2024 年度 卒業論文

モバイル端末の単独動作による牛顔個体識別の複数の端末・特徴抽出器での評価

情報通信工学プログラム

学籍番号 60212711

橋元 勝美

指導教員　椋木 雅之 教授

2025 年 2 月 7 日

目次

1 はじめに

近年、農業のさまざまな課題を解決する手段として、スマート農業が注目を集めている。農林水産省によると、スマート農業は「ロボット技術や情報通信技術（ICT）を活用し、省力化・精密化や高品質生産を実現することを推進している新たな農業」[1]と定義されている。畜産農業においても、スマート農業への取り組みが活発である。例えば、cntxts Inc. が提供する Smart Cattle®[2] では、牛にセンサデバイスを装着し、そのデバイスから得た情報をスマートフォンなどで受信するシステムを提供している。1 頭の牛に 1 つのセンサデバイスを取り付けて管理するこの方法は、牛の個体情報や位置情報を正確に管理できるが、導入コストが高いため、多くの畜産農家にとって利用しやすい方法とは言えない。コストがあまりかからない方法としては、株式会社ファームノートが提供するクラウド牛群管理システム Farmnote Cloud[3]がある。Farmnote Cloud では、繁殖対象の牛やワクチン接種が必要な牛などを絞り込んだり、アカウントを通じて獣医師など外部の関係者とデータをやり取りしたりして、詳しい牛の情報を管理できる。しかし、耳のタグに書かれた個体識別番号を手動で入力したり、文字情報だけで情報を出力したりするなど、使いにくい部分もある。兒玉[4]の研究では、クライアントサーバシステムを利用した「CowFindAR」を構築している。このシステムは、カメラ映像の取得やユーザに対する情報の提示を行うためのクライアントシステムと、深層学習を用いた牛顔個体識別を行うためのサーバシステムからなる。クライアントで撮影した牛顔画像をサーバに送信する。サーバでは、受信した画像に対して、牛顔検出、牛顔特徴抽出、登録データとの照合を行い、照合結果をクライアントに返信する。クライアントは、返信結果をライブ映像上に重ねて情報提示する。クライアントサーバシステムを採用にすることにより、計算コストのかかる深層学習技術をサーバ上で実行でき、拡張性の高いシステムになっている。しかし、このシステムの利用には通信インフラ（例：Wi-Fi 環境）の整備が必要である。また、通信を行う際のオーバーヘッドによって識別処理の速度が低下する可能性があり、リアルタイム性が求められる場面では課題となる。そのため、軽量化したモデルを使用することでモバイル端末上でも一定の精度を維持できるようなシステム設計が必要となる。

そこで本研究では、より導入コストが低いシステムとして、モバイル端末の単独動作による牛顔個体識別を提案する。このシステムは、牛撮影、牛顔検出、牛顔特徴量抽出、登録データとの照合、識別結果の提示までのすべての処理を、単独のモバイル端末（スマートフォン）上で実行する。牛舎は、市街地から離れた場所にあり、通信ネットワークが整備されていないことも多い。提案システムはそのような実環境でも通信インフラなしに使用可能である。また、画像を送信するオーバーヘッドも生じないため、処理速度の面でも優位性がある。

一方で、モバイル端末の単独動作では計算能力が低いため、速度や精度に影響が出る。このため、使用する深層学習手法の選定が重要となる。本研究では、ResNet[5]、MobileNet[6]、EfficientNet[7]について精度・速度を評価する。さらに、複数のモバイル端末上で精度・速度評価を行い、現状の機器上における提案システムの有効性を調査する。

以下、第 2 章では牛個体情報を扱う従来研究や本研究で使用する特徴抽出器などについて述べる。第 3 章では構築したモバイル端末の単独動作による牛顔個体識別システムについて述べる。

第 4 章ではシステムの基本的な性能について評価する。第 5 章では研究の内容をまとめるとともに、今後の展望について述べる。

2 顔画像による個体識別

2.1 顔画像による個人・個体の識別

現在、顔画像による個体識別は、様々な分野で活用されている。世界トップと言われている最先端の顔認証技術を開発した NEC の顔認証システムは、高精度と高速処理を特徴としており、独自のアルゴリズムにより、膨大なデータベースから瞬時に個人を識別することが可能である。特に NIST（米国国立標準技術研究所）の「顔認証ベンチマークテスト」(FRVT）において、複数回にわたりトップの精度を記録している[8]。2022 年のテストでは、誤認証率（False Non-Match Rate: FNMR）が 0.1%未満という驚異的な数値を達成した。この精度は、顔の向きや表情の変化、照明条件の違いなど、さまざまな要因にも対応し、安定した性能を維持している。また、高速処理能力も持ち合わせており、1 秒間に 100 万人以上の顔を照合できる性能がある。これにより、空港のセキュリティチェックや大規模イベントでのリアルタイム認証が求められる状況にも対応可能である。

この技術の進化により、顔認証は単なるセキュリティ用途にとどまらず、個人識別や健康管理など、より広範な分野に適用されている。例えば、東京オリンピック・パラリンピックでは、30 万人以上の大会関係者を対象にした顔認証システムが導入され、大規模な運営の支えとなった。コロナ禍では、非接触型認証が注目され、より安全で効率的な社会インフラの整備に貢献している。

このように、顔認証技術は人間社会で幅広く活用されているが、近年では動物個体の識別にも応用が進んでいる。

ニホンザルの個体識別の研究[9]では、ディープラーニングを活用したニホンザルの個体識別システムを開発した。42 個体を対象に約 15 時間の映像データを収集し評価実験を行い、検出精度 82.2%、識別精度 83%を達成した。

クマの個体識別の研究[10]では、個体ごとの明確な模様を持たないヒグマの識別を目的に、ニューラルネットワークによる顔検出、ResNet34 による特徴抽出を行った。また、SVM（サポートベクターマシン）による分類を組み合わせたクマ顔識別システム「BearID」を開発した。4,674 枚の画像で検証した結果、83.9%の識別精度を達成し、個体追跡や個体数推定、人間との軋轢管理などの生態学的応用が可能であることを示した。

2.2. 牛顔画像による個体識別

近年、動物の顔画像を用いた個体識別に関する研究が進められており、その一環として、牛の顔画像を利用した個体識別手法の開発が注目されている。

牛顔画像識別の研究[11]では、牛の個体識別精度を向上させるために、軽量な MobileNet を特徴抽出のニューラルネットワークとして採用した。また、牛顔検出モデルには RetinaFace と識別精度向上のための損失関数 ArcFace を統合した CattleFaceNet を提案した。従来の手法と比較した結果、ArcFace を採用した CattleFaceNet が 91.3%の識別精度を達成した。さらに、処理速度においては 24FPS を達成し、リアルタイムでの家畜管理やトレーサビリティ向上に貢献する可能性を示した。

