東京都立産業技術高等専門学校

ものづくり工学科　情報システム工学コース

卒業研究論文

機械学習を用いた転倒検知システムの開発

SDN を用いた動的ルーティング制御の一検討

堀川 風花

2026 年 2 月

目 次

図 目 次

表 目 次

1　序論

大学キャンパスや企業内 LAN ではネットワーク利用者の増加に伴い，ネットワークトラフィックが急増している．その結果，混雑による遅延やパケットロスが発生し，通信品質の低下が課題となっている．

Software Defined Networking（SDN）は，ネットワーク機器の制御プレーンとデータプレーンを分離することで，集中管理と柔軟な制御を可能とする技術である．SDN を活用すれば動的ルーティングによりトラフィック最適化が期待できる．

1.1　背景

従来のネットワークでは静的ルーティングに依存しており，トラフィック集中時の動的な回避が困難である．

〇〇

1.2　研究目的

〇〇

〇〇

本研究の目的は、○○○

2　関連研究・技術

2.1　姿勢推定を用いた転倒検知

AlphaPose、OpenPose、BlazePose、OpenPifPaf、HRNet を用いてフレームデータから人間の骨格を抽出し、機械学習モデルを構築した. 具体的には RNN、LSTM、ST-GCN、Vision Transformers などの機械学習モデルを作成し, 転倒イベントを識別した．さらに、高度な姿勢推定フレームワークである ViTPose と Trans-Pose を導入することで、調査範囲を拡大した．実験は 3 つの異なるデータセットを用いて、転倒検知技術の全体的な評価をしていた. 様々なポーズ抽出器について詳細な分析を行い, 各手法で生成された明確なスケルトンポイント（骨格位置）を機械学習モデルを学習させた.

ここではフレームに対する人の各関節の絶対位置であるスケルトン情報の前処理やスライディングウィンドウに触れたい．この論文では前処理として正規化手法や絶対位置からの相対位置変換を紹介している.

姿勢検出手法は、環境ノイズ, カメラの位置、屋内物体からの干渉の影響を受けやすい. そのため、これらの前処理により、特に複雑な屋内環境において, 姿勢検出の精度と信頼性を大幅に向上させる.

最小-最大正規化を適用してデータを区間 [0, 1] にスケーリングし、モデルの収束を早めてモデルの精度を向上させます．新しいスケールデータは式 に従って計算される.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \in [0, 1] \tag{2.1}$$

元のデータセットの値は X で表され、最小値は X_{min} で表され、最大値は X_{max} で表される.

ただし、データセットには欠損値を含むインスタンスが含まれている．全てのグループに欠損データがあるわけではないが、値が 0 である場合は X_{min} も 0 であることを示す．この元のデータ分布の変化により、不確実な属性が生じる．したがって、式および式 (7) で概説した正規化手法の使用を提案する.

$$x_c^f = \frac{W}{2} y_c^f = \frac{H}{2} \tag{2.2}$$

元の座標位置を, x_n^f と y_n^f 座標系における同等の位置に変換し、比較性を高める．値が 0 の点は欠損点と指定され、計算から除外される．欠損点が存在する場合、その位置には計算された中心点が代入され

る．変位が必要な距離（x_{dis}, y_{dis}）は式 (8) および式 (9) によって定義される．

$$x^f_{dis} = x^f_8 - x^f_c y^f_{dis} = y^f_8 - y^f_c \tag{2.3}$$

変位は第 8 関節点 (x^f_8, y^f_8) の中心点へ、基準点は股関節中心円周 (x^f_c,y^f_c) となる．変位値が負の場合、物体は右方向へ移動する．変位値がゼロより大きい場合、オブジェクトは左に移動する．相対位置への変位を計算するための接合点の更新座標（rx^f_n, ry^f_n）は以下のように与えられる：

$$rx^f_n = (x^f_n - x^f_{dis}) ry^f_n = (y^f_n - y^f_{dis}) \tag{2.4}$$

相対位置正規化手法には二つの明確な利点がある．第一に、元のデータ分布を保持すること．第二に、欠損データの同時計算を不要とすること．さらにこの処理は冗長な特徴量を排除する．特筆すべきは、横方向移動や近接性といった人間の移動特性が骨格位置を安定的に維持する点である．この特性によりモデルに連続的な動きが与えられ、学習プロセスに有益であることが実証されている．

転倒と非転倒事象の判別において、特定の骨格関節は識別特性を持たないとみなされる．処理の効率化と明瞭化のため、これらの冗長な関節は除去される．最終的に保持される訓練用関節セットには、鼻、肩、肘、手首、首、股関節、膝、足首が含まれる．

BlazePose は主に 2D 画像特徴量に依存して骨格を識別するため、これらの特徴量が欠落すると不安定性や予測誤差が生じます．人体が隠蔽されたり形状が不明瞭だったり、上下肢が覆われている場合、骨格構築時にデータ損失や不正確さが発生する可能性があります．提案手法では、線形補間を用いて欠損データを補正することでこの問題に対処する．十分なデータセットを確保するため、100 枚の画像を集約し、欠損値のない初期データを用いて後続画像の欠損部分を補完する．この補間プロセスにより、元のデータセットから予測データ値を導出する．ただし、初期関節点の数が不十分な場合、予測値と実際の欠損値に大きな乖離が生じる可能性がある．図 10 は検出されなかった骨格関節の事例を示す．線形補間では、2 つの既知データ点を用いてそれらの間の傾きを計算し、Y（式 (12)）のおおよその値を推定する．

$$Y = Y_0 + \frac{Y_1 - Y_0}{X_1 - Y_0}(X - X_0) \tag{2.5}$$

推定値と実測値の差異は、遷移点が完全な状態では統計的に有意な差を示さない．RP 正規化前、データセットには 22.38%の重要点の欠落が認められた．RP 正規化後、この欠落率は 13.4%に減少した．さ

らに、補間処理と RP 正規化を経て、欠落する重要点の発生率は全データセットのわずか 1.8%にまで低下した.

動画から機械学習モデルに入力するための時系列データを作成するため、まず UR-Fall、UPFall、または Le2i データセットから個々の人体骨格を取得する．次のステップでは、AlphaPose、OpenPose、BlazePose、HRnet、OpenPifPaf といった多様な姿勢抽出器を用いて人間の姿勢を推定します．これらの姿勢抽出器は重要なキーポイントの特定に役立つ．その後、これらのキーポイントを主要特徴量として利用するモデルを構築しする．トレーニング過程では、転倒または非転倒活動を示すシーケンスの予測を可能にするため、様々な機械学習モデルを活用する．この手法により、多様な姿勢推定技術と機械学習モデルを活用し、様々なデータセットにおける転倒検出の探索・分析が可能となる．フレームシーケンスを決定するため、スライディングウィンドウが連続する骨格特徴を結合する．この手法では、ウィンドウサイズは動画レート（fps）にウィンドウサイズ（$swl = fps\times$ 秒）を乗じた画像シーケンスをカバーするように設定される．例として、1 秒間隔で 25 フレーム/秒の動画の場合、各ウィンドウは 25 フレーム（swl = 25）をカバーする．各ウィンドウは前のウィンドウの右隣のフレームに位置し、動画の最終フレームがウィンドウでカバーされるまでこの配置が続く．スライディングウィンドウの総数は、以下の式を用いて決定できる：

$$sw_m = FrameVideo - swl + 1 \tag{2.6}$$

本手法における総分岐数は、動画フレーム処理に「スライディングウィンドウ」と呼ばれる概念を用い、sw で表される．動画の総フレーム数は FrameVideo で表される．ここで FrameVideo は動画内の総フレーム数を表す．各スライディングウィンドウは、個人の動作を捕捉する連続したフレーム群を包含する．この構成により、各スライディングウィンドウには様々な活動に従事する個人の骨格データが含まれることが保証される．(1,swl, n) 形式の各フレームは、n 個の識別された骨格で構成され、ここで n は検出された骨格の数を示す．スライディングウィンドウ手法に関して、我々のアプローチでは、スライディングウィンドウ内の各フレームセットが分類推論を受ける．通常、各スライディングウィンドウから単一の分類結果が生成される．1 つの分類結果に寄与するフレーム数（または HPE モデルが返す関節キーポイントのセット数）は、ウィンドウサイズ（swl）と一致します．swl は、時間（秒）に動画フレームレート（fps）を乗じた値です．したがって、推論レートが 25fps の場合、各スライディングウィンドウのフレームセットには 25 フレーム、または単一の分類出力生成に寄与する関節キーポイントのセットが含まれる．

これらのデータ前処理やスライディングウィンドウによってどの姿勢推定技術と機械学習モデルの組み合わせでもモデルの精度は 90%を超える結果となった.

この研究は本研究と似ている部分もあるが、この研究は検知できる人数は 1 人を想定しており、複数人での利用は想定していなかった.

そのため、本研究ではこれらの知見を踏まえ，スケルトンデータの正規化や時系列データに適した機械学習モデルの選定を行い、複数人でも対応できるシステムを作成する．また、本研究では数字的なモデルの精度の値のみではなく、実際の利用を想定した際の処理遅延やモデルの転倒検知の精度や偽陰性率を評価したい.

2.2　３次元姿勢推定技術による転倒検知

この研究では、異なるデータセット間における 3D 人体姿勢推定の汎化性能の低下に着目している.

RCB 画像からのの 3D 人体姿勢推定アの研究は多く存在するが、通常、被写体、姿勢、カメラ、照明など、多くの要因に関して多様性が制限されたデータセットで評価されているため、実際の任意の条件（「in-the-wild」）で動作させると性能が低下すると考えられる．カメラの視点、被写体、およびデータセット間の汎化を大幅に改善するスケール正規化手法を提案している．追加の実験では、カメラの増減、複数のデータセットを用いた学習、提案された解剖学に基づく姿勢検証ステップの適用、そして 3D 姿勢推定の基盤として OpenPose を使用することによる効果を調査している．実験結果は、データの前処理（異なる姿勢推定による関節検知の結果の差を緩和するため)、スケール正規化、そして仮想カメラの拡張が、モデルの汎化を大幅に改善する上で有用であることを示している.

