V posledních letech došlo k významnému pokroku v oblasti strojového vidění, což umožňuje efektivní a přesné rozpoznávání objektů v různých prostředích. Jednou z nejmodernějších knihoven, která si získala širokou popularitu, je YOLOv8 (You Only Look Once version 8). Tato knihovna přináší rychlé a přesné metody detekce objektů a je ideální pro implementaci v reálném čase. V této sekci se zaměříme na to, jak využít YOLOv8 ke specifickému úkolu: rozpoznávání slepic v kurníku.

Automatizované rozpoznávání umožňuje sledování počtu slepic. Použití YOLOv8 přináší následující výhody:

1. **Rychlost a Přesnost**: YOLOv8 je navrženo pro rychlou detekci objektů s vysokou mírou přesnosti, což je ideální pro aplikování v reálném čase na farmách.
2. **Jednoduchost Implementace**: Díky snadnému rozhraní a rozsáhlé dokumentaci je integrace YOLOv8 do existujících systémů velmi přímočará.
3. **Flexibilita**: Můžete trénovat model na vlastním datasetu slepic, což zajistí optimální rozpoznávání i v specifických podmínkách vašeho kurníku.

V následujících kapitolách si podrobně projdeme kroky, jak připravit dataset, trénovat vlastní model se zaměřením na rozpoznávání slepic a jak jej implementovat do našeho systému monitorování.

Obrázkový dataset

Obrázkový dataset je strukturovaná kolekce obrazových dat, která se používána pro potřeby strojového učení a počítačového vidění. Tyto datasety obsahují jednotlivé obrázky, které jsou často spojeny s dodatečnými informacemi nebo anotacemi, jež jsou potřebné pro trénink modelů. Níže jsou uvedeny hlavní aspekty, které charakterizují obrázkové datasety:

1. **Obrázky**: Základní komponentou datasetu jsou samotné snímky, které mohou být v různých formátech (např. JPEG, PNG). Rozlišení a počet barev se může lišit v závislosti na účelu datasetu.
2. **Anotace**: Kromě samotných obrázků obsahují datasety take anotace. Tyto anotace zahrnují následující informace
   * **Klasifikační štítky**: Které identifikují, co se na obrázku nachází (např. „kočka“, „pes“, „slepice“).
   * **Ohraničující boxy**: Které označují polohu specifických objektů na obrázku.

Datasety jsou klíčové pro vývoj a zlepšování modelů strojového učení, protože poskytují potřebné tréninkové a testovací údaje. Příklady známých obrázkových datasetů zahrnují ImageNet, COCO (Common Objects in Context), a MNIST (pro ručně psané číslice).

V kontextu využití datasetů pro strojové učení je důležité zajistit, aby datasety byly kvalitní, rozmanité a dostatečně rozsáhlé, což pomáhá modelům převážně se zobecněním a výkonem v různorodých situacích.

Pro specifické nasazení v babičině kurníku jsem připravil vlastní dataset, který jsem použil pro doučení již existujícího základního modelu, který poskytuje Yolo8. Záběry jsem vyfotil mobilem v prostředí babiččina výběhu a kurníku.

Existuje mnoho systému pro tvorbu anotaci anglicky image labeling. Já jsem zvolil Azure AI - Azure Machine Learning studio. Pro práci je velmi intuitivní a nechá spustit v několika krocích.

Nejdříve jsem pomocí průvodce vytvořil nový projekt.

A screenshot of a computer

AI-generated content may be incorrect.

Dále jsem vybral zdroj obrázků, které bylo potřeba klasifikovat. Vzhledem k tomu, že se jedná o cloudovou službu bylo třeba obrázky slepiček na Azure naimportovat.

A screenshot of a computer

AI-generated content may be incorrect.

V dalším kroku jsem nadefinoval kategorie, které budu chtít v obraze identifikovat. V mém případě jsem zvolil label „slepice“

A screenshot of a computer

AI-generated content may be incorrect.