モバイル端末の先行研究として、兒玉[4]は「CowFindAR」を提案し、その基本性能の評価を行った。CowFindAR は、クライアントサーバシステムを採用し、深層学習によって牛の顔画像から個体を識別し、個体識別番号を提示するシステムである。このシステムは、カメラ映像を取得しユーザに情報を提示するクライアントシステムと、深層学習を用いた画像識別を行うサーバシステムの 2 層構成となっている。クライアントシステムは画像をサーバシステムに送信し、サーバ側で識別処理を行い、その結果を返す。評価実験では、登録用 27 枚の画像を用意し、検出用 27 枚の識別精度を検証した。CowFindAR による撮影画像と、手作業で切り出した画像を識別器に入力し、識別性能の劣化度を比較した結果、撮影画像 26 枚のうち 53.8%を正しく識別し、切り出した画像を入力した場合は 61.5%の識別精度を示した。また、レスポンスタイムの計測では、牛顔検出に成功した場合の平均レスポンスタイムは 279 ミリ秒、失敗した場合は 183 ミリ秒であり、識別処理に要する時間は約 100 ミリ秒と推定された。

この研究では、モバイル端末単独でのリアルタイム識別は考慮されておらず、サーバを用いることで計算負荷を軽減している。しかし、モバイル端末単独での動作を実現するためには、軽量な識別モデルの導入や端末の処理能力に適したアルゴリズムの選定が必要となる。

2.3 モバイル端末による牛顔個体識別

CowFindAR の課題として、識別までのレスポンスタイムが遅いことが挙げられる。システムの高速化には通信を行う際のオーバーヘッドによる識別処理の速度の低下を改善しなければならない。

そのような現状を打破するために、本研究では、モバイル端末の単独動作による牛顔個体識別システムを提案する。このシステムは、カメラの映像を取得して YOLO[12]による牛顔検出を行う。牛顔が検出されると、切り出された牛顔を特徴抽出器に入力し、特徴ベクトルを出力する。この特徴ベクトルとデータベースに登録された特徴ベクトルの類似度を求め、最大の類似度を持つ個体番号を出力することで個体識別を行う。様々な特徴抽出器を使用して精度と速度を評価す

ることで、より実用化に適した最適なシステムの構築が可能となる。また、複数のモバイル端末上における精度・速度評価を行い、現状の機器上での有効性を調査する。

本システムにおいて、牛顔検出を行うモデルには、YOLOv8 を使用する。顔認識を行う手法としては、ArcFace[13]用いる。また、牛顔識別を行う特徴抽出器には、ResNet, MobileNet, EfficientNet を使用する。

2.4 ArcFace と特徴抽出器

2.4.1 ArcFace

ArcFace[13]は、顔識別の精度を高めるために設計された深層学習モデルである。このモデルは、顔の特徴を高次元の特徴ベクトルとして表現し、それを高次元ベクトル空間の超球の表面に配置する。特徴ベクトルの類似度は角度として計測され、さらにこの角度にマージン（余裕）を加えることで、異なる顔同士の距離を広げる。この手法により、ノイズの多いデータに対しても高い耐性を持ち、大規模な顔認証システムにおいても優れた性能を発揮する。こうした特性から、ArcFace は監視カメラやセキュリティシステムなど、さまざまな場面で活用されている。

この ArcFace の学習には、式（1）の Additive Angular Margin Loss という損失関数が用いられている。

$$-\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n} e^{s\,cos\theta_j}} \quad \cdots(1)$$

ここで、yiはサンプルiの真のクラスラベル、θjはサンプルiの特徴ベクトルとj番目のクラスの中心ベクトルとの角度、Nはミニバッチ内のサンプル数、nは学習時のクラス数である。sはスケールパラメータ、mはマージンパラメータで、どちらも学習時に設定されるハイパーパラメータである。この損失関数の目的は、特徴ベクトルのクラス内分散を小さくし、クラス間分散を大きくすることである。同じクラスの特徴ベクトル同士の角度を小さくし、異なるクラスの特徴ベクトル同士の角度を大きくすることで、クラス間の識別性を高めることを目指している。

ArcFace の構造を図 1 に示す。ArcFace はバックボーンとヘッドからなっている。バックボーンは、特徴抽出器の役割を持つ CNN が使用される。画像を入力すると特徴ベクトルが得られる。

入力画像を畳み込みニューラルネットワーク（CNN）に通し、特徴ベクトルを抽出する。次に、この特徴ベクトルをヘッドと呼ばれる全結合層に入力し、分類器へと送る。最終的に、分類器を用いて個体識別のための学習を行い、識別に適した特徴表現を得る。

識別では、学習時と同様に入力画像を CNN に通して特徴ベクトルを取得する。データベースの特徴ベクトルと比較し、コサイン類似度を用いて類似度を計算することで、個体識別を行う。

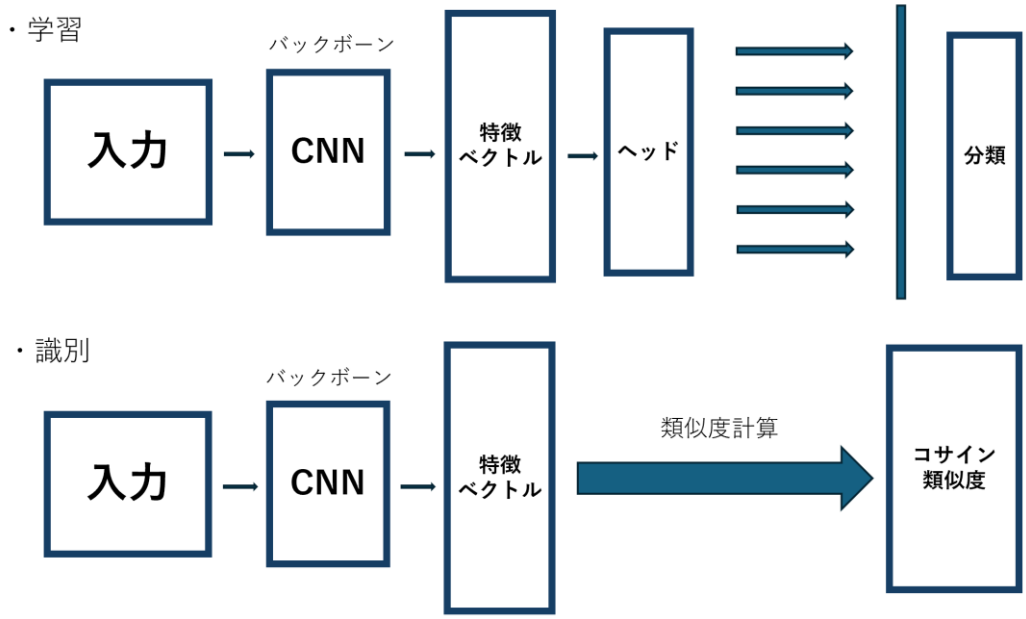


図 1: ArcFace の構造

2.4.2 特徴抽出器

本研究では、特徴抽出器に ResNet, MobileNet, EfficientNet を使用して、精度・速度の面から評価する。以下では、それぞれの特徴抽出器の概要について述べる。