スケール正規化では具体的に推定した人のスケルトンポイントの正規化や相対位置変換を行っている．計算手順は以下の通りである．各ポーズサンプルの絶対関節座標 p_i は、ローカル座標系の原点にあるヒップの中心点 p_0 に対するスケルトンの相対的な関節位置に基づいて個別に正規化された．スケール s は、原点と N 個の関節位置すべてのユークリッド距離の平均を計算することで定量化した.

$$s = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{p}_i - \mathbf{p}_0\| \tag{2.7}$$

その後、すべての関節位置座標をスケールで割ってスケルトンのサイズを変更し、正規化されたスケールを 1 にした．正規化された関節位置 p_i は次のように計算された.

「肩～足首の距離」など特定の部位基準ではないので、学術的に厳密な「身長正規化」とは少し異な

るが、人物の大きさに依存しない形に正規化をした.

$$\hat{\mathbf{p}}_i = \frac{1}{s}(\mathbf{p}_i - \mathbf{p}_0) \tag{2.8}$$

本研究では転倒検知を目的としており，タスク自体は異なるものの，動画から推定した関節座標を時系列データとして学習する点で共通する課題を有する.

そのため，本研究においても Rapczyński らの知見を参考に，人物および撮影条件に依存しない特徴抽出を実現するため，関節座標に対してスケール正規化を適用する．前処理として，各フレームで推定された人体のスケルトン関節位置に正規化と相対値変換が存在している.

2.3 姿勢推定を用いた転倒検知

3　原理

3.1　システム全体構成

本研究では，姿勢推定により得られた関節座標の時系列情報を用いて，転倒動作を分類する時系列分類モデルを構築した。

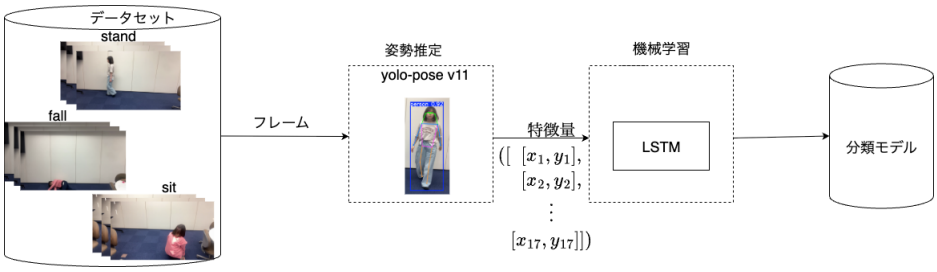

図 3.1 分類モデル作成の流れ

分類モデルに学習させるデータとして以下の 3 つの状態に分類される 1〜3 秒の動画データを各クラス 700 個ずつ、合計 2100 個の動画データを作成した。動画は私以外の人にも協力してもらい、異なる方向や異なる角度から撮影し、なるべく多様なデータになるようにした。姿勢推定には yolo-pose を採用し、動画データの各フレームごとからスケルトン情報を取得する。

- sit（床に座っている、しゃがみ込んでいる状態）
- stand（歩行している状態）
- fall（転倒している状態）

このスケルトン情報を入力としてあらかじめ付与したラベルを教師データとして学習させたモデルを用いて，リアルタイム転倒検知システムを構築した。推論時には入力映像に対して YOLOv8-Pose を用

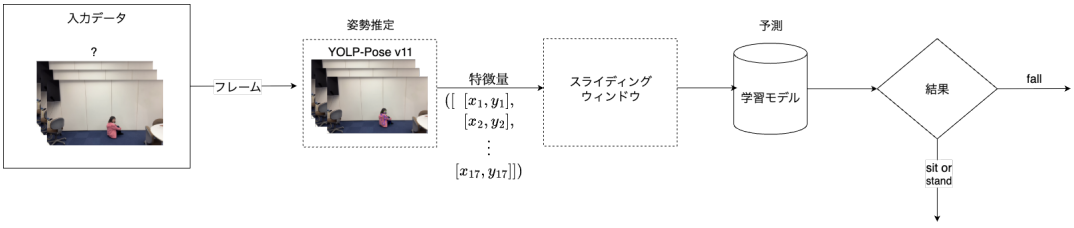

図 3.2 システムの流れ

いて人物検出および姿勢推定を行い，YOLO-Tracking により人物 ID を時系列で追跡する。各人物ごとに関節座標系列を蓄積し，30 フレーム分の特徴量が揃った時点で LSTM モデルによる動作分類を行う。

3.1.1 スライディングウィンドウ技術

モデル学習時には長さの異なる動画を入力するためにフレームの数を 30 フレームに固定した。各動画を等間隔で 30 フレームずつサンプリングし、一つの時系列データとして扱った。また、姿勢推定では 13 個のキーポイントを取得し、各キーポイントは x 座標と y 座標で表すので、各フレームは 26 次元の入力となる。つまりモデルの入力は 26×30 次元の時系列特徴量になる。

推論時にはスライディング法を用いて特徴量を構成する。ウィンドウサイズは W=30 フレームとし、各人物について直近 30 フレーム分の特徴量を FIFO（First-In First-Out）形式で保持する。新たなフレームが入力されるたびに最新の特徴量を追加し、ウィンドウが満たされた状態では最も古いフレームを破棄することで、連続した動作情報を維持する。ウィンドウが 30 フレームに達した時点で、その時系列特徴を LSTM モデルに入力し、stand、sit、fall といった行動クラスを推定する。推定結果は短時間の揺らぎを抑制するため，直近 5 フレームの予測結果に対して多数決による平滑化処理を行った。

3.1.2 LSTM を用いた転倒検知

分類モデルには，時系列データの依存関係を学習可能な LSTM（Long Short-Term Memory）ネットワークを用いた。ネットワーク構成は，LSTM 層（128 ユニット）と LSTM 層（64 ユニット）を直列に配置し，過学習抑制のために Dropout 層（0.3）を各層後に挿入した。LSTM から得られた特徴ベクトルに対し、ReLU 活性化関数を用いた全結合層（32 ユニット）を適用し、非線形変換を行う。最終層には Softmax 関数を用いた全結合層を配置し、fall,stand，sit の 3 クラスに対する確率分布を出力する。

学習には，交差エントロピー損失関数と Adam 最適化手法を用いた。エポック数は 30，バッチサイズは 16 とした。

3.1.3 データ前処理

フレームから yolo-pose を使って取得したスケルトン情報は環境ノイズや遮蔽や位置によって影響を受けやすい。これモデルに入力する前に複数の前処理を挟むことで複雑な屋内環境においてモデルの予測精度と信頼性を大幅に向上させことができる。

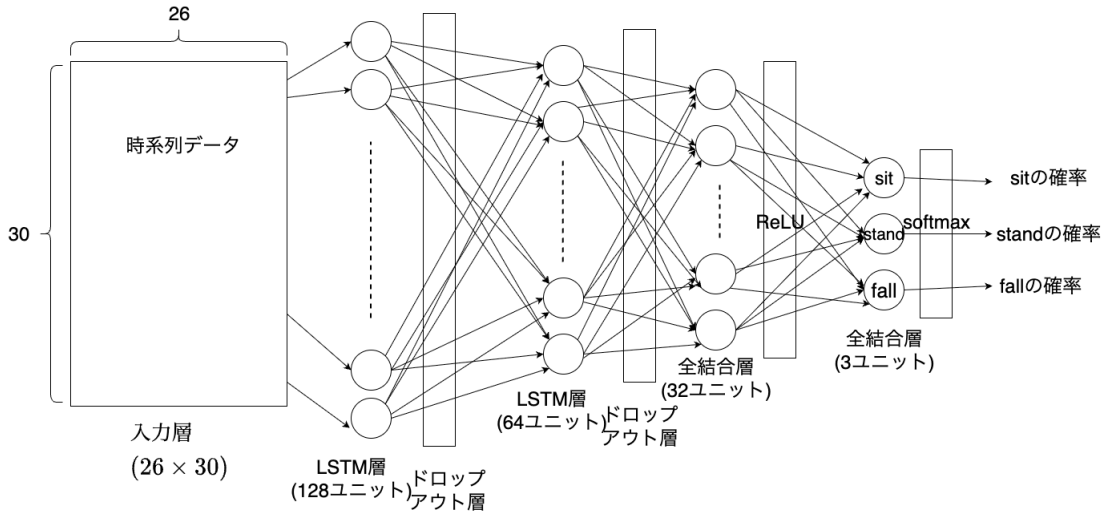

図 3.3 LSTM モデルの構成

キーポイントの抽出

yolo-pose ではキーポイントとして、鼻、目、耳、肩、肘、手首、股関節、膝、足首の合計 17 種（鼻以外の部位は左右を持つ）が含まれる。転倒と非転倒事象の判別において、特定の骨格関節は識別特性を持たないとみなされる。処理の効率化と明瞭化のため、これらの関節は除去する。また、このシステムの運用は介護施設や障害者施設を想定しており、そのような場所ではマスクをつけている利用者が多い。そのため、鼻のキーポイントの検出が難しい状況が想像できるので鼻のキーポイントは除去し、代わりに左右の目の中央のキーポイントという新たな特徴量を作成する。左右の目の中央のキーポイントはシンプルに左右の目の平均をとって算出した。最終的に保持される訓練用関節セットには、目の中央、肩、肘、手首、股関節、膝、足首の合計 13 種のキーポイントが含まれる。

線形補完

姿勢推定に基づく関節座標は，遮蔽や検出失敗により一部フレームで欠損が生じることがある。本研究では，時系列特徴量の連続性を維持するため，学習時に検出に失敗した関節点に対して線形補完を用いた欠損補完処理を行った。

