Fast Background Subtraction

Rohit Kumar

Overview

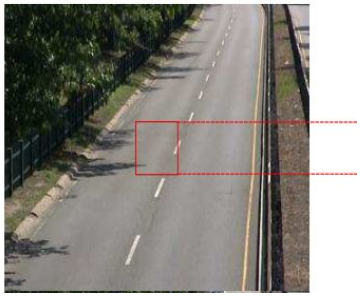
- Problem statement
- Literature Survey
- Our Work
- Future scope and conclusion

Problem statement

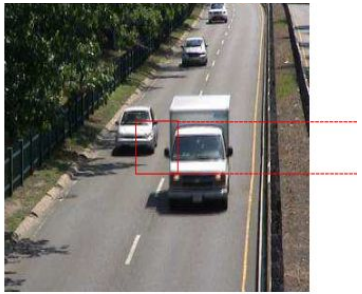
Data view

- Consider the camera mounted at the security gate.
- We have a static image i.e the image to be considered as background.
- We have frames having object coming and going.

Background image



Input frame



What is background subtraction ?

- **Foreground detection** is one of the major tasks in the field of computer vision and image processing whose aim is to detect changes in image sequences.
- **Background subtraction** is any technique which allows an image's foreground to be extracted for further processing (object recognition etc.).
- Background subtraction is a widely used approach for detecting moving objects in videos from static cameras.
    - As a preprocessing step of object detection
    - Surveillance tracking
    - Human pose estimation etc.

Task

- Task is to develop a deep learning model that do the background subtraction.
- In addition considering the constraints
  - BGS is a preprocessing step in real application systems.
  - It is inherent need of systems for it to be fast.
  - It should not be specific to a scene. Generalisation is needed.
  - Run nearly 200 fps with a batch of 10.

Challenges to consider

Model should be robust to

- ➔ Illumination Changes - even in static scenes, lightning changes is a problem.
- ➔ Camouflage - similar colors of background and foreground
- ➔ Night videos - As most pixels have a similar color in a night scene, recognition of foreground objects and their contours is difficult
- ➔ Hard Shadows - Dark, moving shadows that do not fall under the illumination change category

We need not to worry about :

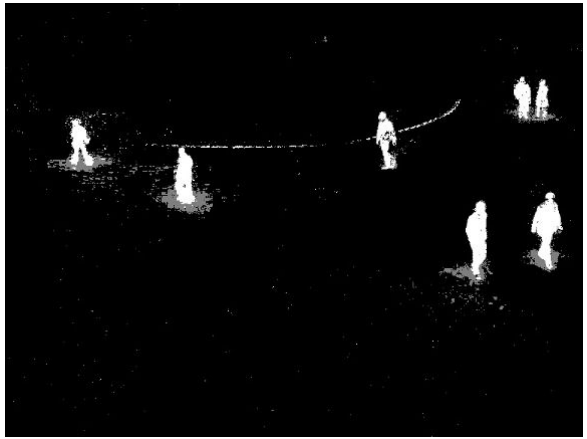
- ➔ Dynamic background - not applicable for the case of static scenes
- ➔ Camera Jitter
- ➔ Ghost/ Intermittent Object motion - Foreground objects that are embedded into the background scene and start moving after background initialization are the so-called ghosts.

Literature Survey

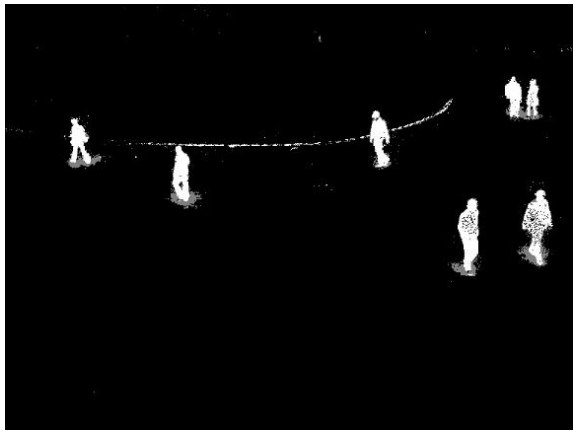
Classical methods

- There are classic algorithms like MOG, MOG2, KNN etc. are simple threshold based.
- They do not address
  - Camouflage
  - illumination changes
  - Night videos and other challenges

MOG



KNN



State of the art

- FgSegNet and FgSegNet2 are on the top of CDNet benchmark.
- They are are deep neural networks not ready for deployment in real-time.

Others

- Paper : A Deep Convolutional Neural Network for Background Subtraction
  - Proposes a lightweight CNN model but
    - Scene specific
    - Patch based approach
    - Input size dependant

- Paper : BSUV-Net: A Fully-Convolutional Neural Network for Background Subtraction of Unseen Videos
  - Too deep to be applicable for real-time preprocessing step.

- The limitations of classical methods and complexity of challenges pushes us to use CNN based models.
- Our need for a model that generalises for any { background, input_Image } pair.