ResNet とは、Microsoft Research が開発した深層学習モデルで、主に画像分類に使用されている。このモデルの特徴は、「残差接続」と呼ばれる仕組みを導入し、非常に深いネットワークでも効果的に学習できる点である。これにより、従来の深層学習モデルで問題となっていた「勾配消失問題」を軽減し、152 層以上の深いネットワークでも高い性能を発揮する。

MobileNet とは、モバイルデバイス向けに設計された軽量な畳み込みニューラルネットワーク（CNN）である。主に、計算リソースが限られた環境での使用を目的としており、深さ可変の畳み込み層を使用して、モデルのサイズと計算量を削減する。MobileNet は、特にリアルタイムの画像分類やモバイルアプリケーションでの利用に適しており、精度と効率のバランスを取ることが特徴である。

EfficientNet とは、Google が提案した CNN アーキテクチャで、モデルのスケーリングを系統的に行うことで高い精度を実現している。深さ、幅、解像度をバランスよく調整する複合係数を用いており、従来のモデルよりも少ないパラメータで高い性能を発揮する。特に、画像分類タスクにおいて最先端の精度を達成している。

2.5 YOLO

本研究では、牛の顔検出に高速かつ高精度な物体検出モデルである YOLO（You Only Look Once）[12] を使用する。YOLO は、画像全体を一度の処理で解析し、物体の位置（バウンディングボックス）と物体の種類を同時に予測できるため、一般的な物体検出手法と比べて速度が大幅に向上する特徴を持つ。また、検出した物体に対して物体の種類とその物体が存在する確率を同時に計算することで、誤認識を減らし精度を向上させることが可能である。これらの特性により、YOLO はリアルタイムでの物体検出に適している。

3. モバイル端末の単独動作による牛顔個体識別

3.1 システム全体の構成

本論文では、モバイル端末の単独動作による牛顔個体識別システムを構築し、評価実験を行う。

本システムは、カメラの映像を取得して YOLO[12]による牛顔検出を行う。牛顔が検出されると、切り出された牛顔を特徴抽出器に入力し、特徴ベクトルを出力する。この特徴ベクトルとデータベースに登録された特徴ベクトルの類似度を求め、最大の類似度を持つ個体番号を出力することで個体識別を行うシステムである。全体の基本的な構成、処理手順を以下に示す。

① ArcFace による事前学習
- 1-1 特徴抽出器の選択
- 1-2 学習率やエポック数の指定
- 1-3 学習の開始

② 登録
- 2-1　モバイル端末のカメラから画像を取得し、YOLO により牛顔検出
- 2-2　ArcFace により特徴量抽出
- 2-3　特徴ベクトルをデータベースに登録

③ 個体識別
- 3-1 モバイル端末のカメラから画像を取得し、YOLO により牛顔検出
- 3-2 検出された牛顔から ArcFace により特徴量抽出
- 3-3 抽出された特徴ベクトルとデータベースとの類似度を求める
- 3-4 個体識別番号と処理結果（類似度）をユーザに提示

本システムにおいて、深層学習の実装には NCNN[14] を使用する。NCNN とは、モバイル端末や組み込みデバイス向けに最適化された軽量で高速なニューラルネットワークライブラリである。特に Android などのリソースが限られた環境でも効率的に AI モデルを動作させることが可能である。C++ で実装されており、さまざまなプラットフォームでのコンパイルが可能なため、画像認識や物体検出などリアルタイム処理が求められるアプリケーションに適している。

一方、モデルの学習には PyTorch[15] を使用した。PyTorch は、機械学習やディープラーニングのためのフレームワークであり、柔軟なモデル構築と GPU による高速計算が可能である。また、timm[16]（Torch Image Models）という PyTorch 向けの画像認識モデルライブラリがあり、ResNet や EfficientNet などの事前学習済みモデルを簡単に利用できる。

PyTorch は主に研究や開発向けに設計されており、柔軟な機能を豊富に備えているのに対し、NCNN はスマートフォンや組み込み機器での使用を重視し、軽量で高速な動作を特徴としている。このため、本システムでは NCNN を使用する。

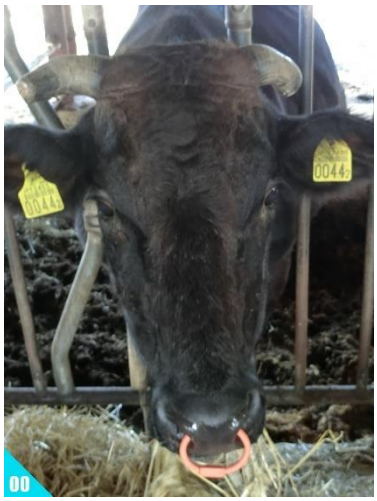
PC 上で行ったモデル学習では、PyTorch や timm を用いたため、出力ファイルの形式は「.pth」となっている。.pth ファイルは PyTorch の学習済みモデルの重みを保存する形式であり、そのままでは NCNN で使用できない。NCNN で利用するためには、ネットワーク構造を定義する「.param」と、モデルの重みを格納する「.bin」の 2 つのファイルが必要となる。しかし、PyTorch の .pth ファイルと NCNN の .param および .bin ファイルには互換性がないため、一度「.onnx」形式に変換し、それを経由して NCNN 用のファイル形式に変換を行った。

3.2 YOLO による顔検出

特徴ベクトルを抽出するには、牛の顔を適切に切り出す必要がある。これは、特徴抽出時に不要な背景を取り除き、識別精度を向上させるためである。切り出し前の画像を図 2、切り出し後の画像を図 3 に示す。