V následujícím kroku následovala nejméně zábavná činnost. Bylo třeba projít jednotlivé snímky a ručně vybrat oblasti, kde já jako člověk vidím slepici. Oblast, kterou jsem označil se pak následně ukládá jako metadata k obrázku.

A screenshot of a computer

AI-generated content may be incorrect.

Když se podařilo projít celý obrázkový dataset, mohl jsem vyexportovat „labeling“ data. Jak jsem psal výše, mít kvalitní učící dataset je důležité, proto jsem exportoval pouze data, která se podařilo úspěšně označit. Do fotogalerie se mi dostalo i několik nejasných snímků, případně těch, kde slepice nebyla vůbec. Takové snímky jsem vyřadil. Anotace jsem exportoval ve formátu COCO => dat odkaz do slovniku

Formát COCO (Common Objects in Context) je standardní formát pro anotaci obrázků používaný ve strojovém učení a počítačovém vidění. Jeho hlavním cílem je poskytnout strukturu pro uložení informací o objektech v obrázcích, což umožňuje efektivní trénink a hodnocení modelů detekce objektů, segmentace a dalších úloh počítačového vidění.

**A screenshot of a document

AI-generated content may be incorrect.**

Soubor obsahuje několik klíčových sekcí: images, annotations, a categories.

Zde je popis jednotlivých částí daného JSON souboru ve formátu COCO:

1. **images**: Tato sekce obsahuje informace o každém obrázku v datasetu. Pro tento konkrétní dataset je k dispozici pouze jeden obrázek:
   * id: Unikátní identifikátor obrázku (zde 1).
   * width a height: Rozměry obrázku (zde 534 pixelů x 291 pixelů).
   * file\_name: Cesta k souboru obrázku včetně názvu souboru.
   * coco\_url, absolute\_url: Adresy, kde je obrázek uložen, ať už v cloudovém sandboxu (coco\_url) nebo přes URL pro přímý přístup (absolute\_url).
   * date\_captured: Čas a datum zachycení obrázku v UTC formátu.
2. **annotations**: Tato sekce popisuje anotace neboli poznámky k jednotlivým objektům na obrázcích. Zde je uvedena jedna anotace:
   * id: Unikátní identifikátor anotace (zde 1).
   * category\_id: Odkazuje na kategorii objektu (zde 1, což odpovídá "slepice").
   * image\_id: ID obrázku, na který se tato anotace vztahuje.
   * area: Relativní plocha objektu na obrázku (zde 0.016). Tato hodnota je často normalizována vzhledem k velikosti obrázku.
   * bbox: Vektor čtyř čísel, který definuje obdélník (bounding box) okolo objektu na obrázku. Hodnoty jsou obvykle normalizované, takže se pohybují mezi 0 a 1 a odpovídají relativním souřadnicím a rozměrům (x, y, šířka, výška).
3. **categories**: Tato sekce obsahuje seznam kategorií, do kterých mohou objekty na obrázcích patřit. Zde jsou dvě kategorie:
   * id: Unikátní identifikátor kategorie.
   * name: Jméno nebo popis kategorie (zde "slepice" a "unknown").

Tento JSON tedy popisuje dataset obsahující obrázek s jedním označeným objektem, který patří do kategorie "slepice". COCO formát umožňuje uložení komplexních a strukturovaných dat o objektech na obrázcích, což je velmi užitečné pro trénování a testování modelů strojového učení.

A black screen with orange lines

AI-generated content may be incorrect.

Nyní jsme si vytvořili zdroje dat a jsme připraveni, vyučit existující model. Doučovat doma existující model je relativně složité, ale díky výborné dokumentaci na strankach <https://docs.ultralytics.com/> jsem to zvladnul.

Základem pro úspěšné a efektivní učení je hardware. Hlavně GPU, protože trénování modelů hlubokého učení je výpočetně náročné. Vypozoroval jsem, že největší vliv na učení má velikost paměti grafické karty. V mém případě byla použita Nvidia RTX 4070.

O stavbě a následném učení neuronových sítí by se nechalo desítky stránek, ale to není předmětem mojí práce. Rád bych alespon představil blokové schema učícího procesu.