Our work

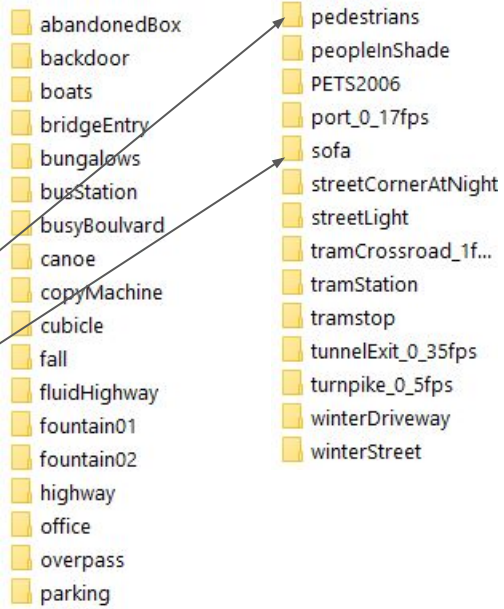
Datasets

Human activities

- There are nearly 31 datasets available for background subtraction divided into three categories
  - Article
  - Conference
  - Project
- A list of those can be found here
- CDNet is the most complete dataset as per our needs.

CD-Net

- Change detection dataset
- Dataset is of video frames
- Each frame is labelled by foreground mask.
- After filtering for our target we got these subsets.



- Total image ~48,000
- Dataset
  - Test set - [sofa, pedestrians] ~3,500 images
  - Train set - [all except the test set folders] ~ 44,500 images

Background subtraction

Background subtraction has the following steps

1. Background modeling
2. Background initialization
3. Background maintenance
4. Foreground detection

BS : Background modeling & initialization

- Taking mean of first 10 frames proves to be sufficient model for initial background.
- For each sub - dataset
    - Generate a BG_IMAGE as mean of the first 10 frames.

BS : Background Maintenance

Since our target does not include modelling { dynamic background, camera jitter, and intermittent object motion } background maintenance is not required for our task.

- We assume that initial background created will remain same
- However, the illumination changes, which are natural even for static backgrounds are induced as a preprocessing step.

BS : Foreground Detection

O

Background image

Input frame

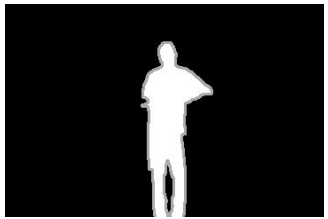
Preprocessing

A 6 channel input

Algorithm

Predicted Mask

True Output Mask



Loss

Optimisation

Capturing challenges in data

- Illumination Changes:
  - Adding a random illumination change in the preprocessing step.
- Camouflage :
  - In dataset
- Night videos :
  - In dataset
- Hard Shadows :
  - In dataset

Preprocessing

- Our model is size independent, however for training with batches
  - Random crop of size 224
- Five types of labels in CDNet i.e each pixel in true mask is marked with one of the five.

0 : static	50 : hard shadow	85 : Outside ROI	170 : Unknown motion	255 : Motion
Background	Background		Outside ROI	Foreground

- We don't train on the pixels that are outside ROI.

Preprocessing

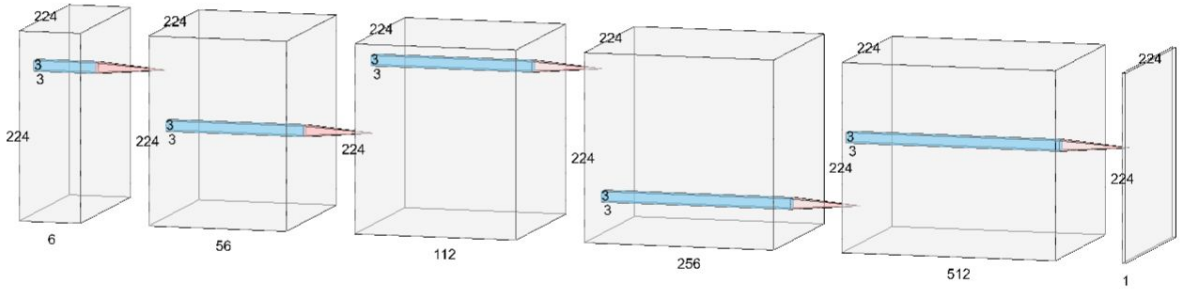
- Random illumination change.
- Normalise the image to the range [0, 1].
- Make the mask single channel and binary [0, 1] .
    - Include value 0.85 for outside ROI regions.

Model

- A fully connected CNN (FCNN) model with 5 layers.
  - FCNN is important because we want our model to be independent of input size.
  - Kernel size : 3 with pad : 1 so that width and height remains same.
  - ReLU nonlinearity after Convolution
  - Batch Normalisation after non-linearity
- Sigmoid after last layer to map the outputs to probability distribution for each pixel to be background or foreground.

Algorithm

```
Net(
  (b1): Sequential(
    (0): Conv2d(6, 56, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(56, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (b2): Sequential(
    (0): Conv2d(56, 112, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(112, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (b3): Sequential(
    (0): Conv2d(112, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (b4): Sequential(
    (0): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (1): ReLU()
    (2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
  (final_layer): Sequential(
    (0): Conv2d(512, 1, kernel_size=(1, 1), stride=(1, 1))
    (1): Sigmoid()
  )
)
```



Loss function

- We used Jaccard distance as our loss measure.

- We smoothed our loss to prevent loss decay to zero.
- The pixels that are Outside ROI are not considered in loss and hence in training.

