令和 3 年度茨城大学大学院理工学研究科情報工学専攻
修士学位論文

半教師あり物体検出モデル CSD を利用した 追加クラスの学習

所属　　理工学研究科情報工学専攻
著者　　結城洸太 (20NM734T)
指導教員　　新納浩幸教授

令和 4 年 2 月 4 日 (金)

令和 3 年度茨城大学大学院理工学研究科情報工学専攻
修士学位論文

半教師あり物体検出モデル CSD を利用した追加クラスの学習

著者

結城洸太 (20NM734T)

指導教員

新納浩幸教授

論文要旨

本論文では, 十分に学習された物体検出モデルを利用してそのモデルに新たなクラスを追加したものを作成したい場合に対して,CSD を利用する方法を提案する.

物体検出のモデル作成は深層学習を用いられるのが主流となっているが, その学習には一般に膨大な訓練データが必要である. そのため, データセットの作成にコストが掛かるという問題がある.

上記の問題を解消するため, 低いコストの訓練データで検出器を学習する方法を提案する. 物体検出の技術を利用する際, 既に学習されたクラスに対して, 新たなクラスを追加したい場合が考えられる. 本論文では, その場合を想定し, 4 クラスの十分な量のデータで学習された SSD のモデルをベースにして, 1 クラスを追加した 5 クラスの CSD のモデルを, 低いコストのデータを用いて作成した. そして, 訓練データの量を変更し実験を行い, その違いによる精度の差を検証した.

その結果,CSD を用いた方法は, パフォーマンスの向上には有効ではあるが不安定であり, 実用的な方法ではないと結論づけた.

Master's in Scholastic 2021,
Major in Computer and Information Sciences,
Granduate School of Science and Engineering, Ibaraki University

Learning additional classes used CSD, a Semi-supervised learning method for object detection

Author

Kota Yuuki (20NM734T)

Adviser

Prof. Hiroyuki Shinnou

Abstract

In this paper, we propose a method o use CSD when you want to create a new class added to the model using a well-learned object detection model. Deep learning is used for the learning of object detectors. In general, a huge amount of training data is necessary for the training. Therefore, there is a problem that it costs to create a dataset.

We propose a method to learn detectors with low cost training data, in order to solve the above problems. When using object detection technology, it is possible to add a new class to the learned detector. In this paper, assuming this case, based on a model of SSD learned with a sufficient amount of data of four classes, a 5-class CSD model with 1 class added was created using low cost data. Then, the amount of training data was changed and experiments were carried out, and we verified the difference in accuracy due to the difference.

As a result, The method using CSD is effective but unstable for improving performance, we concluded that it was not a practical method.

目次

図目次

表目次

1 序論

1.1 背景

物体検出は, 画像中に映る物体の位置と種類を検出するタスクである. 物体検出の主な使用としては, 自動運転や顔認証などがあり, 様々な分野で利用されている技術である. 物体検出のモデル作成は深層学習を用いられるのが主流となっていおり, それにより近年精度が大きく向上している.

その学習には, 一般に教師あり学習が用いられる. 物体検出を含む教師あり学習では, 膨大な訓練データが必要であり, 特に物体検出の訓練データは, 訓練画像に加えて, 正解クラスと正解ボックスのアノテーションが必要である. そのうち, 正解ボックスは画像中の座標を用いて正確に表す必要があり, アノテーション作業が困難である. 同じ画像のタスクである画像識別は, アノテーションは正解クラスのみであるのに対し, 物体検出のアノテーションは, 正解ボックスが加わることで, データ作成のコストが比較的高く,1 つの物体のラベリング作業には約 10 秒かかるといわれている [1]. そのため, 物体検出のデータセットの作成にかかるコストは, この分野で大きな問題とされている.

1.2 概要

本論文では, 学習済みモデルにクラスを追加したい場合に対して, 訓練データのコストを削減することを目的に, 低コストで物体検出モデルを学習する手法を提案する.

具体的な方法としては, まず, 十分な 4 クラスのデータを用いて,SSD のモデルを作成する. SSD は, 教師あり学習を用いた物体検出のモデルであり, 十分な精度の検出を行うことができる. この学習済みモデルの検出対象である 4 クラスに,1 クラス追加した 5 クラスの検出が可能となるモデルを作成する. このとき, 学習済みの 4 クラスの SSD モデルを利用し,CSD を用いた低コストの訓練データでの学習を行う.

これにより, 学習の用いる訓練データのコストを抑えたうえで, 十分な検出精度のモデルの作成を期待する. また, 訓練データの量を変え, 各モデルの精度を比較する. そして, データ量によってどの程度の精度向上が見られるかを検証する.

2　物体検出

物体検出は, 画像処理のタスクの一つである. 物体検出では, 画像を入力し, クラス確信度とバウンディングボックスを出力する. クラス確信度は, 検出された物体のクラスがどの程度確かであるかを最大値 0, 最小値 1 で表したものである. バウンディングボックスは, 物体の位置を正方形のボックスで表現したものである. 一般にバウンディングボックスは, 中心座標 (x, y), 幅 w, 高さ h で表される. これにクラス確信度を付与したものを 1 つの出力のペアとする. 物体検出では, 画像中に対象となる物体が 1 つに定まるとは限らず, 出力として位置とクラスの組が複数存在する場合がある.

物体検出の学習は, 通常は教師あり学習で行われる. 一般的な手法としては, R-CNN, YOLO, SSD の 3 つが挙げられる. これらの手法はシンプルかつ高い精度を実現しており, 現在, ほとんどの物体検出の手法は, この R-CNN 系, YOLO, SSD を合わせた 3 つの手法を派生させたものとなっている.

第 1 章で述べたように, 物体検出のラベリングには莫大なコストが掛かる. それを解決するため研究されたものに, 弱教師あり学習と半教師あり学習を用いる手法がある.

2.1　Fast R-CNN

fast R-CNN は教師あり学習を用いた物体検出の手法の一つである. この手法は R-CNN のモデルを実行速度の面で改良したものであり, R-CNN の派生としてできたモデルは, R-CNN 系と分類される.

R-CNN 系では, 物体の領域候補を探してからクラス分類をするため, 実行速度の面で YOLO や SSD には劣る手法である. しかし今では, 領域候補提案のアルゴリズムが改良されたことにより, 動画を用いた物体検出をするのに十分な実行速度が R-CNN 系のモデルでも実現されている.

検出精度の面では, YOLO や SSD に比べ R-CNN 系は高い精度であり, これは, R-CNN が比較的単純な構造であるためである. 近年では, YOLO 系の精度が向上し, YOLO 系のモデルが最も高い精度を実現している.

2.1.1　Fast R-CNN のネットワーク

Fast R-CNN のネットワーク (図 1) は, 領域候補提案部分と, クラス分類・検出部分に分かれる. 領域提案の生成は, Edge boxes などの外部アルゴリズムを利用する. これにより, 物体らしいボックスのみを厳選した領域提案を得る. 次に,RoIPooling 層で領域提案に対する CNN 特徴量をプーリングする. その後, 全結合層 (FC) を経て, 分類・検出のネットワークに分岐する. クラス分類のネットワークは, softmax 層で正規化され, クラス確信度の出力 (outputs) を得る. 検出のネットワークは, rerressor 層を経て, 領域提案

とのボックス誤差の出力 (bbox) を得る.

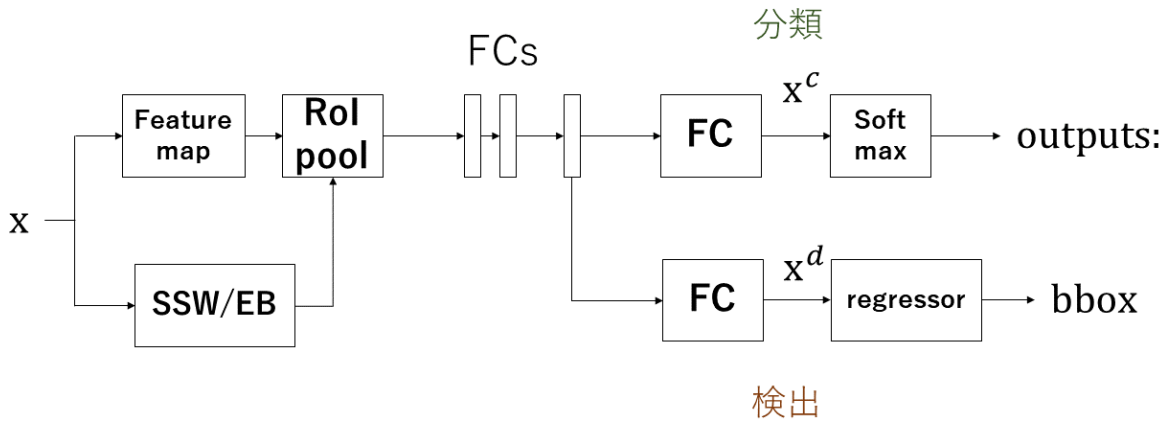

図 1　Fast R-CNN のネットワーク構造

2.1.2　Fast R-CNN における学習の処理概要

Fast R-CNN 及び一般的な教師あり学習の物体検出では, 以下の処理が繰り返される.