まず，各動画から等間隔に抽出した 30 フレームについて関節座標を取得し，すべての関節点が検出されなかったフレームを欠損フレームとして判定した。欠損が存在する場合には，検出が成功した前後フレームの関節座標を用い，各関節点の x および y 座標について時間軸方向の線形補間を行った。フレーム時刻 t における関節 j の座標成分 $p_j^{(k)}(t)$ ($k \in \{x, y\}$) が欠損している場合，直前および直後の有効フレーム $t_1, t_2 (t_1 < t < t_2)$ を用いて，以下の線形補間式により推定した。

$$p_j^{(k)}(t) = p_j^{(k)}(t_1) + \frac{t - t_1}{t_2 - t_1} \left(p_j^{(k)}(t_2) - p_j^{(k)}(t_1) \right) \tag{3.1}$$

なお，動画の先頭および末尾において有効フレームが存在しない場合には，最初または最後に検出された関節座標を用いた定数補完を行い，外挿による不安定な推定を防止した。

相対位置正規化

最小値-最大値正規化を適用してデータを区間 [0, 1] にスケーリングし、人物の位置による大小関係や人物の身長の差による精度の差を無くすことができると考える。

$$s = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{p}_i - \mathbf{p}_0\| \tag{3.2}$$

その後、すべての関節位置座標をスケールで割ってスケルトンのサイズを変更し、正規化されたスケールを 1 にした．正規化された関節位置 $\hat{\mathbf{p}}_i$ は次のように計算された．

「肩〜足首の距離」など特定の部位基準ではないので、学術的に厳密な「身長正規化」とは少し異なるが、人物の大きさに依存しない形に正規化をした．

$$\hat{\mathbf{p}}_i = \frac{1}{s}(\mathbf{p}_i - \mathbf{p}_0) \tag{3.3}$$

また、各キーポイントを体の中心からの相対位置に変換することで、人物のフレームの中の位置によらす正しく分類できるようにする。体の中心を左右の股関節の中央の位置 $[p^x_{HipCenter}, p^y_{HipCenter}]$ とする。すると、相対位置は式 3.4 のように計算できる。

$$\begin{aligned} rp_n^x &= (p_n^x - p_{HipCenter}^x) \\ rp_n^y &= (p_n^y - p_{HipCenter}^y) \end{aligned} \tag{3.4}$$

3.2 畳み込みニューラルネットワーク（CNN：Convolutional Neural Network）

画像認識において頻繁に使われる畳み込みニューラルネットワーク（CNN：Convolutional Neural Network）は畳み込み層やプーリング層、全結合層などから構成される．そのため、まずニューロンモ

デルや畳み込み演算や特徴抽出などの用語や新たな概念、処理方法について説明する．

ニューロンモデル

ニューロンモデルとは脳の神経細胞（ニューロン：neuron）を模した数理モデルである．これによって、データの予測や簡単な物体検出が可能になる．

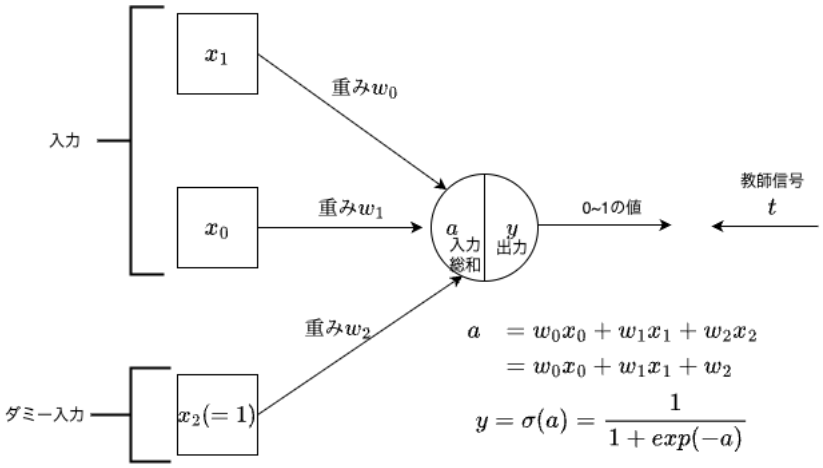

図 3.4 ニューロンモデル

1 つのニューロンに 2 つの入力 x_0 と x_1 が入力信号として入ってくる（図 3.4）．各入力と各入力に対する重み w_0 と w_1 を掛け算し、合計をとり、最後に定数 w_2 を加えた値を入力総和 a とする．$x_2 = 1$ とすると、入力総和 a を式 3.5 のように表せる．

$$\begin{aligned} a &= w_0x_0 + w_1x_1 + w_2x_2 \\ &= \sum_{i=0}^{2} w_ix_i \end{aligned} \tag{3.5}$$

その後、この入力総和 a を活性化関数（この例ではシグモイド関数）に通した式 3.6 がニューロンの出力値 y となる．

$$y = \sigma(a) = \frac{1}{1+exp(-a)} \tag{3.6}$$

このように，1 つのニューロンは入力データ $\boldsymbol{x}$ に対して，重み付き和と活性化関数による変換を行い，出力値 y を生成する．この出力値 y は，入力データがあるクラスに属する確率や，数値を予測した結果として解釈される．

ニューロンモデルの学習方法

本研究で扱うニューロンモデルの学習は，入力データとそれに対応する正解ラベル（教師データ）が与えられる教師あり学習に基づいて行われる．学習時には、入力データ $\boldsymbol{X}$ をニューロンに入力し、ニューロンが出力した y（予測データ）と教師データ $\boldsymbol{T}$ の誤差が最小になるようにこの重み $\boldsymbol{W}$（以下、パラメータという）を調整していく．

また、ニューロンが出力した y（予測データ）と元の教師データの誤差を平均交差エントロピー誤差 $E(\boldsymbol{w})$ という（式 3.7）．この誤差関数は，予測結果 y_n が教師データ t_n に近づくほど小さな値をとる．

$$E(\boldsymbol{w}) = -\frac{1}{N}\log P(\boldsymbol{T}|\boldsymbol{X}) = -\frac{1}{N}\sum_{n=0}^{N-1} t_n log y_n + (1-t_n) log(1-y_n) \tag{3.7}$$

パラメータの偏微分（式 3.8）が 0 になったときのパラメータが平均交差エントロピー誤差が最小になるパラメータだと考える．

$$\frac{\partial E}{\partial w_i} = \frac{1}{N}\sum_{n=0}^{N-1}(y_n - t_n)x_{ni} \tag{3.8}$$

パラメータの調整には勾配降下法を使って行う（式 3.9）．誤差関数を各パラメータで偏微分することで，誤差が増加する方向を求め，その逆方向にパラメータを更新する方法である．この操作をすべての学習データに対して繰り返すことで，モデルは入力データと教師データの対応関係を学習する．α を学習率といい、一般的には 0.01 といった極めて小さい値となる．

$$w_i(\tau+1) = w_i(\tau) - \alpha\frac{\partial E}{\partial w_i} \tag{3.9}$$

活性化関数

活性化関数とはニューロンの入力総和から出力を決定するための関数である．活性化関数は複数の種類があり、用途によって使い分ける．ここでは活性化関数の中でも頻繁に使われるシグモイド関数とソフトマックス関数、ReUL 関数を説明したい．

- シグモイド関数

  負から生の実数をを 0 から 1 までの値に変換する関数である．実数を確率に変換する際に用いられる．

$$\sigma(x) = \frac{1}{1+exp(-x)} \tag{3.10}$$

- ソフトマックス関数

  シグモイド関数の多次元に拡張した関数であるため、多項分類器に使われることが多い.

$$h(x_k) = \frac{exp(x_l)}{\sum_{l=0}^{L} exp(x_l)} \tag{3.11}$$

- ReUL 関数

  入力が 0 以下なら 0 を、0 より大きければその値をそのまま出力する関数である．ニューラルネットワークの活性化関数で使われることが多い.

$$relu(x) = max(0, x) \tag{3.12}$$

ニューラルネットワーク

単一のんツーロンモデルでは単純な分類しか行えないが、複数のニューロンを層状に接続することで、より複雑な分類が可能になる．このニューロンモデルの集合体をニューラルネットワークという．ニューラルネットワークは主に入力層、中間層（隠れ層ともいう)、出力層に分かれる．図 3.5 は 2 次元の入力（x_2 はダミー入力のため数に含めない）を受け、３つのカテゴリーに分ける多項分類モデルである．入力層では観測データがそのまま入力され，中間層では入力データの特徴を抽出する処理が行われる．出力層では，中間層で得られた特徴量をもとに，各クラスに属する確率が計算される.

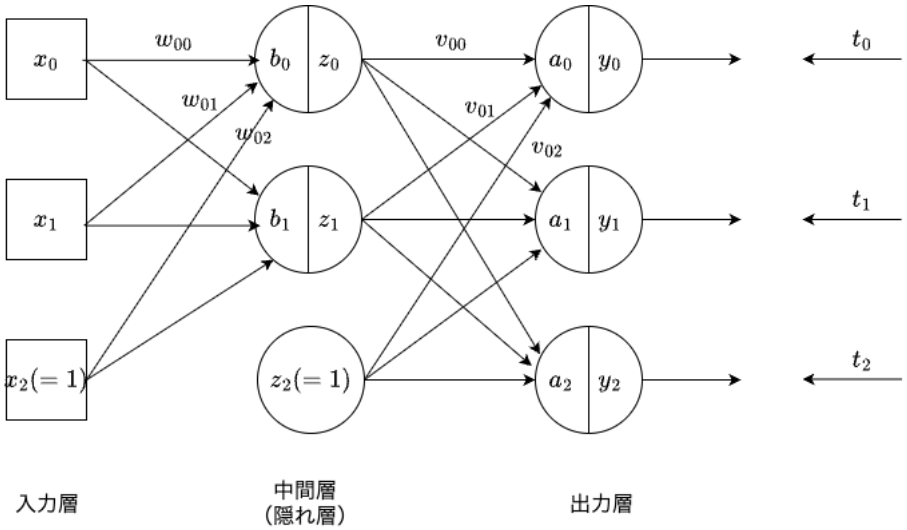

図 3.5 ニューラルネットワーク

入力層、中間層（隠れ層ともいう)、出力層の次元をそれぞれ D、M、K とすると、ネットワークは次

のように定義される．

$$\text{中間層の入力総和：} b_j = \sum_{i=0}^{D} w_{ji} x_i \tag{3.13}$$

$$\text{中間層の出力：} z_j = h(b_j) \tag{3.14}$$

$$\text{出力層の入力総和：} a_k = \sum_{j=0}^{M} v_{kj} z_j \tag{3.15}$$

$$\text{最終的な出力：} y_k = \frac{exp(a_k)}{\sum_{k=0}^{K-1} exp(a_k)} \tag{3.16}$$

最終的な出力（式 3.16）にはソフトマックス関数が使われる．
CNN では，中間層の一部を畳み込み層やプーリング層に置き換えることで，画像の空間的特徴を効率的に抽出する．