Jaccard Index $$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}.$$

Jaccard Distance $$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}.$$

```
def forward(self, y_true , y_pred):
    '''
    y_true : consists the roi details also with pixel value 85 indicating as outside ROI
    y_pred : predicted vector of same shape as y_true.
    '''

    batch_size = y_true.shape[0]

    # filter ROI
    roi_indices = torch.where ( y_true != 0.85 )
    y_true = y_true[roi_indices]
    y_pred = y_pred [ roi_indices ]

    # jaccard intersection
    intersection = (y_true * y_pred ).abs().sum()
    sum_ =  ( y_true.abs() + y_pred.abs() ).sum()
    jac = ( intersection + self.smooth ) / ( sum_ - intersection + self.smooth )

    # jaccard distance
    d = ( 1 - jac ) * self.smooth / batch_size
    return d
```

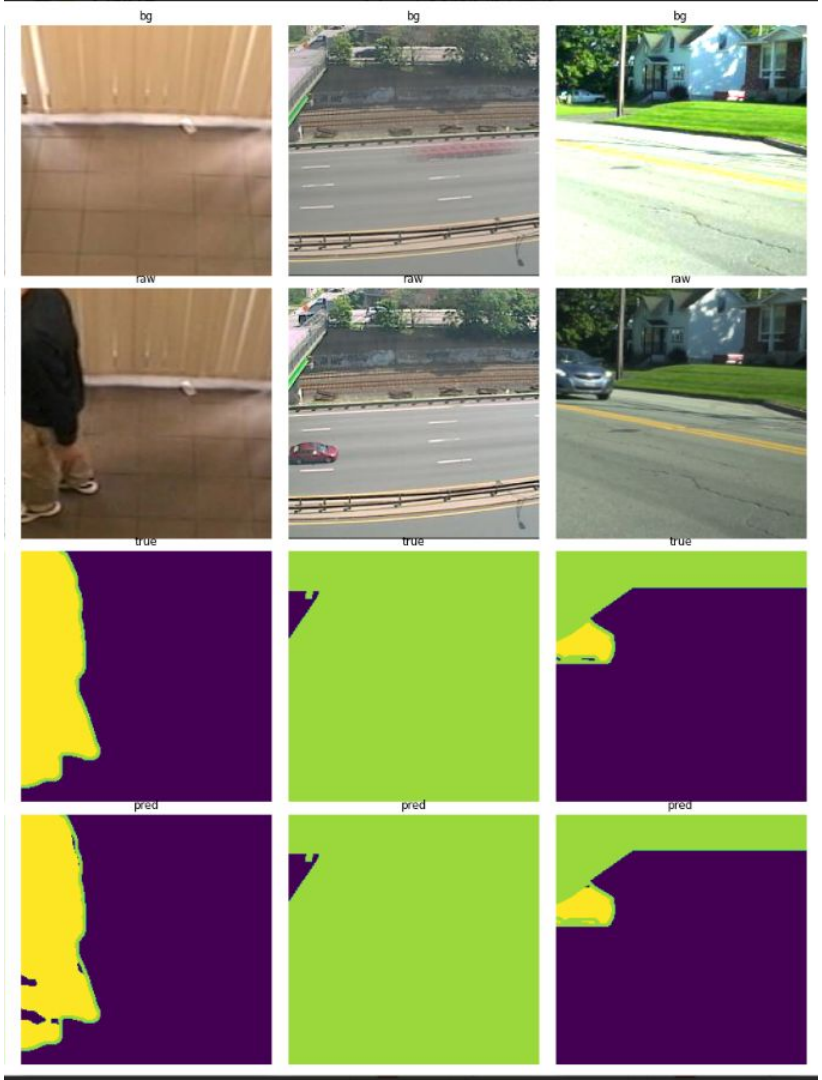
Optimizer

- Adam with standard learning rate of 0.001.

```
optimizer = optim.Adam(net.parameters(),
                       lr=0.001,
                       betas=(0.9, 0.999),
                       eps=1e-08,
                       )
```