1. 訓練データの読み込み
2. ネットワークでの推論
3. 正解座標とボックスのマッチング
4. 損失関数の計算

ネットワークの推論では, 訓練データの画像を入力し, 出力としてクラス確信度とボックス位置を得る. 正解座標とボックスのマッチングは, 推論で得たボックスと正解ボックスを IoU(重複面積) で比較し, 通常 IoU が 0.5 以上を Positive として, 損失関数を計算する.

損失関数は, クラス推定の損失関数と, 位置推定の損失関数に分かれる. クラス推定の損失関数は, クロスエントロピーを用い, 位置推定の損失関数は, smoothL1 正規化を用いて計算される.

2.2 Faster R-CNN

Faster R-CNN は Fast R-CNN をより高速に改良したモデルである. 領域候補提案のアルゴリズムを畳み込みニューラルネットワーク (CNN) で組み込むことで, 前モデル Fast R-CNN の約 8 倍の速さ, かつ end-to-end での検出を実現している.

2.2.1 Faster R-CNN のネットワーク

Faster R-CNN のネットワーク (図 2) は, Fast R-CNN とほぼ同じネットワークであり, 改良された領域候補提案部分が異なる. Faster R-CNN では, 領域候補提案を RPN(Region Proposal Network) と呼ばれる CNN 層で構成される. RPN では, 特徴マップの出力を利用し領域提案を出力する. 外部アルゴリズムではなく CNN に組み込むことで高速化を実現し, データに合わせて訓練・調整される.

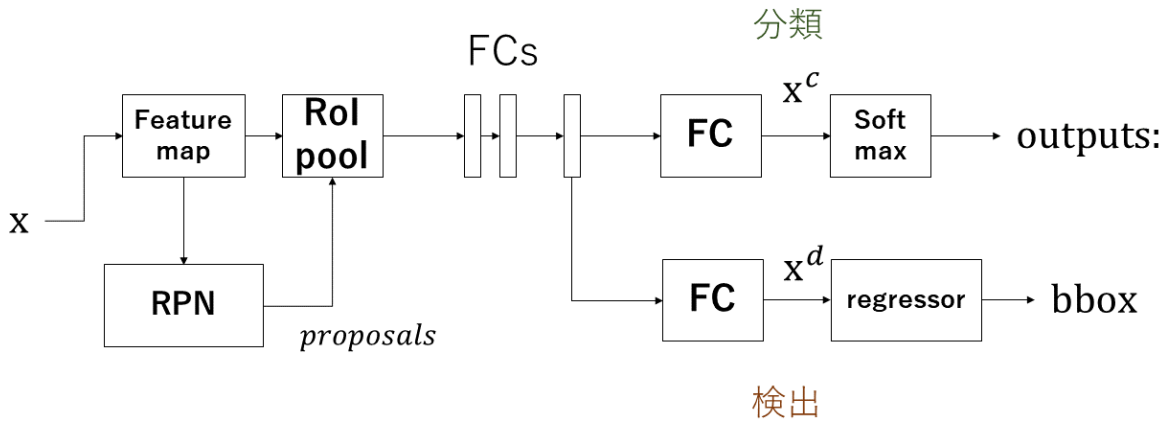

図 2 Fast R-CNN のネットワーク構造

2.2.2 Faster R-CNN における学習の処理概要

Faster R-CNN の処理は, Fast R-CNN と同様の流れで行われる. ただし, 変更点である RPN 部分で異なる処理が行われる. RPN では, 特徴抽出器の出力を利用した CNN に

より, 自動生成されたボックスを厳選・誤差調整し, 領域提案が出力される. 生成された領域提案は, Fast R-CNN 同様 RoLPooling 層に渡される.

また, 損失関数として, クラス推定と位置推定に加え, RPN の損失計算が追加される. これにより, 領域候補部分のネットワークが訓練され, データに沿った領域提案が可能となる.

2.3 SSD

SSD(Single Shot MultiBox Detector) [2] は, 物体検出の手法の一つである. R-CNN 系は, 物体の領域候補を探してからクラス分類を行っており,2 段階の処理に分かれていた. それに対して SSD での物体の検出方法は, ボックスをあらかじめ用意することで位置推定とクラス分類を同時に実行しており, R-CNN 系で 2 段階で行われた処理を 1 段階で行っている. この様なアルゴリズムを Single Shot と呼ぶ. また, それに対して R-CNN 系を Two Shot と呼ぶ. これにより, R-CNN 系と比較して優れた実行速度を実現している. 同じく Single Shot 系の物体検出手法としては YOLO が挙げられる. YOLO は, R-CNN 系の弱点である実行速度を改善する手法として開発されたが, 検出精度の面では R-CNN 系には及ばなかった. そこで, Single Shot の手法を用いて精度の向上を図るため開発されたのが SSD である. SSD ではデフォルトボックスの作成を工夫することで, 同じく実行速度の面で優れた手法である YOLO より高い検出精度を実現した.

2.3.1 SSD の仕組み

SSD の位置推定の方法について説明する. SSD における物体の位置はバウンディングボックスで表され, これらはデフォルトボックスとオフセットの組み合わせによって表される. デフォルトボックスは以下の式に従い, デフォルトの設定では 1 つの画像に対して 8732 個生成される.

k 番目の特徴マップのスケール s_k

特徴マップの数 m

$s_{max} = 0.9, s_{min} = 0.2$(入力画像のスケールを 1.0)

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1), k \in [1, m]$$

アスペクト比 a_r

$$a_r = 1, 2, 3, \frac{1}{2}, \frac{1}{3}$$

$$\text{幅 } (w_k^a = s_k\sqrt{a_r}), \text{高さ } (w_k\ a = s_k/\sqrt{a_r})$$

アスペクト比が 1 の場合は以下のスケールを追加

$$s'_k = \sqrt{s_k s_{k+1}}$$

各デフォルトボックスの中心座標 k 番目の特徴マップのサイズ f_k

$$(\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|})$$

$$i, j \in [0, |f_k|)$$

SSD では, 複数の異なる分割数の特徴マップに対してデフォルトボックスを作成する. 分割数が変わると特徴マップのセルの大きさが変わり, その大きさに応じてボックスのスケールを定める. これによりスケールの違うボックスが生成される. スケールの違いにより, 様々な大きさの物体に対応し検出することができる.

さらに, ボックスは一つのセルにおいても複数個生成される. スケールは同じだがアスペクト比の異なるボックスを生成することで, 様々な形の物体に対応している.

これによって生成された 8732 個のデフォルトボックス $priors(cx, cy, w, h)$ ごとに, オフセット $loc\_p(\Delta cx, \Delta cy, \Delta w, \Delta h)$ とクラス確信度 $conf\_p(c0, c1, c2, \cdots, cn)$ を推論する. そして, クラス確信度がしきい値を超えたボックスを, バウンディングボックス $(cx + \Delta cx, cy + \Delta cy, w + \Delta w, h + \Delta h)$ として表示する.

ただし, この方法では多くのデフォルトボックスがしきい値を超え, 一つ物体に対して複数のバウンディングボックスが集中してしまう. その問題を解消する方法として, NMS(Non Maximum Suppression) がある. これは, 表示されたバウンディングボックス同士の重複面積がしきい値を超えた場合, クラス確信度が最大のボックス以外を削除するという手法である. これにより, 一つの物体に対して表示されるバウンディングボックスは一つに定まる.

2.3.2 SSD のネットワーク

SSD のネットワークは CNN で構成されている. 図 3 は, SSD のネットワーク構造を図式化して示したものである. SSD のネットワークは, ベースネット層, 追加ネットワーク層, 位置推定層, クラス推定層に分けることができる.

ベースネット層は,vgg16 などの画像認識のモデルから最終層の全結合層を除いたネットワークとなっている. この部分は, 既存のモデルを利用することになる. 特徴マップは, 入力として 300×300 の画像が入力され, 複数の CNN 層への入出力を経て, 最終的に 38×38 の特徴マップを出力する.

追加ネットワーク層は, SSD 独自のネットワークで, 畳み込みにより 38×38 の特徴マップを 1×1 まで小さくしていく.

