



**AGH**

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

---

---

# System wykrywania i identyfikacji zwierząt na podstawie obrazu z kamery monitoringu

Krzysztof Dudek, Maciej Pieniążek, Weronika Hilaszek

Kraków, 22 października 2024

# **Spis treści**

<b>1</b>	<b>Wprowadzenie</b>	<b>1</b>
<b>2</b>	<b>Cel projektu</b>	<b>2</b>
2.1	Zastosowania . . . . .	2
<b>3</b>	<b>System</b>	<b>3</b>
3.1	Architektura systemu . . . . .	3
<b>4</b>	<b>Detekcja i identyfikacja</b>	<b>5</b>
4.1	Zbiór danych . . . . .	5
4.2	Przygotowanie danych . . . . .	5
4.3	Wykorzystany model . . . . .	7
<b>5</b>	<b>Aplikacja</b>	<b>9</b>
5.1	Komponenty aplikacji . . . . .	9
5.2	Korzystanie z aplikacji . . . . .	9
5.3	Rezultaty predykcji . . . . .	10
<b>6</b>	<b>Podsumowanie</b>	<b>12</b>
6.1	Otrzymane rezultaty . . . . .	12
6.2	Wnioski . . . . .	13
6.3	Możliwe dalsze kierunki rozwoju . . . . .	13

# 1 Wprowadzenie

Wraz ze wzrostem liczby obszarów chronionych oraz dążeniem do lepszego zrozumienia eko-systemów, potrzeba zaawansowanych systemów monitoringu dzikiej przyrody staje się coraz bardziej istotna. Nasz projekt ma na celu stworzenie systemu wykrywania i identyfikacji zwierząt na podstawie obrazu z kamery monitoringu, który wykorzystuje modele głębokiego uczenia.

System ten jest zaprojektowany z myślą o monitorowaniu dzikiej fauny w polskich lasach, umożliwiając precyzyjne rozpoznawanie i śledzenie różnych gatunków zwierząt. Kluczowym elementem projektu jest integracja wyników detekcji z aplikacją webową, co pozwala na łatwe zarządzanie danymi i szybką reakcję na wykryte incydenty.

Nasza aplikacja wykorzystuje metody analizy obrazu, w tym techniki uczenia głębokiego, aby zapewnić wysoką skuteczność w identyfikacji zwierząt. Do oceny efektywności systemu używamy precyzyjnych metryk, takich jak precision, recall oraz średnia precyzja (mAP). Dzięki temu możemy optymalizować działanie naszego modelu i dostosowywać go.

Projekt składa się z trzech głównych komponentów: modułu predykcji wideo, części serwerowej/brokerowej oraz systemu odstraszania. Moduł predykcji wideo odpowiedzialny jest za analizę obrazu w czasie rzeczywistym, serwer pełni rolę węzła komunikacyjnego, a system odstraszania, zbudowany na platformie Raspberry Pi, reaguje na wykryte incydenty, emitując odpowiednie dźwięki.

W naszej dokumentacji podzieliliśmy treść na kilka kluczowych sekcji, aby szczegółowo przedstawić każde zagadnienie projektu. Po wprowadzeniu, w sekcji 2 omówiliśmy pokrótkę istotę tego projektu i możliwe zastosowania. Kolejna sekcja, 3 opisuje projekt i architekturę systemu. Następnie, w sekcji 4, opisaliśmy model wykorzystywany do wykrywania zwierząt, w tym również dane wykorzystywane do uczenia i sposób ich przygotowania. Sekcja 5 poświęcona jest aplikacji, w tym omówieniu frontendu i backendu. Ostatnia sekcja, 6, prezentuje otrzymane wyniki i szczegółową analizę, wyciągnięte wnioski oraz zaproponowane przez nas możliwe dalsze kierunki rozwoju projektu.

## 2 Cel projektu

Celem projektu jest opracowanie systemu wykrywania i identyfikacji zwierząt na podstawie obrazów z kamer monitoringu, wykorzystującego zaawansowane techniki przetwarzania obrazu i głębokiego uczenia, aby umożliwić dokładną identyfikację gatunków zwierząt w ich naturalnym środowisku. Dodatkowym celem jest implementacja modułu odstraszania, który reaguje na wykryte incydenty, emitując dźwięki mające na celu odstraszenie zwierząt, co może prowadzić do zwiększenia bezpieczeństwa, m.in. w sytuacji interakcji między ludźmi a zwierzętami lub sytuacji zagrożenia dla zwierząt.