Results - Training

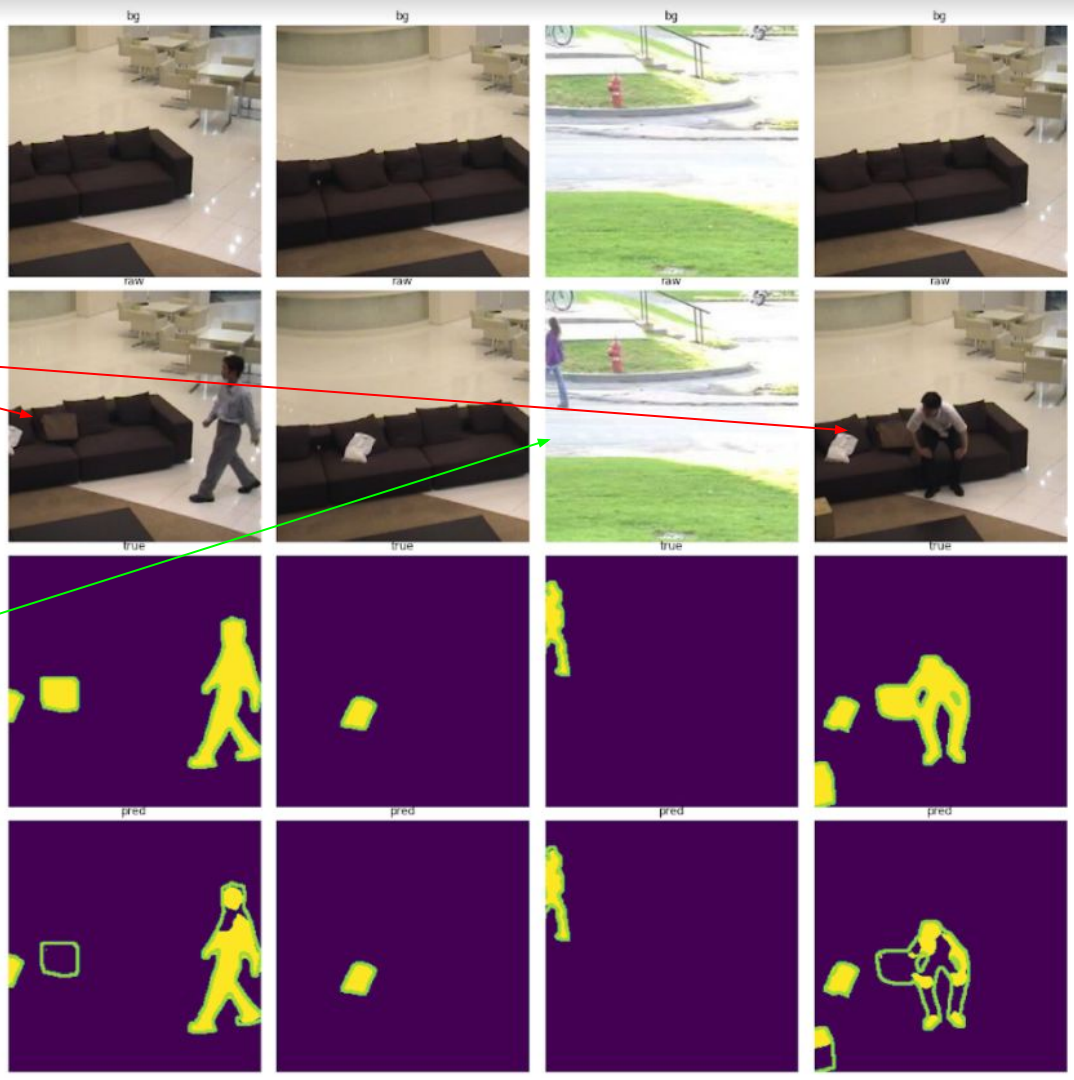
Results - Training



Results - Validation

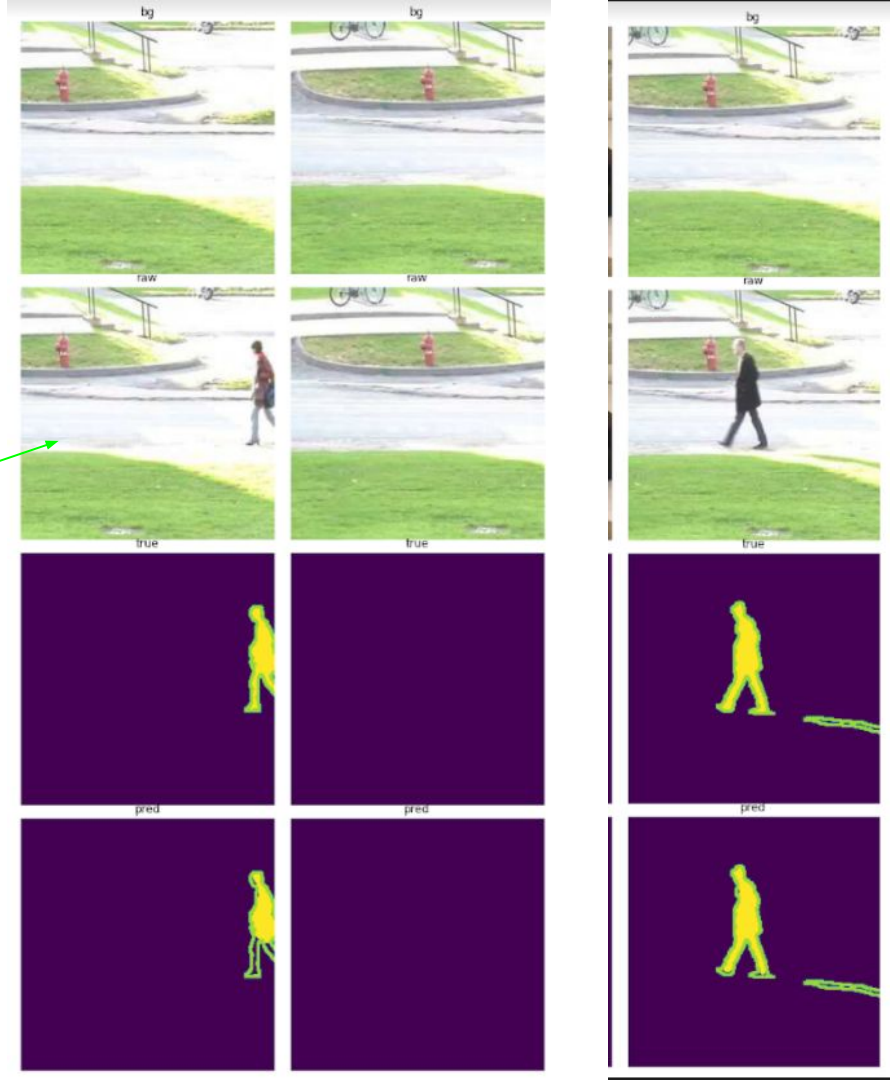
Unable to capture camouflage

Able to capture good background



Results - Validation

Able to capture other than camouflage.



