Při chovu slepic by bylo velmi užitečné vědět, kolik vajec která slepice snáší, protože to umožňuje farmáři sledovat produktivitu jednotlivých slepic. Díky tomu může:

* **Optimalizovat chov**: Identifikací slepic s vysokou a nízkou snáškou může farmář rozhodnout o selekci nebo o zvláštní péči pro méně produktivní slepice.
* **Zlepšit zdraví slepic**: Nízká snáška může být indikátorem zdravotních problémů. Včasná identifikace umožňuje podniknout kroky k léčbě nebo prevenci nemocí.
* **Efektivněji plánovat krmení a zdroje**: Sledování výkonu pomáhá v rozhodování o výživě a péči, aby byla maximalizována produkce vajec při optimálních nákladech.
* **Zvýšit ziskovost**: Monitorováním snášky lze zlepšit celkovou produktivitu farmy a tím i její ekonomickou efektivitu.

Celkově tedy znalost počtu vajec od každé slepice pomáhá v efektivním řízení chovu a zajišťuje lepší výsledky jak po stránce produkční, tak ekonomické.

Abych toho dosáhl, potřebuji být schopen jednotlivé slepice identifikovat a tuto informaci propojit s momentum, kdy slepice sense v kurniku vejce.

V sekci pridat odkaz “Vaha”, je popsan mechanismus, kdy pravidelnou analyzou casove rady, kterou poskytuje vaha ziskame informaci o casove udaji, kdy bylo vejce sneseno.

Pak jiz jen staci identifikovat, ktera z nasich slepic byla v tu dobu v hnízdě. Existuje několik variant, jak slepice identifikovat, já jsem se rozhodl využít k identifikaci obrazovou analýzu.

Postupuji následujícím způsobem.

Nejprve procházím videozáznam a identifikuji oblasti, kde se slepice nacházejí. Tyto oblasti následně vystřihnu a uložím do samostatných souborů. Poté procházím tyto menší obrázky a pomocí segmentace vystřihnu plochu, kde je slepice, zatímco ostatní části zůstanou bílé.

Takto vyříznuté obrázky převádím na tenzory. Převod na tenzor znamená, že dvourozměrný obraz (matice pixelů) převedu do vícerozměrné datové struktury, která dokáže reprezentovat různé vlastnosti obrazu a je vhodná pro strojové učení a další matematické zpracování. Tímto způsobem získám numerickou reprezentaci obrazu, se kterou mohu efektivně pracovat.

Získané tenzory porovnávám se vzorovými vektory uloženými v databázi. Pokud je vektor shodný, znamená to, že jsem našel odpovídající slepici, a podařilo se mi ji tak identifikovat.

Script nosnice\_2.py

 Tento kód slouží k segmentaci objektů na obrázcích pomocí modelu YOLO (You Only Look Once) verze 8 pro segmentaci. Níže je shrnutí jeho funkcionality:

1. **Importy**:
   * Importuje knihovny potřebné pro práci s obrázky (OpenCV, NumPy) a pro použití modelu YOLO.
   * Importuje funkci list\_jpg\_files z modulu reidentification.util, která pravděpodobně vrací seznam všech souborů s příponou .jpg z daného adresáře (tato funkce však není v kódu definována).
2. **Funkce apply\_mask a apply\_mask2**:
   * Obě funkce aplikují masku na obrázek.
   * apply\_mask2 používá metodu cv2.bitwise\_and pro aplikaci masky.
   * apply\_mask vytváří výstupní obrázek s bílým pozadím a pak kopíruje části původního obrázku tam, kde se maska aplikuje, zajišťuje, že maska je binární.
3. **Funkce segment\_countour**:
   * Inicializuje model YOLO pro segmentaci pomocí natrénované sady váhy yolov8x-seg.pt.
   * Načte seznam všech obrázků s příponou .jpg pomocí funkce list\_jpg\_files.
   * Pro každý obrázek v seznamu:
     + Načte obrázek pomocí OpenCV.
     + Provede segmentaci pomocí modelu YOLO.
     + Extrahuje masky a souřadnice ohraničujících rámečků, konkrétně se zaměřuje na třídu označenou jako 'chicken' (třída s ID 14 v COCO datasetu).
     + Pokud je důvěra (confidence) vyšší než 0.5, aplikuje masku na obrázek a výsledek uloží jako nový obrázek s příponou .png.
4. **Spuštění skriptu**:
   * Skript je spuštěn voláním funkce segment\_countour s argumentem d:\\kurnik\\chicken, což by měl být adresář obsahující obrázky k analýze.

Pokud to shrneme, skript analyzuje obrázky, vyhledává objekty, které pravděpodobně představují kuřata, a segmentuje je pomocí masky, přičemž výstup ukládá jako nový obrázek s aplikovanou maskou.