牛の顔を正確に切り出すためには、画像内の牛顔領域を特定する必要がある。この領域の検出には、物体検出アルゴリズムである YOLOv8 を使用した。

本研究では、リアルタイム処理において高速かつ効率的に動作することを重視し、軽量モデルである「YOLOv8s」を選択した。「s」は YOLOv8 の中でもスピードを重視した小型モデルであることを意味する。学習には、「牛全体」と「牛顔」の 2 つのクラスにラベルを付けた 5921 枚の画像を使用し、バッチサイズ 32 で 50 エポックの訓練を実施した。



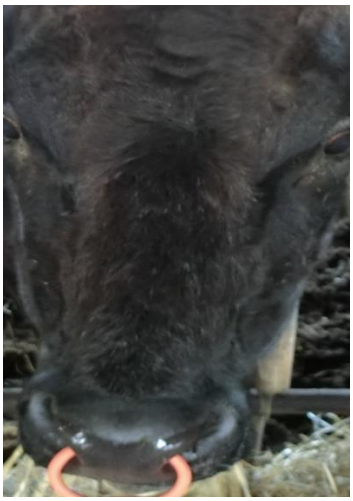


図 2: 牛顔切り出し前の画像　図 3: 牛顔切り出し後の画像

3.3 ArcFace による顔識別

3.3.1 特徴抽出器の選択

本研究では、様々な特徴抽出器間でどのような違いがあるのかを評価することを目的とし、ResNet152、ResNet34、ResNet18、MobileNetV3_Small、MobileNetV3_Large、MobileNetV4_Small、EfficientNet_B0 を使用した。これらの特徴抽出器について、フロップ数や特徴ベクトルの次元数などの比較結果を、以下の表 1 に示す。

モデルの評価においては、ImageNet が広く用いられる。ImageNet は、物体認識のための大規模な画像データセットであり、約 1400 万枚以上の画像と 1000 種類以上のクラスがラベル付けされている。特に、ImageNet Large Scale Visual Recognition Challenge（ILSVRC）は、モデルの性能を測る基準として知られており、Top-1 エラーと Top-5 エラーが評価指標として使われる。Top-1 エラーは、モデルが最も高い確率で予測したクラスが実際のクラスと一致しない割合を示し、Top-5 エラーは、モデルが最も高い確率で予測した上位 5 つのクラスに実際のクラスが含まれない割合を示す。

また、モデルの性能を評価する際には、精度だけでなく計算効率や複雑さも重要な要素となる。その指標として、フロップ数（FLOPs）やパラメータ数がある。フロップ数は、モデルが 1 回の推論で行う浮動小数点演算の回数を示し、FLOPs が少ないほど計算リソースを抑えた効率的なモデルと言える。一方、パラメータ数は、モデルが学習する重みやバイアスの数であり、多いほどモデルの表現力が向上し、高精度な予測が可能になるが、その分計算コストも増加する。

表 1: 各特徴抽出器の情報

モデル名	フロップ数（約）	パラメータ数（約）	特徴ベクトル次元数	Top1 エラー (%)	Top5 エラー (%)
ResNet152	11.3GFLOPs	60.2M	2048	21.69	5.94
ResNet34	3.6GFLOPs	21.8M	512	26.70	8.58
ResNet18	1.8GFLOPs	11.7M	512	30.24	10.92
MobileNetv3_small	66MFLOPs	2.5M	1024	32.40	12.50
MobileNetv3_large	219MFLOPs	5.4M	1280	24.80	7.50
MobileNetv4_small	-	2.52M	1280	-	-
EfficientNet_b0	390MFLOPs	5.3M	1280	22.90	6.70

3.3.2. モデルの学習

特徴抽出器のモデル学習では、ArcFace [13] のバックボーンとして、ResNet152、ResNet34、ResNet18、MobileNetv3_small、MobileNetv3_large、MobileNetv4_small、EfficientNet_b0 の 7 種類を選定し、それぞれに対して学習を行った。バックボーンとは、入力された顔画像から特徴抽出するための深層畳み込みニューラルネットワーク（CNN）の部分を指す。学習データには、宮崎県内の農場で撮影された牛顔画像を使用し、3,148 クラス、合計 5,997,217 枚のデータを用いた。学習の設定として、学習回数を 60 エポックに設定した。ハイパーパラメータには式（1）のmとsがある。個体ごとの特徴を明確に分離するため、クラス間マージンmを 0.05 に設定し、識別精度を向上させつつ、過剰な分離が発生しないようにした。さらに、特徴ベクトル間の類似度を強調するためにスケーリングファクターsを 30.0 に設定した。

3.3.3 特徴ベクトルの抽出

特徴ベクトルの抽出は、入力画像から特定の領域を切り出すことから始まる。まず、牛顔を含む矩形領域を左上の座標（x, y）を基準に幅と高さを指定して抽出し、この部分画像を保存する。次に、ニューラルネットワークが適切に処理できるよう、画像を 224×224 ピクセルにリサイズする。その後、ピクセル値の平均を引き算して正規化を行う。これにより、モデルの入力に適した形式へと整えられた画像が特徴抽出の処理へと進む。ニューラルネットワークは、この前処理

済み画像を基に、特徴抽出器を用いて解析し、特徴ベクトルとして出力する。この特徴ベクトルの次元数は使用する特徴抽出器に依存し、最終的に牛の個体識別に利用される。

3.3.4. 牛顔画像の識別

牛顔画像の識別には、データベースに格納されたそれぞれの特徴ベクトルと YOLO により検出し、特徴抽出器によって抽出された特徴ベクトルとのコサイン類似度を求める。 データベース内の特徴ベクトルとコサイン類似度が最大となった個体識別番号を出力することで識別を行う。

コサイン類似度は、式（2）で示される。2 つのベクトルの類似度を表す指標で、結果は -1 から 1 の範囲で出力される。値が-1 に近いほど類似性が低く、1 に近いほど類似性が高いことを意味する。

$$Cosine\ Similarity(A,B) = \frac{A \cdot B}{||A||||B||} \quad \cdots(2)$$

3.4. アプリの使用方法

3.4.1 牛顔画像の登録

本システムでは、牛顔画像の登録を行い、個体識別番号と特徴ベクトルを保存し、管理する必要があるためデータベースを用いる。ファイルが文字とカンマで区切られており、軽量であるため、データベースのファイル形式として CSV ファイルを使用する。

モバイル端末上で牛顔画像から個体識別を可能にするため、牛顔検出後、個体識別に必要な特徴量を抽出する処理を 100 回繰り返す。特徴抽出は、安定した角度から行えるように 100 回ごとに行い、識別時には検出された角度から安定して実施できるように工夫した。抽出された特徴量と個体識別番号は、0 から順に配列に上書きされ、データベースに登録される(図 4)。