誤差逆伝搬法（バックプロパゲーション）

ニューラルネットワークに学習を行わせる方法として、主に誤差逆伝搬法（バックプロパゲーション）がある．誤差逆伝搬法は、平均交差エントロピー誤差を使って、入力方向とは逆向きに出力層の重みから入力層の重みへ順に更新していく方法である．しかし、誤差逆伝搬法は新しい最適化手法ではなく，ニューロンモデルの学習方法である勾配降下法をニューラルネットワークに適用させた計算手順である．実際に、勾配降下法と同じように平均交差エントロピー誤差関数をパラメータ w_{ji} と v_{ji} で偏微分させた関数を使ってパラメータを更新する．

一つのデータ n に対する交差エントロピー誤差 E_n を式 3.17 のように定義すると、平均交差エントロピー誤差は式 3.18 のように簡潔に表せる．

$$E_n(\boldsymbol{W}, \boldsymbol{V}) = -\sum_{k=0}^{K-1} t_{nk} log y_{nk} \tag{3.17}$$

$$E(\boldsymbol{W}, \boldsymbol{V}) = -\frac{1}{N} \sum_{n=0}^{N-1} \sum_{K=0}^{K-1} t_{nk} log y_{nk} = \frac{1}{N} \sum_{n=0}^{N-1} E_n(\boldsymbol{W}, \boldsymbol{V}) \tag{3.18}$$

ここでは、式の見やすさのため E_n を E とし、n を省力する．また、E を v_{kj} で偏微分した式が連鎖率より、式 3.19 である．δ_k は、出力層の各ニューロンにおける出力 y_k（予測データ）とそれに対する

教師データ t_k の誤差の大きさを表す量であり，誤差逆伝搬法における基本的な指標である．この値を用いることで，各重みが誤差に与える影響を定量的に評価できる．

$$\begin{aligned}\frac{\partial E}{\partial v_{kj}} &= \frac{\partial E}{\partial a_k}\frac{\partial a_k}{\partial v_{kj}} \\ &= (y_k - t_k)z_j \\ &= \delta_k z_j\end{aligned} \tag{3.19}$$

v_kj の更新規則はニューロンモデルと同じように式 3.20 になる．

$$v_{kj}(\tau+1) = v_{kj}(\tau) - \alpha\frac{\partial E}{\partial v_{kj}} \quad = v_{kj}(\tau) - \alpha\delta_k z_j \tag{3.20}$$

この意味は、v_{kj} の変化の大きさが最終的な誤差 δ_k と中間層の出力 z_j の積で決まるということである．もし、はじめのニューロンの出力 y_n がとそれに対する教師データ t_n が一致し、誤差 δ_k が 0 の場合は v_{kj} は変化しない．z_{kj} に対しては値が大きいほど、出力 y_k への寄与が大きいとして、v_{kj} の変化量をその分大きくするよう働く．

次に、E の w_{ji} による偏微分を求める．まず、連鎖率により分解することができる（式 3.21）．

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial b_j}\frac{\partial b_j}{\partial w_{ji}} \tag{3.21}$$

ここで、式 3.20 の類似性からとりあえず、$\frac{\partial E}{\partial b_j}$ を式 3.22 のように定義する．

$$\frac{\partial E}{\partial b_j} = \delta_j \tag{3.22}$$

すると、連鎖率により、E の w_{ji} による偏微分は式 3.23 である．

$$\begin{aligned}\frac{\partial E}{\partial w_{ji}} &= \frac{\partial E}{\partial b_j}\frac{\partial b_j}{\partial w_{ji}} \\ &= \delta_j x_i\end{aligned} \tag{3.23}$$

δ_j について中間層の活性化関数を $h()$ とすると、具体的に式 3.24 と表せる．

$$\delta_j = h'(b_j)\sum_{k=0}^{K-1} v_{kj}\delta_k \tag{3.24}$$

$h'(b_j)$ は活性化関数の微分を $\sum_{k=0}^{K-1} v_{kj}\delta_k$ は出力先の誤差を各重みで掛け合わせた総和を示している．つ

まり、δ_j は出力先での誤差を逆伝搬し、計算している（図 3.6）.

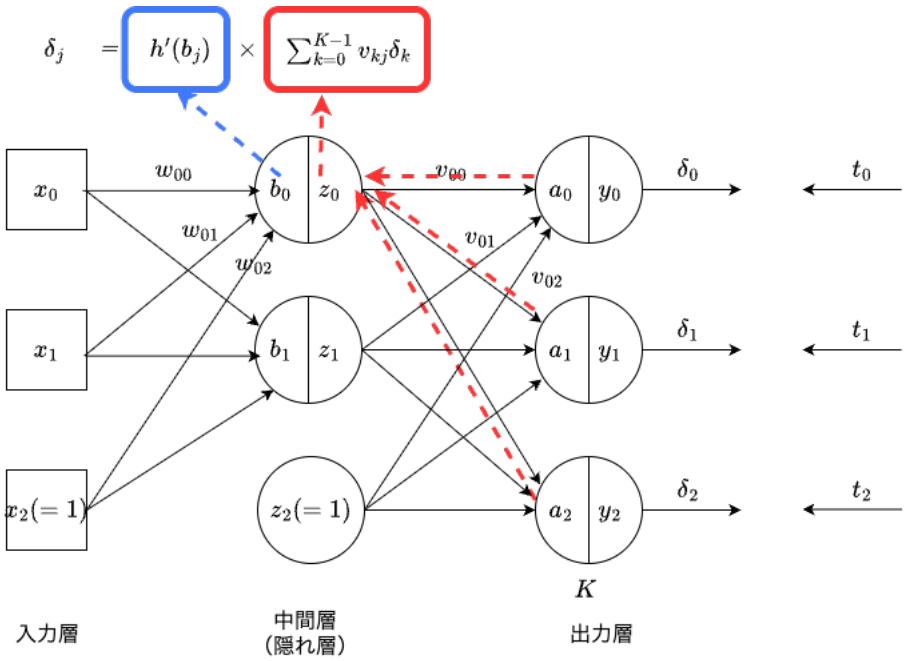

図 3.6 誤差の逆伝搬

w_{ji} は式 3.25 に従って更新される．この式から、w_{ji} も結合先の誤差 δ_j と結合元の出力 x_i の積によって変化していることがわかる．

$$w_{ji}(\tau+1) = w_{ji}(\tau) - \alpha\frac{\partial E}{\partial w_{ji}} = w_{ji}(\tau) - \alpha\delta_j x_i \tag{3.25}$$

以上の計算をまとめると，誤差逆伝搬法は以下の手順で学習を進める．

1. ネットワークに x を入力し、出力層のニューロンごとに y を出力する．
2. 出力層のニューロンごとに出力 y と教師データ t の間の誤差を計算する．
3. 出力層の誤差から中間層の誤差を計算する．
4. 中間層の重みを結合元の重みや結合先の誤差を使って更新する．
5. 1〜4 の作業を繰り消し、重みを改善していく．

3.2.1 カーネルと畳み込み演算

CNN を説明するにあたって、二次元配列による画像の表現方法について、いくつか触れておく．画像は画素（ピクセル）という小さい点によって格子状に構成されている．そして、その各画素に割り当

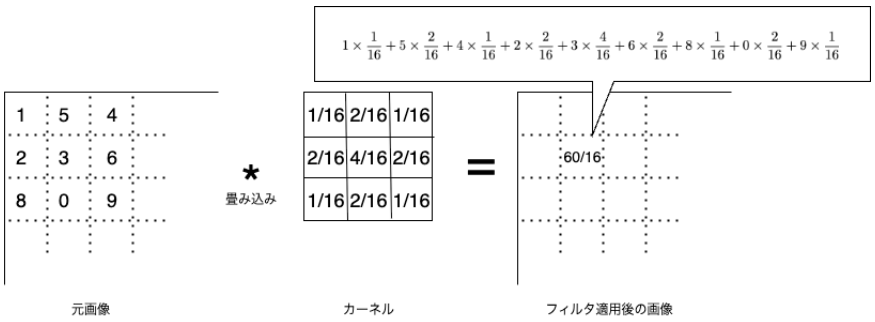

図 3.7 畳み込み演算

てられる色を表す値を画素値という．画像の左上を原点 $(0,0)$ として右方向と下方向をそれぞれ x 軸と y 軸の正方向とする．また、各画素値に割り当てられる色を表す軸をチャネルと呼ぶ．カラー画像は一般的に 3 チャネルの二次元配列でできており、3 つの値はそれぞれ赤、緑、青の色の強度を表している．具体的にカラー画像の配列の形は $(512, 512, 3)$((画像の高さ, 画像の幅, チャネル数))、カラー画像の画素値を $(196, 200, 165)$((赤, 緑, 青)) とされる．前節までで説明した全結合型のニューラルネットワークは, 入力次元が大きくなるとパラメータ数が急増するという課題がある．CNN ではこの問題に対処するため，局所的な領域に着目した平滑化フィルタや畳み込み演算を用いて特徴抽出を行う．