### 2.1 Zastosowania

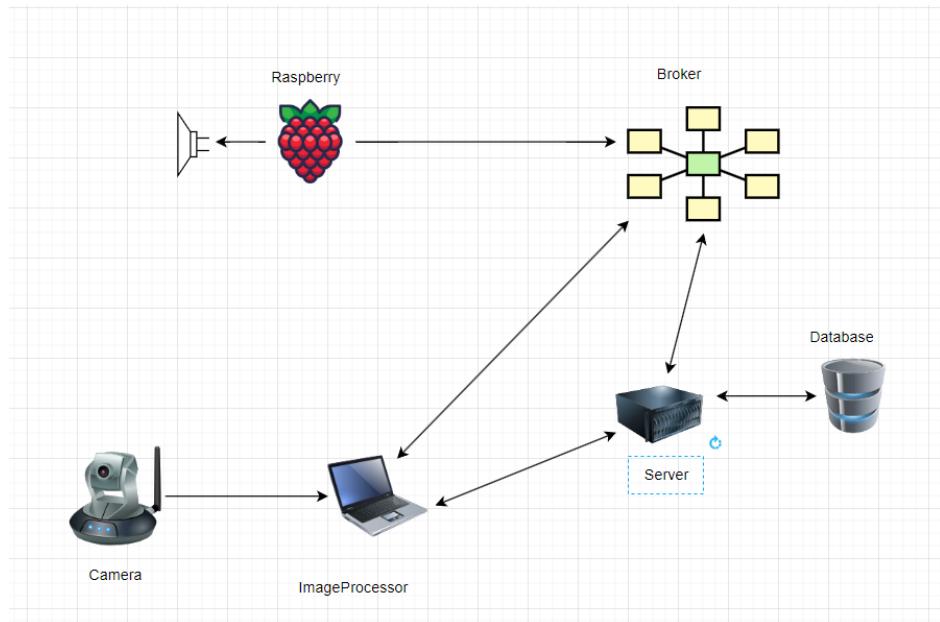
- **Monitorowanie przyrody:** Śledzenie populacji i zachowań zwierząt w rezerwatach i parkach narodowych.
- **Ochrona gatunków:** Identyfikacja zagrożonych gatunków i monitorowanie ich liczebności.
- **Rolnictwo i leśnictwo:** Monitorowanie dzikich zwierząt w celu ochrony upraw i lasów oraz stosowanie modułu odstraszania do zabezpieczenia tych obszarów przed szkodnikami.
- **Bezpieczeństwo:** Detekcja i identyfikacja zwierząt w pobliżu dróg i infrastruktury, aby zapobiegać wypadkom oraz wykorzystanie systemu odstraszania, aby minimalizować ryzyko kolizji.
- **Edukacja i badania:** Narzędzie do badań naukowych i edukacji, wspomagające w analizie dzikiej przyrody, a także w ocenie efektywności metod odstraszania.

# 3 System

## 3.1 Architektura systemu

Cały system składa się z 3 głównych komponentów:

- Predykcji wideo
- Części serwerowej/brokerowej
- Systemu odstraszania



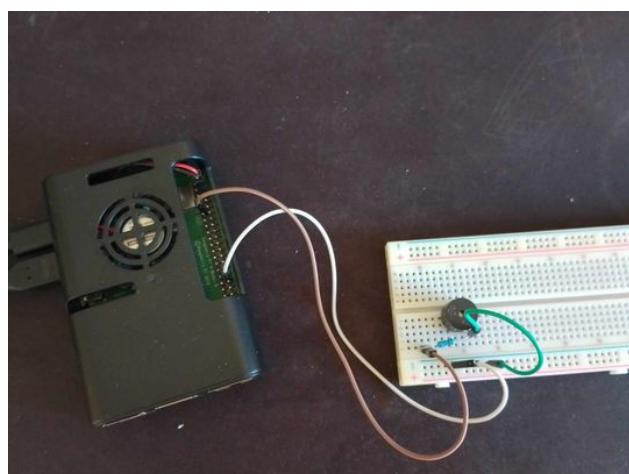
Ilustracja 1: Poglądowa architektura systemu

