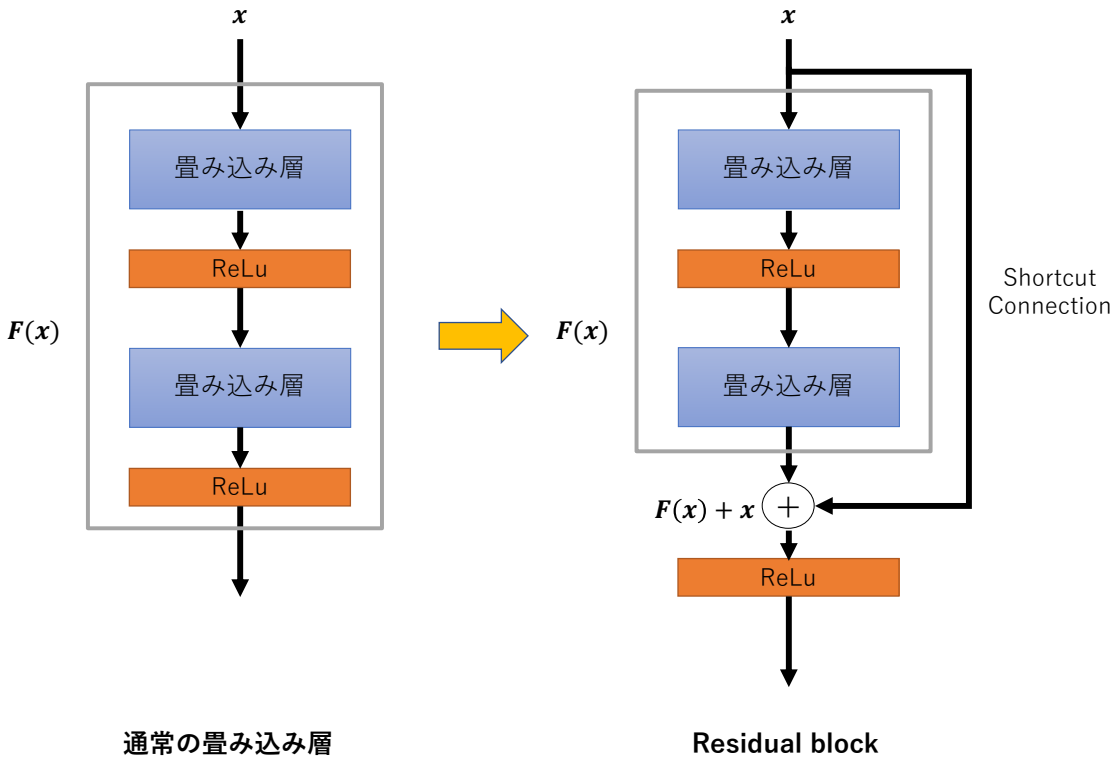

図 6　Residual block の例

表 1　ResNet のネットワーク構造 [6]

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

表 2　ResNet と DarkNet の精度比較 [9]

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [15]	74.1	91.8	7.29	1246	**171**
ResNet-101[5]	77.1	93.7	19.7	1039	53
ResNet-152 [5]	**77.6**	**93.8**	29.4	1090	37
Darknet-53	77.2	**93.8**	18.7	**1457**	78

３.２.２　物体検出ネットワーク

本研究では、YOLO v3 を参考にして、物体検出のネットワークを構築した。提案手法のネットワーク構造を図 7 に示す。物体検出ネットワークでは、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マップを利用する。提案手法では、3 フレーム分の入力に対して、同様に３段階の大きさの特徴マップを抽出し、3 フレーム分を結合して利用する。特徴マップの大きさは特徴量抽出器である ResNet に入力されるフレーム（画像）の大きさに依存する。フレームの大きさが 608×608 の場合、表 1 の各畳み込み層 conv5_x、conv4_x、conv3_x の特徴マップの大きさはそれぞれ 19×19、38×38、76×76 となる。

conv5_x からの特徴マップは 1 フレームあたり 2048 チャネルで、これを 3 フレーム分結合した 6144 チャネルの特徴量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 19×19、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 19×19、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（38×38）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、conv4_x からの特徴マップ（大きさ 38×38、1024 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 3584 チャネルの特徴量を使用する。同様に、物体検出畳み込み層 3 では、conv3_x からの特徴マップ（大きさ 76×76、512 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 1792 チャネルの特徴量を使用する。