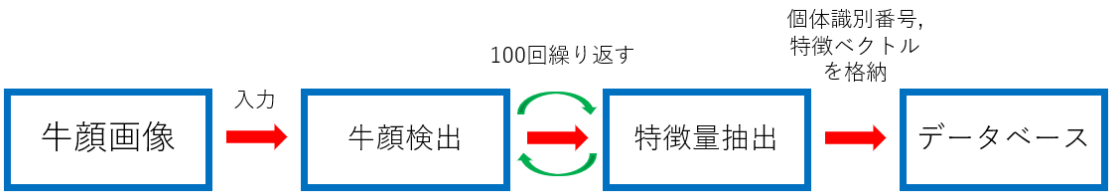


図 4: 仮登録機能

実装されたシステムによる登録前と登録後の様子を以下に図 5、図 6 に示す。図中の青枠は検出した牛顔領域を表す。青枠下の「cow_id」が識別結果の個体識別番号、「detect」が識別結果のコサイン類似度を示している。図中の牛の個体識別番号は「0」である。図 5 は、登録する前であり、登録用画像の番号と違う個体識別番号が表示されている。図 6 のように登録後は、登録用画像の番号と同じ個体識別番号が表示されていることが分かる。

図 5：登録完了前

図 6：登録完了後

3.4.2. 牛顔画像の識別

牛顔画像の識別は、登録された牛顔画像の特徴ベクトルと、検出された牛顔画像の特徴ベクトルのコサイン類似度を算出し、データベース内の情報と比較することで行われる。最も高い類似度を持つ個体の識別番号が出力され、識別結果として表示される。識別の様子を以下に図 7 に示す。図 7 は、識別用画像を使用し、識別結果として個体識別番号と検出された牛とのコサイン類似度を表示している。

識別を正確に行うためには、まず牛の顔を適切に検出する必要がある。正面からの撮影で検出がされない場合は、異なる角度から撮影する必要がある。

図 7：識別結果の出力

4. 複数の端末・特徴抽出器での評価実験

4.1. 実験方法

本実験では、複数の端末および特徴抽出器を使って、牛顔画像の登録から識別までの精度・速度評価を実施した。実験には、Pixel 8a[17]、Pixel 7a[18]、Zenfone 8[19]の 3 種類のスマートフォンを使用した。実験で使用した各端末の処理性能や特性を比較するために、それぞれのスペックを表 2 に示す。シングルコアスコア、マルチコアスコアはともに Geekbench6[20]のスコアを参照している。シングルコアスコアは、スマートフォンの 1 つのコアでどれだけ速く計算できるかを示す数値で、普段の操作や軽いアプリの動作に影響を与える。また、マルチコアスコアは、複数のコアを同時に使ったときの計算の速さを示す数値で、ゲームや動画編集などの重い作業の快適さに関係する。スコアは高いほどよいため、性能が良い順番としては、Pixel 8a > Zenfone 8 > Pixel 7a となる。

表 2：端末のスペック

項目	Pixel 8a	Pixel 7a	Zenfone 8
CPU	Google Tensor G3	Google Tensor G2	Qualcomm® Snapdragon 888 5G Mobile Platform
マルチコアスコア	4194	3320	3716
シングルコアスコア	1639	1359	1502

本研究では、11 頭の牛について、登録用と識別用としてそれぞれ 1 枚ずつ、重複しないよう牛顔画像を選択した。選択した画像を紙に印刷し、スマートフォンで撮影する。各牛顔画像には 00～10 の番号を割り当て、これを個体識別番号とする。登録用画像と識別用画像は、同じ番号が同じ個体を表している。登録用の画像を図 8、識別用の画像を図 9 に示す。登録用の画像は左下に赤色、識別用の画像は青色で番号を表示している。画像を識別するために、登録用の牛には「R」、識別用の牛には「C」を付け、その後に 00～10 の番号を連結する。例えば、5 番目の登録用牛顔画像は「R05」と表記する。

評価実験のシナリオを以下に示す。

①登録用の牛顔画像に関して、番号順に登録する。

②登録用の牛顔画像に関して、番号順に識別する。

③識別用の牛顔画像に関して、番号順に識別する。

④識別用の牛顔画像に関して、正面からできなかった場合に様々な角度から識別する。

精度評価においては、識別用の牛顔画像に関してそれぞれ牛顔の検出に成功するまで撮影する。正面から検出されない場合は、牛顔画像をやや見下ろす形で撮影する。それでも識別されない場合は様々な角度から撮影する。どの角度から検出された場合であっても、成功とみなす。

速度評価においては、FPS とレスポンスタイムの面から評価する。FPS（Frames Per Second）とは、1 秒間に処理できる画像の数を示す指標である。特に、リアルタイム処理においては重要な指標であり、値が高いほどスムーズな動作が可能になる。また、本研究におけるレスポンスタイムは、牛顔画像の検出を開始点とし、特徴抽出、コサイン類似度の計算を経て、最も高い類似度と対応する個体識別番号が出力されるまでの時間とする。評価実験における識別処理を進める際の特徴抽出を行ったときの FPS とレスポンスタイムを記録し、それぞれの平均値を評価指標として用いる。

図 8： 登録用牛顔画像

図 9: 識別用牛顔画像

4.2. 精度評価

本研究では、複数の端末・特徴抽出器における牛顔個体識別の精度を評価した。評価には、PC、Pixel 8a、Pixel 7a、Zenfone 8 の 4 種類の端末を使用し、ResNet152、ResNet34、ResNet18、MobileNetv3_small、MobileNetv3_large、MobileNetv4_small、EfficientNet_b0 の合計 7 つの特徴抽出器の識別精度を比較した。結果を表 3 に示す。

PC においては、すべてのモデルが 90%以上の識別精度を達成し、特に ResNet152 以外のモデルでは 100%の精度を記録した。一方、モバイル端末ではモデルごとに精度のばらつきがある。

ResNet152 は PC で 90.91%の精度を示したものの、モバイル端末では Pixel 8a が 72.72%、Pixel 7a が 81.82%、Zenfone 8 が 54.55%と、端末による影響を受けやすい結果となった。対照的に、MobileNet、EfficientNet といった軽量モデルは、モバイル端末でも比較的高い精度を維持した。特に EfficientNet_b0 はすべての端末で 100%の精度を達成し、計算リソースの制限がある環境でも高い識別性能を示した。

以上の結果から、PC では計算量の多い ResNet のモデルも高精度で動作するが、モバイル端末では軽量な特徴抽出器の方が高い識別精度を維持しやすいことが確認された。特に EfficientNet_b0 は、PC・モバイル端末のどちらにおいても安定した高精度を示しており、モバイル環境での個体識別に適したモデルであると考えられる。

