

物体の誤検出傾向を用いた画像認識による在庫管理システムの開発

嶋田 滉平¹ 山本 雄平² 須山 敬之¹

概要: 近年, IoT の利活用が推進され, IoT を家電に適用したスマート家電が普及している. 特にスマート冷蔵庫では在庫管理をはじめとする様々な機能が搭載されている. 既存製品では庫内カメラから状況を目視で確認できるものがあり, 既存研究では重量センサを用いた在庫管理の手法が提案されている. しかし, 既存製品ではカメラの死角の確認が取れなかったり, 既存研究では物の入出庫は一度に一つまでの制限があり, 利便性が低い問題点がある. 本稿では死角の無い冷蔵庫上部からのカメラ動画を画像認識技術で解析することで在庫管理システムを実現し, また開発したシステムは画像の認識結果に大きく依存するため学習データの組み合わせを変え, 認識精度に与える影響について検討を行った.

1. はじめに

現在, IoT の利活用が推進されており, IoT を掃除機や冷蔵庫, テレビ等に適用したスマート家電の普及率は上がってきている. 特にスマート冷蔵庫は在庫管理をはじめ, 献立の提案, 近隣スーパーの特売情報の通知など様々な機能が搭載されている. 在庫管理機能により冷蔵庫内の状況を扉の開閉なく確認できると, 消費電力の削減や冷蔵庫本体の長寿化が期待でき, 利用者の無駄な動作も減少できる. また, 冷蔵庫内に何の食材がどのくらい残っているかを買った物先で確認できた場合, 過剰購入による食品ロスの減少, 買い忘れの抑制が期待できる.

在庫管理機能を搭載した既存製品では Samsung 電子株式会社の Family Hub [1] がある. この製品は庫内搭載のカメラから冷蔵庫の庫内状況をスマートフォンの専用アプリを通して目視で確認できる.

既存研究では重量センサを用いた手法 [2] が提案されており, 食材の重量を計ることで在庫管理を実現している. しかし, 食材の入出庫は1回につき1個までの条件があるなど, 利便性が低い.

本研究では冷蔵庫の入出庫の様子を撮影した映像を解析し, 冷蔵庫の在庫管理支援を試みる.

2. 研究概要

既存研究では重量センサを用いた冷蔵庫の在庫管理システムが提案されている. このシステムでは食材の入出庫は

1つずつ行うことを前提とし, 冷蔵庫内の重量センサで食材の重量を計る. 計った食材の重量が在庫リストデータベースの中で合致する物を入庫または出庫した食材と推定し, 在庫管理を行う. もし, データベースに合致する物がなかった場合はユーザーに確認を行うことで修正を行なっている. しかし, 山下らの研究では食材の入出庫は同時に1点しかできない前提のため利便性が低い.

Family Hub では庫内搭載のカメラから在庫状況の確認ができるが, これでは食材がカメラの死角に入った時に確認が難しい問題がある.

そこで本研究では画像処理分野の物体検出技術を用いて冷蔵庫上部から食材の入出庫の様子を撮影し, それを解析することで冷蔵庫の在庫管理の自動更新の支援を試みる. 冷蔵庫上部からの撮影であるため一度の入出庫に物の数に制限はなく, 利便性の向上が期待でき画像認識の精度によっては山下らの既存研究と比較して高精度に在庫管理を行える可能性がある. また, 本システムは画像の認識結果に大きく依存するため, 学習に用いるデータの組み合わせを変え, 認識精度に与える影響についての検討も行った.

3. システムの概要

本システムは, 画像中の食材の認識機能, 食材の軌跡作成機能, 入出庫履歴推定機能から構成される. 画像中の食材の認識機能は物体検出技術を用いて画像中のどこに何が写っているのかを判定し, 情報を出力する. 食材の軌跡作成機能は前段階で出力された情報を用いて認識された物体の移動軌跡を作成する. 入出庫履歴推定機能では前段階で作成された軌跡の先頭と末尾の位置を評価し, それが入出