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。

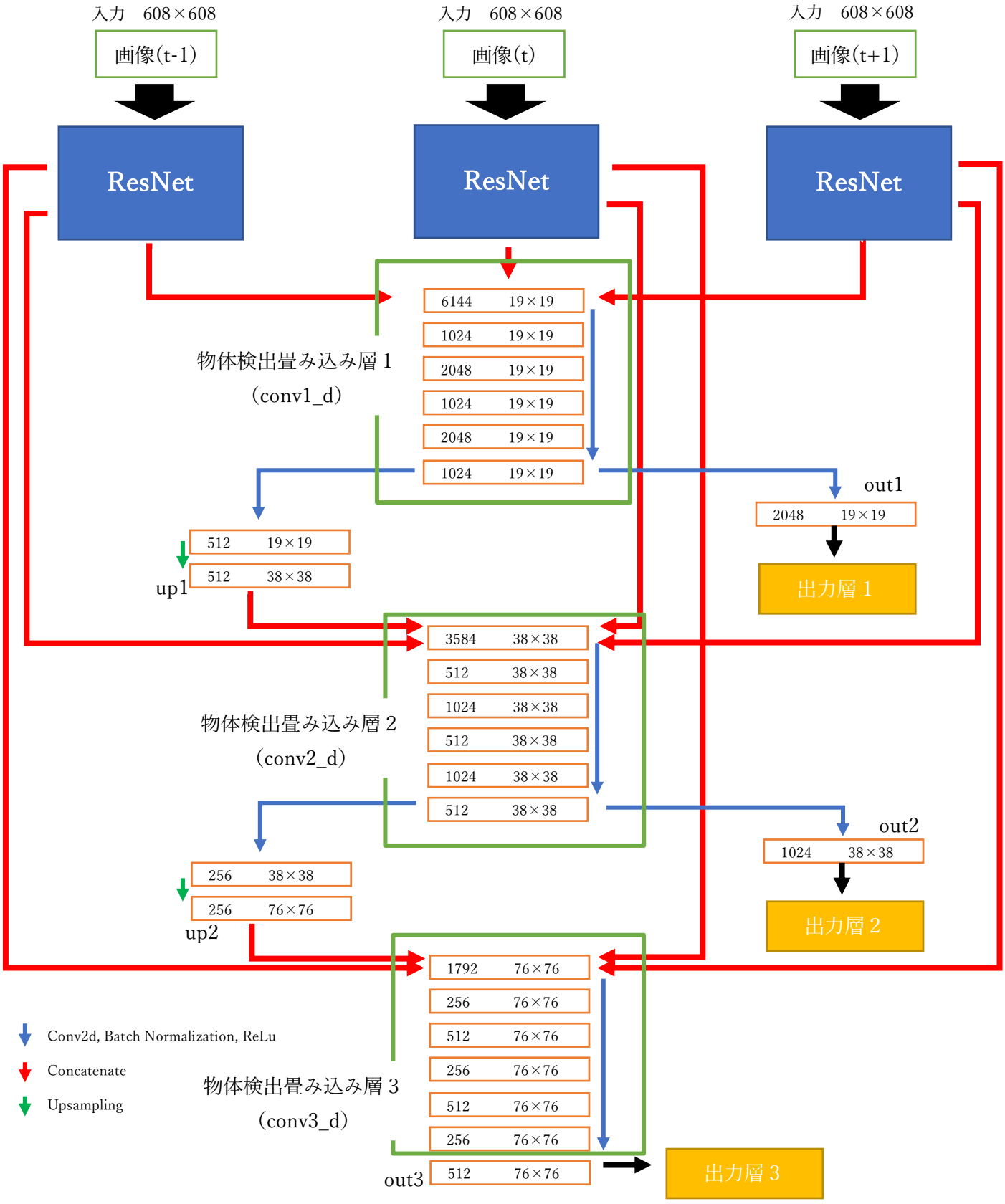

図 7　提案手法の物体検出ネットワーク構造

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、提案手法が従来手法（YOLO v3）と比べて動画像での物体検出がどの程度安定するかを調査するために３つの実験を行う。

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 35 本の馬の動画像[17]を用いる。実験データに用いる馬の画像例を図 8 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 8 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 3 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。クリップを作成する理由としては、実験２で連続検出の精度評価を行う際、クリップを使用するためである。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）を示すデータを LabelImg[18]というツールを使って人手により作成した。

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 9）。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。本研究の実験１、実験３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。

特徴量抽出に用いる ResNet は学習済みのモデルを使用する。学習は提案手法と従来手法（YOLO v3）それぞれで行い、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。

図 8　実験データに用いる馬の画像例

表 3　実験データの情報

	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

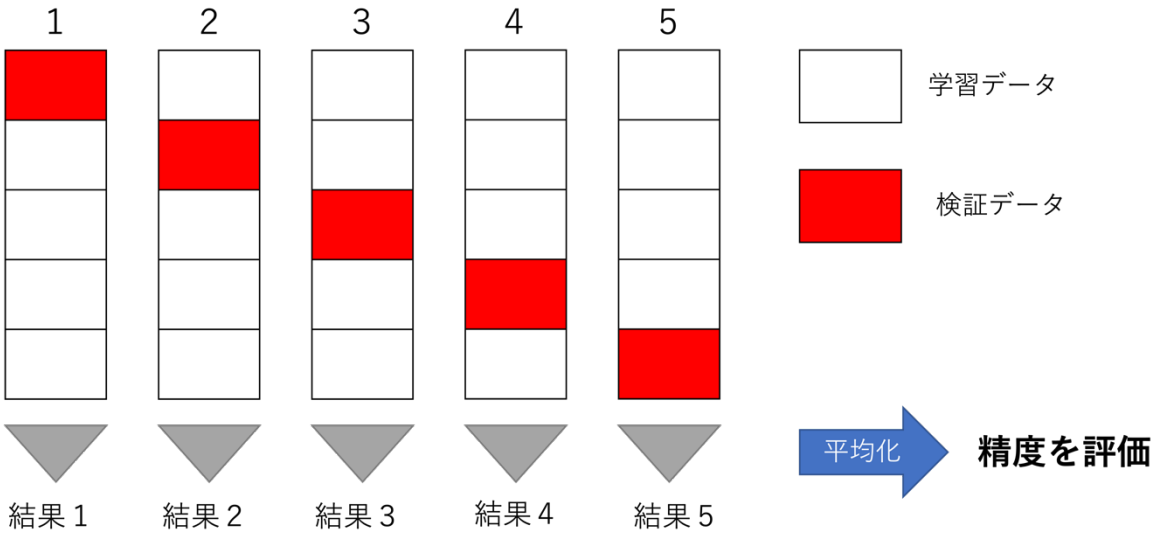

図 9　クロスバリデーションの例

４.２　実験１

実験１では提案手法と従来手法（YOLO v3）との平均適合率（AP）を用いた比較を行う。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 10）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（本実験では 0.5）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。