Insights

- Dataset is not sufficient.

Plan

- We need more dataset.
  - There are not much real datasets available for the task.

- Since our target is similar to creating a general background and input frame comparator, we decided to create a synthetic dataset.

Insights

- Camouflage is a difficult situation to capture.
    - Our dataset have few images to capture the pattern.

Plan

- We need to ensure good amount of patterns related to camouflage in our synthetic dataset.

Insight

- The whole training was done on Google Colab, while reading data from Google Drive storage.
    - Each iteration takes ~2hrs
    - I trained for ~30 epochs with iterative improvement.

Plan

- We need more and fast computing power.
- A dedicated system or Cloud service.

Synthetic dataset

- Take a good dataset like COCO.
- Take two random image from COCO : R_IMG1 and R_IMG2
- Take a random mask from CDNet : R_MASK

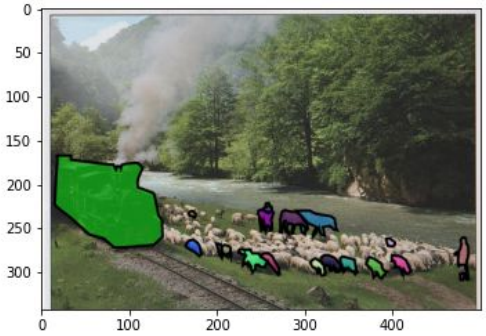
```
Use R_IMG2 as background.

Cut out the R_MASK from R_IMG2 and paste over background.

     Target_input = R_MASK * R_IMG1 + ( 1 - R_MASK ) * R_IMG2
```

- Now we have synthetic target_input, Background image and true mask.
- We believe that this will improve our algorithm for general background subtraction.

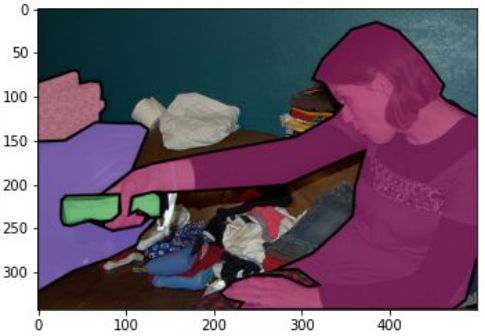
Please ignore the segmentation mask over the images.

This is due to web-viewer image in COCO are by default segmentation covered.