¹ 大阪工業大学大学院 情報科学研究科

² 関西大学環境都市工学部

庫のどちらなのかを判定する。最後に入出庫判定された食材の種類と数がシステムの出力として表示される。概要図を図 1 に示す。

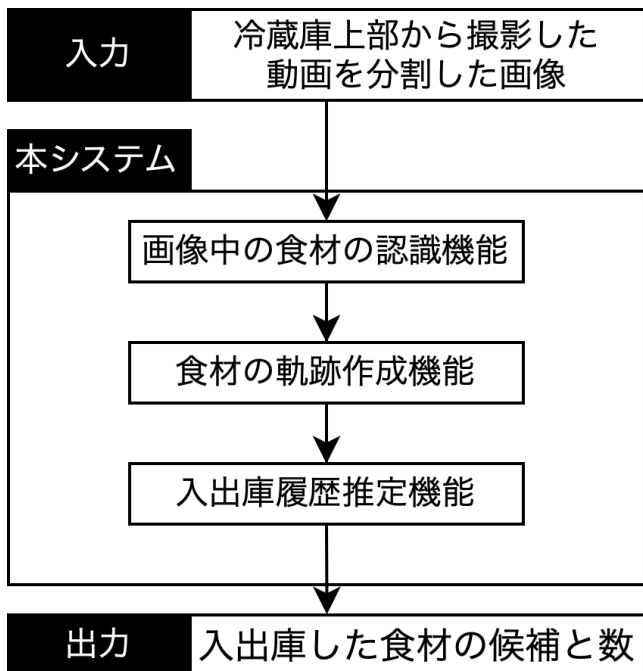


図 1 本システムの概要

3.1 システムへの入力

本システムに入力するデータは冷蔵庫上部から撮影した動画をフレームごとに分割した画像である。撮影には iPhone11 を使用し 4K, 60fps で撮影を行った。撮影した動画を FFmpeg [3] で 1 フレームごとに分割を行い、それらをシステムの入力データとする。図 2 は冷蔵庫上部から下方向に引き出し式の冷蔵庫の内部を撮影をしている様子である。



図 2 撮影風景

3.2 画像中の食材の認識機能

画像中の食材の認識には YOLOv4 [4] を用いる。YOLOv4 とは Object Detection の手法の 1 つで検出精度が高く、リアルタイム物体検出が可能なアルゴリズムである。

判定は独自に用意したモデルを用いる。本稿では大根、人参、玉ねぎ、キュウリ、トマト、キャベツ、レタス、白菜、ナス、リンゴの 10 クラスについて学習を行い認識モデルの作成を行った。YOLO では検出された物体の中心座標、幅、高さからなる矩形情報と信頼度、クラス番号が出力される。作成したモデルで判定した結果の例を図 3、判定後に得られる JSON 形式の検出データの一部を図 4 に示す。



図 3 判定結果

```

"frame_id":1,
"filename":"D:\YoloVbuild\darknet\X64\shimada\Video\Test\Ytest\Y0000193.jpg",
"objects":[
{
  "class_id":31,
  "name":"ringo",
  "relative_coordinates":{"center_x":0.296126,"center_y":0.662537,
    "width":0.145183,"height":0.266826},
  "confidence":0.999681},
{
  "class_id":6,
  "name":"tomato",
  "relative_coordinates":{"center_x":0.300830,"center_y":0.179934,
    "width":0.112314,"height":0.202575},
  "confidence":0.999666}
]

```

図 4 検出データの一部

検出データのプロパティを表 1 に示す。

3.3 食材の軌跡作成機能

本機能では前段階で作成した JSON を解析することで食材の移動した軌跡の作成を行う。撮影速度を 60fps に設定

表 1 検出データのプロパティ

frame_id: 判定結果の連番		
filename: 判定画像のパス		
objects	class_id: 判定クラスの ID	
	name: 判定クラス名	
	confidence: 信頼度	
	relative_coordinates	center_x: 検出矩形の中心 x 座標
		center_y: 検出矩形の中心 y 座標
width: 検出矩形の幅		
height: 検出矩形の高さ		

しているため、任意のフレームに存在した矩形は過去数フレーム以内にも存在すると仮定して、処理を行う。軌跡を繋ぐ際の条件は、現在の矩形中に過去矩形の中心座標が存在するか否かとした。例を図 5 に示す。