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

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。

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

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。図 11 に学習回数毎の loss 関数の変化を示す。1000 回まで急激に減少した後、ある程度の幅で変動しながら徐々に減少している。学習の途中階段での挙動も比較するために、学習回数 3000 回、5000 回、10000 回での結果をパターン毎に比較する。提案手法と YOLO v3 での実験結果を表 4 に示す。

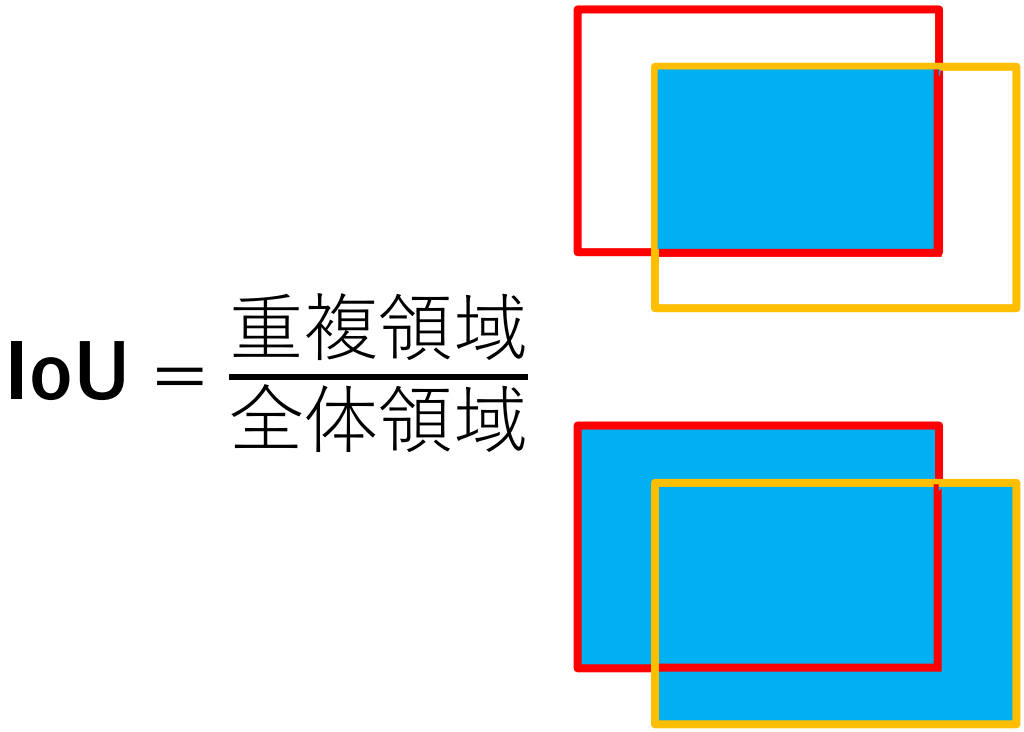

図 10　IoU の概念（赤枠：正解領域、橙枠：検出領域）

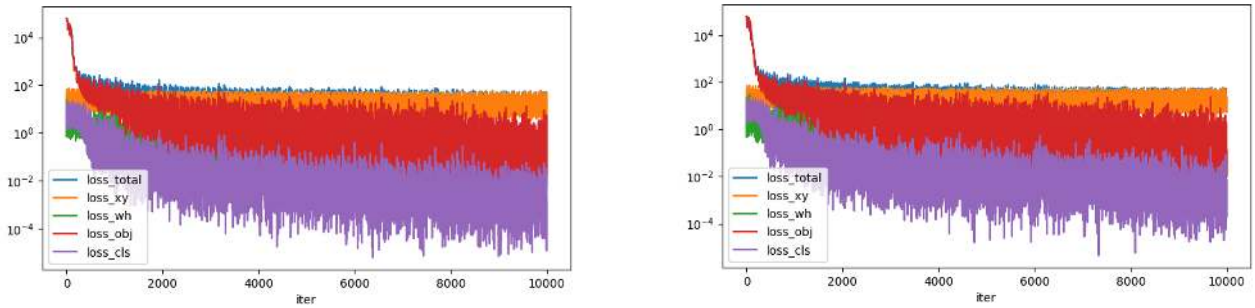

図 11　loss 関数の推移（左：提案手法、右：YOLO v3）

実験 1 の実験結果の考察

表 4 から、パターン 4 以外のパターンにおいて、提案手法の結果が YOLO v3 の結果よりも優れていることがわかる。パターン 4 では YOLO v3 の学習回数 5000 回での結果の値が最も優れているが、パターン毎の平均値において提案手法の最も優れている結果（10000 回）と YOLO v3 の最も優れている結果（10000 回）を比較すると提案手法は YOLO v3 よりも約 0.04 上回っている。このことから、全体的に、提案手法の検出精度が YOLO v3 よりも優れていることがわかる。また、学習回数ごとの結果で見てみると学習回数 10000 回が概ね優れた結果を示している。パターン 4 での YOLO v3 の結果では、10000 回よりも 5000 回の方が良い結果になっているが、平均値で見ると学習回数が多いほど精度が良くなっていることがわかる。