最後に, 位置推定層, クラス推定層は, どちらもパディング 1, 3×3 のフィルタの畳み込み層 6 層からなり, 特徴マップの大きさを一定に保持したまま, ネットワーク最終層まで

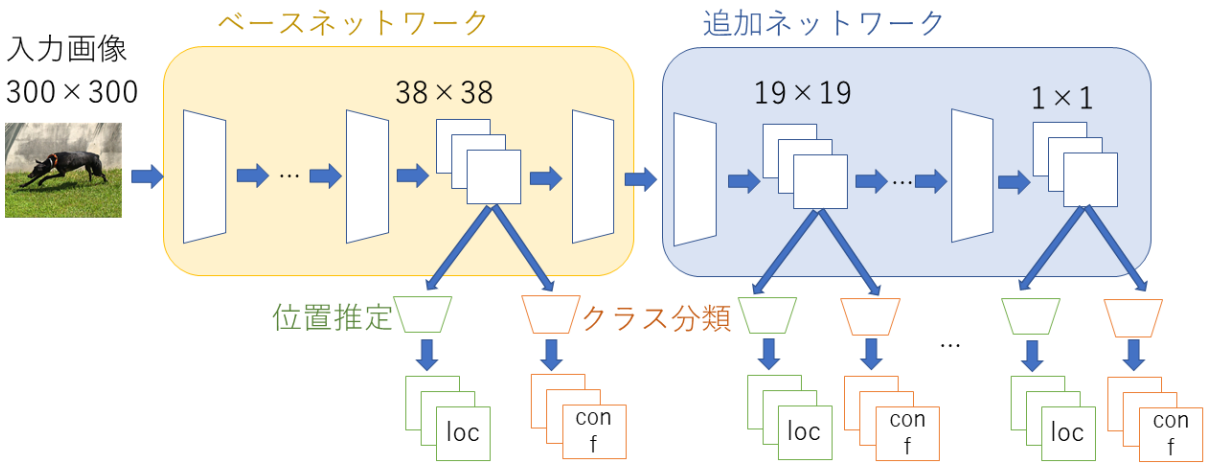

図 3　SSD のネットワーク構造

つながる．この二つの層は, ベースネット層, または追加ネットワーク層から分岐している．異なるサイズの特徴マップから分岐することで, 様々な物体の大きさに対応したボックスの作成を実現している．

2.3.3　SSD における学習の処理概要

SSD における学習では以下の処理が繰り返される．

1. 訓練データの読み込み
2. ネットワークでの推論
3. 正解座標とボックスのマッチング
4. 損失関数の計算

ステップ 3 を除いては, 基本的な機械学習の流れと同じとなっている．

ステップ 1 では, 訓練データから, 画像 $images$ とアノテーション $target$ のテンソルを作成する．$image$ は 300×300 にリサイズされ, (C, H, W) の画像データ形式に変更される．

$images = (\text{バッチサイズ} : B, Channel : 3, Height : 300, Width : 300)$

$targets = (\text{バッチサイズ} : B, \text{物体数} : O, (xmin, ymin, xmax, ymax, label\_ind) : 5)$

ステップ 2 では, ネットワークの推論を行う. ステップ 1 で作成した $images$ のデータからネットワークで推論を行い, 推論結果 out を作成する. 推論結果 out は, 以下で構成されている.

オフセット $loc\_p : (\Delta cx, \Delta cy, \Delta w, \Delta h)$
クラス確信度 $conf\_p : (c0, c1, c2, \cdots, cn)$
ボックス座標 $priors : (cx, cy, w, h)$

ステップ 3 では, ボックスのマッチング処理を行う. ステップ 1 で作成したアノテーション $targets$ から, 正解座標 $truths$ と正解ラベル $labels$ を作成する. 正解座標 $truths$ と正解ラベル $labels$, ステップ 2 で作成した推論結果 out を用いて, マッチング処理を行う.

マッチング処理は 8732 個のデフォルトボックスに対して行われ, ボックス座標 $priors$ とオフセット loc_p から求められるデフォルトボックスの面積と正解座標 $truths$ の面積を IoU でマッチングする. IoU は, 二つの領域がどれくらい重なっているかを表す指標である. IoU については, 節 5.1.2 で詳しく説明する. マッチングにおける IoU の閾値は 0.5 で, それを超えた場合マッチング成功となる.

マッチング成功 (Positive) の場合, 正解ラベル $conf\_t$: ラベル番号 $[1, n]$, 正解オフセット $loc\_t : (\Delta cx, \Delta cy, \Delta w, \Delta h)$ を作成する.

マッチング失敗 (Negative) の場合, 正解ラベル $conf\_t$: ラベル番号 0 を作成する.

ステップ 4 では, 損失関数の計算を行う.

SSD における損失関数は, 位置推定の損失関数とクラス確信度の損失関数を組み合わせた, 以下の式で表される.

損失関数 $L(x, c, l, g)$

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g))$$

ただし,

$$\text{位置推定の損失関数} : (localazationloss) L_{loc}(x, l, g)$$

$$\text{クラス確信度の損失関数} : (confidenceloss) L_{conf}(x, c)$$

$$\text{位置推定とクラス分類の重要度を制御するパラメータ} : \alpha (\text{ここでは} \alpha = 1)$$

マッチングに成功した $Positive$ のボックスの数 N (N が 0 の場合, 例外として損失関数は 0)

また, 位置推定の損失関数 $L_{loc}(x, l, g)$ は, 以下の式で表される.

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{(cx, cy, w, h)} x_{ij}^{k} smooth_{L1}(l_i^m - \hat{g}_j^m) \tag{1}$$

ただし,

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w$$

$$\hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^w = \log(\frac{g_j^w}{d_i^w})$$

$$\hat{g}_j^h = \log(\frac{g_j^h}{d_i^h})$$

$$smooth_{L1}(x) = \begin{cases} -0.5x^2, & |x| < 1 \\ |x| - 0.5, & |x| \geq 1 \end{cases}$$

式 3 において, i はデフォルトボックス 8732 個を区別し, j は画像中の物体の正解座標を区別している. l_i^m は推論結果のオフセット, $\hat{g}_j^m$ は正解座標のオフセットを表す. x_{ij}^k は i と j がマッチング成功なら 1, それ以外は 0 となる. $\hat{g}_j^{cx}$ は, 正解座標 g_j^{cx} とデフォルトボックス座標 d_i^{cx} のオフセットを,d_i^w で正規化している. $\hat{g}_j^w, \hat{g}_j^h$ は, ボックス幅 d_i^w と高さ d_i^h で正規化され,log スケールになっている.

位置推定の損失関数は, $Negative$ の場合 x_{ij}^k が 0 となるので, $Positive$ のボックスが計算対象であり, $smooth_{L1}$ で位置推定の誤差を計算する.

クラス確信度の損失関数は $L_{conf}(x, c)$ は, 以下の式で表される.

$$L_{conf}(x, c) = -\sum_{i \in Pos}^{N} x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg}^{N} \log(\hat{c}_i^0) \tag{2}$$

$$where \hat{c}_i^p = \frac{exp(c_i^p)}{\sum_p exp(c_i^p)}$$

式 2 において i はデフォルトボックス 8732 個を区別し, j は画像中の物体の正解座標を区別している. x_{ij}^k は i と j がマッチング成功なら 1, それ以外は 0 となる. p はクラスのラベル番号で,1 項目はデータセットの 1 から n のクラス番号, 2 項目はバックグラウンドのクラス番号 0 になる. $\hat{c}_i^p$ は $softmax$ 関数で正規化している.

$Positive$ ボックス (1 項目) と $Negative$ ボックス (2 項目) が計算対象であり, その合計が損失関数となる. クラス分類において一般的なクロスエントロピーで, クラス確信度 $conf\_p$ と正解ラベル $conf\_t$ の誤差を計算している.

なお, 損失関数の計算に使う $Negative$ のボックスは, クラス確信度の上位 30 個のみを使用する. これにより, 正解と間違いやすい不正解データを学習することができ, 精度を向上させる効果がある.

2.4 弱教師あり学習

弱教師あり学習は, 教師あり学習や教師なし学習に並ぶ, 機械学習の手法の一つである. 弱教師あり学習では,「弱教師データ」を用いて学習する.「弱教師データ」は, 教師あり学習で用いる教師データに比べて情報量の少ないものであり, データ作成のコストが低い. そのため, 教師あり学習に比べて訓練データのコストを低く抑えることができる.

具体的には, 教師あり学習では推論したいものと同じ項目の正解データを用意し学習させる必要がある. 例えば, 画像を入力し物体の位置とクラスを出力する物体検出のモデルを学習するには, 訓練データとして, 画像と, それに対する物体の位置とクラスの情報を含む正解アノテーションを用意する必要がある.

それに対し, 弱教師あり学習では, 推論したいものと同じ項目の正解データを用意し学習させる必要がなく, 正解データの一部を含まないものを使用し学習する. 物体検出においては, 物体の位置の情報を含まないデータであったり, 物体の位置を 1 点で表したデータなどを用いて, 物体の位置を推論するモデルを学習する.

2.4.1 WSOD

WSOD とは, 弱教師あり学習を用いた物体検出 (Weakly Supervised Object Detection) であり, 物体検出のタスクの一つである. 通常の教師あり学習における物体検出での学習では, 訓練データとして, 画像データとアノテーションデータの 2 つを用いる. アノテーションは, 画像に存在する物体の種類であるクラスラベルと, 物体の位置であるボックス座標の情報が含まれる. WSOD は, ボックスの情報が全くない訓練データを用いて物体検出を行うタスクのことである. つまり,WSOD における訓練データは, 画像と, そこに存在する物体のクラスラベルのみである.