表 3：識別結果

モデル名	PC（%）	Pixel8a（%）	Pixel7a（%）	Zenfone8（%）
ResNet152	90.91	72.72	81.82	54.55
ResNet34	100.00	90.90	81.82	100.00
ResNet18	100.00	90.90	100.00	90.90
MobileNetv3_small	100.00	100.00	100.00	90.90
MobileNetv3_large	100.00	100.00	90.90	81.82
MobileNetv4_small	100.00	81.82	90.90	90.90
EfficientNet_b0	100.00	100.00	100.00	100.00

4.3. FPS 評価

図 10 は、異なる端末（Pixel 8a、Pixel 7a、Zenfone 8）と異なる特徴抽出器（EfficientNet_b0、MobileNetv3_small、MobileNetv3_large、MobileNetv4_small、ResNet152、ResNet34、ResNet18）における FPS（Frames Per Second）の比較を示した棒グラフである。このグラフでは、横軸に各端末と特徴抽出器の組み合わせ（例: pixel8a_effi_b0, pixel7a_ResNet152 など）、縦軸に FPS の値を配置している。各棒の高さは FPS の値を表し、値が高いほど高速な処理が可能であることを示す。

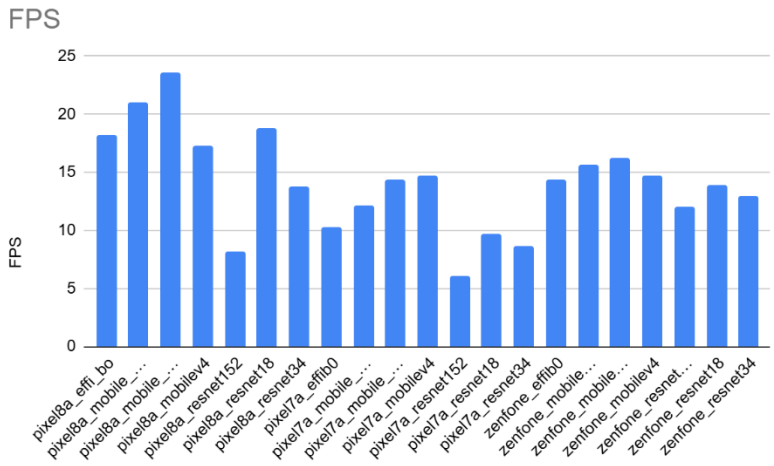


図 10: FPS

図 11 は、異なる端末（Pixel 8a、Pixel 7a、Zenfone 8）の間における FPS の平均を示している。Pixel 8a が最も高い平均 FPS を記録しており、次いで Zenfone 8、最後に Pixel 7a が続いている。この結果から、モバイル端末の性能順にパフォーマンスが良いことが分かる。特にグラフィック性能やリアルタイム処理を必要とするタスクにおいて、Pixel 8a は選択肢となる。一方で、Pixel 7a のパフォーマンスが他の 2 つに比べて劣っている。

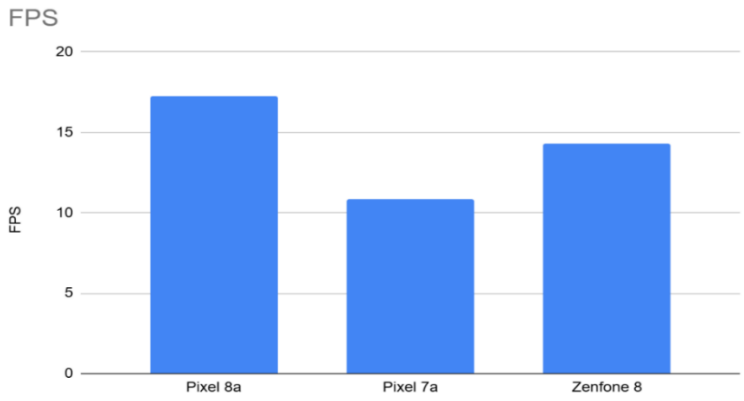


図 11：端末ごとの平均 FPS

図 12 は、特徴抽出器ごとの平均 FPS を示す。MobileNetv3 small が最も高い FPS を示しており、軽量で高効率な特徴抽出を可能にしている。一方、ResNet152 は最も低い FPS を記録しており、計算量が多いために処理速度が低下していると考えられる。これに対して、MobileNet や EfficientNet は高い効率性を発揮しており、特にリアルタイム性が重要なシステムに適していると考えられる。

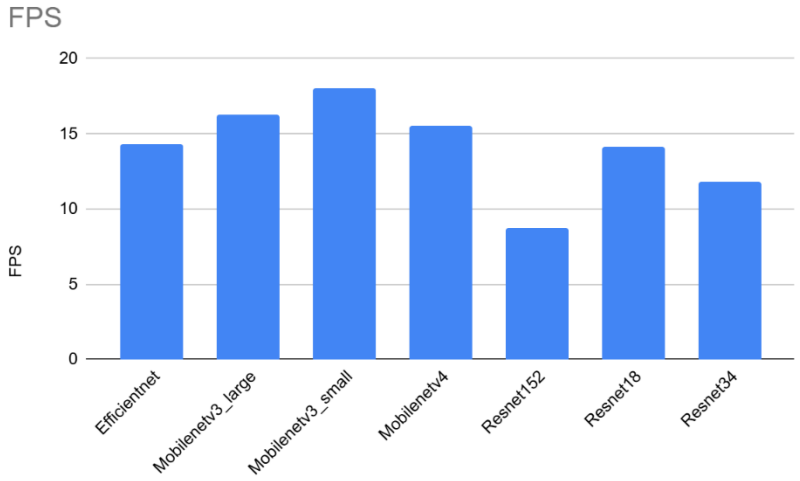


図 12:　特徴抽出器ごとの平均 FPS

4.4. レスポンスタイム

図 13 は、異なる端末（Pixel 8a、Pixel 7a、Zenfone 8）と異なる特徴抽出（EfficientNet_b0、MobileNetv3_small、MobileNetv3_large、MobileNetv4_small、ResNet152、ResNet34、ResNet18）におけるレスポンスタイムの比較を示した棒グラフである。このグラフは、横軸に各端末とネットワークモデルの組み合わせ（例: pixel8a_effi_b0, pixel7a_ResNet152 など）、縦軸にレスポンスタイムの値を配置している。各棒の高さはレスポンスタイムの値を表し、値が高いほど処理に時間がかかっていることを示している。評価実験において、ResNet152 のレスポンスタイムが非常に長く、同じスケールで比較できないため、ResNet152 を含まない結果を図 14 に示した。