**Predykcja wideo** obejmuje kamerę wideo wraz z uruchomionym oprogramowaniem predykcyjnym, które ma na celu aktywną analizę obrazu w czasie rzeczywistym. W przypadku detekcji, oprogramowanie archiwizuje klatki, zapisując istotne momenty wykryte na nagraniu. Dodatkowo, odpowiada ono za przesyłanie informacji o rozpoczęciu i zakończeniu detekcji do brokera, co pozwala na natychmiastową reakcję systemu na zdarzenia. Po zakończeniu archiwizacji, tworzy pełne wideo z zapisanych klatek i przesyła je do backendu, zapewniając dostęp do materiału wideo dla dalszej analizy i przetwarzania. Dzięki temu system jest w stanie efektywnie zarządzać danymi wideo i utrzymywać wysoką jakość monitoringu.

**Serwer** stanowi węzeł komunikacyjny pomiędzy pozostałymi komponentami systemu, zapewniając płynny przepływ informacji. Przechowuje on ustawienia predykcji wideo oraz listę możliwych do detekcji klas, umożliwiając łatwe zarządzanie konfiguracją systemu. Dodatkowo, serwer odpowiada za przechowywanie nagrani z incydentów, wzbogaconych o dodatkowe dane kontekstowe, co ułatwia późniejszą analizę. Komunikuje się również z brokerem, zapewniając, że wszystkie zdarzenia i aktualizacje są natychmiast przekazywane pomiędzy

komponentami. Serwer pełni więc kluczową rolę w integracji wszystkich elementów systemu i utrzymaniu jego sprawnego działania.

**System odstraszania** składa się z płytki Raspberry Pi 4b z podłączonym głośnikiem, co pozwala na emitowanie dźwięków odstraszających. Na uruchamianym na płytce programie łączymy się z brokerem, nasłuchując wiadomości o incydentach, aby odpowiednio na nie reagować. Po odebraniu informacji o incydencie, system aktywuje głośnik, emitując dźwięk mający na celu odstraszenie intruza lub zwrócenie uwagi na zdarzenie. Dodatkowo, system może być skonfigurowany do rejestrowania reakcji na incydenty, co umożliwia analizę skuteczności podjętych działań. Dzięki takiej konfiguracji, system odstraszania jest zarówno elastyczny, jak i skuteczny w reagowaniu na potencjalne zagrożenia.



Ilustracja 2: Zdjęcie poglądowe płytka z podłączonym głośnikiem

**Podsumowanie:** cały system, składający się z serwera, modułu predykcji wideo oraz systemu odstraszania, został zaprojektowany z myślą o elastyczności, łatwości implementacji i możliwości rozwoju, co pozwala na integrację nowych funkcji oraz dostosowanie do zmieniających się potrzeb. Nie jest to finalny produkt, lecz demonstracyjny prototyp, który ilustruje potencjał i skuteczność takiego typu systemów w praktycznych zastosowaniach.

## 4 Detekcja i identyfikacja

### 4.1 Zbiór danych

W celu trenowania i testowania naszego systemu wykrywania i identyfikacji zwierząt na obrazach z kamery monitoringu, wykorzystaliśmy zbiory danych przygotowane przy użyciu platformy Roboflow. Roboflow to narzędzie do zarządzania danymi obrazowymi i adnotacjami, które ułatwia przygotowanie, augmentację oraz eksport zbiorów danych do różnych formatów, w tym formatu YOLO. Dane zostały wyszukane w publicznych datasetach, a następnie podzielone na wybrane przez nas klasy, łącznie wykorzystano ponad 2000 zdjęć. Każdy element w zbiorze danych składa się z:

- **obrazu:** Kolorowego zdjęcia w formacie JPEG/PNG, przedstawiającego różne gatunki zwierząt w ich naturalnym środowisku.
- **adnotacji:** Plików w formacie YOLO zawierających informacje o lokalizacji zwierząt na obrazach (bounding boxes) oraz etykiety klas identyfikujące gatunki zwierząt.