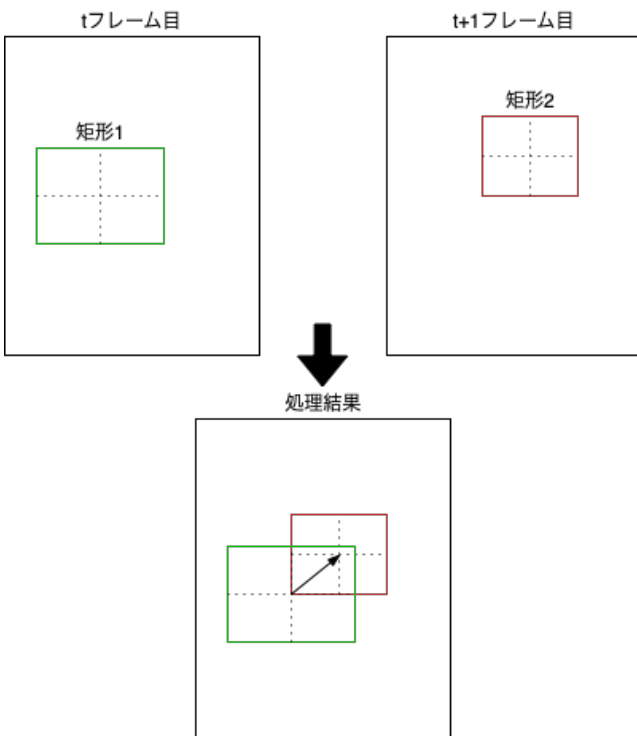


図 5 軌跡を繋ぐ際の条件

また、図 6 のように現在のフレームの矩形内に過去のフレームの矩形が複数個存在するようなことが想定される。

動画の撮影速度が 60fps であり、1 フレーム間に移動できる食材の距離は十分小さいと考えられるため、矩形が複数存在する場合には現在のフレームの矩形とそれぞれの矩形とのユークリッド距離を計算し、一番距離が小さい矩形を該当フレームと仮定し、軌跡を接続する。

3.4 入出庫履歴推定機能

本機能では作成した食材の軌跡の先頭と末尾の位置を評価することで入庫または出庫のいずれかを判定する。位置の評価では予め定義した入出庫判定境界を用いる。入出庫判定境界の位置は冷蔵庫の引き出しの前面の位置とし、先