Pro učení jsem použil script, ktery jsem si pripravil.

**A computer screen shot of a program

AI-generated content may be incorrect.**

**Krok 1 - CleanWorkspace**

Tento krok je zodpovědný za přípravu pracovního prostředí před zahájením tréninkového procesu.

Odstranuji v něm archivaci stávajících dat a modelů, které mi ve workspace zůstaly z předchozich běhů a již nejsou potřebné.

Čištění pomáhá zabránit kolizím s předchozími seancemi a udržet v něm pořádek, což zajišťuje, že nové modely a datové soubory jsou aktuální a správně organizované.

**Krok 2 - GroupCreator**

Tento krok se zaměřuje na organizaci dat do různých skupin pro trénování a testování modelu.

Data mám rozdělena v několika různých adresářích a tento krok má na starost jejich správné složení.

Stejná data používám výuků modelu na počítání slepic, detekce vetřelce a plánované reidentifikace slepic.

**Krok 3 - DatasetSplitter**

Proces rozdělení datasetu je klíčový pro oddělení trénovací, validační a testovací sady.

Rozdělení může být prováděno v daných poměrech, například 70% trénovací data, 15% validační data a 15% testovací data.

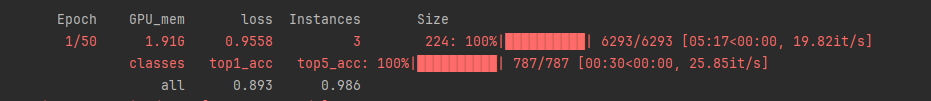
Správné rozdělení dat je důležité pro zajištění objektivního hodnocení modelu, aby se předešlo přeškolení.

Tento krok také zajišťuje, že žádné datové soubory nejsou opominuty a že rozdělení je náhodné a reprezentativní.

**Krok 4 - TrainYolo**

Jedná se o hlavní krok, kde je model YOLOv11 trénován na trénovacích datech.

Proces zahrnuje několik iterací (epoch) přes trénovací data, během kterých model aktualizuje své váhy podle chybného odhadu.



Během trénování se sleduje výkonnost modelu na validačních datech, která zabraňují přeškolení. Přeškolení si můžeme představit například u modelu, který rozpoznává kačeny, kočky a slepice. I když všechny konfigurační parametry byly správně, systém v některých případech špatně identifikoval kočku. Až zpětnou analýzou trénovacích dat bylo zjištěno, že došlo k přeškolení. System identifikoval kachnu nikoliv podle tvaru jejího těla, případně zabarvení, jako to udělá člověk, ale podle toho, zda na obrázku byla ci nebyla tráva. To pak vedlo k tomu, že pokud byla kočka na trávě, byla označena za kachnu.

Spusti trénovací smyčku je jednoduché. Bylo třeba předat jen několik základních parametrů.

**Training Parameters**

**Epochs**: 50

**Batch Size**: 16

**Pretrained**: True (using pretrained weights)

**Data Path**: c:\kurnik\klasifikace\kurnik\_learn\_temp\train

50 Epoch známená kolik poběží učících cyklů.

Batch Size – specifikuje kolik vláken poběží paralelně, je třeba dbát na velikost pameti grafické karty

Pretrained – říká, že budeme rozšiřovat již naučený model

Data Path – specifikuje odkud se mají vzit data k učeni.

**Krok 5 - YOLOTester**:

Po úspěšném tréninku modelu je nutné jeho testování na dříve oddělené testovací sadě.

Tento krok zahrnuje vyhodnocení výkonnosti modelu pomocí metrik, jako je přesnost, citlivost.

Testování nám poskytuje objektivní zpětnou vazbu o tom, co za objekty model umí ve snímcích rozpoznat. Důležíté je předkladat pro testování snímky, které během tréninku neviděl.

Na základě testovacích výsledků jsem pak následně prováděl úpravy učicích parametrů, případně jsem dodával další data do učícího datasetu.

A graph of a train loss

AI-generated content may be incorrect.