Zbiór danych obejmuje następujące klasy zwierząt:

1. Człowiek	7. Krowa	13. Królik
2. Pies	8. Kura	
3. Jeleń/Sarna	9. Lis	14. Jeż
4. Koń	10. Orzeł	15. Niedźwiedź
5. Kot	11. Wiewiórka	
6. Wilk	12. Ryś	16. Dzik

W jednej konfiguracji modelu usunięto następujące klasy: człowiek, krowa, kura, koń.

### 4.2 Przygotowanie danych

Proces przygotowania danych obejmował kilka kluczowych kroków:

- **Anotacja obrazów:** Ręczne oznaczanie lokalizacji zwierząt na obrazach za pomocą narzędzi do anotacji, w tym narzędzi oferowanych przez platformę Roboflow.
- **Weryfikacja jakości:** Sprawdzanie poprawności i jakości anotacji oraz usuwanie obrazów o niskiej jakości.
- **Augmentacja danych:** Zastosowanie technik augmentacji danych za pomocą funkcji dostępnych w Roboflow, aby zwiększyć różnorodność zbioru treningowego. W każdej

konfiguracji zastosowano resize do 640x640 oraz automatyczne dostosowanie kontrastu; w dwóch konfiguracjach zastosowano również odwracanie obrazu, dodano noise oraz blur.

#### 4.2.1 Przykłady obrazów

Poniżej znajdują się przykładowe obrazy ze zbioru danych, po wykonaniu augmentacji i nałożeniu etykiet:



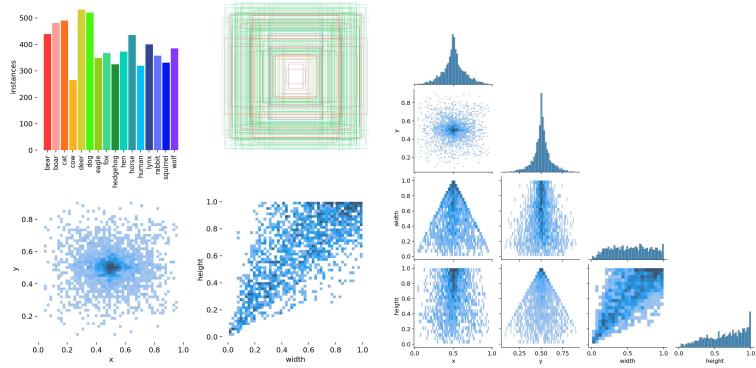
Ilustracja 3: Przykładowe obrazy zwierząt z naszego zbioru danych.

#### 4.2.2 Podział danych

Zbiór danych został podzielony na trzy części:

- **Zbiór treningowy:** obrazy używane do trenowania modelu, zazwyczaj 70-80% zbioru.
- **Zbiór walidacyjny:** obrazy używane do walidacji i dostrajania modelu podczas treningu, zazwyczaj ok. 10-15% zbioru.
- **Zbiór testowy:** Używany do oceny ostatecznej wydajności modelu po zakończeniu treningu, zazwyczaj 10-15% zbioru.

Poniżej prezentujemy statystyki w przypadku pierwszej konfiguracji modelu, zawierającej największy zbiór danych:



Ilustracja 4: Statystyki naszego zbioru danych.

### 4.3 Wykorzystany model

W naszym projekcie do wykrywania i identyfikacji zwierząt na obrazach z kamery monitoringu wykorzystaliśmy model YOLOv8, który jest nowoczesnym i zaawansowanym narzędziem w detekcji obiektów. YOLOv8 został wybrany ze względu na jego wydajność, szybkość i wysoką dokładność.

#### 4.3.1 Konfiguracje modelu

W ramach projektu przetestowaliśmy różne konfiguracje modelu YOLOv8, aby ocenić ich wydajność i dostosować model do naszych specyficznych potrzeb. Poniżej przedstawiono szczegóły trzech z tych konfiguracji:

- **Konfiguracja 1:**

- Model: YOLOv8m (medium)
- Dane: Pełny zbiór danych
- Augmentacje: Resize do 640x640, automatyczne dostosowanie kontrastu, odwracanie obrazu, dodanie szumu (noise) i rozmycia (blur)
- Cel: Zrównoważona wydajność i dokładność

- **Konfiguracja 2:**

- Model: YOLOv8s (small)
- Dane: Pełny zbiór danych
- Augmentacje: Resize do 640x640, automatyczne dostosowanie kontrastu
- Cel: Szybkie przetwarzanie z akceptowlaną dokładnością