物体の位置の情報なしに物体の位置を推定できる理由としては, 画像分類の CNN が, 暗黙的に物体の位置情報を含んでいるからである. 画像分類は, 画像からクラスラベルを推論するタスクであり, 学習データには画像とラベルを用いる. 画像分類の CNN は, 画像中の物体をパーツとして学習している. そのため, 画像中のどこに物体があったとしても, 画像の一部分を対象の物体と認識し, ラベルを推論することができる. つまりこのネットワークは, 画像のどの部分が該当するかという位置に関する意味表現を暗黙的に学習している. これにより, 物体の位置の情報なしで物体の位置の推定が可能となる.

2.5 WSDDN

WSDDN(Weakly Supervised Deep Detecttion Networks) [4] は, 弱教師あり学習を用いた物体検出 (WSOD) の手法の一つである. このモデルは,Fast R-CNN が基となっており,WSOD のモデルのなかでも, シンプルなアルゴリズムである. また, このモデルは

WSOD の分野で初期に精度を出したモデルであり, 以降の WSOD モデルは, このモデルを基に作成されたり, 精度比較の対象となったりしている.

WSDDN は 2015 年に発表された古い手法であり, 精度としては, m AP が 40 程度である. これは, 当時の弱教師あり学習としては非常に高い精度 (当時の最先端手法は mAP=20 程度) であったが, 教師あり学習の手法と比較するとかなり低いものであった.

WSDDN 以降 WSOD の精度は向上し, 最先端の手法で mAP は 60 程度になっている.

2.5.1 WSDDN のネットワーク

WSDDN のネットワーク構 (図 4) は,Fast R-CNN を基にしている. ただし, 層の出力部分に大きな変更点がある. Faster R-CNN では,fc 層後の分岐した層で, クラス推定の softmax と位置推定の softmax を行い, クラススコアとボックス誤差を出力していた. WSDDN では, ボックス誤差推定の softmax を行わず, 次元軸を変えたクラス推定の softmax を行っている. 通常のクラス推定の softmax では, 各領域候補に対してクラススコアを C 次元ベクトルでマッピングする. つまり, 一つのボックスごとに全クラスのスコアの総和が 1 になるのに対し, 軸を入れ替えた softmax では, クラスごとに領域候補の確率分布を R 次元ベクトルでマッピングする. つまり, クラスごとに全ボックスの総和が 1 になる. 前者はそのボックスがどのクラスに当てはまるのかを意味する値であるのに対し, 後者は, どのボックスが, より指定したクラスの物体に近いのかを意味する. 弱教師あり学習では, 正解として与えられるのはその画像に含まれる物体のクラスのみのため, 正解クラスに最も当てはまるボックスはどれかという情報は重要度が高い. 最終的にこれらの softmax 層の出力のアダマール積をとり, 最終的なスコア y として出力される.

2.5.2 WSDDN における学習の処理概要

WSDDN における学習では,Fast R-CNN 同様に以下の処理が繰り返される.

1. 訓練データの読み込み
2. ネットワークでの推論
3. 正解座標とボックスのマッチング
4. 損失関数の計算

WSDDN では, 後半の正解ボックスのマッチング及び損失関数の計算の処理が, 教師ありの R-CNN 系と異なる. WSDDN の学習は, ボックス座標の正解データを使わないで行うため, ステップ 3 の正解座標とボックスのマッチングができない. そのため, 次のようなアルゴリズムを用いる.

まず, 領域候補提案で得られた物体のから, 最もスコアが高いものを, 仮の正解ボックスとする. 仮の正解と他の領域候補提案をマッチングし, 重複面積 (IoU) が 0.6 以上のものを positive とする. その結果を基に, 損失関数を計算する.

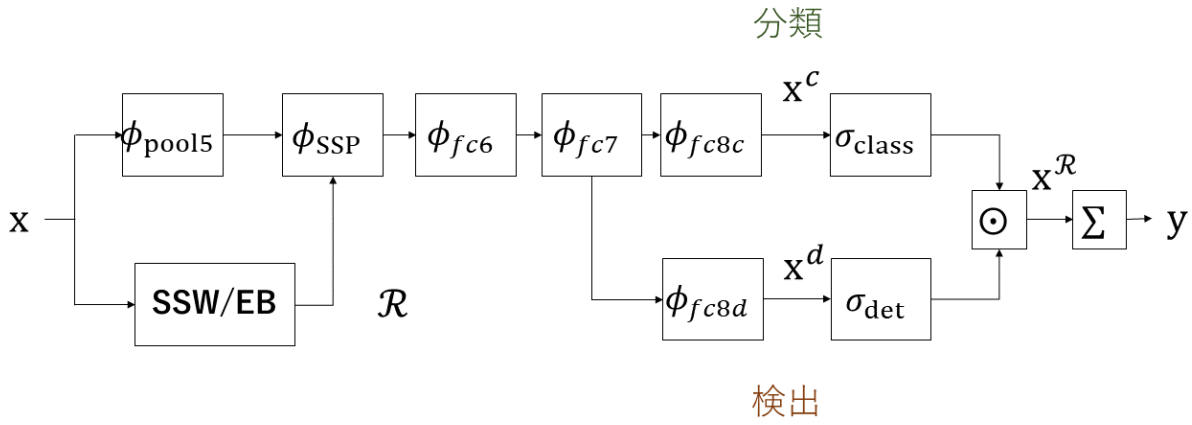

図 4　WSDDN のネットワーク構造

損失関数は, クラス推定の損失関数と位置推定の損失関数に分かれる．クラス推定の損失関数 L_{cls} は, R-CNN 同様にクロスエントロピーを用いる．位置推定の損失 L_{loc} は,Spatial Regulariser と呼ばれる以下の式によって計算される.

位置推定の損失 L_{loc} を以下の式で計算する.

$$L_{loc} = \frac{1}{nC} \sum_{k=1}^{C} \sum_{i=1}^{N_k^+} \frac{1}{2} (\phi_k^y)^2 (x_i|w) (\phi_{kp}^{fc7} - \phi_{ki}^{fc7})^T (\phi_{kp}^{fc7} - \phi_{ki}^{fc7}) \tag{3}$$

ただし,N_k^+ は, クラス k においてマッチングで Positive であったボックス, kp は $\arg\max_j \phi_k^y$(仮の正解ボックス),

仮の正解ボックス kp とマッチングで Positive を示したボックス i で, 特徴抽出直後の層 $fc7$ における出力 ϕ^{fc7} を比較する．もし, kp が正しいボックスならば, それと似た結果である i と kp では, $fc7$ での出力は似た値を示すという考えから, 損失値として $(\phi_{kp}^{fc7} - \phi_{ki}^{fc7})$ を用いている.

2.6 半教師あり学習

半教師あり学習は, 教師あり学習や弱教師あり学習に並ぶ, 学習の手法である. 半教師あり学習の訓練には, 教師ありデータと教師なしデータの両方を用いる.

半教師あり学習の手法はいくつかあるが, 主なものに, Self-training と Consistency regularization がある.

Self-training の手法は, まず教師ありデータを使用してモデルをトレーニングする. そして, 教師なしデータで予測を行う. その入力 xu の上位 1 つの予測値 y^- がしきい値 σ を超える場合, xu の疑似ラベルとしてスコアが最大であるクラス y^- として, 訓練データとして学習に利用する. これを繰り返すことで, モデルを学習を進める. ただし, しきい値 σ の設定次第では, 追加するデータ量に大きく差が出て学習が不安定になる恐れがある. そのうえ, 低すぎる値であれば, 誤った疑似ラベルを採用する可能性があり, 精度が向上しない可能性がある. また, 用意する教師なしデータに量によってもパフォーマンスに影響がでる可能性があり, 少なすぎたり, 多すぎたりする場合にも精度の低下の要因となることがある.

Consistency regularization の手法は, 入力にノイズを加えたものを用意し, それらの出力の差を最小化する手法である. 入力画像 x に対して, データ拡張などを適用し, ノイズを入れた画像 $x0$ を取得する. そして,x と $x0$ をそれぞれ入力し, 得られた出力予測 $f(x)$ と $f(x0)$ の差を最小化する. これは, 画像 x と $x0$ は, 同じ画像がもとになっていることから, 得られる出力は同じものであることが最適であるという考え方によるものである. この最小化操作は, 出力間の差によって決定するため, ラベルの有無によらず行うことができ, 教師あり・なしデータの両方で行うことができる. 後述する CSD で使用されている手法であり, 詳細なアルゴリズムは節 3 で説明する.