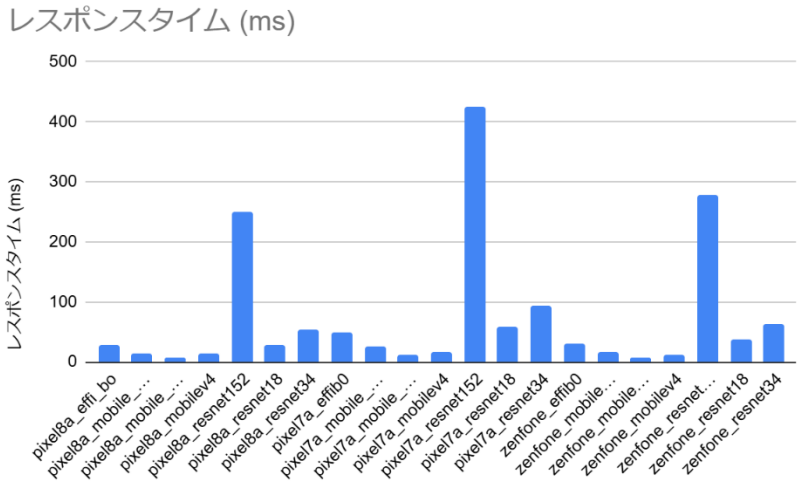


図 13: レスポンスタイム

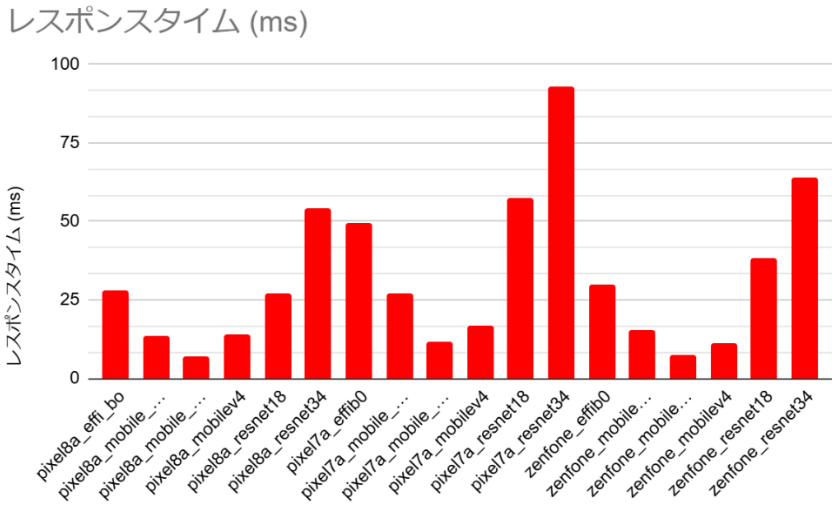


図 14: レスポンスタイム（ResNet152 を除く）

端末の処理能力が高いほどレスポンスタイムは短縮される傾向にあり、軽量なモデルほど高速な処理が可能となる。一方、ResNet のような深層モデルでは計算負荷が高く、レスポンスタイムが大幅に増加する。

図 15 は、Pixel 8a、Pixel 7a、Zenfone 8 の平均レスポンスタイム（ms）を比較したものである。Pixel 7a が約 100 ミリ秒と最も遅いレスポンスタイムを示している。一方、Pixel 8a と Zenfone 8 は約 60 ミリ秒前後と短く、Pixel 7a よりも高速であることが明らかである。さらに、スペック表を参考にすると、Pixel 8a はマルチコアスコアとシングルコアスコアで最も高い値を記録しており、その短いレスポンスタイムと一致する。対照的に、Pixel 7a は性能スコアが最も低いため、レスポンスタイムも最も長い。また、Zenfone 8 はスコアが Pixel 8a より劣るものの、Pixel 8a に近いレスポンスタイムを実現している。

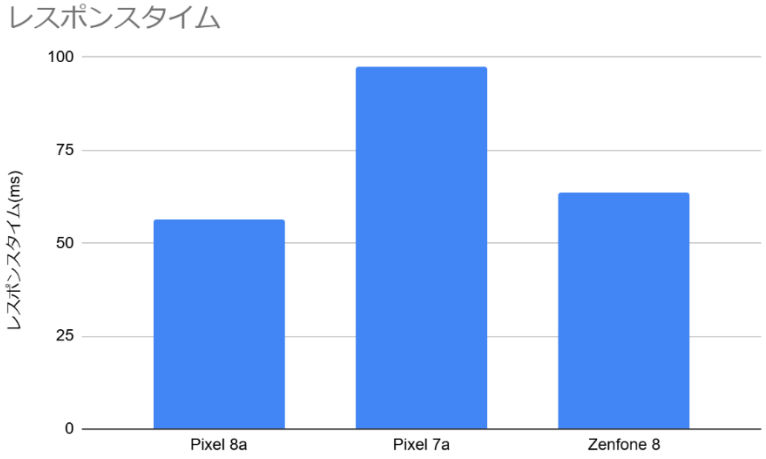


図 15：端末ごとの平均レスポンスタイム

図 16 は、ResNet152 を除いた特徴抽出器ごとの平均レスポンスタイム（応答時間）を示している。ResNet34 が最長時間を記録しており、処理効率が低いことを示している。一方で、MobileNetv3_small は最短時間を示し、EfficientNet や MobileNet も比較的短い時間で処理が可能である。この結果から、リアルタイム性が要求されるシステムでは MobileNetv3_small が最適である。

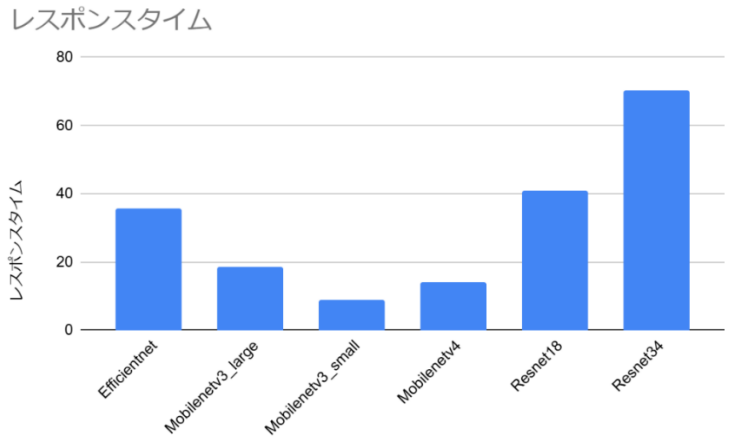


図 16:　特徴抽出器ごとの平均レスポンスタイム（ResNet152 を除く）

4.5. 考察

端末性能が高いほど処理速度が向上し、特に軽量モデル（EfficientNet や MobileNet）はパフォーマンスを維持しやすい傾向がある。Pixel 8a は最も高い性能スコアを持ち、レスポンスタイムや FPS において優れた結果を示しており、Tensor G3 チップが軽量・大規模モデル両方で高い性能を発揮する。Pixel 7a は性能が低く、深層モデル使用時に負荷が大きく、レスポンスタイムが遅くなる。Zenfone 8 は Snapdragon 888 の性能により、Pixel 7a より深層モデルで優位に立ち、レスポンスタイムも約 60 ミリ秒で良好である。