表 4　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
YOLO v3	3000	0.5012	0.3072	0.5746	0.6864	0.3589	0.4857
	5000	0.5873	0.3607	0.6358	**0.7072**	0.4454	0.5473
	10000	0.6162	0.4214	0.6838	0.6460	0.4933	0.5721
提案手法	3000	0.5366	0.2045	0.6440	0.6112	0.3407	0.4674
	5000	0.6307	0.3588	0.6702	0.6917	0.4860	0.5675
	10000	**0.6362**	**0.4325**	**0.7240**	0.7050	**0.5533**	**0.6102**

実験 1 の追加実験

実験 1 では、提案手法が YOLO v3 よりも検出精度において優れていることを示した。しかし、提案手法の学習回数 10000 回でのパターンごとの平均値は 0.6102 となっており、あまり良くない結果である。そこで、MSCOCO[19]で学習を行なった YOLO v3 と実験データを 10000 回学習した提案手法での比較を行った。MSCOCO は馬のクラスを含む 80 種類のクラスの約 10 万枚の画像で構成されるデータセットである。この MSCOCO の学習済みモデルを用いた YOLO v3 の AP と提案手法における学習回数 10000 回での AP の比較を表 5 に示す。

表 5 から MSCOCO での学習済みモデルを用いた YOLO v3 の結果が提案手法における学習回数 10000 回での結果よりも大きく上回っていることがわかる。パターンごとの平均値で見ると、約 0.23 上回っている。この原因の一つとして、提案手法での学習データ数が少なかったことが考えられる。今回の実験では、実験データに 10 枚の連続した画像群であるクリップを 140 個使用している。1 つのクリップ内の画像はほとんど変化のない画像のため、異なるタイプの画像としては 140 種類のみであると考えられる。ここからクロスバリデーションの手法を用いると、1 つのパターンで約 110 種類しか学習に使用されない。動画像で考えると 28 本の動画しか学習に使えていない。MSCOCO は約 10 万枚の画像で構成されていることを考えると、結果が良くなかった原因の 1 つとして、実験データの数が少なかったことが考えられる。単一の画像データは近年比較的収集が容易になっているが、提案手法では少なくとも連続した 3 フレームの動画像が必要となる。学習データの収集の困難さが課題の一つである。

表 5　MSCOCO を用いた YOLO v3 と提案手法（学習回数 10000 回）との比較

	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
YOLO v3 (MSCOCO)	**0.7571**	**0.8396**	**0.9021**	**0.9978**	**0.7026**	**0.8398**
提案手法	0.6362	0.4325	0.7240	0.7050	0.5533	0.6102

4.3　実験 2

実験 2 では、提案手法と従来手法（YOLO v3）での連続検出の精度を評価する。実験データは実験 1 と同様に、140 個のクリップを用いる。 1 クリップには連続する 10 フレーム分の画像がある。この 10 フレーム内で検出された矩形領域（BBox）の数がどのくらい変わったかを比較することで、連続検出の精度を評価する。具体的には、図 12 のように、あるフレームで検出された BBox の数とその次のフレームで検出された BBox の数の差分を取る。この差分を対応する馬の数ごとに 140 クリップ分取り、差分の合計数を提案手法と YOLO v3 で比較する。差分の合計数が小さいほど、連続での検出が行えていることになる。信頼度スコアの閾値は 0.4 とする。実験の結果を表 6 に示す。

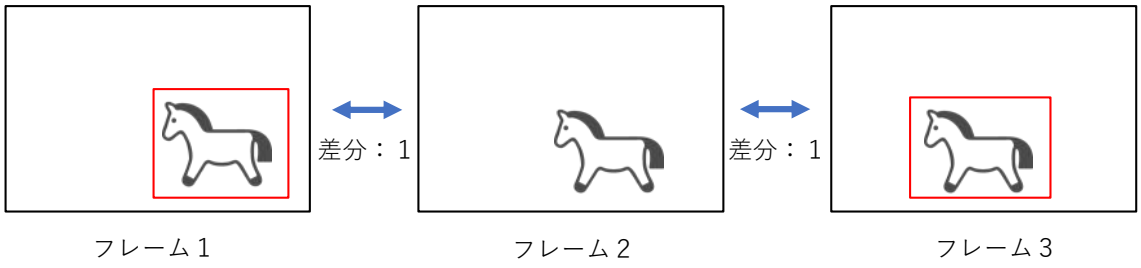

図 12　連続フレームごとの検出された BBox の数の差分の例（赤枠：検出成功）※1

※1 馬のアイコン：https://icooon-mono.com/about-icoon-mono/

実験２の実験結果の考察

表 6 から提案手法の方が YOLO v3 よりも差分の合計数が少ない結果であった。YOLO v3 では BBox の数の変化が 251 回あったのに対して、提案手法では、約半分の 133 回であった。このことから、提案手法は YOLO v3 に比べて、連続での検出が行えており、時間方向での物体検出結果が安定していると考える。

表 6　実験２の結果

	差分の合計数
YOLO v3	251
提案手法	**133**

次に、実験データのクリップの中から３つの例を選び、10 フレームで検出される矩形領域の数の比較を行った。使用したクリップの画像を図 13 に示す。

例 1　　例 2　　例 3

図 13　使用したクリップの画像

例 1 の物体検出結果

表 7 に例 1 でのフレームごとの正しい矩形領域の数と従来手法での BBox の数、提案手法での BBox の数を示す。また、YOLO v3 で物体検出を行なった結果のクリップを図 14、提案手法で物体検出を行なった結果のクリップを図 15 に示す。例 1 では、画像中に 1 頭の馬が存在するため、この 1 頭の馬の位置に矩形領域が付加されていれば検出が成功となる。YOLO v3 では 1 フレーム、3 フレーム、9 フレームで検出漏れが生じているのに対して、提案手法では全てのフレームで検出ができている。