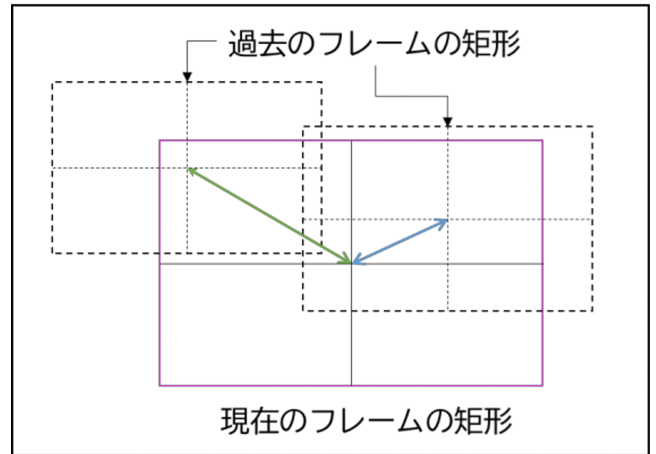


図 6 矩形が複数個存在する場合の例

頭が境界外で末尾が境界内なら入庫、先頭が境界内で末尾が境界外なら出庫と判定する。

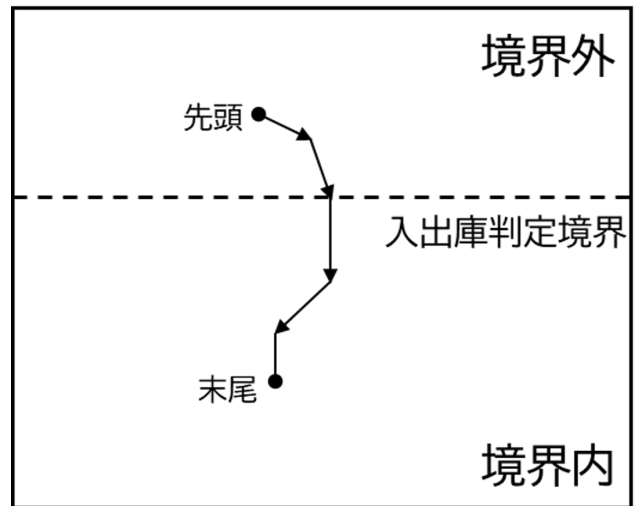


図 7 入庫判定の例

また、作成した軌跡の食材推定は各検出情報の信頼度の和を検出数で割り平均値を計算し、最も高い値を持つクラスをその軌跡の食材とした。

図 8 の場合、クラス A, B, C があり、平均値がそれぞれ 0.910, 0.694, 0.893 となり、この軌跡はクラス A の軌跡であると判定される。

また、入出庫判定は入庫となるので、結果としてはクラス A が入庫したと判定される。

3.5 システムからの出力

前節までの処理を行った結果を表形式で食材の候補と入出庫した数が表示される。表 2 に出力の例を示す。

表 2 の場合、大根が+1 とは大根が 1 個入庫したことを表し、白菜-1 は白菜が 1 個出庫したことを表す。

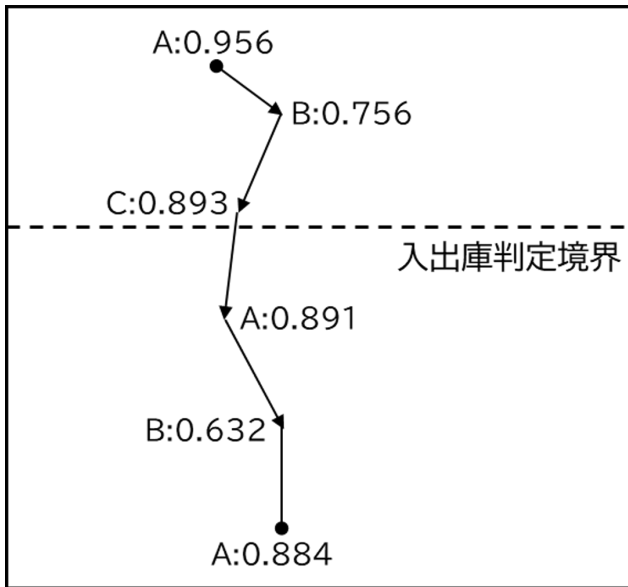


図 8 食材推定の例

食材名	入出庫数
大根	+1
人参	+2
白菜	-1
玉ねぎ	-4

4. 開発したシステム

4.1 開発環境

システムの開発には以下を用いた。

- IDE : Visual Studio Community 2019
- フレームワーク : .NET Framework Ver 4.7.2
- 言語 : C# Ver 9.0
- ライブラリ : Newtonsoft.Json Ver 12.0.3
- 物体検出ライブラリ : Darknet(YOLOv4)

4.2 画像中の食材の認識機能の画面

画像中の食材の認識機能のユーザインタフェースを図 9 に示す。

認識に用いるモデルの設定は Weights ファイル項目で行い、検出時の最低信頼度を検出閾値項目で設定する。最低信頼度とは検出された各矩形の信頼度の最低値であり、この値未満の検出矩形は出力されなくなる。また、検出情報の出力先を JSON パス項目で設定でき、検出に用いる画像フォルダは画像パス項目に設定する。

4.3 食材の軌跡作成機能の画面

食材の軌跡作成画面のユーザインタフェースを図 10 に示す。

追加ボタンでは解析対象の JSON ファイルの指定を行う



図 9 食材の認識機能の UI



図 10 食材の軌跡作成の UI

ことができ、誤って追加してしまったファイルを削除ボタンで除くことができる。遡行フレーム数では遡るフレームの最大値を設定することができ、また、判定結果に一瞬だけ映り、後に映らないようなノイズ情報を除去するには最短軌跡長を設定することで実現できる。最短軌跡長を 5 に設定した場合、図 11 の左図の軌跡の両端が右図のように除去することができる。