Graf který je automaticky generování při učení nám umožnuje sledovaz, jak dobře model učí vzhledem k časové ose tréninku. Tento graf typicky zobrazuje ztrátovou funkci (loss) na svislé ose (Y) a počet trénovacích epoch nebo kroků na vodorovné ose (X).

A graph with blue lines and orange dots

AI-generated content may be incorrect.

Tento graf se zaměřuje na metriku přesnosti, konkrétně na "top-1 accuracy," která měří, jak často je nejpravděpodobnější (nejvyšší hodnocená) předpověď modelu správná.

Nyní jsem si pověděli jak vznikl model pro počítání slepic v kurníku a nyní se podíváme jak jsem připravil model pro segmentaci slepice z obrazu.

**Nejlepsi a nejhorsi nosnice**

Image segmentation je technika v počítačovém vidění, která má za cíl rozdělit obrázek na několik segmentů nebo oblastí pro jednodušší analýzu. Cílem image segmentation je identifikovat a rozlišit různé objekty nebo části obrázku, což může být užitečné pro různé aplikace, jako je rozpoznávání objektů, detekce hranic, klasifikace a další analytické úlohy.

V mém případě jsem chtěl využit tuto techniku k tomu, abych identifikoval která slepice snesla vajíčko abych tak získal statistiku o tom v jaké kondici slepice jsou a jak dobře snáší.

Existuje několik možností jak slepici identifikovat, já jsem se rozhodl pro rozpoznání slepice z obrazu.

Stejným způsobem jak jsem popsal v předchozím kroku, natrénoval jsem **YOLO pro detekci slepic**:

V tomto případě byla tvorba anotacniho souboru významě složitější protože bylo třeba nejenom identifikova, že slepice je na obraze, ale identifikovat jeji konkrérní tvar, případně jenom část těla.

A screenshot of a computer

AI-generated content may be incorrect.

Dale jsem vybral vhodnou verzi YOLO, aktualne ve verzi 11, který jsem natrénoval na mém modifikovaném d datasetu, aby byl schopný detekovat slepice v různých pozicích a podmínkách.

**Implementace Instance Segmentation**:

YOLO knihovna podporuje segmetnaci, sám o sobě není segmentační model, ale může být použit ve spojení s technikami jako Mask R-CNN nebo můžete použít rozšíření YOLO, jako je YOLACT či Yolov8, která podporují segmentaci.

* + Zaměřte se na rozpoznání jednotlivých instancí slepic, zejména pokud chcete rozlišit mezi jednotlivými ptáky.

**Reidentifikace s použitím Feature Embeddings**:

* + Pro reidentifikaci slepic potřebujete vypočítat a uložit feature embeddings (vlastnosti) pro každou detekovanou slepici. To může zahrnovat například barvu peří, velikost či jiné vizuální charakteristiky.
  + Použijte hluboké učení na další analýzu těchto vlastností a k porovnání, zda nově detekovaná slepice odpovídá dříve detekované instanci.

1. **Implementace sledování**:
   * Použití algoritmů pro sledování (např. SORT, Deep SORT) pro sledování pohybu slepic mezi jednotlivými snímky. Tyto algoritmy využívají informace o pozici a vlastnostech (embeddings) pro udržování identity každé slepice.
   * Pokud je to nutné, použijte kamerové záznamy nebo uchovávejte přehled o polohách slepic, abyste měli historické údaje pro zlepšení resultace reidentifikace.
2. **Hodnocení a Úprava Modelu**:
   * Pravidelně hodnoťte přesnost vašeho systému pomocí testovacích dat, zahrnujících různé podmínky osvětlení, úhly pohledu a překážky.
   * Vylepšujte model na základě zpětné vazby, jemně nalaďte hyperparametry, a pokud je potřeba, rozšiřujte dataset o nové situace z reálného prostředí.

Využívání YOLO pro segmentaci a reidentifikaci vyžaduje integraci několika technologií a metod pro dosažení optimálních výsledků, zejména když je třeba dosáhnout reálné přesnosti v dynamickém a často složitém prostředí kurníku.