

Artificial Intelligence Project

Edvard Khromov - Maksym Patrushev - Matsvei Volkau

Informatyka stosowana II ROK IV SEM 2024



Treść dokumentacji

1.	Wstęp	2
2.	Wstępne przetwarzanie danych	3
3.	Przebieg pracy	5
4.	Ocena modelu	6
5.	Źródła	9



1. Wstęp

W obecnym klimacie geopolitycznym poprawne wykrywanie pojazdów wojskowych, zwłaszcza tych pochodzenia rosyjskiego, odgrywa kluczową rolę z perspektywy bezpieczeństwa narodowego i planowania strategicznego. Artykuł opisuje technikę wykrywania rosyjskich pojazdów wojskowych przy użyciu modelu YOLOv8 (You Only Look Once). YOLOv8 to niezależny system rozpoznawania pojazdów na wysokim poziomie, który dzięki szybkiemu przetwarzaniu obrazów z pomocą oświetlenia, szybko wykrywa i identyfikuje pojazdy.

Narzędzie YOLOv8 będzie wykorzystywane do analizy obrazów i ekstrakcji metadanych w celu identyfikacji pojazdów wojskowych w różnych scenariuszach. Różne scenariusze identyfikacji pojazdów wojskowych będą oznaczane poprzez metadane wyekstrahowane z analizy obrazów za pomocą YOLOv8. Istnieje kilka znaczących etapów w rozwoju tego praktycznego i niezawodnego systemu wykrywania:

Fazy Rozwoju Systemu Wykrywania



System ten umożliwia rozwój zautomatyzowanego systemu wykrywania rosyjskich pojazdów wojskowych, który pozwoli na wzmocnienie zdolności strategicznych w zakresie dow odzenia i kontrolowania zagrożeń geopolitycznych.



2. Wstępne przetwarzanie danych

Stworzenie poprawnego datasetu dla treningu i walidacji modelu zaczyna się od zbierania zdjęć z internetu. Wybierając odpowiednie źródła, takie jak wyszukiwarki obrazów i strony z darmowymi zasobami, można zgromadzić dużą liczbę różnorodnych obrazów reprezentujących interesujące nas kategorie. Następnie zdjęcia te są organizowane i filtrowane, aby zapewnić wysoką jakość i różnorodność danych.

Kolejnym krokiem jest użycie narzędzia RoboFlow do oznaczania zebranych obrazów. RoboFlow umożliwia łatwe tworzenie etykiet i adnotacji, co jest kluczowe dla poprawnego trenowania modelu. W procesie oznaczania użytkownik ręcznie przypisuje odpowiednie kategorie do każdego obrazu, co pozwala modelowi na naukę rozpoznawania wzorców i cech charakterystycznych dla każdej kategorii.

RoboFlow automatyzuje wiele zadań związanych z przetwarzaniem danych, takich jak normalizacja obrazów, augmentacja danych (np. obracanie, zmiana jasności), co dodatkowo zwiększa efektywność i dokładność modelu. Po zakończeniu oznaczania, dataset jest dzielony na zestawy treningowe i walidacyjne. Zestaw treningowy służy do nauki modelu, natomiast walidacyjny pozwala na ocene jego wydajności i ogólności.

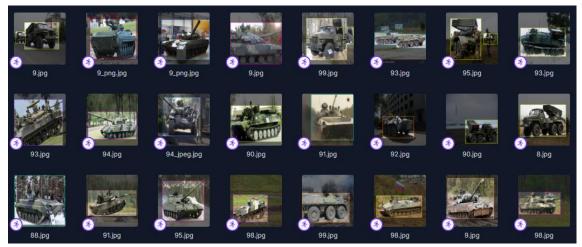
Cały proces, od zbierania danych po ich oznaczanie i przetwarzanie, zapewnia stworzenie solidnego i wszechstronnego datasetu, który jest kluczowy dla skutecznego trenowania i walidacji modeli machine learning.

Strona projektu na Roboflow ③ Need help? △ MVDetection IST PBSMP » Dataset Health Check Generated on June 01, 2024 at 12:28 am. 2 Regenerate Average Image Size 993 1,254 0.17 mp 416x416 0 missing annotations 0 null examples 1.3 per image (average) Across 10 classes ≣ Manage Classes 🖺 Rebalance Splits Class Balance all train valid test Size Distribution



Przykłady zbiorów danych dla modelu

Trening:



Validacja:



Testowanie:





3. Przebieg pracy

Przede wszystkim, wykorzystano bibliotekę Ultralytics, która dostarcza implementację modelu YOLOv8. Model ten jest znany ze swojej wydajności w czasie rzeczywistym, co czyni go idealnym do zastosowań wymagających szybkiego przetwarzania obrazu, takich jak systemy monitoringu czy autonomiczne pojazdy.

Początek procesu polegał na przygotowaniu środowiska, w tym instalacji wymaganych bibliotek. Następnie pobrano zbiór danych z platformy Roboflow, co umożliwiło dostęp do zestawu obrazów i odpowiadających im adnotacji w formacie YOLO. Kluczowym elementem było poprawne skonfigurowanie ścieżek dostępu do danych oraz modelu.

Model YOLOv8 został następnie skonfigurowany do treningu z użyciem specyficznych parametrów, takich jak liczba epok treningowych, rozmiar obrazów oraz ścieżka do pliku konfiguracyjnego danych. Proces treningu obejmował 100 epok, co zapewniło odpowiednią ilość iteracji do nauki modelu na podstawie dostarczonych danych.