MobileNet は軽量で FPS とレスポンスタイムが高いが、精度は ResNet や EfficientNet に劣る。EfficientNet は高精度を保ちながら安定した性能を発揮し、実用性において有効である。

実用面において、特徴抽出器には高精度・中速である EfficientNet が最適である。EfficientNet を採用した場合、最も遅い Pixel 7a 上でも 10FPS 以上のフレームレートが出ており、遅延を気にせず使用できた。このことから、現状のミドルレンジのスマートフォン上であっても、提案システムは速度・精度の両面から有効に動作すると言える。

5. おわりに

本論文では、モバイル端末の単独動作による牛顔個体識別システムを提案した。

評価実験では、スマートフォンのカメラで撮影した牛顔画像を基に、深層学習モデルを活用して牛個体を検出・識別し、様々な特徴抽出器と複数の端末における有効性を調査した。さらに、登録機能については、個体識別番号順に登録できることを確認した。評価実験では、異なる端末や特徴抽出器における識別精度、FPS、レスポンスタイムの傾向を分析した。

実験結果から、MobileNet は軽量なネットワークであるため FPS が高く、レスポンスタイムも短いが、ResNet や EfficientNet と比較すると識別精度が低下する場合があることが判明した。一方で、EfficientNet は高精度を維持しつつ、処理速度も比較的高速であり、安定した識別性能を発揮することから、現時点での実用性において有効なモデルであると考えられる。

また、モバイル端末間においては、端末性能が高いほど処理速度が向上する。そのため、スペックが高い Pixel 8a > Zenfone 8 > Pixel 7a の順に現時点での実用性において有効なモバイル端末であると考えられる。

今後の課題としては、データベースへの登録件数が増加した際の識別処理の最適化や、高速計算アルゴリズムの導入を検討する必要がある。また、実用性を向上させ、ユーザにとって利便性の高いシステムを実現するためには、登録機能の改良が求められる。さらに、本システムの畜産農業施設への導入を見据えた運用試験を行い、実環境下での有効性や課題を明確にすることも今後の重要な取り組みとなる。

謝辞

本研究を行うにあたり、ご指導・ご協力いただいた椋木雅之教授に深く感謝申し上げます。指導教員である椋木雅之教授には、研究に関する相談やアドバイスなど、丁寧かつ熱心にご指導いただきました。

ご指導・ご協力いただいた皆様や、共に切磋琢磨した椋木研究室のメンバーの皆様に深く感謝を申し上げるとともに、皆様のご健勝とますますのご活躍をお祈り申し上げます。

参考文献

[1] 農林水産省,「スマート農業」, https://www.maff.go.jp/j/heya/sodan/17009/02.html (2025 年 2 月 4 日アクセス).
[2] cntxts Inc., 「Smart Cattle®」, https://smartcattle.net/home/index (2025 年 2 月 4 日アクセス).
[3] 株式会社ファームノート, 「クラウド牛群管理システム『Farmnote Cloud』」, https://farmnote.jp/cloud/ (2025 年 2 月 4 日アクセス).
[4] 兒玉光平,「CowFindAR:牛顔個体識別を用いたモバイル端末向け管理情報提示システム」, 宮崎大学大学院工学研究科 修士論文, 2021.
[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, 2016.
[6] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for MobileNetV3," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1314-1324, 2019.
[7] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks" *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 6105-6114, 2019
[8] NEC, 「顔認証ベンチマークテスト（FRVT）でトップの精度を記録」, https://jpn.nec.com/press/202402/20240208_01.html (2025 年 2 月 4 日アクセス).
[9] J. Paulet, A. Molina, B. Beltzung, T. Suzumura, S. Yamamoto, and C. Sueur, "Deep Learning for Automatic Facial Detection and Recognition in Japanese Macaques: Illuminating Social Networks," *Primates*, vol. 65, no. 4, pp. 265-279, 2023.
[10] M. Clapham, E. Miller, M. Nguyen, and C. T. Darimont, "Automated Facial Recognition for Wildlife that Lack Unique Markings: A Deep Learning Approach for Brown Bears," *Ecology and Evolution*, vol. 10, no. 22, pp. 12883-12892, 2020.
[11] B. Xu, W. Wang, L. Guo, G. Chen, Y. Li, Z. Cao, and S. Wu, "CattleFaceNet: A Cattle Face Identification Approach Based on RetinaFace and ArcFace Loss," *Computers and Electronics in Agriculture*, vol. 193, Article 106675, 2022.
[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779-788, 2016.
[13] J. Deng, J. Guo, J. Yang, N. Xue, I. Kotsia, and S. Zafeiriou, "ArcFace: Additive Angular Margin Loss for Deep Face Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 5962-5979, 2022.
[14] Tencent,「ncnn: High-performance neural network inference framework」, GitHub repository, Available at: https://github.com/Tencent/ncnn (2025 年 2 月 4 日アクセス).

[15] PyTorch, 「An open source machine learning framework」, Available at: https://pytorch.org/ (2025 年 2 月 4 日アクセス).
[16] R. Wightman, 「timm: PyTorch Image Models」, GitHub repository, Available at: https://github.com/rwightman/pytorch-image-models (2025 年 2 月 4 日アクセス).
[17] Google, 「Pixel 8a」, Available at: https://store.google.com/product/pixel_8a (2025 年 2 月 4 日アクセス).
[18] Google, 「Pixel 7a」, Available at: https://store.google.com/product/pixel_7a (2025 年 2 月 4 日アクセス).
[19] ASUS, 「Zenfone 8」, Available at: https://www.asus.com/Phone/Zenfone-8/ (2025 年 2 月 4 日アクセス).
[20] Geekbench, 「Geekbench 6」, Available at: https://www.geekbench.com/ (2025 年 2 月 4 日アクセス).