3 CSD

Consistency-based Semi-supervised learning method for object Detection（CSD）[1] は, 半教師あり学習を用いた物体検出手法の 1 つである．これは, 物体検出におけるアノテーションコストの問題を解決する目的で提案された手法である．このモデルは, 教師あり物体検出モデルである SSD などをベースに半教師ありに拡張させる手法である．半教師あり学習を用いることで, 物体検出におけるアノテーションのコストを軽減しつつ, 教師なしデータを最大限活用し, 精度の向上を実現している．

3.1 CSD の仕組み

CSD は, SSD などの Single Shot の検出器と R-CNN 系の Two Shot の検出器の両方に適用できる学習アルゴリズムである．Consistency regularization の正則化の手法を用いて, 教師なしデータでの学習を可能とする．

3.1.1 検出アルゴリズム

CSD の検出アルゴリズムは, ベースとなる検出器によって 2 つのカテゴリに分類される．R-CNN 系のような Two Shot 検出器では, RPN を使用し, 物体を含む可能性が高い領域候補提案に対してのみ検出を行う．SSD のような Single Shot 検出器では, RPN を使用しないため, 特徴マップのすべての領域候補提案に対して分類と位置推定を行う．

3.2 CSD のネットワーク

CSD のネットワークは, ベースとなる検出器が Single Shot か Two Shot かによって異なる．

3.3 CSD における学習の処理概要

CSD における学習では以下の処理が繰り返される．

1. 訓練データの読み込み
2. ネットワークでの推論
3. 出力と反転データの出力のマッチング (ラベル付きデータとラベルなしデータの両方)
4. 正解座標とボックスのマッチング (ラベル付きデータのみ)
5. 損失関数の計算

CSD の学習の詳細について説明する．CSD の学習は, ラベル付きの教師ありデータと

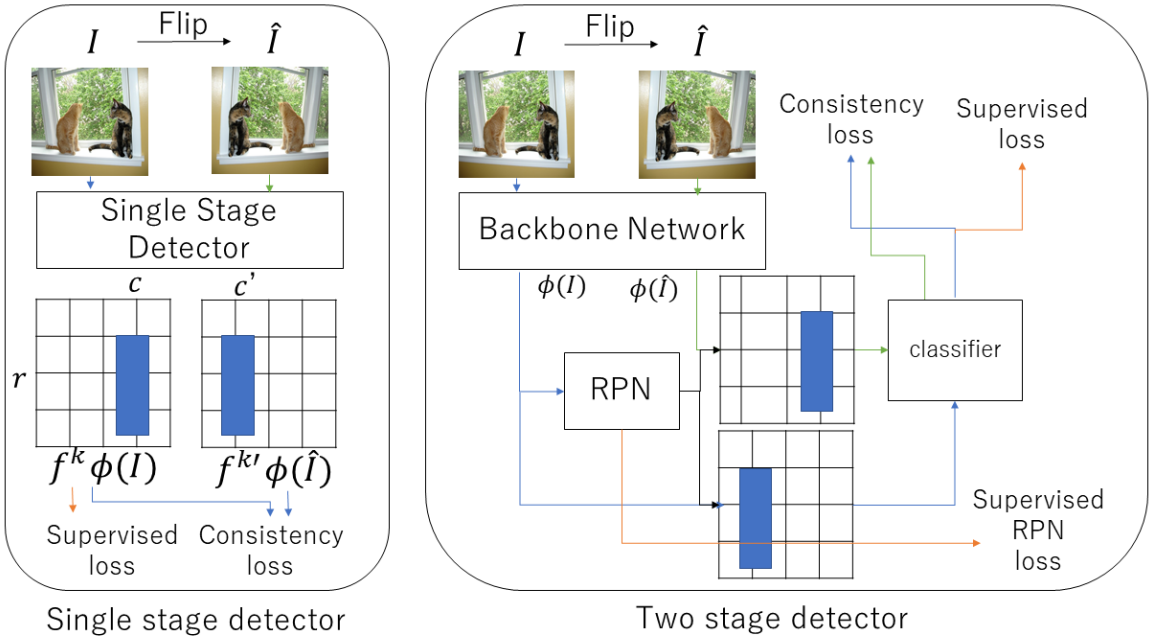

図 5　CSD のネットワーク構造 (Jisoo Jeong ら [1])

ラベルなしの教師なしデータを用いて行う．CSD の動作は, ベースとなる検出器が Single Shot か Two Shot かによって異なる．入力データは, 元の画像 I とそれを水平方向に反転した複製画像 $\hat{I}$ を用意する．入力 I と $\hat{I}$ を用いて, Consistency regularization の手法を行う．これらの出力であるバウンディングボックスは同じクラスである必要があり, 位置も一致している必要がある．トレーニングに用いるデータは, ラベルありとラベルなしの両方が含まれており, Consistency regularization の処理はどちらの場合でも行われる．

次に, クラス分類における Consistency 損失を説明する．画像 (I) の入力で, p 番目のピラミッド, r 番目の行, c 番目の列, d 番目のデフォルトボックスに対する Softmax 演算後の出力クラス確率ベクトルを $f_{cls}^{p,r,c,d}(I)$ で示す．また, 反転画像 $\hat{I}$ に対する確率ベクトルを $f_{cls}^{p,r,c',d}(\hat{I})$ 示し, これらの予測は同等である必要がある．

位置 (p,r,c,d) を k, 水平方向に反転した位置 (p,r,c',d) を k' と表記したとき,Consistency 損失は次の式 4 で表される．

$$l_{con\_cls}(f_{cls}^{k}(I), f_{cls}^{k'}(\hat{I})) = JS(f_{cls}^{k}(I), f_{cls}^{k'}(\hat{I})) \tag{4}$$

このとき, JS はイェンセン-シャノン発散 (JSD) であり, 分布間の類似度を表すもので, 二つの確率ベクトルがどの程度似ているかを示すものである．

また, 分類の全体的な Consistency 損失は, 全てのボックスのペアからの損失値の平均で表される (式 5).

$$L_{con\_c} = E_k[l_{con\_cls}(f_{cls}^k(I), f_{cls}^{k'}(\hat{I}))] \tag{5}$$

次に, 位置推定における Consistency 損失を説明する. 画像 I に対する位置推定の結果を, $f_{loc}^k(I)$ で表され, $[\Delta cx, \Delta cy, \Delta w, \Delta h]$ で構成される. $\Delta cx, \Delta cy$ は, デフォルトボックスからの中心座標のずれ, $\Delta w, \Delta h$ は, デフォルトボックスの幅, 高さのずれを表す.

クラス分類の結果である $f_{cls}^k(I), f_{cls}^{k'}(\hat{I})$ のペアとは異なり, 位置推定の結果 $f_{loc}^k(I)$ と $f_{loc}^k(I)$ は互いに水平方向に反転させた画像であるため, 同等になるように変更を行う. 具体的には, 以下の修正を行う.

$$\begin{aligned} \Delta cx^k &\Leftrightarrow -\Delta \hat{cx}^{k'} \\ \Delta cy^k, \Delta w^k, \Delta h^k &\Leftrightarrow \Delta \hat{cy}^{k'}, \Delta \hat{w}^{k'}, \Delta \hat{h}^{k'} \end{aligned} \tag{6}$$

1 組のバウンディング ボックスに使用される位置の Consistency 損失は次のようになる.

$$\begin{aligned} l_{con\_loc}(f_{loc}^k(I), f_{loc}^{k'}(\hat{I})) =& \frac{1}{4}(||\Delta cx^k - (-\Delta \hat{cx}^{k'})||^2 + ||\Delta cy^k - \Delta \hat{cy}^{k'}||^2 \\ &+ ||\Delta w^k - \Delta \hat{w}^{k'}||^2 + ||\Delta h^k - \Delta \hat{h}^{k'}||^2) \end{aligned} \tag{7}$$

バウンディングボックスの各ペアの位置損失と, 全体の Consistency 損失は, クラス分類の場合と同じように計算される.

$$L_{con\_l} = E_k[l_{con\_loc}(f_{loc}^k(I), f_{loc}^{k'}(\hat{I}))] \tag{8}$$

そして, クラス分類と位置推定を合わせた全体の Consistency 損失は, 以下のようになる.

$$L_{con} = L_{con\_c} + L_{con\_l} \tag{9}$$

また, ラベルありデータを入力した場合, 元の検出器と同様に損失を計算する.

最終的に, 損失 L は, 上記の Consistency 損失 L_{con} に, 元の検出器のクラス分類損失 L_c と位置推定損失 L_l を加えて, 以下で表される.

$$L = L_c + L_l + w(t) \cdot L_{con} \tag{10}$$

4　提案手法

本論文では, 十分に学習された物体検出モデルを利用してそのモデルに新たなクラスを追加したものを作成したいような場合を想定する．そして, 4 クラスの十分な量のデータで学習されたモデルをベースにして, 1 クラスを追加した 5 クラスのモデルを, 低いコストのデータを用いて作成する方法を提案する．

まず, 十分な 4 クラスのラベル付きデータを用いて, SSD の学習を行う．そして, 十分な精度の 4 クラスの学習済みモデルを作成する．

次に作成した SSD の事前学習モデルを用いて,5 クラスの検出器を作成する．これは,5 クラスの少数のラベル付きデータとラベルなしデータを使用し, SSD ベースの CSD での学習を行う．

本実験では, ラベルありデータの量を固定し, ラベルなしデータの量を変化させた場合でそれぞれ学習を行い, 比較する．

4.1　実装方法

SSD の学習及び, 学習済み SSD をベースにした CSD の学習を実装する手順を説明する．

なお, SSD 及び CSD モデルのソースコードは以下のリンク先から取得したもの利用し, 一部変更することで実装した．このコードは pytorch で書かれており, 実際に推論を行う Demo 部分は, Jupyter Notebook で作成されたノートブック形式のプログラムである．

```
https://github.com/soo89/ISD-SSD
Copyright (c) 2017 Max deGroot, Ellis Brown
Released under the MIT license
```

5　実験

5.1　実験設定

5.1.1　実験の流れ

実験の前段階として CSD の精度の検証を行った.

