Problem Statement:

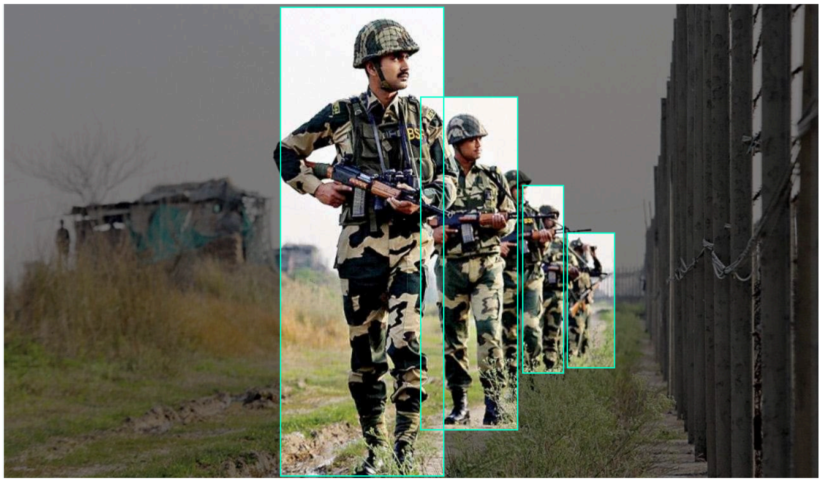
- Identify Soldier based on the Uniform (CRPM,BSF,JAmmu &amp; Kashmir Police) take `random image of soldier to application and which force they Belongs to using Deep Learning Concept.
- Edge Scenario is Bonus

Data Preparation

I have used an open dataset which has three different types of uniforms, i.e CRPF, BSF and Jammu and Kashmir police.
Train images: 251
Validation images:27
Test images:24
Each have a label of a list of strings, where each element contains in a .txt format.
<class_id> <x_center> <y_center> <width> <height>

Example of dataset

Border Security Forces (BSF)

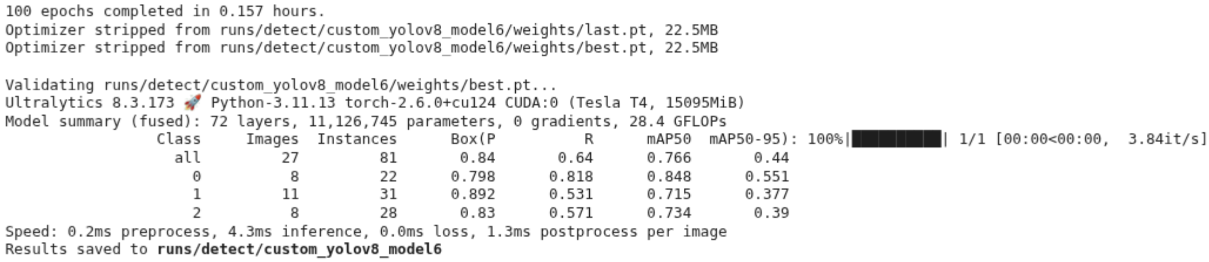


Target features for the model:

Feature	CRPF	BSF	JK Police
Camo Pattern	Green/blue jungle camo	Beige-green digital	Plain khaki
Headgear	Helmet/camo cap	Khaki cap/beret	Khaki or maroon beret
Badge/Insignia	"CRPF" tag	"BSF" label	"J&K Police" badge
Common Settings	Riot, jungle, city	Border/jungle patrol	Urban/police work