平滑化フィルタとは画像の各画素値を周りの画素値を考慮した上で更新する方法である．これを行うことで画像のノイズ除去が実現できる．平滑化フィルタを行う際は、ある画素値に対してどのように周辺の画素値を収集するか決める数学的な行列または配列であるカーネルを用いる．また、カーネルには目的によって異なる行列や配列を用いる．カーネルを用意したら、カーネルと画像との間で各画素値に対して、図 3.7 のような畳み込み演算を行う．この処理を画像全体に施すことで、フィルタ処理が完了する．畳み込み演算は式 3.26 のように表せる．ここで、*は畳み込み演算の演算子を、$I_g(x,y)$ と I_o、K はフィルタ適用後の画像と元画像、カーネルを表し、W と H は画像の幅と高さを表している．

$$I_g(x,y) = K * I_o = \sum_{u=-[W/2]}^{[W/2]} \sum_{v=-[H/2]}^{[H/2]} K(u+[W/2], v+[H/2]) I_o(x+u, y+v) \tag{3.26}$$

3.2.2 特徴抽出

画像の畳み込み演算を使った特徴抽出について触れたい．まずは微分を使ったエッジ特徴量の抽出から始める．エッジ特徴量とは画素値の変化が大きい部分であり、これを行うことで物体の輪郭抽出が可能になる．垂直方向のエッジ特徴量と水平方向のエッジ特徴量は式 3.28 で計算できる．実際には、図

3.8 の（a）x 軸方向の 1 次微分カーネルと（b）y 軸方向の 1 次微分カーネルを使うことで計算できる．

$$E_x(x,y) = \left|\frac{\partial}{\partial x} I(x,y)\right| \simeq |I(x+1,y) - I(x,y)| \tag{3.27}$$

$$E_y(x,y) = \left|\frac{\partial}{\partial y} I(x,y)\right| \simeq |I(x,y+1) - I(x,y)| \tag{3.28}$$

最終的に垂直方向と水平方向のエッジ特徴量を合わせて式 3.29 の一次微分のエッジ特徴量を算出する．

$$E(x,y) = \sqrt{E_x(x,y)^2 + E_y(x,y)^2} \tag{3.29}$$

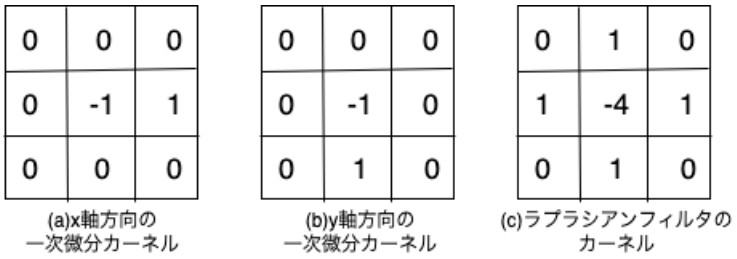

図 3.8 特徴抽出用カーネル

1 次微分値をさらに微分した 2 次微分を計算することで、輝度の変化が大きい部分を抽出することが可能である．座標 (x,y) における 2 次微分を行ったエッジ特徴量 $E_L(x,y)$ は式 3.30 になる． 2 次微分の特徴抽出は図 3.8 の（c）ラプラシアンフィルタのカーネルを使うことで計算できる．

$$\begin{aligned}
E_L(x,y) &= \delta I(x,y) \\
&= \frac{\partial^2}{\partial x^2} I(x,y) + \frac{\partial^2}{\partial y^2} I(x,y) \\
&\simeq \frac{\partial}{\partial x} I(x+1,y) - I(x,y) + \frac{\partial}{\partial y} I(x,y+1) - I(x,y) \\
&\simeq I(x+1,y) - I(x,y) - I(x,y) - I(x-1,y) + I(x,y+1) - I(x,y) - I(x,y) - I(x,y-1) \\
&= I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) - 4I(x,y)
\end{aligned} \tag{3.30}$$

このようにカーネルと畳み込み演算を使った処理によって画像から特徴量を抽出することができる．この性質は CNN に利用されている．CNN では，このような特徴抽出を一つのカーネルだけで行うのではなく，複数の異なるカーネルを同時に用いる．各カーネルは異なるパターン（エッジ，角，テクスチャなど）に反応するため，同一の入力画像に対して複数種類の特徴マップが生成される．例えば，垂直方向のエッジを強調するカーネル，水平方向のエッジを強調するカーネル，斜め方向のエッジを検出するカーネルを同時に適用することで，画像中の輪郭構造を捉えることが可能となる．

3.2.3 畳み込み層

畳み込み層では畳み込み演算を基本とする処理になる．畳み込み層では出力チャネル数、カーネルの幅と高さ及びストライドはハイパーパラメータとなり、カーネルはパラメータとなり、学習により決定される．また、CNN では x 軸と y 軸に加えて、チャネル方向にも畳み込みを行う．入力画像が RGB の 3 チャネルの場合、チャネルごとに異なるフィルタを用意し、畳み込み演算した結果を全部加算し、その結果を出力マップとする．そのため、出力される特徴マップの数は入力のチャネル数に直接影響されない．また、1 つの特徴マップに複数のカーネルを用いることで，出力される特徴マップのチャネル数は増加する．このようにして，層を重ねるごとにより多様で抽象的な特徴表現が獲得される．

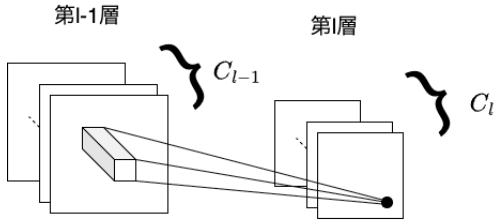

図 3.9 2 層間の畳み込み演算

図 3.9 のように第 l-1 層が C_{l-1} チャネルで構成されている場合、C_{l-1} 層のカーネル高さ, カーネル幅,C_{l-1} チャネルの直方体に畳み込み演算を行い、第 l 層のある一点が計算される．

特徴マップとは、畳み込み演算により生成される [高さ, 幅, チャネル数] の形の特徴量である．ストライドとは畳み込み演算を適用する間隔である．
図 3.10 の通り、ストライドが 2 の場合は、カーネルを x 方向、または y 方向に 2 画素ずつずらして畳み込み演算を行い、結果の値を特徴マップに代入していく．そのため、畳み込み演算適用前と畳み込み演算適用後の特徴マップは幅と高さが異なる．

畳み込み層では畳み込み演算のほかにパディングという処理も必要であることがある．パディングとは特徴マップの端をある適当な値で埋める処理を指す．特徴マップが削られることを防ぐことが目的である．また、パディングで埋める値は 0 であることが多い．

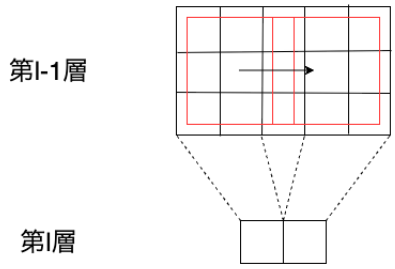

図 3.10 ストライド 2 の場合の畳み込み演算

3.2.4 プーリング層

プーリング層では、カーネルの値をまとめる処理を行う．プーリング処理には最大プーリングや平均プーリングなどが存在するが、今回は最大プーリングのみ紹介する．図 3.11 はカーネルを [3,3] とし、ストライドを 2 とした場合の最大プーリングの処理の様子である．カーネルと同じサイズの C_{l-1} 層のブロックの中で最大の値を次の C_l 層の画素の値とする．プーリングはカーネルを移動させながら行うが、チャネルごとに独立して行われるなど、畳み込み層とは異なる部分もある．

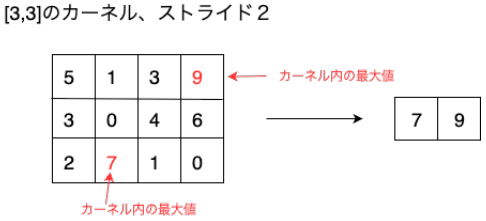

図 3.11 最大プーリング

CNN では、一般的に畳み込み層やプーリング層のストライドを用いて特徴マップを縮小しながら、特徴抽出を行う．特徴マップを縮小させることで計算量を減らすことができる．

3.2.5 全結合層

全結合層は，畳み込み層やプーリング層によって抽出された局所的な特徴を全体として統合し，最終的な判断を行う役割を担う．畳み込み層が画像中の局所的なパターンを検出するのに対し，全結合層はそれらの特徴の組み合わせから，ニューラルネットワークとして学習された重みにより、画像全体がどのクラスに属するかを判定する．全結合層に入力するためには、畳み込み層やプーリング層の出力である特徴マップを一次元に変換する必要があり、この処理を Flatten という．この仕組みは YOLO においても同様であり、畳み込み層で抽出された特徴量を全結合層を通じて人物の位置やクラス確率といった検出結果へ変換するために用いられている．

3.2.6　畳み込みニューラルネットワークの構成

図 3.12 は CNN の基本的な構成の例である．畳み込み層とプーリング層を相互に複数回適用しながらニューラルネットワークを構築し、出力層の前に全結合層を用いた．一回の畳み込み層ではなく、複数回の畳み込み層を適用することで、単純な低次元特徴マップから、より複数の特徴量を含む高次元特徴マップの検出が可能となる [1].