表 7　例 1 のフレーム毎の矩形領域の数

	1	2	3	4	5	6	7	8	9	10
正解	1	1	1	1	1	1	1	1	1	1
YOLO v3	0	1	0	1	1	1	1	1	0	1
提案手法	1	1	1	1	1	1	1	1	1	1

赤・・・馬 1 の矩形領域の数

図 14　例 1 に対して YOLO v3 で物体検出を行なった結果のクリップ

図 15　例 1 に対して提案手法で物体検出を行なった結果のクリップ

例２の物体検出結果

表 8 に１頭目の馬を赤色、２頭目の馬を青色の数字で、例２でのフレームごとの正しい矩形領域の数と従来手法での BBox の数、そ提案手法での BBox の数を示す。また、YOLO v3 で物体検出を行なった結果のクリップを図 16、提案手法で物体検出を行なった結果のクリップを図 17 に示す。例２では、画像中に２頭の馬が存在するため、２頭の馬ごとの位置に矩形領域が付加されているかどうかで検出を判断する。YOLO v3 では２フレーム目において２頭目の馬の検出漏れが生じているが、提案手法では 2 フレーム目も検出漏れなく全てのフレームで検出ができている。

表 8　例２のフレーム毎の矩形領域の数

	1	2	3	4	5	6	7	8	9	10
正解	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1
YOLO v3	1,1	1,0	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1
提案手法	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1	1,1