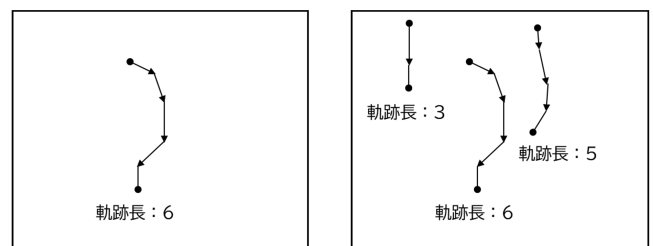


図 11 最短軌跡長が及ぼす効果の例

実際に作成した軌跡の例を図 12 に示す。

図 12 は人参を冷蔵庫へ入庫した時の軌跡を示している。

4.4 システムからの出力の画面

システムからの出力画面のユーザインタフェースを図 13 に示す。

入出庫した食材の数の修正には +1 ボタンと -1 ボタンを用いることで実現できる。追加ボタンと削除ボタンでは任意の食材の追加と削除を行うことができ、編集ボタンで食材の変更ができる。最後に決定ボタンでデータベースの更新を行う。

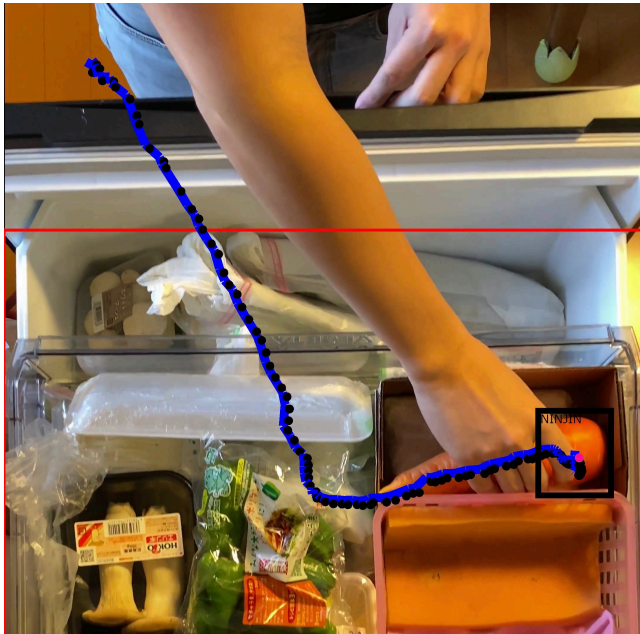


図 12 作成した軌跡の例

ManagementResult		
クラス名	入出庫結果	
HAKUSAI	+1	+1
kyuri	+2	+1
nasu	+1	-1
NINJIN	0	
RINGO	0	
tamanegi	+3	
TOMATO	0	

図 13 システムからの出力の UI

5. システムの評価

第 4 章で開発したシステムで処理を行い、実際に入出庫した結果と処理結果の比較を行う。また、システムの評価と並行してモデルの学習パターンについても検証を行った。

5.1 モデルの学習パターン

本章で用いたモデルの学習パターンは計 3 つあり、それは以下の通りである。

- (1) 1 クラスを 1 モデルにしたもの (パターン 1)
- (2) 全クラスを 1 モデルにしたもの (パターン 2)
- (3) 色・見た目が似ているクラスを 1 モデルにしたもの (パターン 3)

パターン 3 に関して、本稿ではキャベツとレタス、トマトとリンゴを一つのモデルとし学習を行った。

5.2 パターン 1 の評価結果

パターン 1 の評価結果を表 3 に示す。真値は実際に入出

庫した各食材の数を表しており、システム列は処理を行った結果の入庫数と出庫数、その結果を示している。システムの入庫と出庫は正答と漏れ、過剰の 3 つで構成されており、正答は正しく判定を行えた数、漏れはみ検出できなかった数、過剰は多く検出してしまった数を表している。