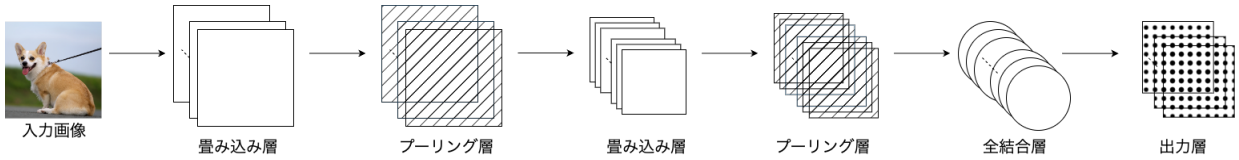

図 3.12 CNN の構成

3.3　YOLO による物体検知

本研究では物体検知手法として YOLO（You Only Look Once）を用いる。YOLO は高速かつ高精度な物体検出手法であり，現在までに v11 まで発展している。本節では，YOLO の基本的な原理を理解するため，最初に提案された YOLO v1 の検出アルゴリズムを中心に説明する。従来の物体検出では、分類器を利用して行ってが、YOLO は物体検出をバウンディングボックスとそれに関連するクラス確率への回帰問題として扱う。単一のニューラルネットワークが、1 回の評価で画像全体からバウンディングボックスとクラス確率を直接予測できる。各バウンディングボックスは、x、y、w、h、および信頼度の 5 つの予測値で構成される。物体を検知する原理としては, まず入力された画像を S × S 個の正方形（グリッドセル）に分割する．物体の中心がグリッドセルに収まる場合、そのグリッドセルがその物体の検出を担当する。画像分類における予測確率を信頼度として、モデルがそのバウンディングボックスに物体が含まれていると予想した確率 $Pr(Object)$ とボックスの位置の正確さを表した $IOU_{\frac{\text{truth}}{\text{pred}}}$ を掛け合わせた $Pr(Object) * IOU_{\frac{\text{truth}}{\text{pred}}}$ を定義する。ここで定義される信頼度は学習時の目標値であり，推論時にはネットワークが IOU を直接計算するのではなく，IOU を近似する値として信頼度を予測する。

各グリッドセルは B 個の、物体を囲んだ部分領域であるバウンディングボックスと各信頼度スコアを予測する．信頼度スコアは 0 に近いほど, 背景であることを示し,1 に近いほど, 物体であることを示す．各グリッドセルは、C 個の条件付きクラス確率 Pr(Classi—Object) も予測する。式 3.31 のように各バウンディングボックスは、「中心座標を含むセルに物体がある」という条件付きのクラス確率 $Pr(Class_i|Object)$ とボックスの正確さを示す信頼度 $Pr(Object) * IOU_{\frac{\text{truth}}{\text{pred}}}$ を掛け合わせて, クラス固有の信頼度スコアが

得られる.

$$Pr(Class_i|Object) * Pr(Object) * \\ IOU_{\frac{\text{truth}}{\text{pred}}} = Pr(Class_i) * IOU_{\frac{\text{truth}}{\text{pred}}} \tag{3.31}$$

このスコアは, そのクラスがボックスに出現する確率と, 予測されたボックスがオブジェクトにどれだけ適合するかの両方を表す.

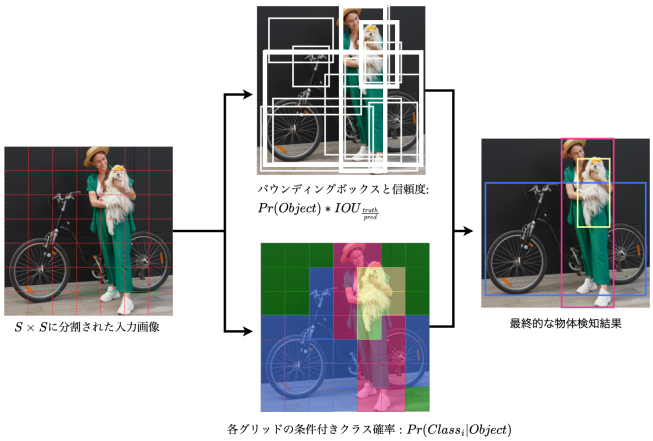

図 3.13 モデルの検出フロー

最後に Non-Maximum Suppression（NMS） を用いて, 重複するバウンディングボックスの中から, 最終的にクラス別に最も正確なバウンディングボックスを選び抜く [10]。

3.3.1 IoU（intersection over Unio）

物体検出における物体の位置は、物体の上端と下端に水平の線を、右端と左端に垂直な線を引いて得られる長方形である外接矩形（bounding box）で表される．矩形は矩形の左上端点を原点として右方向を x 軸の正方向、下方向を y 軸の正方向とする．矩形の大きさは幅を w、高さを h と表す．外接矩形の表現方法として 2 通りある．一つ目は中心座標 (x, y) と大きさ (w, h) で表す方法である．二つ目は矩形の左上端点の座標 (x_{min}, y_{min}) と右下端点の座標 (x_{max}, y_{max}) で表す方法である．

物体検出器が予測する外接矩形は位置に誤差が生じるため、必ずしも実際の物体の位置と重なるとは限らない．より物体の位置を正確にするため、実際の物体の位置と予測される外接矩形の当てはまり度合いに IoU（intersection over Union）という指標を使います．具体的には学習時に損失関数を計算したり、評価時に検出矩形が正しいか判定する用途で使われる．IoU の計算方法は式 3.32 の通りです.

$$IoU = \frac{\text{積集合の面積}}{\text{和集合の面積}} \tag{3.32}$$

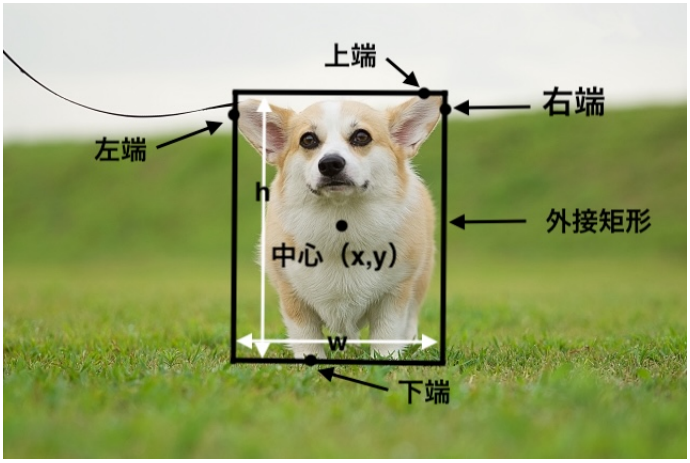

図 3.14 物体検出で使われる外接矩形

IoU は二つの矩形の和集合の面積に対しての積集合の面積の割合さと言える．IoU は 2 つの矩形の中心位置が近く、かつ大きさが等しいほど、大きくなる．

3.3.2 Non-Maximum Suppression（NMS）

Non-Maximum Suppression（NMS）とは１つの物体に対して複数のバウンディングボックスが存在するとき、最も正確なボックスを選び、その他の余計なボックスを取り除くためのアルゴリズムである。まず、すべてのボックスをスコアに基づいてソートする。最大スコアを持つボックス M が選択され、その他のボックスとの IOU が計算される。IOU が事前定義された閾値を超えた場合に、その二つのボックスは同じものを検知しているとして、重複しているボックスは削除される [2]。この作業をくり返していくことで、余計なボックスを取り除いていく。

3.3.3 ネットワーク設計

本研究では YOLO v1 のネットワーク構成を示す（図 3.15)。ネットワークは 24 層の畳み込み層と、それに続く 2 層の全結合層で構成されており、$-s-$ はストライド間隔を示している。ネットワークの最初の畳み込み層は画像から特徴を抽出し、全結合層は出力確率と座標を予測する。最終的な出力は、7 × 7 × 30 の多次元配列です。

3.4 YOLO-Pose による姿勢推定

YOLO-Pose は物体検出モデルである YOLO v5 を基盤とし, 画像における共同検出と 2D 複数人物ポーズ推定を行う手法である。具体的には、検出された人物ごとに複数のキーポイント（関節点）の位置と

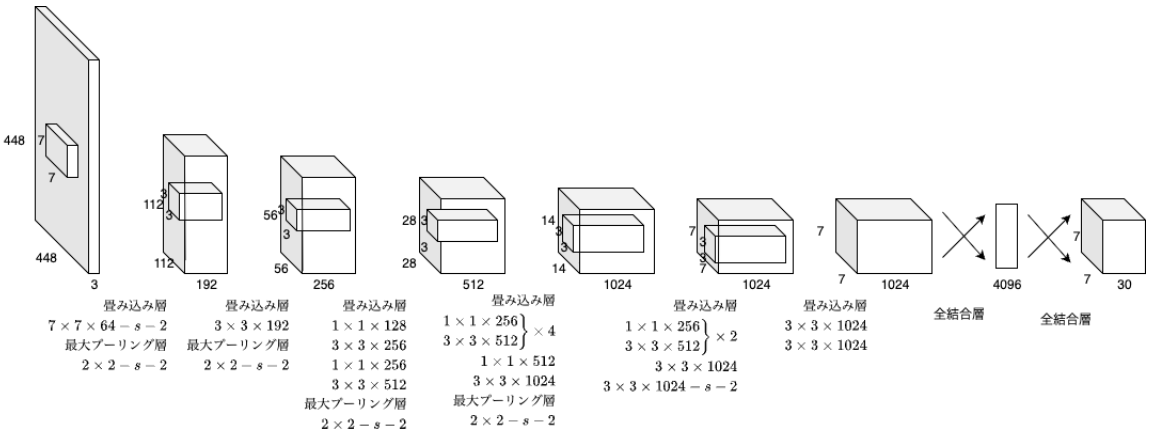

図 3.15 ネットワーク設計

信頼度を回帰的に推定する．そのため、YOLO-Pose は 2 つの異なる出力を持っており、一つはバウンディングボックスを予測し, もう一つは人の骨格を構成するキーポイントを推定する。人物のすべてのキーポイントをアンカーに関連付けられる。

一般的に既存の姿勢推定には、トップダウン方式とボトムアップ方式の 2 つに分類される。yolo-pose ではトップダウン方式とボトムアップ方式の両方の長所を取り入れている。

学習時には、正解となるバウンディングボックスに一致するアンカーボックスまたはアンカーポイントは、バウンディングボックスの位置とともに、2 次元姿勢を保存する。

異なる人物の 2 つの類似した関節は、空間的に互いに近接している可能性がある。従来のボトムアップ方式で用いられるヒートマップを使用すると、異なる人物の空間的に近い 2 つの類似した関節を区別することは困難である。

しかし、これら 2 人の人物を異なるアンカーと照合すれば、空間的に近接した類似のキーポイントを区別するのは容易になる。アンカーに関連付けられたキーポイントは既にグループ化されているため、さらにグループ化する必要はない。このように、トップダウン方式の人物単位の識別能力とボトムアップ方式の一定時間での処理という両方の利点を導入することで、一定の実行時間とシンプルな後処理を両立できる。