Zrzut ekranu przy treningu modelu:

		' '		0 -				
0	%cd {HOME}							
	!yolo task=de	tect mode=	train model	=yolov8s.pt	data=/cont	ent/dataset	s/MVDetecti	ction_IST_PBSMP-1/data.yaml epochs=100 imgsz=416 plots=True
₹	Epoch 82/100	GPU_mem 1.94G Class all	0.4916	cls_loss 0.5059 Instances 0	dfl_loss 1.009 Box(P 0			16: 100% 44/44 [00:15<00:00, 2.88it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 2.01it/s]
	Epoch 83/100	GPU_mem 1.95G Class all	0.4697	cls_loss 0.4937 Instances 0	1.003 Box(P	28 R		16: 100% 44/44 [00:09<00:00, 4.58it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 3.07it/s]
	Epoch 84/100	GPU_mem 1.94G Class all	0.4822	cls_loss 0.5009 Instances 0	dfl_loss 1.004 Box(P	28 R	mAP50	ze 16: 100% 44/44 [00:10<00:00, 4.01it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 2.37it/s] 0
	Epoch 85/100	GPU_mem 1.95G Class all	0.4791	cls_loss 0.4853 Instances 0	dfl_loss 1.015 Box(P	33 R		16: 100% 44/44 [00:08<00:00, 4.91it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 4.38it/s]
	Epoch 86/100	GPU_mem 1.94G Class all	0.4714	cls_loss 0.4869 Instances 0	dfl_loss 1.001 Box(P	34 R		l6: 100% 44/44 [00:12<00:00, 3.45it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 3.62it/s]
	Epoch 87/100	GPU_mem 1.95G Class all	0.475	cls_loss 0.4857 Instances 0	dfl_loss 1.007 Box(P	Instances 41 R 0		16: 100% 44/44 [00:12<00:00, 3.58it/s] 50 mAP50-95): 100% 2/2 [00:00<00:00, 3.88it/s]

W trakcie treningu model uczył się rozpoznawania i lokalizowania obiektów na obrazach, optymalizując swoje parametry w celu minimalizacji funkcji straty. Funkcja ta mierzyła różnicę między przewidywaniami modelu a rzeczywistymi etykietami w zbiorze treningowym, uwzględniając zarówno lokalizację obiektów, jak i klasyfikację.

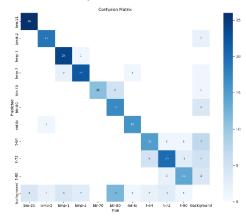


4. Ocena modelu

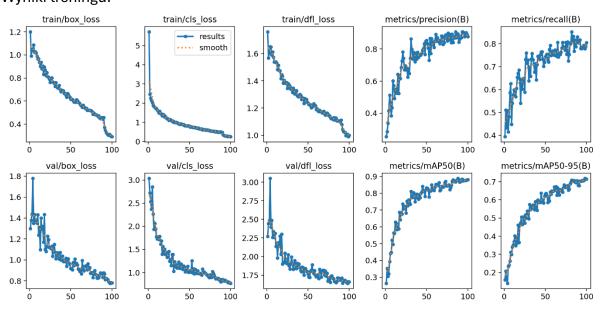
Ocena modelu detekcji obiektów YOLOv8 dostarcza istotnych informacji na temat jego wydajności i skuteczności. Po zakończeniu treningu, model został poddany walidacji i testowaniu na oddzielnych zbiorach danych, co pozwoliło na dokładną ocenę jego możliwości.

Wyniki walidacji zostały wizualizowane za pomocą kilku kluczowych grafik. Macierz konfuzji przedstawia, jak dobrze model rozpoznaje różne klasy obiektów, wskazując na liczbę poprawnie i niepoprawnie sklasyfikowanych przykładów. Ogólne wyniki treningu, przedstawione na wykresie, pokazują zmiany w wartości funkcji straty oraz innych metryk w trakcie kolejnych epok treningowych, co pozwala ocenić, jak model się uczył i czy proces treningu przebiegał prawidłowo.

Macierz konfuzji:



Wyniki treningu:







Wizualizacje wyników predykcji na partii walidacyjnej pokazują konkretne przykłady obrazów, na których model dokonał detekcji obiektów. Te obrazy zawierają prostokąty otaczające (bounding boxes) oraz etykiety klas, co umożliwia ocenę, jak dokładnie model lokalizuje i klasyfikuje obiekty na nowych danych.

Ostateczne testowanie modelu zostało przeprowadzone na zbiorze testowym, który nie był używany w trakcie treningu ani walidacji. Wyniki tego testu dostarczają ostatecznej oceny skuteczności modelu, mierzonej przy użyciu metryk takich jak mAP (mean Average Precision). Metryka ta mierzy średnią precyzję modelu w detekcji obiektów, uwzględniając zarówno dokładność lokalizacji, jak i klasyfikacji.



Wyniki testowania:















Całokształt ocen wskazuje, że model YOLOv8 jest w stanie skutecznie i precyzyjnie wykrywać obiekty na obrazach, co jest potwierdzone zarówno przez wizualne inspekcje wyników, jak i obiektywne metryki. Dzięki tym ocenom można stwierdzić, że model jest dobrze przygotowany do praktycznych zastosowań w rzeczywistych scenariuszach detekcji obiektów.

5. Źródła

https://app.roboflow.com/militarydetection-v1

https://github.com/roboflow/notebooks

https://blog.roboflow.com/how-to-train-yolov8-on-a-custom-dataset/

https://docs.roboflow.com/

https://github.com/ultralytics/ultralytics

https://universe.roboflow.com/capstoneproject/russian-military-annotated

 $\underline{\texttt{https://colab.research.google.com/github/roboflow-ai/notebooks/blob/main/notebooks/train-yolov8-object-detection-on-custom-dataset.ipynb}$

https://en.wikipedia.org/wiki/Convolutional_neural_network