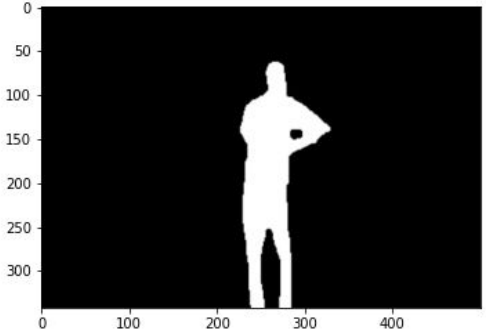
R_IMG1



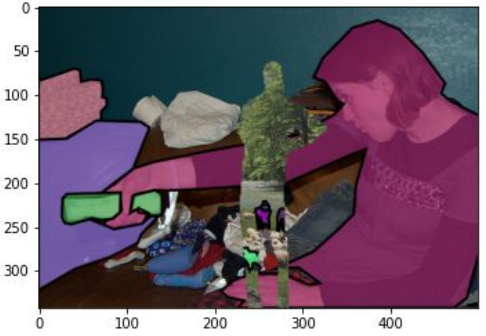
R_IMG2



R_MASK



Target input



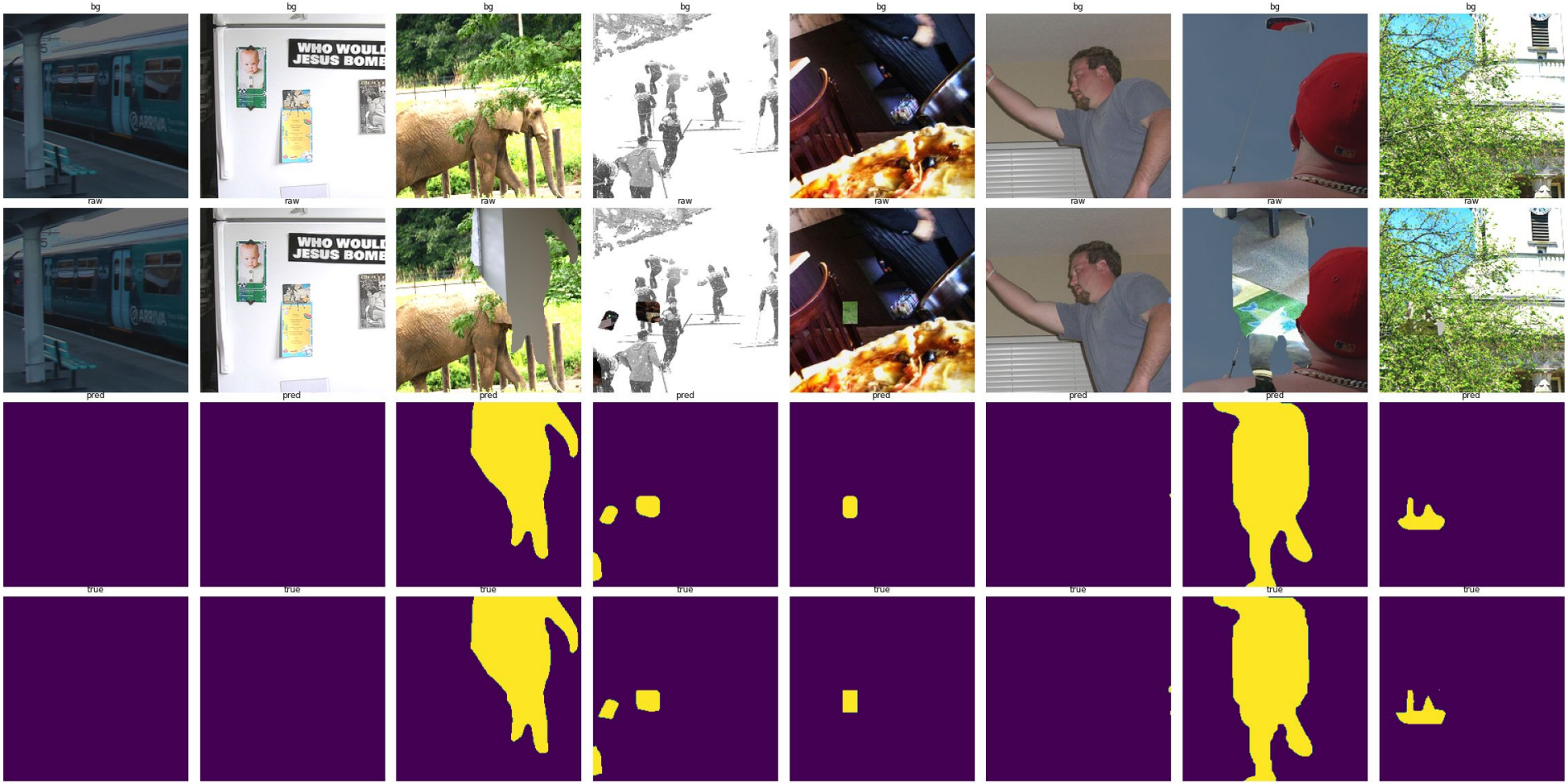
COCO dataset

- We used COCO 2014
- COCO train Images ~84K
- COCO val images ~41k