表 3 の玉ねぎに関しては過剰に検出傾向が見られ、また大根や白菜、キャベツ、レタスは手と重なる割合が少ないにも関わらず誤判定する傾向があった。

5.3 パターン 2 の評価結果

パターン 2 の評価結果を表 4 に示す。

パターン 2 ではパターン 1 と比較して在庫差が小さい傾向が見られたが、大根やキャベツ、レタスなどのように検出できていない食材がある結果となった。

5.4 パターン 3 の評価結果

パターン 3 の評価結果を表 5 に示す。

表 5 の玉ねぎはパターン 1 と同様に過剰な検出がされた。しかし、パターン 1 と比較して 1 モデルにまとめたキャベツとレタス、トマトとリンゴの在庫差は小さい傾向であることがわかった。

6. 実験

第 5.1 章で示したように学習データの組み合わせを変更することで検出結果に影響を与えることがわかった。その結果を踏まえて本章では組み合わせを変更したモデルについて検証をした。

6.1 実験方法

モデルの精度評価では学習に含まれていない 10,392 枚の画像で行い、得られた図 4 のような JSON データを解析して評価を行った。また、クラスごとの F1 スコアを計算し、その平均である Macro-F1 スコアを評価指標として用いた。

F1 スコアとは Recall と Precision の調和平均であり、Macro-F1 スコアはクラスごとに計算された F1 スコアの合計をクラス数で割った平均である。F1 スコアの計算には真と予測して実際に真である True Positive(以下 TP)、真と予測して実際は偽である False Positive(以下 FP)、偽と予測して実際は真である False Negative(以下 FN) の 3 つの値が用いられる。Recall は再現率と呼ばれ、実際に真であるデータの内、真と予測されたデータの割合を示す。Precision は適合率と呼ばれ、真と予測したデータの内、実際に真であるデータの割合を示す。

本稿でもこれらの値を用いるため、検出された矩形と真値の矩形との IoU と検出された矩形のクラスと真値のクラスを比較することで評価を行った。IoU とは物体検出で用いられる評価指標の一つで 2 つの矩形の重なり割合で表

表 3 パターン 1 の結果

	真値			システム						結果	
	入庫	出庫	在庫	入庫			出庫			入出庫 推定	在庫差
				正答	漏れ	過剰	正答	漏れ	過剰		
大根	1	0	1	0	1	0	0	0	0	0	-1
人参	3	1	2	1	2	0	1	0	0	0	-2
玉ねぎ	2	1	1	2	0	25	1	0	12	14	13
キュウリ	3	2	1	3	0	3	2	0	0	4	3
白菜	1	0	1	1	0	5	0	0	0	6	5
ナス	2	1	1	2	0	1	1	0	1	2	1
キャベツ	1	0	1	0	1	0	0	0	1	-1	-2
レタス	1	0	1	1	0	3	0	0	0	4	3
トマト	3	2	1	3	0	0	0	2	0	3	2
リンゴ	3	2	1	0	3	0	1	1	0	-1	-2

表 4 パターン 2 の結果

	真値			システム						結果	
	入庫	出庫	在庫	入庫			出庫			入出庫 推定	在庫差
				正答	漏れ	過剰	正答	漏れ	過剰		
大根	1	0	1	×	1	0	×	0	0	0	-1
人参	3	1	2	0	3	0	0	1	0	0	-2
玉ねぎ	2	1	1	2	0	7	1	0	5	3	2
キュウリ	3	2	1	3	0	1	2	0	0	2	1
白菜	1	0	1	1	0	0	0	0	0	1	0
ナス	2	1	1	2	0	0	1	0	0	1	0
キャベツ	1	0	1	×	1	0	×	0	0	0	-1
レタス	1	0	1	×	1	0	×	0	0	0	-1
トマト	3	2	1	1	2	0	1	1	0	0	1
リンゴ	3	2	1	0	3	0	0	2	0	0	-1

される。全く重なっていないと 0 となり、完全に一致しているとは 1 となる。

検出された矩形と全ての真値の矩形で IoU を計算し、閾値を超えてかつ、クラスが一致していれば TP を一つ加算し、閾値を超えてもクラスが一致していない場合は FP を一つ加算する。本稿で用いた IoU の閾値は 0.5 とした。