(bicycle, car, cat, house) の 4 クラスで SSD,CSD のモデルを作成し, それぞれの精度を比較した. 4 クラスのラベル付きデータを用いて SSD モデルを学習し, ラベル付きデータとラベルなしデータを用いて CSD のモデルを学習した. さらに, ラベルなしデータの量を様々な値に変更し, 学習を行った. そして, その違いによってどの程度精度に変化が生じるかを検証した.

以下のデータの組み合わせで,4 クラスの SSD-CSD の学習を行った.

- ラベルあり 1348 枚／ラベルなし 0 枚
- ラベルあり 1348 枚／ラベルなし 200 枚
- ラベルあり 1348 枚／ラベルなし 2261 枚
- ラベルあり 1348 枚／ラベルなし 3174 枚

CSD で用いるデータセットは, ラベルありデータとラベルなしデータを一つにデータセットに集約したうえで, ランダムにシャッフルし入力する. また, バッチサイズは 8 を設定し, ミニバッチ内のラベルありとラベルなしの各データ量は, 全体の割合と同じになるように設定した.

次に, 本論文の提案手法の実験を行った. 事前学習済みの 4 クラスモデルとして, 前段階で作成した (bicycle,car, cat, house) の 4 クラスのデータ (ラベルあり 1348 枚／ラベルなし 0 枚) を学習させた SSD のモデルを利用した.

そして,(bicycle, car, cat, house, train) の 5 クラスで SSD, CSD のモデルを作成し, それぞれの精度を比較した.

以下のデータの組み合わせで,5 クラスの SSD, CSD の学習を行った.

- ラベルあり 250 枚／ラベルなし 0 枚
- ラベルあり 250 枚／ラベルなし 250 枚
- ラベルあり 250 枚／ラベルなし 500 枚
- ラベルあり 250 枚／ラベルなし 1000 枚
- ラベルあり 250 枚／ラベルなし 2000 枚

5.1.2 評価指標

物体検出の出力は, 物体の種類を表すクラスと, 物体の位置を示すボックスである. クラスの推論結果は, 学習するクラスの一覧から選択されるため, 正解不正解が明確に定まる. それに対して, ボックスの推論結果は出力ボックスに実際のボックスとの多少の誤差が生じるため, どのくらいの誤差であれば, 正解か不正解かが曖昧である. 物体検出の正誤評価は, この様な性質上, 一般的に IoU(Intersection over Union) を用いて正誤判定が行われる.

IoU は, 重複面積を意味しており, 実際のボックスと検出したボックスがどれくらい重なっているかを表す指標で, 以下の式で表される.

$$IoU = \frac{AreaofOverlap}{AreaofUnion}$$

$$AreaofOverlap = 2 \text{ つのボックスの共通部分}$$

$$AreaofUnion = 2 \text{ つのボックスの全体部分}$$

IoU が 1 のとき, 二つのボックスは完全に重なっている. また,IoU が 0 のとき, 二つのボックスに重なってる部分はない. SSD では IoU の閾値を 0.5 として, それを超えたものを正解 (positive) としている.

物体検出の精度の評価は mAP(mean Average Precision) といった評価値が用いられる. m AP は, [3] にある方法で導出される.

まず,mAP の導出における予備知識として, TP(True Positive), FP(False Positive) , FN(False Negative), TN(True Negative) の指標を示す.

TP (True Positive)　実際にボックスに対して, 検出されたボックスが存在し, 重なっている.($IoU > 0.5$).

FP(False Positive)　実際にボックスに対して, 検出されたボックスが存在するが, 外れた位置にある.($IoU < 0.5 \bigcap IoU \neq 0$)

FN(False Negative)　実際はボックスがあるが, 検出したボックスはない.

TN(True Negative)　検出したボックスも, 実際のボックスもない.

次に,$Precision$ と $Recall$ を示す. $Precision$ は, 検出した物体がどれくらい正しいのかを表している. $Recall$ は, 検出すべき実際の物体がどれくらい検出できたかを表している. これらは,TP,FP,FN によって, 次のように表される.

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

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

以上を踏まえたうえで, mAP の導出する.

まず, 推論で得られた結果をクラス別に確信度の高いボックス順に並べる. そして, 表 1 にあるように上から順にボックスを見ていき, 検出されたボックスを現在のボックスまでとしたうえでの $Precision$ と $Recall$ を求める.

このとき,$Precision$ と $Recall$ は次の式で表される.

$$Precision = \frac{\text{現在まで正解したボックス数}}{\text{現在まで見た検出ボックス数}}$$

$$Recall = \frac{\text{現在まで正解したボックス数}}{\text{すべての実際のボックス数}}$$

ボックスの正解不正解は, IoU によって決定する. IoU のしきい値は, データセットによって定まっている.

> pascalVOC データセットの場合, IoU しきい値は IoU0.50 または IoU0.75 を用いる.
>
> COCO データセットの場合, 0.50:0.95(0.50〜0.95 を 0.05 刻みで mAP 導出し, その平均をとる) また, 0.50 と 0.75 を用いた mAP を使用する.

なお, 同じ正解を示すボックスが複数検出された場合は, 確信度が一番高いものを正解とし,

表 1　$Precision$ と $Recall$ の導出例

順位	クラス確信度 (%)	正誤	$Precision$	$Recall$
1	100	正解	$1/1 = 1$	$1/10 = 0.1$
2	99	正解	$2/2 = 1$	$2/10 = 0.2$
3	96	不正解	$2/3 = 0.67$	$2/10 = 0.2$
4	92	不正解	$2/4 = 0.5$	$2/10 = 0.2$
5	89	不正解	$2/5 = 0.4$	$2/10 = 0.2$
6	88	正解	$3/6 = 0.5$	$3/10 = 0.3$
7	80	正解	$4/7 = 0.57$	$4/10 = 0.4$
8	77	正解	$5/8 = 0.63$	$5/10 = 0.5$
⋮	⋮	⋮	⋮	⋮

求めた $Precision$ と $Recall$ をもとに, 縦軸を $Precision$, 横軸を $Recall$ とした,PR 曲線 (Precision-Recall Curve) を描く. すると, AP は次の式で表される. ただし,p は $Precision$ の変数, r は $Recall$ の変数とする.

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

なお, これは PR 曲線の下側の面積を表している. 次に, 積分を簡単にするため, グラフを以下の式で凸凹を均す.

$$p_{interp}(r) = \max_{\tilde{r}:\tilde{r}\geq r} p(\tilde{r})$$

最後に積分を求めるが, 補間適合率の点を取ることで, 簡略化して計算する. 補間適合率の点の取り方は, データセットによって標準値が決まっている.

pascalVOC データセットの場合,$r = (0.0, 0.1, \cdots, 1.0)$ の計 11 箇所の点を取る.
COCO データセットの場合,$r = (0.00, 0.01, \cdots, 1.00)$ の計 101 箇所の点を取る.

そして, その点についてのみ計算を行う. これにより, 縦 × 横の掛け算で簡単に求めることができる.

すると, 以下の式で AP が求められる.

$$AP = \frac{1}{11} \sum_{r\in(0,0.1,\cdots,1)} p_{interp}(r)$$

つまり,

$$AP = \frac{1}{11} * (p_{interp}(0) + p_{interp}(0.1) + \cdots + p_{interp}(1.0))$$

これをすべてのクラスに対して導出し, 全てのクラスの AP の平均が mAP となる.

mAP は, IoU しきい値を添えて表記されることがあり, 例えば IoU か 0.50 をしきい値とした場合, 以下のように表すことがある.

- mAP@0.50
- $mAP_{0.50}$
- mAP_{50}
- mAP50
- $mAP(IoU = 0.50)$

本実験では PascalVOC データセットを用いるため, IoU しきい値を 0.50 とし, 補間適合率の点は 11 点を取る方法で導出した mAP を用いてモデルを評価する.

5.2 実験データ

今回使用したデータセットは,PASCAL Visual Object Classes(VOC) 2007, 2012 である. このデータセットは, 物体検出で一般的に使用されているものである. 画像ファイル (.jpg) と画像内の物体の位置とクラスを表すアノテーションを示すファイル (.xml) の組で構成されている. 以下の URL からダウンロードできる.

`http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtrainval_06-Nov-2007.tar`
`http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtest_06-Nov-2007.tar`

http://host.robots.ox.ac.uk/pascal/VOC/voc2012/VOCtrainval_11-May-2012.tar

VOC は計 20 種類のクラスのデータからなり, {cat, dog} といった動物クラス, {car, ship} といった乗り物クラス, {chair, tvmonitor} といった家具クラスを含む.

- VOC2007 は, trainval のデータ数 5011 枚と test のデータ数 4952 枚で構成される.
- VOC2012 は, trainval のデータ数 11540 枚で構成される.