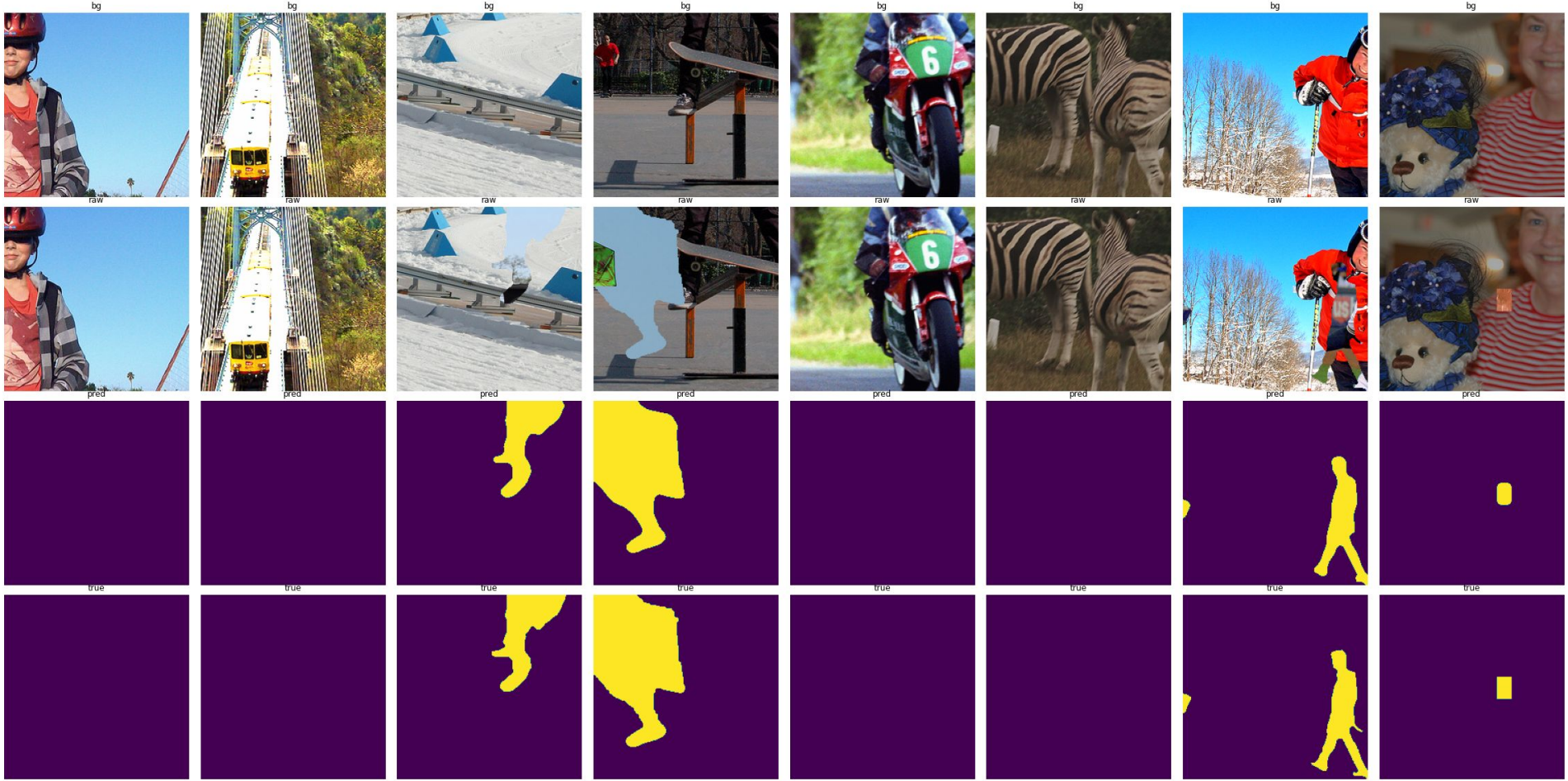
Synthetic Dataset

- Train size 200_000 random tuples of (randomImage1, randomImage2)
- Validation size 5_000 random tuples of (randomImage1, randomImage2)

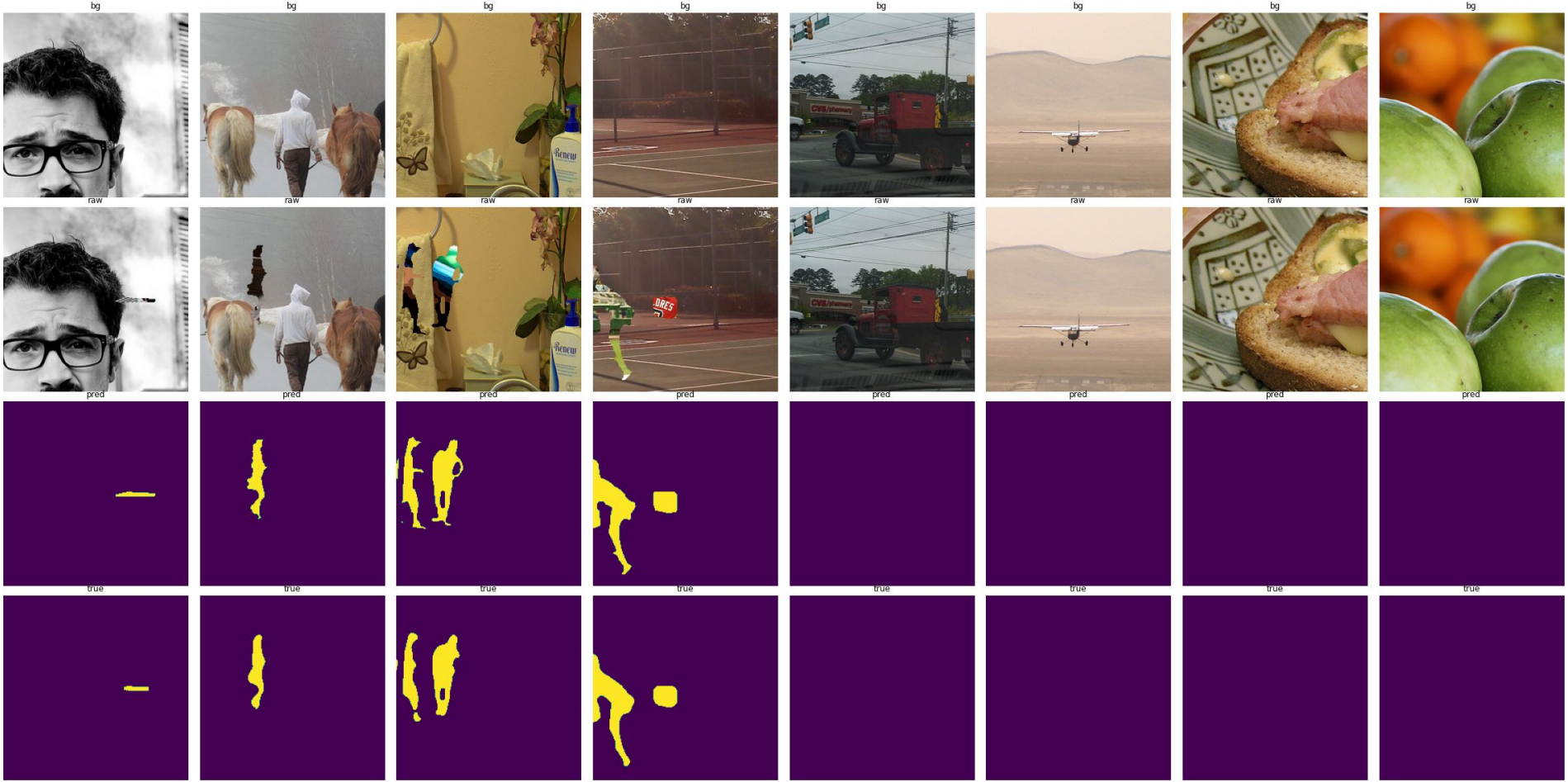
Training results on Results on Synthetic Data