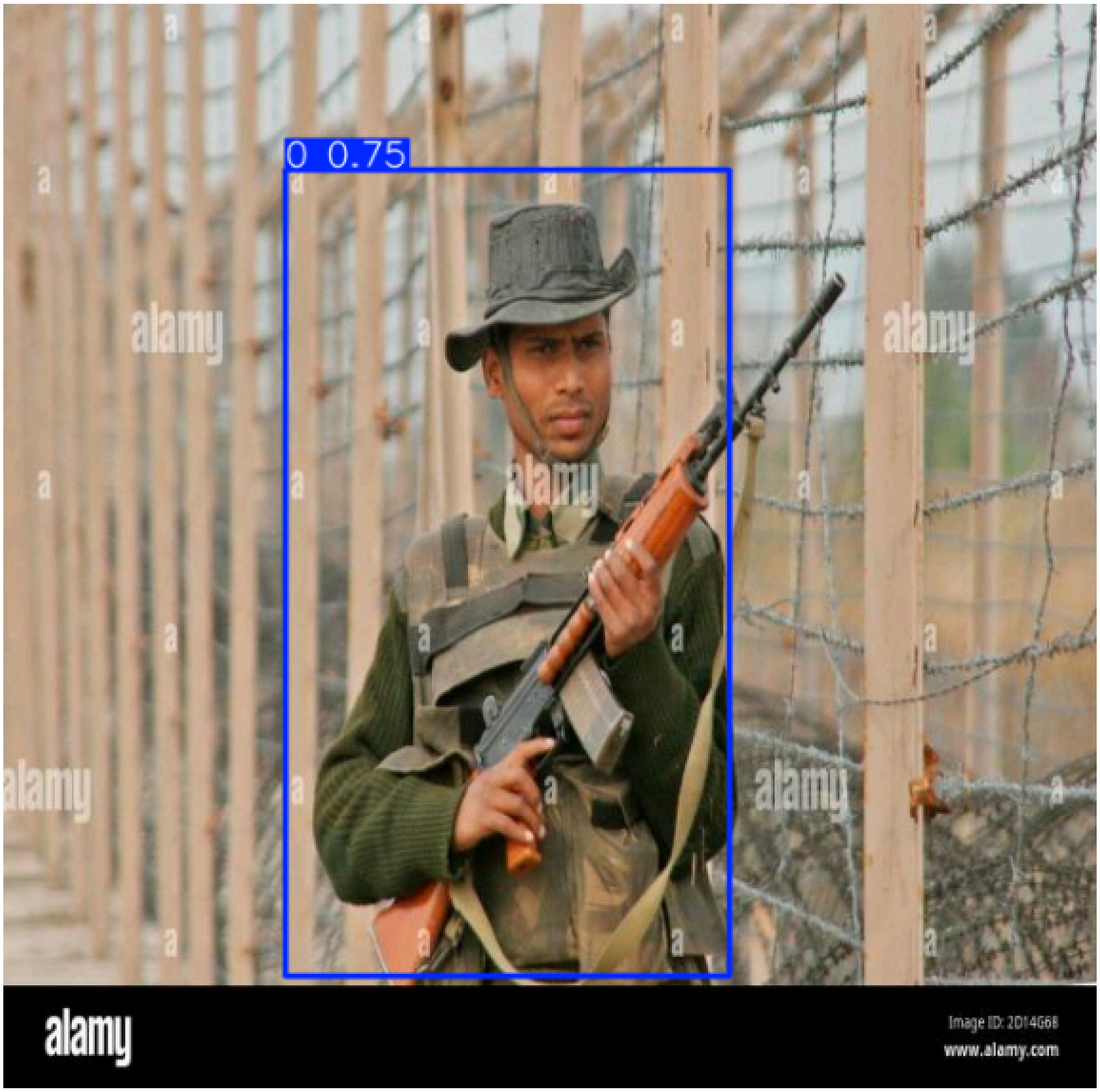
Model

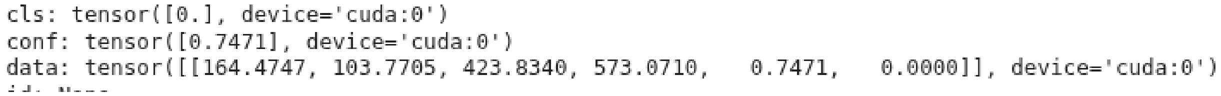
Train Yolov8 model on the described above dataset. Used Ultralytics library for the training,
Class 0 BSF
Class 1 CRPF
Class 2 J&K Police
Training Model summary



```
100 epochs completed in 0.157 hours.
Optimizer stripped from runs/detect/custom_yolov8_model6/weights/last.pt, 22.5MB
Optimizer stripped from runs/detect/custom_yolov8_model6/weights/best.pt, 22.5MB

Validating runs/detect/custom_yolov8_model6/weights/best.pt...
Ultralytics 8.3.173 🚀 Python-3.11.13 torch-2.6.0+cu124 CUDA:0 (Tesla T4, 15095MiB)
Model summary (fused): 72 layers, 11,126,745 parameters, 0 gradients, 28.4 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100%|██████████| 1/1 [00:00<00:00,  3.84it/s]
                   all         27         81       0.84       0.64      0.766       0.44
                     0          8         22      0.798      0.818      0.848      0.551
                     1         11         31      0.892      0.531      0.715      0.377
                     2          8         28       0.83      0.571      0.734       0.39
Speed: 0.2ms preprocess, 4.3ms inference, 0.0ms loss, 1.3ms postprocess per image
Results saved to runs/detect/custom_yolov8_model6
```





```
cls: tensor([0.], device='cuda:0')
conf: tensor([0.7471], device='cuda:0')
data: tensor([[164.4747, 103.7705, 423.8340, 573.0710,   0.7471,   0.0000]], device='cuda:0')
```

Result:

Showing class 0 i.e BSF with a confidence score of 0.75