VOC を用いた物体検出の標準は, これらを合わせた, trainval を 16551 枚, test を 4952 枚で行われるのは一般的である.

本実験では, その中から次のデータを取り出した.

- 事前学習に用いるラベルあり訓練データ： VOC2007 から,4 クラス (bicycle, car, cat, horse) のラベル付きデータ 1,348 個.
- 事前学習に用いるラベルなし訓練データ： VOC2012 から,4 クラス (bicycle, car, cat, horse) のラベルなしデータ 3174 個.
- 5 クラスの学習に用いる少量のラベルあり訓練データ : 5 クラス (bicycle, car, cat, horse, train) のラベルありデータ計 250 個.
- 5 クラスの学習に用いるラベルあり訓練データ : 5 クラス (bicycle, car, cat, horse, train) のラベルありデータ計 1596 個.
- 5 クラスの学習に用いるラベルなし訓練データ : 5 クラス (bicycle, car, cat, horse, train) のラベルありデータ計 3526 個.
- テストデータ (訓練データと被り無し) : 5 クラス (bicycle, car, cat, horse, train) のデータ計 1787 個.

物体検出では, ベースネットワークとして画像認識の学習済みモデルを特徴抽出器として利用するのが一般的である. 本実験ではベースネットワークとして, vgg16 を使用した. vgg16 は, 画像認識の学習済みモデルで, 20,000 クラスの分類ができる, 高性能なモデルである. ネットワーク構造は, 畳み込み 13 層と全結合 3 層の計 16 層で構成されたニューラルネットワークである.

5.3 実験結果

CSD の検証のための 4 クラスモデルの実験結果を表 2 に, 提案手法の実験結果を表 3 に示す.

各表の bicycle, car, cat, horse, train の項目は, クラス別の AP を示している. すべてのクラスの AP を平均したものが mAP となる.

表 2 のに示すのは,CSD の精度を検証する実験の結果である. ラベルなしデータを 200

枚に設定した場合の mAP が最も高い結果であった.

この結果から, 使用するラベルなしデータの量は, 多すぎても少なすぎても行けず, ある程度適した割合のデータを用意する必要があるといえる.

また, ラベルあり 1348 枚／ラベルなし 3174 枚の学習については, ラベルなしデータの量が一定以上になると, 損失が大きく増加してしまい, 学習がうまくいかないという結果であった. 繰り返し学習を行ったが, 学習はできなかった. これについては, 節 6 で考察する.

表 2　4 クラスデータセットの実験結果

メソッド	ラベルあり	ラベルなし	mAP	bicycle	car	cat	horse
SSD	1348	0	82.2	81.6	84.2	79.5	83.5
SSD-CSD	1348	200	83.3	82.0	84.7	80.6	84.9
SSD-CSD	1348	2261	82.4	81.0	85.7	80.0	82.8

表 3　5 クラスデーセットの実験結果

メソッド	ラベルあり	ラベルなし	mAP	bicycle	car	cat	horse	train
SSD	250	0	72.0	72.6	73.6	71.0	77.2	65.7
SSD	1596	0	83.3	81.8	85.7	79.4	84.4	85.4
SSD-CSD	250	0	77.2	79.3	76.9	79.9	79.7	70.0
SSD-CSD	250	250	72.7	75.5	69.9	72.3	75.6	70.4
SSD-CSD	250	500	70.7	74.2	68.0	69.5	75.6	67.0
SSD-CSD	250	1000	69.9	75.3	67.9	68.1	73.7	64.7

表 3 に示すのは, 提案手法の実験結果である. mAP を比較すると, 同量のラベル付きデータを用いて学習した SSD より精度がよくなったのは, ラベルなしデータを 0 枚, 250 枚で学習したものであり, 後者が最大であった.

また, ラベルあり 250 枚／ラベルなし 2000 枚の学習については, 4 クラスで行った学習と同様に, ラベルなしデータの量が一定以上になると, 損失が大きく増加してしまい, 学習がうまくいかないという結果であった.

また,35 ページ以降の付録資料, 図 6 から図 15 に, 各モデルの epoch ごとの mAP の推移を示す. 必要に応じてクラス別やモデル別に比較したものを示す.

なお, 図のタイトルの括弧書きの値は, 学習で使用したデータの割合を (ラベル付きデータ：ラベルなしデータ) で表記したものである.

6　考察

6.1　データセット

本実験では, データセットの量を変更して学習を行い, それらの違いを検証した．CSD の特徴として, ラベルなしデータが増えるにつれて学習速度が低下することは, Jeong ら [1] の研究で考察されている．本実験で行った, 4 クラスの CSD の検証においても, その傾向はみられた．本節ではそれらの他に, データセットに起因すると考えられる問題点について考察する．

6.1.1　学習初期の精度

CSD で追加クラスの学習を行った実験では, ラベルなしデータの量が増えるにつれて, 学習初期時点で精度が大きく低下した．mAP70 以上の学習済みのモデルをベースに学習したにもかかわらず, 特にラベルなし 1000 枚を用いた場合では, mAP が 0 近くまで低下することが観測された．この原因としては, ラベルなしデータが増えることで,Supervised 損失に比べ Consistency 損失が学習に寄与する割合が増加したことが考えられる．それにより, ただし, 事前学習モデルの効果が完全に失われたのではないと思われる．CSD 学習速度は, ラベルなし画像が増えるに連れて低下する傾向があるが, 追加ラベルの学習ではそのデメリットが軽減され, 最初こそ mAP は低下するが学習速度に関してはラベルなしデータが多い方が優れた結果が得られている．これは, 事前学習の特徴抽出部分が色濃く残っていることに寄与したものであると考えられる．

6.1.2　ラベルなしデータの割合

本実験の結果より, CSD でラベルなしデータが 0 枚の場合が最大の mAP であった．これは, CSD の Consistency regularization の手法が作用したためと推測できる．教師あり学習の Consistency regularization 手法は本来パフォーマンスの低下に繋がるとされていたが, クラスを追加したいという本実験の設定では効果的であったと考えられる．

また,CSD ではラベルなしデータを増やすほど学習速度が低下するため, 今回学習したエポック数では学習が足りていない可能性が考えられる. 本実験以上に時間をかけて学習を進めていくと, ラベルなしデータを用いる場合の精度が向上する可能性がある．

本実験で, ラベルなしデータを一定以上増やしていくと, 学習がうまくいかないことが確認された．今回扱ったデータセットに関しては, 事前学習済みのパラメータをベースに学習すると, ラベルなしのデータの割合が, 全体の 80% を超えたあたりでその現象が見られた．この原因としては, 前節で述べたものと同じで, ラベルなしデータが増えることで, Supervised 損失に比べ Consistency 損失が学習に寄与する割合が増加したことが考えられる．ラベルなしのデータ割合を増やしていくと, 学習初期での精度が下がる傾向が見ら

れたが, 学習不可能な状態はその現象の延長であると考えられる. これを解消する方法としては, Consistency 損失の重みに関するパラメータ設定を小さくすることが考えられる.

この結果から得られる知見としては, CSD を用いる場合はデータセットの設定を繊細に選択する必要があり, また, Consistency 損失のパラメータ設定とラベルなしデータの割合の相関についても考慮する必要がある, ということである. また, その最適な設定は実験に基づき得られるものであり, 用いるデータセットごとにも異なることが考えられる. そのため, パラメータを得るための実験コストを考えると, データ作成のコストを抑えるという本来の目的から逸脱してしまう問題がある.

6.2 CSD

今回利用した CSD では, 精度をわずかに向上させるようなテクニックがいくつか行われている [1]. それによって, 教師あり学習の結果よりも高い精度を実現している. しかし, 一般にテクニックを利用した複雑なモデルは, 汎用性を失う傾向があり, これらはトレードオフの関係になっている. 今回行ったクラス追加のケースでは, それらのテクニックがマイナスに作用した可能性が考えられる. そのため, それらのテクニックを用いた場合と用いなかった場合の比較も検証する必要がある.

7　結論

本研究では, 学習済みの 4 クラスの物体検出の SSD モデルを利用し,1 クラス追加した CSD モデルを低コストの訓練データで作成した. そして, ラベルなしデータの量を変えて学習したモデルを mAP で比較した. それにより, ラベル付きデータを用いない CSD が最大の mAP であるという結果が得られた. 結論としては, 提案手法は不安定且つ繊細なパラメータ調整が必要であり, パフォーマンスを向上させる点においては有効なこともあるが, コストの面で考えると実用的かつ汎用的な方法であるとはいえない.

謝辞

最後に, 本論文を作成するにあたり, 指導, 助言を頂いた, 指導教官の新納浩幸教授に心より感謝申し上げます. また, 研究を進めるにあたり協力していただいた, 日立オートモティブシステムズ株式会社の伊藤浩朗様, 土井宏治様に, 深く感謝申し上げます.

参考文献

[1] Jisoo Jeong, Seungeui Lee, Jeesoo Kim, Nojun Kwak, "Consistency-based Semi-supervised Learning for Object detection"In Advances in Neural Information Processing Systems, pages 10758 - 10767, 2019.