赤・・・馬１の矩形領域の数

青・・・馬２の矩形領域の数

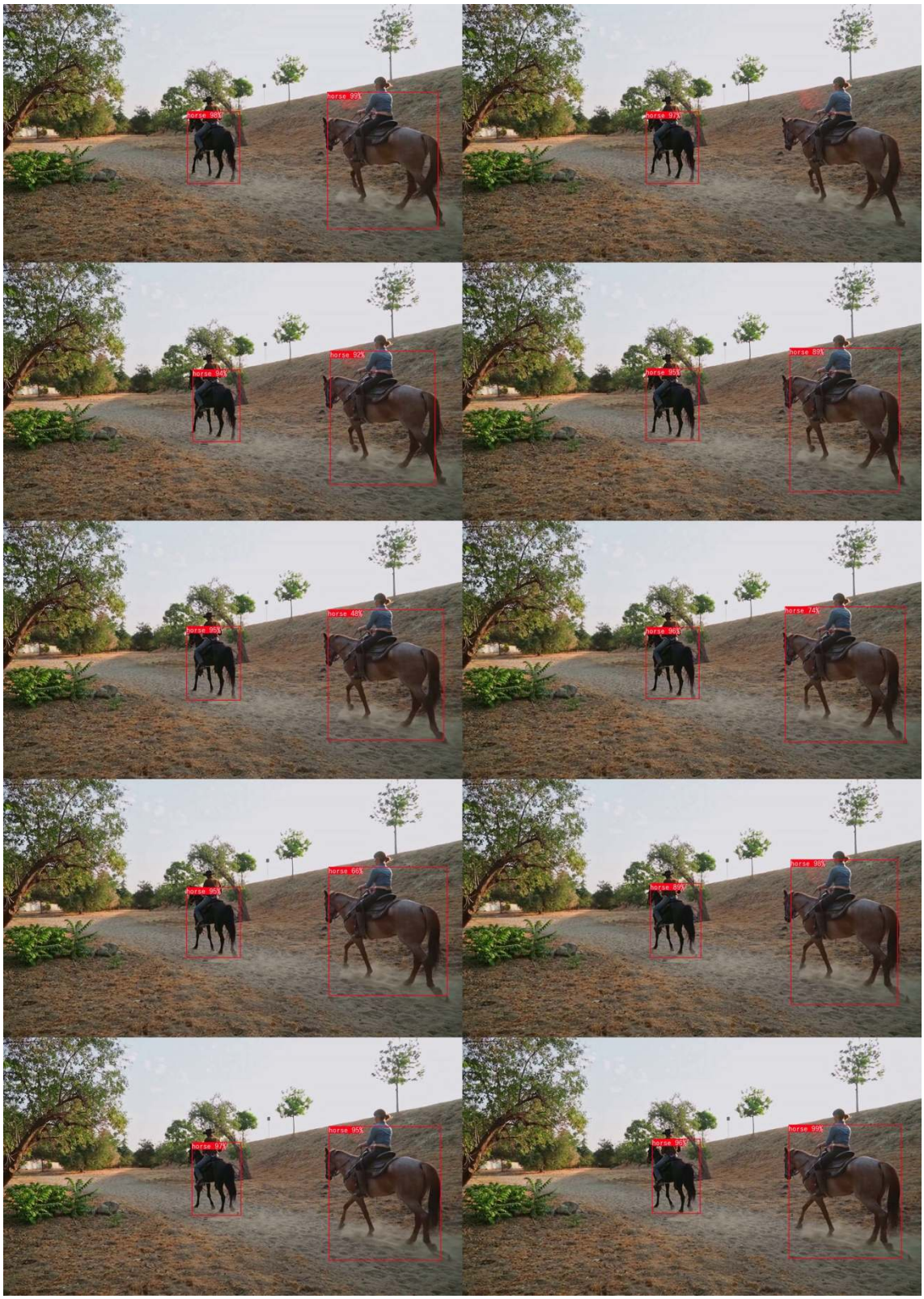

図 16　例 2 に対して YOLO v3 で物体検出を行なった結果のクリップ

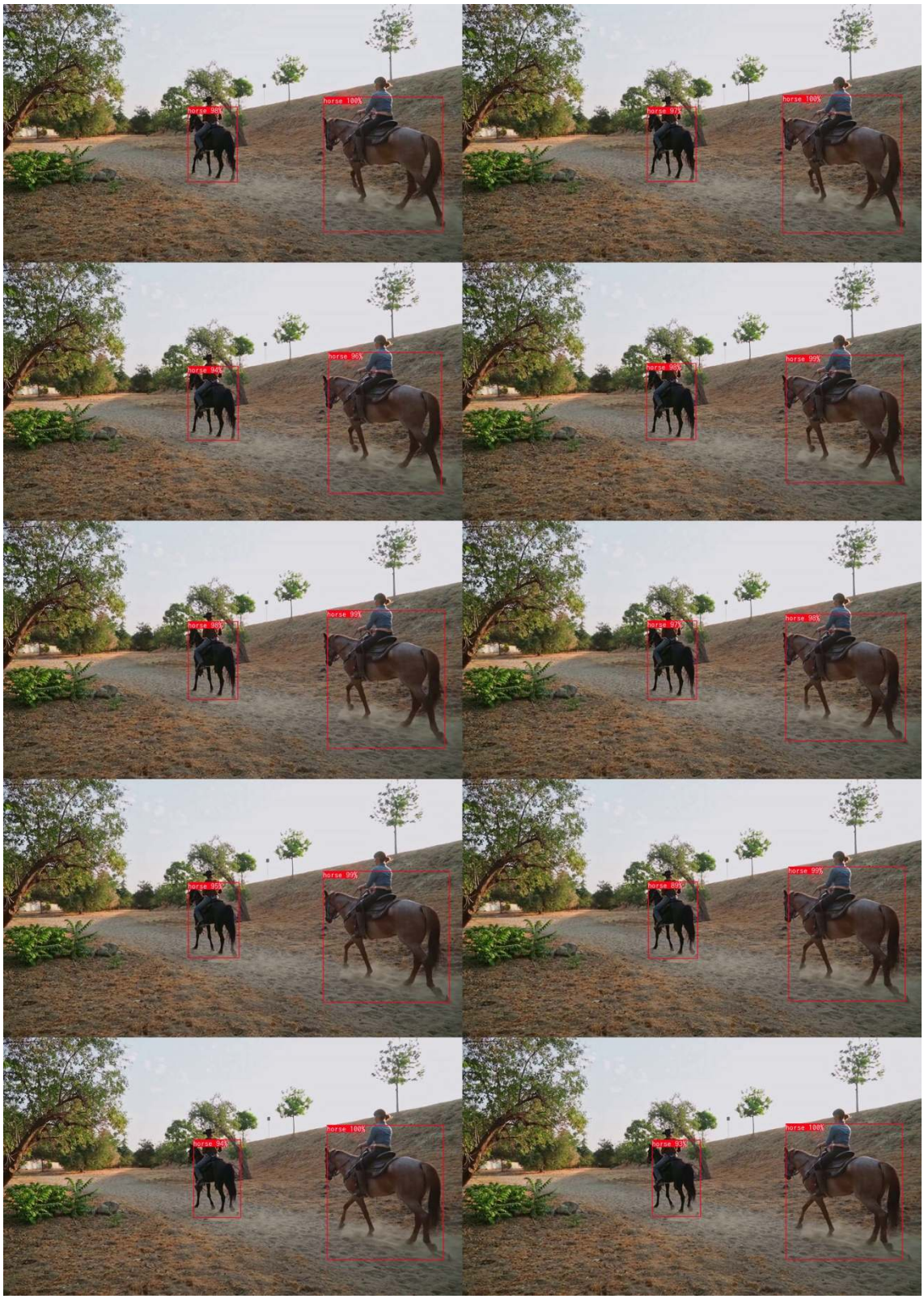

図 17　例 2 に対して提案手法で物体検出を行なった結果のクリップ

例 3 の物体検出結果

表 9 に 1 頭目の馬を赤色、2 頭目の馬を青色、3 頭目の馬を緑色、4 頭目の馬を紫色の数字で、例 3 でのフレームごとの正しい矩形領域の数と従来手法での BBox の数、提案手法での BBox の数を示す。また、YOLO v3 で物体検出を行なった結果のクリップを図 18、提案手法で物体検出を行なった結果のクリップを図 19 に示す。例 3 では、画像中に 4 頭の馬が存在するため、4 頭の馬ごとの位置に矩形領域が付加されているかどうかで検出を判断する。YOLO v3 と提案手法の両方で、馬 3 の検出ができなかったことがわかる。これは馬 2 と馬 3 が重なり合っているため、片方の馬だけ検出され、もう片方の馬は検出ができなかったためである。

表 9　例 3 のフレーム毎の矩形領域の数

	1	2	3	4	5	6	7	8	9	10
正解	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1	1,1,1,1
YOLO v3	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,0,1,1	1,1,0,1
提案手法	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1	1,1,0,1