また、真値の矩形に対して、一つも矩形が検出されなかった場合は FN を一つ加算する。

6.2 モデルの評価結果

玉ねぎクラスを基本とし、他のクラスと組み合わせたモデルの評価結果を表 6 に示す。

この組み合わせでは他クラスと組み合わせたモデルや玉ねぎクラスのみモデルと比較して全クラスモデルの方が精度が高い結果となった。次に大根クラスを基本とし、他のクラスと組み合わせたモデルの評価結果を表 7 に示す。

この組み合わせでは全クラスモデルや大根クラスのみモデルと比較してナスと組み合わせた方が精度が高い結果となった。次にナスクラスを基本とし、他のクラスと組み合わせたモデルの評価結果を表 8 に示す。

この組み合わせも表 7 と同様に全クラスモデルやナスクラスのみモデルと比較して大根と組み合わせた方が精度

が高い結果となった。

以上よりクラスをうまく組み合わせることにより、1 クラスのみの場合や全てのクラスで学習したモデルより認識精度が向上するケースがあることがわかった。大根とナスのモデルのように全クラスモデルと比較して精度が良くなる組み合わせが存在するため、一概に 1 モデルに含まれるクラス数を増やすと精度が向上するわけではないことがわかった。

7. まとめ

本稿では物体検出技術を用いて冷蔵庫の在庫管理システムの開発を行い、計 3 パターンのモデルで入出庫の処理を行った結果、色や見た目が似ているクラスを 1 モデルとした方が在庫差が小さくなる傾向が見られた。それを踏まえてモデルに含まれるクラスを変更し、精度の検証を行った。その結果、モデルに含まれるクラス数が多いと誤検出数が減少する傾向がある一方で全クラスモデルが一概に高精度とはならず、2 クラスのモデルが高精度となる場合があることが判明した。

今後は 3 クラスを組み合わせたものについても検証を行い、より高精度となるクラスの組み合わせを探索することで在庫管理システムの精度向上について検討する。

表 5 パターン 3 の結果

	真値			システム						結果	
	入庫	出庫	在庫	入庫			出庫			入出庫 推定	在庫差
				正答	漏れ	過剰	正答	漏れ	過剰		
大根	1	0	1	0	1	0	0	0	0	0	-1
人参	3	1	2	1	2	0	1	0	0	0	-2
玉ねぎ	2	1	1	2	0	29	1	0	12	18	17
キュウリ	3	2	1	3	0	3	2	0	0	4	3
白菜	1	0	1	1	0	6	0	0	0	7	6
ナス	2	1	1	2	0	1	1	0	1	1	0
キャベツ	1	0	1	1	0	2	0	0	2	1	0
レタス	1	0	1	1	0	4	0	0	1	4	3
トマト	3	2	1	3	0	1	2	0	1	1	0
リンゴ	3	2	1	3	0	1	2	0	0	2	1

表 6 玉ねぎが含まれるモデルの評価結果

		Macro-F1
1 クラス		0.205
10 クラス		0.676
2 クラス	大根	0.512
	ナス	0.467
	リンゴ	0.596
	トマト	0.614

表 7 大根が含まれるモデルの評価結果

		Macro-F1
1 クラス		0.416
10 クラス		0.676
2 クラス	ナス	0.777
	人参	0.522
	リンゴ	0.761
	玉ねぎ	0.512

表 8 ナスが含まれるモデルの評価結果

		Macro-F1
1 クラス		0.655
10 クラス		0.676
2 クラス	大根	0.777
	人参	0.507
	リンゴ	0.682
	玉ねぎ	0.467

謝辞 本研究は JSPS 科研費 JP19138779 の助成を受けたものです。

参考文献

- [1] Samsung 電子株式会社 : Family Hub <https://www.samsung.com/us/explore/family-hub-refrigerator/overview/>.
- [2] 山下 徹, 玉井森彦, 安本慶一: 重量センサを用いた冷蔵庫内のモノ管理システム, 情報処理学会研究報告, Vol. 2014-CDS-10, No. 3, pp. 1-7 (2014).
- [3] FFmpeg: <https://ffmpeg.org>.
- [4] Bochkovskiy, A., Wang, Y. and Liao, M.: YOLOv4: Optimal Speed and Accuracy of Object Detection, *arXiv:2004.10934* (2020).