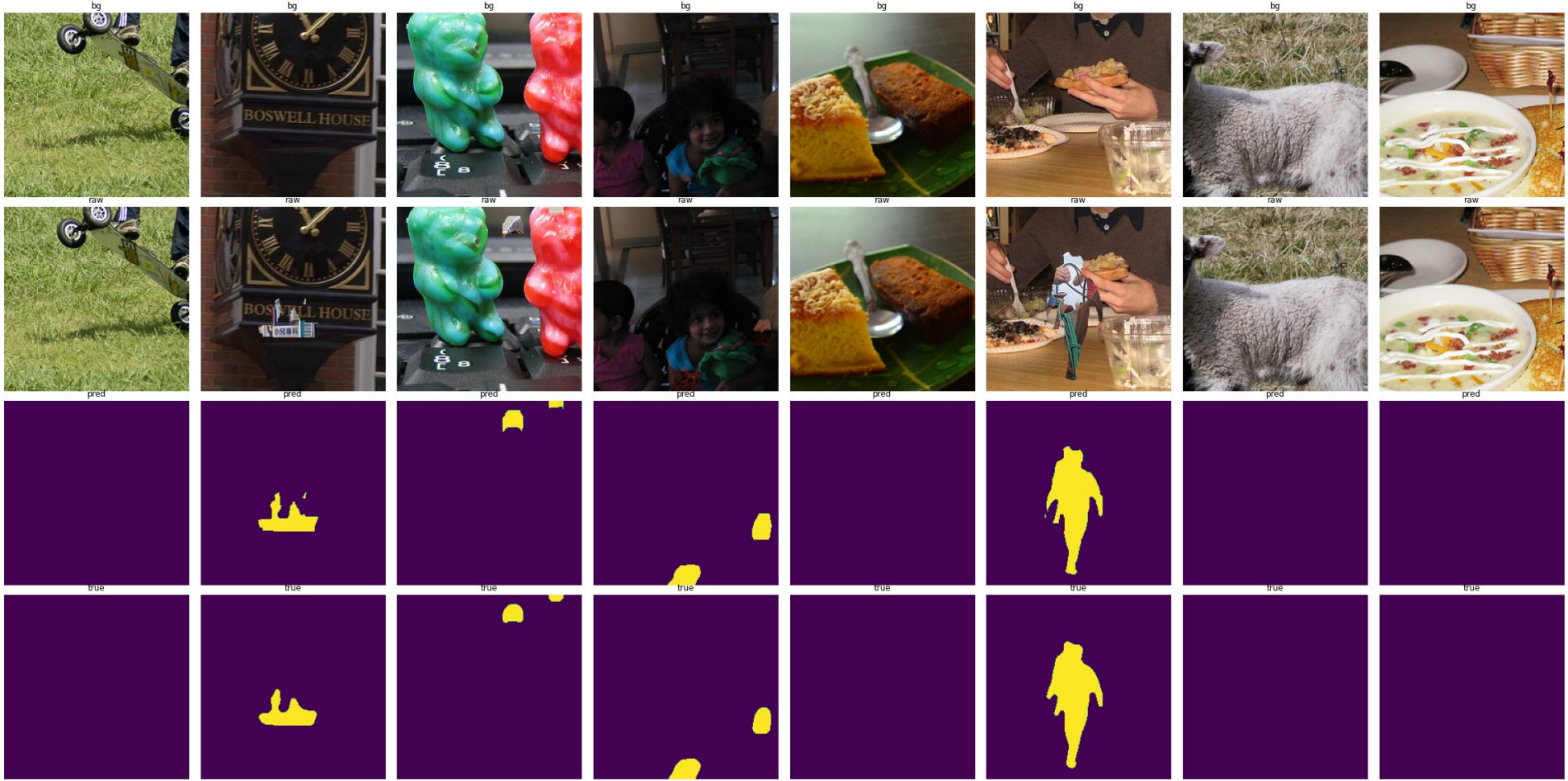
Training results on Results on Synthetic Data



Validation results on Results on Svnthetic Data



Validation results on Results on Synthetic Data



Future Scopes and Work

Dataset

- Making it general and evaluating on real video frames.
- Using more realistic masks and synthetic data
  - Masks of CDNet are not capturing all scenarios
  - Using masks from segmentation datasets like COCO.
  - Generate the dataset similar to previous methods.

- Using computer graphics software to generate data specific to taks.
  - E.g generating background subtraction for human for video conferencing.
  - It’s easy to get multi-view perspectives of the person.
  - Thus enabling us to add generalisation in the BGS in view of camera position.

Application

- Developing an application model that can use it.

Thankyou