赤・・・馬 1 の矩形領域の数
青・・・馬 2 の矩形領域の数
緑・・・馬 3 の矩形領域の数
紫・・・馬 4 の矩形領域の数

図 18　例 3 に対して YOLO v3 で物体検出を行なった結果のクリップ

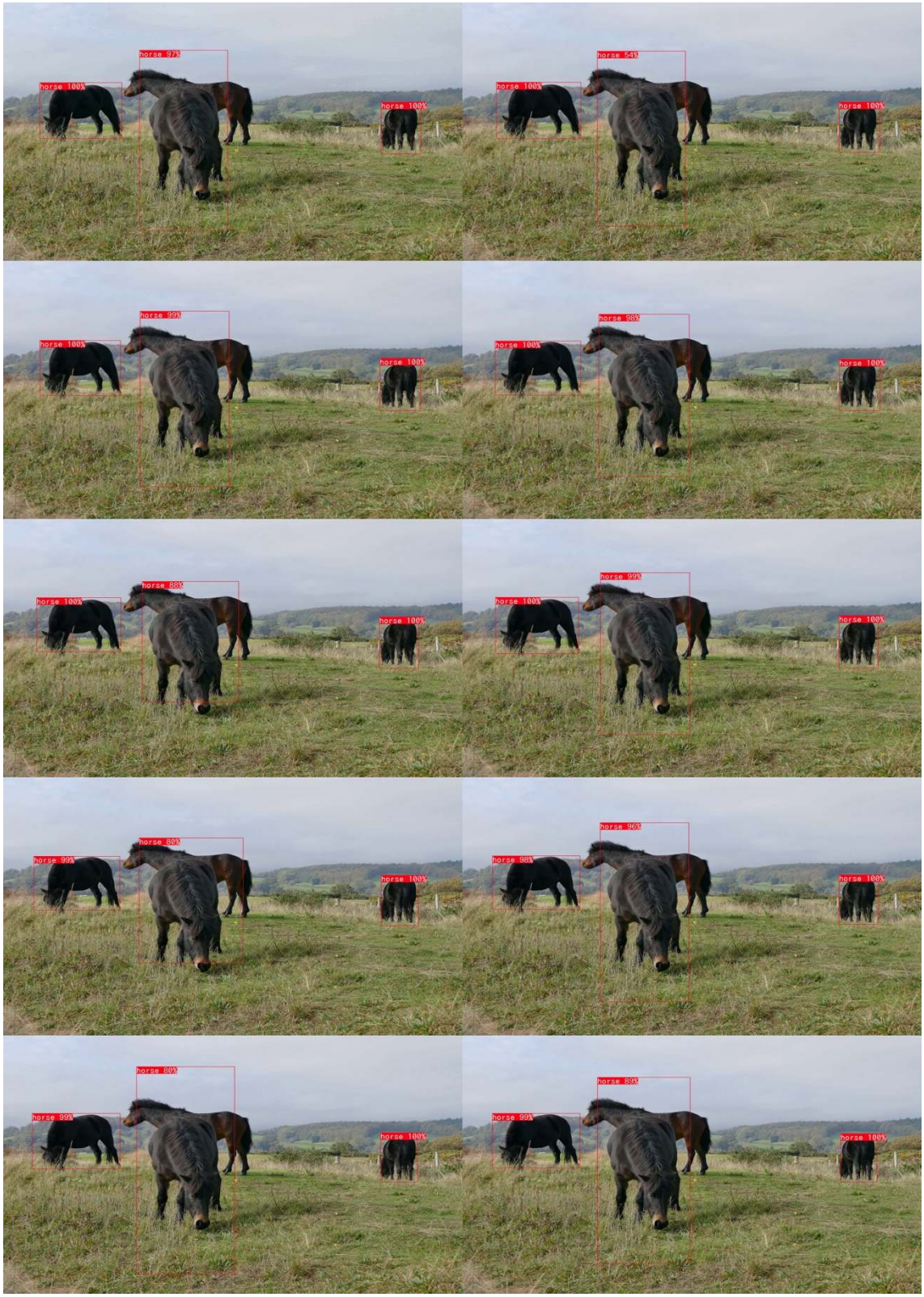

図 19　例 3 に対して提案手法で物体検出を行なった結果のクリップ

３つの例からの考察

例１と例２においては、YOLO v3 での物体検出で検出漏れが生じていたのに対して、提案手法での物体検出では、検出漏れなく全てのフレームで検出ができていた。また、例３では、YOLO v3 と提案手法の両方で、４頭の内１頭の馬の検出がうまくいかないことが確認された。この馬は別の馬と重なり合っているため、両方の馬を別々に検出することが難しかったと考えられる。提案手法では検出するフレームの前後のフレームを物体検出に利用するが、前後のフレームでの検出がうまくいっていなければターゲットとなるフレームの検出もうまくいかない場合が考えられる。このことから、従来手法で複数のフレームを連続して検出できないような物体に対しては、提案手法でも検出することが難しいと考える。

4.3　実験3

実験3では、提案手法の物体検出ネットワークのチャネル数を増やすことで、精度の変化が見られるかどうかを評価した。チャネル数を増やすことで、学習できる特徴量も増えるため、精度の向上が見込める。逆に、チャネル数を減らすことで学習すべきパラメータ数が減り、学習データが少なくても適切に学習が行える可能性もあると考えた。

表 10 に、実験に用いた各ネットワークのチャネル数を示す。1.5 倍ネットでは、物体検出畳み込み層での各チャネル数を全て 1.5 倍、0.5 倍ネットでは全て 0.5 倍とした。物体検出畳み込み層 2 , 3 では、それぞれ物体検出畳み込み層 1 , 2 からの特徴マップを結合して入力としているが、特徴マップのチャネル数がそれぞれ 1.5 倍、0.5 倍に変わるため、結合後の入力となる特徴量のチャネル数もそれに応じて変わる。

実験3も実験1と同様に、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、AP を用いて検証を行う。実験1では、学習回数 3000 回、5000 回、10000 回での AP の値を比較していたが、実験3では、10000 回での AP の値のみで比較する。実験3の実験結果を表 11 に示す。