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

[3] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, "The PASCAL Visual Object Classes (VOC) Challenge" Int J Comput Vis (2010) 88: 303 - 338 Andrew Zisserman

[4] H. Bilen and A. Vedaldi. Weakly supervised deep detection networks. In Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit. (CVPR), pages 2846 - 2854, 2016.

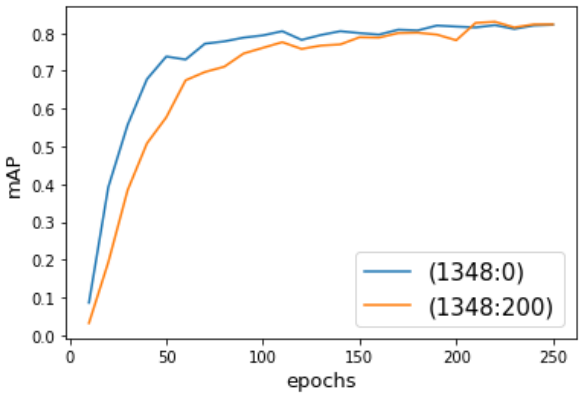

図 6　mAP の推移：CSD 4 クラス

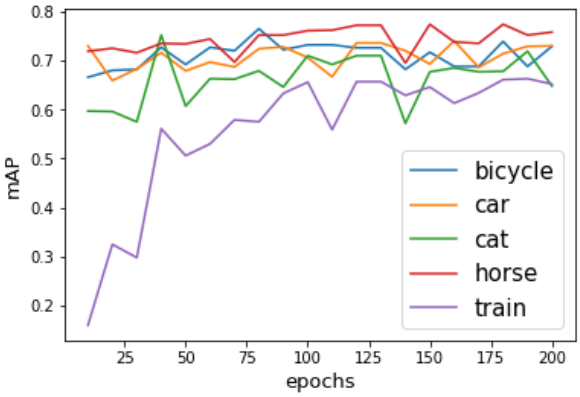

図 7　クラス別 AP の推移：SSD 5 クラス (250:0)

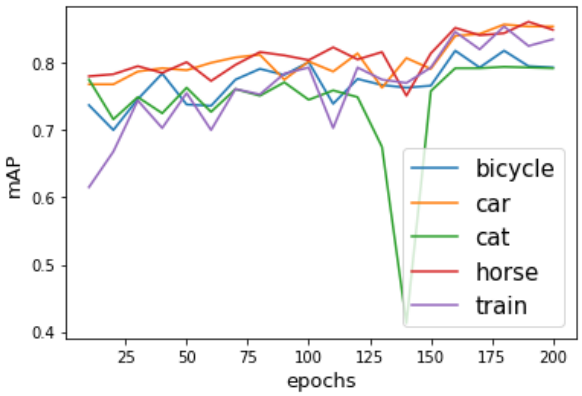

図 8　クラス別 AP の推移：SSD 5 クラス (1596:0)

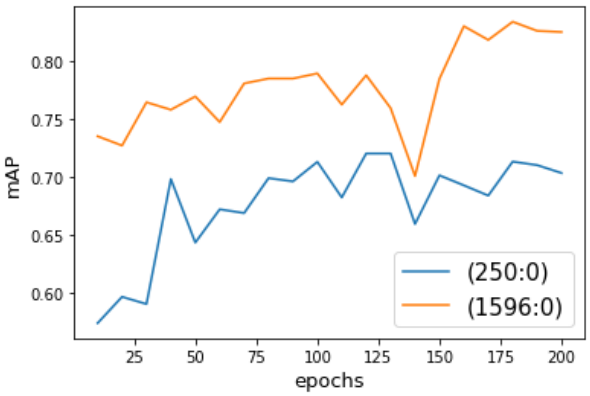

図 9　mAP の推移：SSD 5 クラス

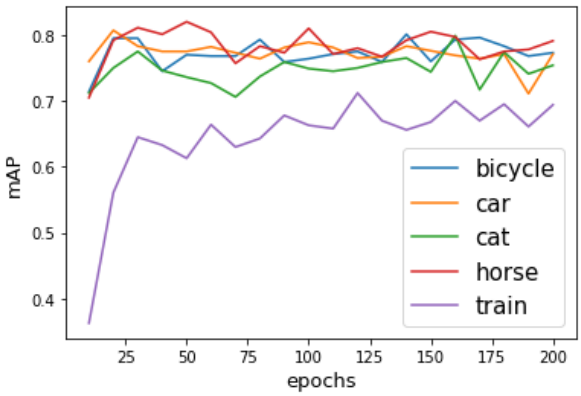

図 10　クラス別 AP の推移：CSD 5 クラス (250:0)

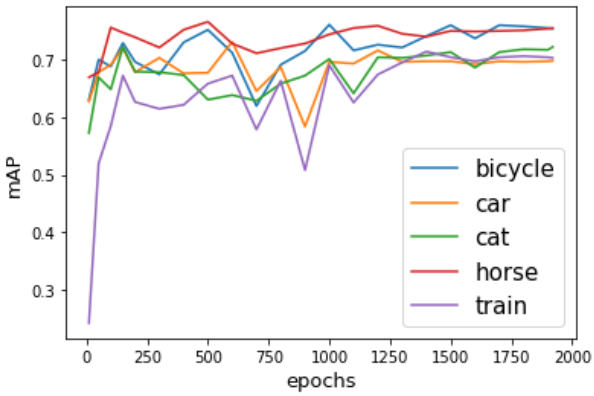

図 11　クラス別 AP の推移：CSD 5 クラス (250:250)

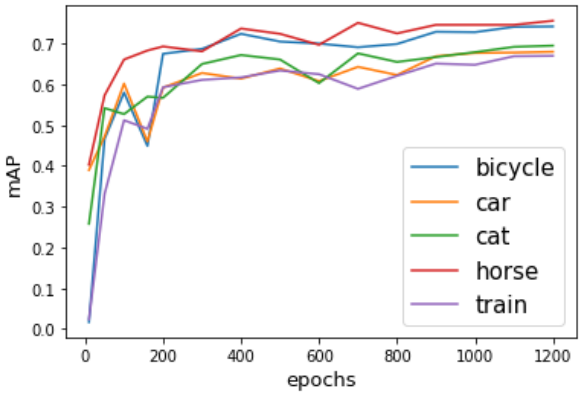

図 12　クラス別 AP の推移：CSD 5 クラス (250:500)

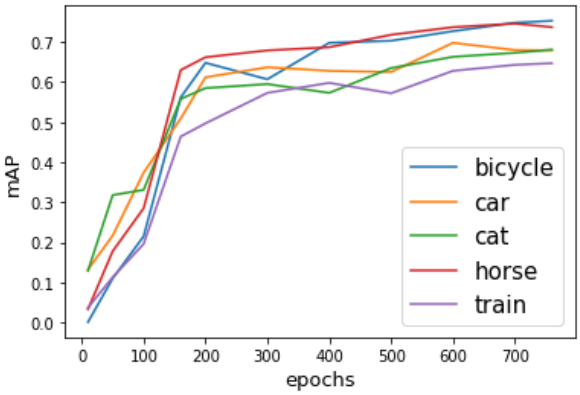

図 13　クラス別 AP の推移：CSD 5 クラス (250:1000)

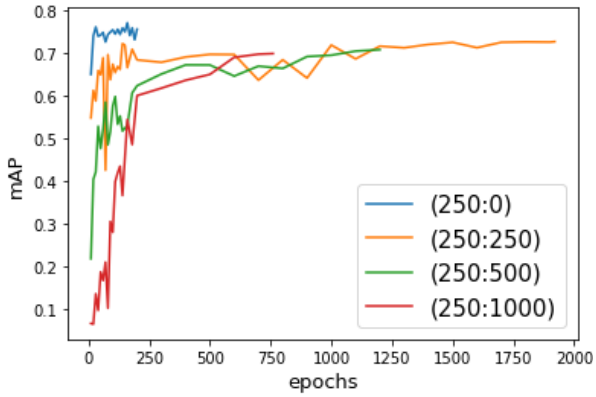

図 14　mAP の推移：CSD 5 クラス

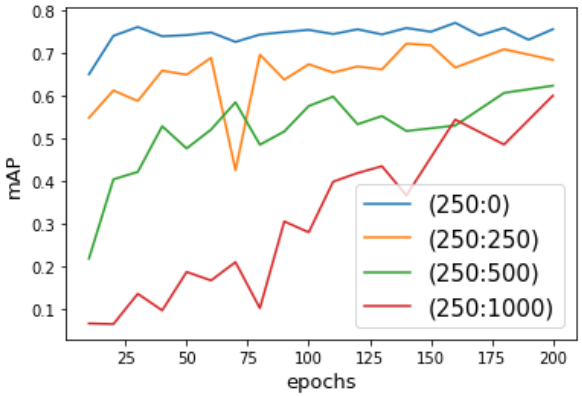

図 15　学習序盤の mAP の推移：CSD 5 クラス