各人物には 17 個のキーポイントがあり、各キーポイントは座標と信頼度で識別される:$\{x, y, conf\}$。したがって、各アンカーについて、キーポイントヘッドは 51（17×3）個の要素を予測し、ボックスヘッドは 6 個の要素（x 座標、y 座標、幅、高さ、物体の信頼度、人クラスの信頼度）を予測する。n 個のキーポイントを持つアンカーの場合、全体的な予測ベクトルは次のように定義される。

C_x, C_y：中心座標　W, H：幅・高さ

box_{conf}：物体の信頼度　$class_{conf}$：人クラスの信頼度

K_x^n, K_y^n：各キーポイントの座標　K_{conf}^n：各キーポイントの信頼度

$$P_v = \{C_x, C_y, W, H, box_{conf}, class_{conf}, K_x^1, K_y^1, K_{conf}^1, ..., K_x^n, K_y^n, K_{conf}^n\} \tag{3.33}$$

キーポイントの信頼度は、そのキーポイントの可視性フラグ $\delta(v_n)$ に基づいて学習される。キーポイントが可視または遮蔽されている場合、信頼度は 1 に設定され、それ以外の場合、視野外にある場合は信頼度は 0 に設定されます。推論中は、信頼度が 0.5 を超えるキーポイントを保持され、それ以外の予測キーポイントはすべて除外される。

また,YOLO-Pose では損失関数として, バウンディングボックスに対しては IOU の拡張である CIOU 損失を用い, キーポイントに対しては Object Keypoint Similarity (OKS) に基づく損失を導入している.

ネットワーク構成としては，図 3.16 の通りになる。

CSP-Darknet53 をバックボーンとし，画像の特徴抽出を行い、様々なスケール P3、P4、P5、P6 で特徴マップを出力する。次の PANet はこれらの複数スケールの特徴マップから人物検出及び姿勢推定に適した特徴表現を生成する。特に，キーポイント推定では局所的な位置精度が重要であるため，PANet によるマルチスケール特徴融合は重要な役割を果たす。PANet の出力は検出ヘッドに送られます。最終的に、異なるスケールに対応した 4 つの検出ヘッドはバウンディングボックスとキーポイントを予測する。

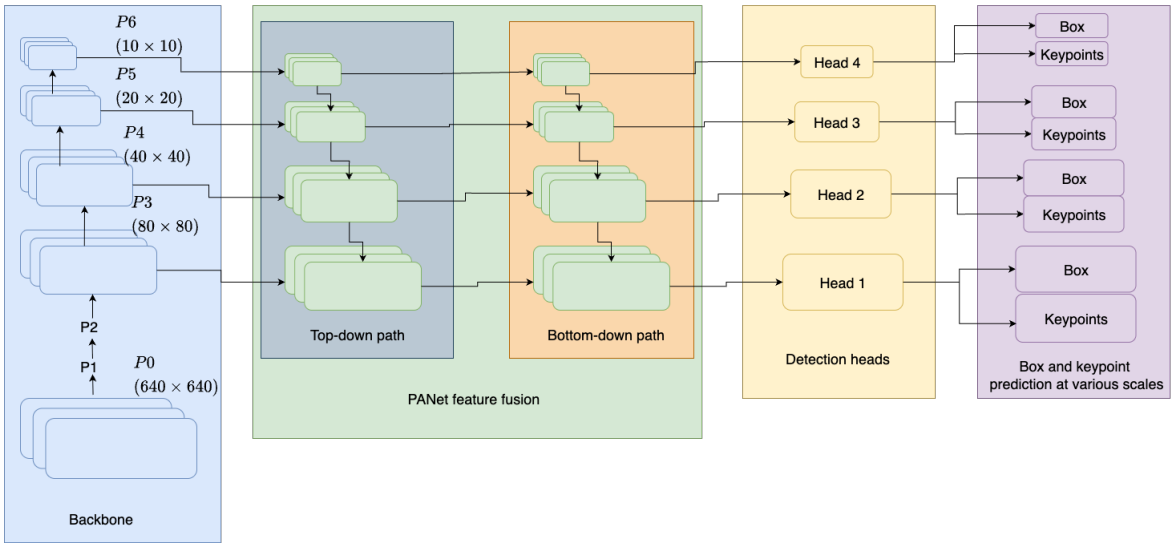

図 3.16 yolo-pose アーキテクチャ

3.4.1 トップダウン方式

トップダウン方式 [8], [12], [13], [19], [20], [21]、または 2 段階アプローチでは、高機能な人物検出器を用いて人物検出を行い、次に検出された人物ごとに 2 次元姿勢を推定する。この方式では，各人物が独立に処理されるため，人物同士が近接している場合や重なりが生じている場合でも，比較的高精度な姿勢推定が可能である。

一方で，画像内の人物数が増加すると，人物検出および姿勢推定の回数が増えるため，計算量が人物数に比例して増大するという欠点を持つ。そのため、トップダウン方式は複雑なタスクや高い精度が必要な場合に有効であり、画像内に存在する物体の位置や形状を正確に検出できるという利点がある。

3.4.2 ボトムアップ方式

ボトムアップ方式 [4], [5], [14], [23], [25] は、画像内の人物全員の識別情報のないキーポイントを 1 回の撮影で見つけ出し、それらを個々の人物インスタンスにグループ化する。ボトムアップ方式は、ヒートマップと呼ばれる確率マップに基づいて動作し、各ピクセルが特定のキーポイントを含む確率を推定する。ボトムアップ方式は、一般的に複雑さが少なく、実行時間が一定であるという利点がある。

しかし，異なる人物の同種のキーポイントが空間的に近接している場合，ヒートマップに基づく識別が困難となり，誤ったグループ化が生じやすいという課題がある。また、トップダウン方式と比較すると、精度が大幅に低下する。

3.4.3 IOU ベースのバウンディングボックス損失関数

現代の物体検出器のほとんどは、ボックス検出において、距離ベースの損失ではなく、GIoU [36]、DIoU、CIoU [37] といった IoU 損失の高度な変種を採用している。バウンディングボックスの検出には CIOU 損失を採用しており、次のように定義される（式 3.34）。Box_{pred}^{ijk} は、位置 (i ,j) とスケール s におけるアンカーの予測ボックスである。

$$\mathcal{L}_{box}(s,i,j,k) = (1 - CIOU(Box_{gt}^{s,i,j,k}, Box_{pred}^{s,i,j,k})) \tag{3.34}$$

3.4.4 Object Keypoint Similarity(OKS)

yolo-pose ではキーポイントを評価するための最も一般的な指標として OKS が用いられる。OKS は、物体検出における IOU（Intersection over Union）に相当する指標として機能し、推定された姿勢が正解の姿勢とどれほど似ているかを 0 から 1 の値で表す。OKS は各キーポイントごとに個別に計算され、その後合計されて最終的な OKS 損失またはキーポイント IOU 損失が得られる（式 3.35）。

$$
\begin{aligned}
d_n &= n \text{ 番目のキーポイントの予測座標と正解座標とのユークリッド距離} \\
k_n &= \text{キーポイントごとに設定される定数（推定が難しいものほど大きい値になる）} \\
s &= \text{人の大きさ} \\
\delta(v_n) &= \text{各キーポイントが画像に写っているか示すフラグ} \\
\mathcal{L}_{kpts}(s,i,j,k) &= 1 - \sum_{n=1}^{N_{kpts}} OKS \\
&= \sum_{n=1}^{N_{kpts}} \frac{exp(\frac{d_n^2}{2s^2k_n^2})\delta(v_n > 0)}{\sum_{n=1}^{N_{kpts}} \delta(v_n > 0)}
\end{aligned}
\tag{3.35}
$$

3.5 YOLO-Tracking による物体追跡

本研究では 1 人ではなく、複数人を対象に転倒を検知するシステムのため、物体追跡技術である YOLO-Tracking の BoT-SORT を用いる。YOLO-Tracking は YOLO によって各フレームの人物を抽出し、その検出結果をもとに時系列方向で人物 ID を対応付けることでマルチオブジェクトトラッキング（Multi-Object Tracking, MOT）を実現する手法である。BoT-SORT[6] には 2022 年 6 月に発表された物体追跡の手法であり、既存の手法、ByteTrack[5] に以下の 3 つの変更を加えたものである。

1. カルマンフィルタ
2. カメラモーション補償（CMC）
3. IoU ReID フュージョン

3.5.1 マルチオブジェクトトラッキング（Multi-Object Tracking, MOT）

一般的に MOT では 2 段階に分けて処理を行う。MOT 1 段階目は物体検出を行い、2 段階目でデータの関連付けを行う。第 1 段階では物体検出を行い，各フレームにおいて人物などの対象物体のバウンディングボックスを取得する。第 2 段階ではデータアソシエーションを行い，異なるフレーム間の検出結果を対応付けることで，時系列に沿った軌跡（トラック）を生成する。データアソシエーションでは、まずトラックレット（動画内の移動物体が一時的に追跡された短い軌跡）と検出ボックス間の類似度を計算し、類似度に応じてオブジェクトに識別子を割り当てる。この類似度には位置情報、動き情報、外観情報が含まれている。特に物体検知による追跡に基づく MOT では，物体検出の精度が追跡性能の上限を大きく左右する。遮蔽やモーションブラーが発生すると検出スコアが低下し，その結果，追跡が途切れる問題が生じやすい。この問題に対処するため，近年の MOT 手法では，低信頼度の検出結果も含めて活用し，より安定したデータアソシエーションを実現する試みが行われている [5]。

3.5.2 Bytetrack

ByteTrack では，BYTE（Binary Association of Every detection）と呼ばれるシンプルかつ汎用的なデータ関連付け手法を提案している。BYTE は高スコア検出ボックスのみを保持する従来の手法 [33, 47, 69, 85] とは異なり、ほぼすべての検出ボックスを保持し、それらを高スコアと低スコアに分離する。まず、高スコア検出ボックスをトラックレットに関連付ける。次に、低スコア検出ボックスと一致しないトラックレットを関連付けることで、低スコア検出ボックス内のオブジェクトを復元し、同時に背景を除去する。