表 10　実験に用いた各ネットワークのチャネル数

	サイズ	提案手法	1.5 倍ネット	0.5 倍ネット
conv1_d	19×19	6144 1024 2048 1024 2048 1024	6144 1536 3072 1536 3072 1536	6144 512 1024 512 1024 512
out1	19×19	2048	3072	1024
up1	38×38	512	768	256
conv2_d	38×38	3584 512 1024 512 1024 512	3840 768 1536 768 1536 768	3328 256 512 256 512 256
out2	38×38	1024	1536	512
up2	76×76	256	384	128
conv3_d	76×76	1792 256 512 256 512 256	1920 384 768 384 768 384	1664 128 256 128 256 128
out3	76×76	512	768	256

実験３の実験結果の考察

表 11 からチャネル数を増やしても検出精度の向上はほとんど見られなかった。パターン３では、チャネル数を増やした物体検出ネットワークでの手法が最も優れた結果になったが、平均すると図 7 の物体検出ネットワークでの手法が最も優れていることがわかる。また、チャネル数を減らした物体検出ネットワークでの手法がどのパターンにおいても、最も悪い結果になっている。実験３の結果から、チャネル数を増やせば増やすほど精度が良くなるわけではなく、本研究での物体検出ネットワークのチャネル数は図 7 で示したものが適切と考える。

表 11　実験３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
提案手法	**0.6362**	**0.4325**	0.7240	**0.7050**	**0.5533**	**0.6102**
1.5倍ネット	0.6329	0.4145	**0.7289**	0.6648	0.5350	0.5952
0.5倍ネット	0.5977	0.3896	0.6322	0.6342	0.5389	0.5585

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、時間方向に連続する数フレームから抽出した特徴量を元に、物体検出を行う手法を提案し、従来の物体検出手法との比較により有効性を示した。

提案手法では、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 フレームから特徴量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ネットワークに与えて、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、前後のフレームの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ようになると考えた。実験の結果から、提案手法は物体検出の精度、時間方向での安定性の 2 つの点で従来手法よりも優れていることを示した。しかしながら、平均適合率の値としてはあまり良くない結果であった。

今後の課題としては、学習データを増やすことが挙げられる。実験として、MSCOCO データセットで学習した従来手法と自身で作成した学習データで学習した提案手法とを比較した結果、提案手法の検出精度は従来手法を下回っていた。MSCOCO が約 10 万枚の画像で構成されているのに対して、作成した学習データは 28 本分の動画像のみであった。動画像のデータは、単一画像に比べて収集が難しく、人手による教師データの付与のコストも高い。この問題を解決し、より大規模なデータセットでの実験が必要である。

また、入力するフレームの数を増やすことも課題として挙げられる。本研究では 3 つの連続したフレームを物体検出に使用した。入力フレームをさらに増やすことで、より多くの特徴量を物体検出に使用することができる。しかし、特徴量を増やしすぎると、GPU のメモリ不足で学習ができなくなる問題があるため、ネットワーク構造の工夫が必要であると考える。

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雅之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をしていただき、研究活動をよりスムーズに行うことができました。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] Viola, P., Jones, M.J.,“Robust Real-Time Face Detection.”, International Journal of Computer Vision 57, 137–154 (2004)

[2] Joseph Redmon , Santosh Divvala, Ross Girshick , Ali Farhadi “You Only Look Once: Unified, Real-Time Object Detection”, CVPR (2016)

[3] DarkNet：https://pjreddie.com/darknet/

[4] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft, “Simple Online and Realtime Tracking”, arXiv:1602.00763 [cs.CV] (2017)

[5] Karen Simonyan, Andrew Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, arXiv:1409.1556 [cs.CV] (2015)

[6] Kaiming He , Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, CVPR (2016)

[7] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, CVPR (2015)

[8] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, “SSD: Single Shot MultiBox Detector”, ECCV (2016)

[9] Joseph Redmon, Ali Farhadi, “YOLOv3: An Incremental Improvement”, arXiv18.04.02767[cs.CV] (2018)

[10] Olaf Ronneberger, Philipp Fischer, Thomas Brox,“U-Net: Convolutional Networks for Biomedical Image Segmentation”, arXiv:1505.04597 [cs.CV] (2015)

[11] Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, “SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation”, arXiv:1511.00561 [cs.CV] (2015)

[12] YOLO v3：https://www.nature.com/articles/s41598-021-81216-5

[13] Bing Shuai, Andrew Berneshawi, Xinyu Li, Davide Modolo, Joseph Tighe, “SiamMOT: Siamese multi-object tracking”, CVPR (2021)

[14] Nicolai Wojke, Alex Bewley, Dietrich Paulus, “Simple Online and Realtime Tracking with a Deep Association Metric”, arXiv:1703.07402 [cs.CV] (2017)

[15] Guanghn Ning, Zhi Zhang, Chen Huang, Zhihai He, Xiaobo Ren, Haohong Wang, “Spatially Supervised Recurrent Convolutional Neural Networks for Visual Object Tracking”, arXiv:1607.05781[cs.CV] (2016)

[16] 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)

[17] 動画ダウンロードサイト：https://www.pexels.com/ja-jp/search/videos/

[18] LabelImg：https://github.com/heartexlabs/labelImg

[19] Tsung-Yi Lin, Micheal Maire, Serge Belongie, Lubomir Bourdev, Ross Girshck, James

Hays, Pietro Perona, Deva Ramanan, C.Lawrence Zitnick, Piotr Dollar, “Microsoft COCO: Common Objects in Context”, arXiv:1405.0312[cs.CV] (2014)