- **Konfiguracja 3:**

- Model: YOLOv8s (small)
- Dane: Pełny zbiór danych minus klasy człowiek, krowa, kura, koń
- Augmentacje: Resize do 640x640, automatyczne dostosowanie kontrastu
- Cel: Szybkie przetwarzanie z akceptowlą dokładnością

#### 4.3.2 Proces uczenia modelu

Proces uczenia modelu składał się z kilku kroków:

1. **Przygotowanie danych:** Dane zostały załadowane i podzielone na zbiory treningowy, walidacyjny i testowy. Wykonano augmentacje, aby zwiększyć różnorodność zbioru treningowego.
2. **Trening:** Model trenowano przy użyciu pretrenowanego modelu YOLOv8 na zbiorze danych COCO. Następnie przeprowadzono dalsze trenowanie na naszym zbiorze danych, dostosowując hiperparametry, takie jak liczba epok oraz rozmiar batcha.
3. **Walidacja:** Po każdej epoce treningowej przeprowadzano walidację modelu, monitorując metryki takie jak `box_loss` (lokalizacja obiektów), `cls_loss` (klasyfikacja obiektów), `dfl_loss` (przewidywanie lokalizacji obiektu) oraz średnia precyza (mAP).
4. **Testowanie:** Po zakończeniu treningu model był testowany na zbiorze testowym, aby ocenić jego ostateczną wydajność.

Poniżej znajduje się kod przykładowego uruchomienia treningu modelu YOLOv8:

```
!yolo task=detect mode=train model=yolov8s.pt data={dataset.location}/data.yaml epochs=5
```

W powyższym przykładzie, model YOLOv8s (small) jest trenowany przez 50 epok na zbiorze danych z obrazami o rozmiarze 640x640 pikseli. Parametr `batch` oznacza liczbę obrazów przetwarzanych jednocześnie podczas jednej iteracji treningu, co wpływa na szybkość i stabilność procesu uczenia. Proces ten został powtórzony dla innych konfiguracji modeli (YOLOv8m), dostosowując odpowiednio parametry.

# 5 Aplikacja

## 5.1 Komponenty aplikacji

Do projektu stworzyliśmy prostą aplikację webową do obsługi detekcji i wyświetlania nagrani.

### 5.1.1 Frontend

Napisany w Angularze 19, pełni rolę interfejsu użytkownika, umożliwiając łatwe i intuicyjne korzystanie z aplikacji. Komunikuje się z backendem w celu wyświetlania wideo oraz aktualnie rozpoznawanych klas, co pozwala na bieżąco śledzić wyniki analizy. Daje również możliwość dynamicznego zmieniania klas do detekcji poprzez porozumiewanie się z backendem, który następnie propaguje informacje o zmianach do brokerka, zapewniając aktualność danych. Automatycznie, co określony krok czasowy, odpytuje backend o nowe wideo, co zapewnia stały dostęp do najnowszych informacji i pozwala na monitorowanie w czasie rzeczywistym.

### 5.1.2 Backend

W naszej aplikacji backend został napisany w Pythonie przy użyciu frameworka Django, co zapewnia solidne fundamenty do dalszego rozwoju. Jego główną rolą jest serwowanie plików wideo i dostarczanie API do komunikacji z frontendem oraz z systemem detekcji wideo, umożliwiając płynną i efektywną wymianę danych. Wykorzystaliśmy przy tym standard REST, dzięki czemu łatwo jest zintegrować różne komponenty i dodawać nowe funkcjonalności bez większych trudności. Backend posiada bazę danych, w której przechowuje informacje o występujących detekcjach, co pozwala na efektywne zarządzanie i przetwarzanie danych. Ponadto, daje możliwość aktualizacji klas do detekcji, co jest kluczowe dla elastyczności i adaptacyjności systemu. Dzięki komunikacji z brokerem, wydarzenia w systemie są przekazywane natychmiastowo, co znacznie usprawnia ogólną architekturę i poprawia szybkość reakcji na zmiany. Taki sposób implementacji zapewnia nie tylko funkcjonalność, ale także skalowalność i łatwość w utrzymaniu systemu.