3.5.3 カルマンフィルタ

SORT [3] では、状態ベクトルは 7 つの要素、$x = [x_c, y_c, a, s, \dot{x}_c, \dot{y}_c, \dot{s}]^T$ とされていました。ここで、(xc, yc) は画像平面における物体中心の 2D 座標であり、s はバウンディングボックスのスケール（面積）、a はバウンディングボックスのアスペクト比である。最近のトラッカーでは、状態ベクトルは 8 つの要素、$x = [x_c, y_c, a, h, \dot{x}_c, \dot{y}_c, \dot{a}, \dot{h}]^T$ に変更されている。しかし、実験により、バウンディングボックスの幅と高さを直接推定する方が性能が向上することがわかった。したがって、KF の状態ベクトルを式 3.36 のように定義し、測定ベクトルを式 3.37 のように定義する。

$$x_k = [x_c(k), y_c(k), w(k), h(k), \dot{x}_c(k), \dot{y}_c(k), \dot{w}(k), \dot{h}(k)]^{\mathrm{T}} \tag{3.36}$$

$$z_k = [z_{x_c}(k), z_{y_c}(k), z_w(k), z_h(k)]^{\mathrm{T}} \tag{3.37}$$

KF の修正は、バウンディングボックスの幅と物体の適合性の向上に寄与している。

3.5.4 カメラモーション補償（CMC）

カメラモーション補償（CMC）は不要なカメラの動き（揺れ、パン、傾き）を修正したり、ぼやけや揺れを軽減する一連の技術である。検出による追跡は、予測されたトラックレットの境界ボックスと検出された境界ボックスの重なりに大きく依存し、データの関連付けを行う。カメラが動いている場合、画像平面上の境界ボックスの位置は劇的に変化する可能性があり、ID スイッチ（フレーム内で追跡している物体の識別 ID が、本来とは異なる別の物体に誤って割り当てられてしまう現象）やなどにつながる可能性がある。これらを防ぐために CMC 機能を追加する。追加のセンサ情報を用いず，隣接フレーム間の画像レジストレーションによってカメラの剛体運動を推定する。具体的には，OpenCV の Video Stabilization モジュールに基づき，Shi–Tomasi 法による特徴点抽出と疎なオプティカルフロー追跡を行い，RANSAC を用いてフレーム k-1 から k へのアフィン変換行列 $A_{k-1}^{k} \in \mathbb{R}^{2\times 3}$ を推定する。推定されたアフィン変換は，Kalman Filter の状態空間に拡張して適用される。回転・スケール成分を含む行列 $\tilde{M}_{k-1}^{k}$と並進成分$\tilde{T}_{k-1}^{k}$ を定義し，予測状態および共分散行列を次式により補正する。

$$*\hat{x}'_{k|k-1} = *\tilde{M}_{k-1}^{k} *\hat{x}_{k|k-1} + *\tilde{T}_{k-1}^{k}$$

$$*P'_{k|k-1} = *\tilde{M}_{k-1}^{k} *P_{k|k-1} *\tilde{M}_{k-1}^{k\top}$$

3.5.5 IoU ReID フュージョン

BoT-SORT では、動き情報と外観情報を統合してデータの関連付けを向上させている。動き情報 IoU と外観情報 $Re-ID$ を統合する新しい手法として、IoU 距離行列とコサイン距離行列を開発しました。まず、IoU スコアでコサイン類似度が低い候補や距離が遠い候補を除外します。次に、

$$\hat{d}_{i,j}^{cos} = \begin{cases} 0.5 \cdot d_{i,j}^{cos}, (d_{i,j}^{cos} < \theta_{emb}) \wedge (d_{i,j}^{iou} < \theta_{iou}) \\ 1, \text{otherwise} \end{cases} \tag{3.39}$$

$$C_{i,j} = min\{d_{i,j}^{iou}, \hat{d}_{i,j}^{cos}\} \tag{3.40}$$

$d_{i,j}^{iou}$ はトラックレット i 番目の予測境界ボックスと j 番目の検出境界ボックス間の IoU 距離であり、モーション コストを表す。$d_{i,j}^{cos}$ は平均トラックレット外観記述子 i と新しい検出記述子 j 間のコサイン距離、つまり新しい外見コストである。θiou はモーションコストの閾値で、0.5 に、θ_{emb} は外見コストの閾値で 0.25 に設定されている。両方が閾値を超えない場合のみ、外観コストを用いて、それ以外は不一致として扱う。トラックレットと検出のありそうもないペアを拒否するために使用されます。そして最終的なコスト行列の各要素は，IoU 距離と修正後の外観距離の最小値として定義される（3.40）。

〇〇〇

4 実験方法

4.1 データセット

○○○

表 4.1 に実験環境を示す.

○○

4.2 実験条件

○○○

図 4.1 に遅延比較結果を示す.

○○○

表 4.1 実験環境

項目	内容
コントローラ	Ryu Controller
スイッチ	Open vSwitch
トポロジ	3 リンク構成
トラフィック生成	iperf3

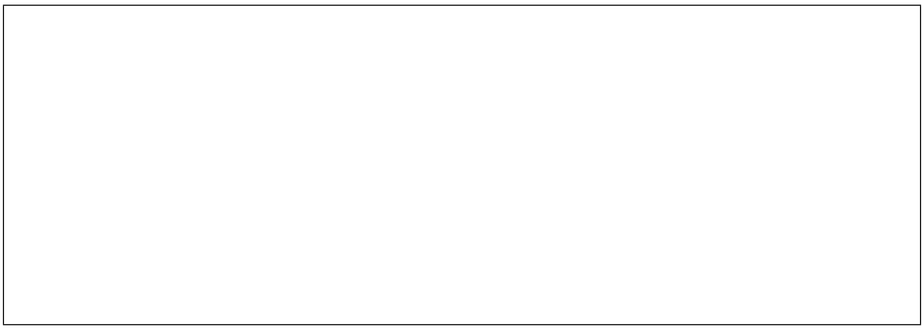

図 4.1 遅延の比較

○○○

○○○

○○○

4.3 評価指標

○○

○○○

○○
○○○

　○○○
○○○
○○○
○○○

　○○○
○○○
○○○
○○○

5　実験結果

5.1　サンプル数と精度の関係

5.2　クラス別性性能評価

5.3　線形補完の効果比較

5.4　判定に有効な距離や人数によるタイムラグ

5.5　転倒検知精度の考察

6　考察

6.1　スケール正規化の有効性

6.2　誤検知・未検知の分析

6.3　実運用時の課題

7　結論

本研究では SDN を用いた動的ルーティングにより，ネットワーク混雑時の遅延改善が期待できることを示した．〇〇

〇〇

〇〇今後は大規模ネットワークや異なるトラフィックパターンへの拡張を検討する．

参考文献

[1] 李銀星. 山田和範. 探検データサイエンス　ニューラルネットワーク入門. 共立出版株式会社,2024,226, 探検データサイエンス

[2] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. Soft-NMS – Improving Object Detection With One Line of Code. Computer Vision and Pattern Recognition (cs.CV),2017.4

[3] Navaneeth Bodla. Bharat Singh, Rama Chellappa, Larry S. Davis. You Only Look Once:Unified, Real-Time Object Detection. Computer Vision and Pattern Recognition (cs.CV),2015.6

[4] Debapriya Maji, Soyeb Nagori, Manu Mathew, Deepak Poddar. YOLO-Pose: Enhancing YOLO for Multi Person Pose Estimation Using Object Keypoint Similarity Loss. Computer Vision and Pattern Recognition (cs.CV),2022.4

[5] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, Xinggang Wang. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. Computer Vision and Pattern Recognition (cs.CV),2021.10

[6] BoT-SORT: Robust Associations Multi-Pedestrian Tracking. Nir Aharon, Roy Orfaig, Ben-Zion Bobrovsky. Computer Vision and Pattern Recognition (cs.CV),2022.6

[7] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, Tae-Kyun Kim. Multiple Object Tracking: A Literature Review.

[8] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft. SIMPLE ONLINE AND REALTIME TRACKING. Computer Vision and Pattern Recognition (cs.CV),2016.2

[9] B. A. A. Nunes et al., “A Survey of Software-Defined Networking,” IEEE Communications Surveys and Tutorials, Vol. 16, No. 3, pp. 1617-1634, 2014.

[10] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: Enabling Innovation in Campus Networks,” ACM SIGCOMM Computer Communication Review, Vol. 38, No. 2, pp. 69-74, 2008.

謝辞

　○○○
○○
○○
○○○

A　SDNコントローラ設定例

```
# Ryu Controller sample

class SimpleSwitch(app_manager.RyuApp):

    def _packet_in_handler(...):

        # install flow dynamically
```

B　追加評価

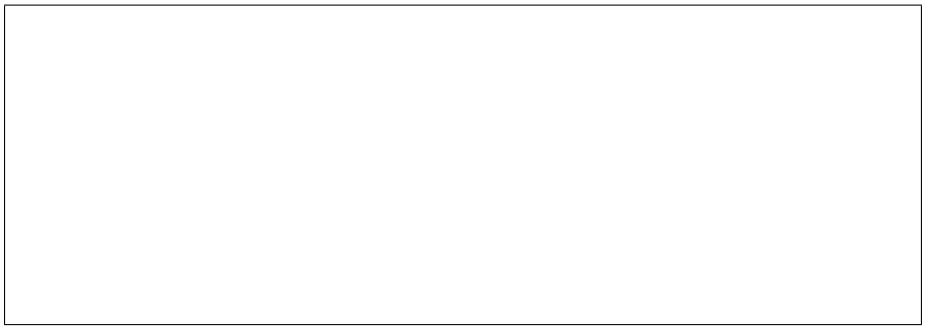

図 B.1 追加評価の図