Script nosnice\_chicken\_cliper.py

**Shrnutí funkcionality:**

1. **Načtení videí:** Skript načte všechny video soubory s příponou .mp4 nebo .avi z určeného adresáře.
2. **Inicializace modelu:** Pro každé video inicializuje YOLO model a připraví video pro zpracování.
3. **Zpracování snímků:**
   * Čte snímky z videa.
   * Každých save\_every (24) snímků provede detekci kuřat.
   * Použije model YOLO k detekci a sledování kuřat (třída 14) s prahovou hodnotou důvěry 0.5.
   * Detekovaná kuřata vystřihne ze snímku a uloží je jako samostatné obrázky.
   * Celé snímky může také ukládat, pokud je to implementováno v metodě save\_image\_frame (v aktuálním kódu se tato metoda nevolá).
4. **Ukládání výsledků:**
   * Vystřižená kuřata jsou uložena do adresářů pojmenovaných podle stream\_id (identifikátor videa) a track\_id (identifikátor sledovaného objektu).
   * Snímky jsou pojmenovány podle stream\_id, track\_id a aktuálního čísla snímku.
5. **Zobrazení výsledků:**
   * Detekovaná kuřata jsou zobrazena v okně s názvem "Chicken". Uživatel může sledovat proces detekce v reálném čase.
   * Program umožňuje ukončení zpracování stisknutím klávesy "q".
6. **Ukazatele výkonu:**
   * Skript kontroluje dostupnost GPU a v případě možnosti využívá akceleraci pomocí CUDA.
   * Omezí zpracování na maximální počet snímků definovaný v self.max\_frame\_count (15 minut videa při 24 FPS).

**Poznámky:**

* **Adresářová struktura:** Výstupy jsou uloženy v podadresářích uvnitř self.output\_path. Pro každé video (stream\_id) a sledovaný objekt (track\_id) je vytvořen samostatný adresář.
* **Předpoklady:** Skript předpokládá, že máte nainstalované potřebné knihovny (ultralytics, torch, opencv-python) a že model yolov9e.pt je dostupný v aktuálním adresáři nebo v cestě, kterou specifikujete.
* **Třída objektu:** V parametru classes=14 může číslo 14 odpovídat specifické třídě v trénovacím datasetu modelu YOLO (např. kuře). Ujistěte se, že číslo třídy odpovídá objektu, který chcete detekovat.

Script v soc .. nosnice\_4.py

Tento skript načítá obrázky z dané složky, předzpracovává je, extrahuje jejich rysy pomocí předtrénovaného modelu ResNet, shlukuje tyto rysy pomocí algoritmu K-means a poté kopíruje obrázky do jednotlivých složek podle jejich příslušnosti k danému shluku.

**Krok za krokem:**

1. **Importy a závislosti**:
   * Načítáme knihovny: os, torch, torchvision, PIL, sklearn, a shutil. Tyto knihovny jsou potřeba pro práci s obrázky, zpracování dat, extrakci rysů a manipulaci s adresáři.
2. **Funkce load\_and\_preprocess\_images(image\_folder)**:
   * Vytváří transformační sekvenci pro předzpracování obrázků: změna velikosti, konverze na tensor a normalizace.
   * Prochází všechny soubory v zadané složce image\_folder a zkontroluje, zda mají obrázkový formát (např. .png, .jpg, .jpeg).
   * Načítá a konvertuje obrázky na tensory, přičemž každý zpracovaný soubor ukládá do seznamu image\_tensors a jeho cestu do image\_paths.
3. **Funkce extract\_features(image\_tensors, model)**:
   * Přijímá seznam tensorů a model.
   * Využívá model bez poslední klasifikační vrstvy k extrakci rysů z každého obrázku.
   * Vrátí rysy obrázků jako tensor.
4. **Funkce cluster\_images(features, num\_clusters=5)**:
   * Převádí extrahované rysy do numpy pole pro použití v knihovně sklearn.
   * Používá algoritmus K-means ke shlukování obrázků do num\_clusters shluků.
   * Vrátí seznam s indexy shluků pro každý obrázek.
5. **Funkce copy\_images\_to\_clusters(image\_paths, clusters, output\_folder)**:
   * Pro každý shluk vytvoří samostatnou složku v output\_folder (př. cluster\_0, cluster\_1, atd.).
   * Zkopíruje každý obrázek do odpovídající složky podle příslušnosti ke shluku.
   * Zajišťuje, že složky pro každý shluk jsou vytvořeny, pokud neexistují.
6. **Hlavní funkce main()**:
   * Definuje vstupní a výstupní složky a počet shluků (num\_clusters).
   * Zavolá funkce pro načtení a zpracování obrázků, extrakci rysů a shlukování.
   * Nakonec zkopíruje obrázky do složek podle shluků.

A group of chickens with numbers

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A chicken standing next to a chicken

AI-generated content may be incorrect.A chicken standing on its hind legs

AI-generated content may be incorrect.A chicken standing on the ground

AI-generated content may be incorrect.