## 5.2 Korzystanie z aplikacji

Część serwerowa aplikacji jest skonteneryzowana, co zapewnia łatwość w stawianiu i zarządzaniu środowiskiem. W naszym przypadku, zdecydowaliśmy się na skorzystanie z usług Google Cloud do hostowania aplikacji na maszynie wirtualnej. Maszyna ta posiadała publiczny adres IP, dzięki czemu wszyscy członkowie zespołu mogli na bieżąco monitorować stan aplikacji podczas jej rozwoju. To rozwiązanie umożliwiło szybkie wdrażanie poprawek oraz zapewniło dostępność aplikacji dla wszystkich zainteresowanych stron.

### 5.3 Rezultaty predykcji

Na podstawie widoku aplikacji przedstawionego poniżej można zobaczyć, że aplikacja wyświetla szczegóły dotyczące incydentów wykrytych przez system. Każdy incydent zawiera następujące informacje:

- **Numer incydentu:** Unikalny numer identyfikujący każdy incydent.
- **Wykryty obiekt:** Klasa obiektu, który został zidentyfikowany, np. "cow".
- **Data i godzina:** Dokładna data i godzina, kiedy incydent został zarejestrowany.
- **Video z detekcją:** Fragment wideo przedstawiający wykryty obiekt z zaznaczonymi obszarami detekcji.

Ponadto, interfejs użytkownika umożliwia zarządzanie flagami (klasami) detekcji po prawej stronie ekranu. Użytkownik może włączać lub wyłączać detekcję poszczególnych klas zwierząt, co jest przedstawione w postaci listy z opcjami wyboru (checkboxami).

#### Incydent numer 81

Wykryty obiekt: **cow**

Data: **20.06.2024**

Godzina: **09:02:15**



#### Manage Flags

- bear
- boar
- cat
- cow
- deer
- dog
- eagle
- fox
- hedgehog
- hen

Ilustracja 5: Widok aplikacji.

### 5.3.1 Predykcja w kolejnych klatkach

System jest zaprojektowany tak, aby śledzić obiekty w kolejnych klatkach wideo. Dzięki temu, obiekt, który porusza się w ramach nagrania, jest ciągle oznaczony poprawnie w każdej kolejnej klatce. Umożliwia to dokładne śledzenie ruchu obiektów i zapewnia, że identyfikacja jest spójna na całej długości wideo. Na podstawie poniższych zrzutów widać, że obiekt (np. człowiek) poruszający się w kolejnych klatkach wideo jest cały czas prawidłowo oznaczany. Ta funkcjonalność jest kluczowa dla monitorowania w czasie rzeczywistym i analizy zachowań.

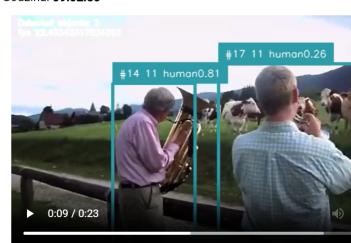
**Incydent numer 82**

Wykryty obiekt: **human**  
Data: **20.06.2024**  
Godzina: **09:02:50**



**Incydent numer 82**

Wykryty obiekt: **human**  
Data: **20.06.2024**  
Godzina: **09:02:50**



**Incydent numer 82**

Wykryty obiekt: **human**  
Data: **20.06.2024**  
Godzina: **09:02:50**



Ilustracja 6: Predykcje w kolejnych klatkach nagrania.

# 6 Podsumowanie

## 6.1 Otrzymane rezultaty

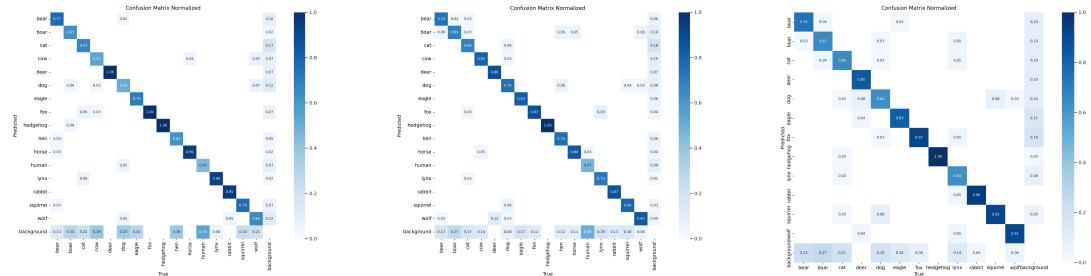
W ramach projektu opracowaliśmy system wykrywania i identyfikacji zwierząt na podstawie obrazów z kamery monitoringu, wykorzystując model YOLOv8. Przeprowadziliśmy szereg eksperymentów z różnymi konfiguracjami modelu, aby ocenić ich wydajność i dokładność. Otrzymane wyniki:

Konfiguracja modelu	mAP	Precision	Recall
YOLOv8m (medium)	80.08%	86.1%	75.2%
YOLOv8s (small)	84.0%	86.8%	75.5%
YOLOv8s (small)	85.4%	88.5%	78.2%

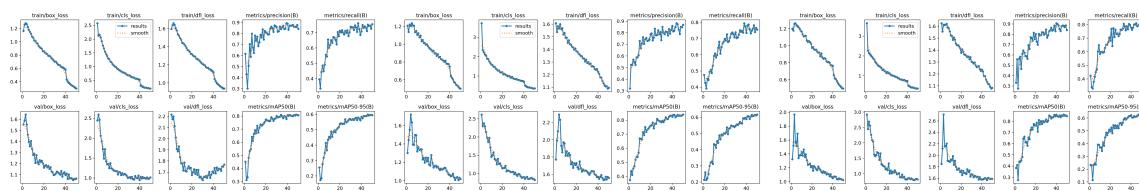
Tabela 1: Wyniki uzyskane dla różnych konfiguracji modelu YOLOv8.

Model w wersji small z pomniejszonym zbiorem danych o 4 klasy sprawdził się najlepiej, otrzymał najlepsze wyniki jeżeli chodzi o precyzję i uczył się w czasie 5 razy mniejszym, niż model w wersji medium z augmentacją danych. Wersja small jest również najlepsza w kwestii przetwarzania w czasie rzeczywistym, co jest istotne dla naszego projektu. W przyszłości, mając dostęp do odpowiednich zasobów obliczeniowych można również wypróbować model w wersji x ze znacznie większą ilością klas i zróżnicowanymi dnaymi.

Poniżej prezentujemy również statystyki porównawcze dla 3 konfiguracji modelu (w kolejności takiej jak w tabeli):



Ilustracja 7: Znormalizowana macierz pomyłek dla różnych konfiguracji modelu YOLOv8.



Ilustracja 8: Wyniki wszystkich metryk

## 6.2 Wnioski

Na podstawie przeprowadzonych eksperymentów możemy wyciągnąć następujące wnioski:

- Wybór odpowiedniej konfiguracji modelu zależy od specyficznych wymagań aplikacji. YOLOv8s jest najlepszy do zastosowań w czasie rzeczywistym.
- Usunięcie niektórych klas (np. człowiek, krowa, kura, koń) w konfiguracji YOLOv8s wpłynęło na zrównoważoną wydajność, umożliwiając bardziej precyzyjną detekcję pozostałych gatunków zwierząt.
- Różne konfiguracje modelu wykazały, że można znaleźć kompromis między szybkością przetwarzania a dokładnością detekcji, dostosowując model do specyficznych potrzeb i ograniczeń sprzętowych.

## 6.3 Możliwe dalsze kierunki rozwoju

Projekt może być dalej rozwijany w następujących kierunkach:

- **Rozszerzenie zbioru danych:** Zwiększenie liczby obrazów oraz klas zwierząt, co pozwoli na trenowanie modelu na bardziej zróżnicowanych i reprezentatywnych danych.
- **Optymalizacja modelu:** Dalsza optymalizacja parametrów modelu, takich jak liczba epok, rozmiar batcha i szybkość uczenia, aby jeszcze bardziej poprawić dokładność i wydajność.
- **Model YOLOv8:** Wybróbowanie większych wersji modelu YOLOv8, w tym wersji l oraz x w celu poprawy dokładności.
- **Integracja z systemami monitoringu:** Implementacja i testowanie systemu w rzeczywistych warunkach monitoringu przyrody, w celu oceny jego skuteczności i niezawodności w praktyce.
- **Rozwój aplikacji webowej:** Udoskonalenie aplikacji webowej poprzez dodanie funkcji takich jak np. interaktywne raporty.