Deep Anomaly

Detection of ‘Abnormal Behaviours’ in crowded scenes

by Saurabh Tote
Akshay Patidar
Shashank Batra

Implementation

Python

PyTorch

Google Colab

Datasets

UCSD Anomaly Detection Dataset

Subway-Exit dataset (Amit Adam et al.)

Some Peculiarities

Transfer Learning

AlexNet trained on ImageNet and MIT places database

First few layers extracted from these

Too much depth avoided

CNNs and FCNs are not adequate

Supervised Learning is not right for this task

Slow (Patch based)

Multiple Gaussian Classifiers used

3 threshold values used

Sparse Autoencoder

Is used between these 2 classifiers
(fig on next slide)

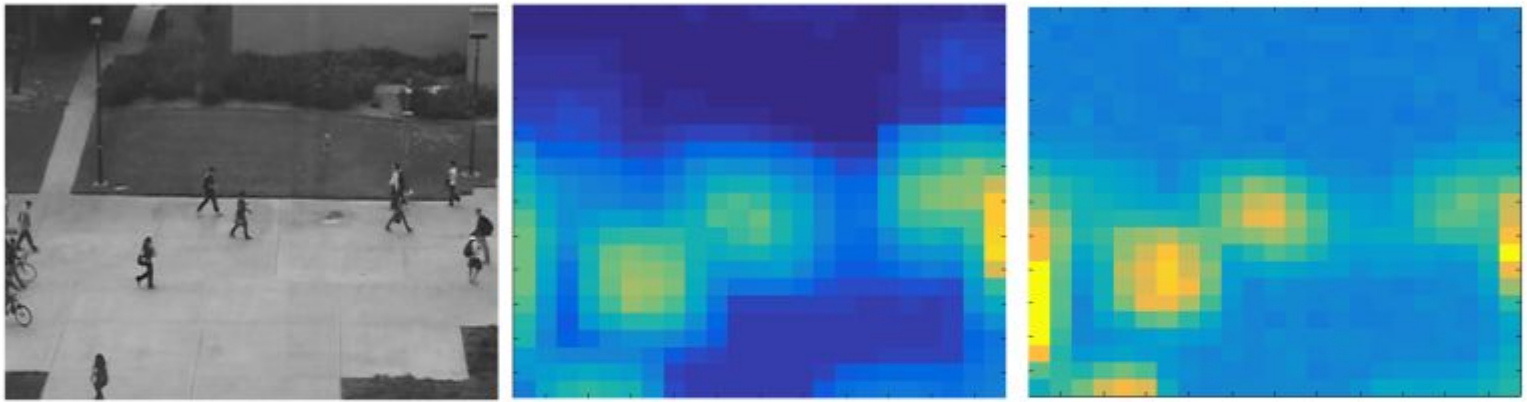
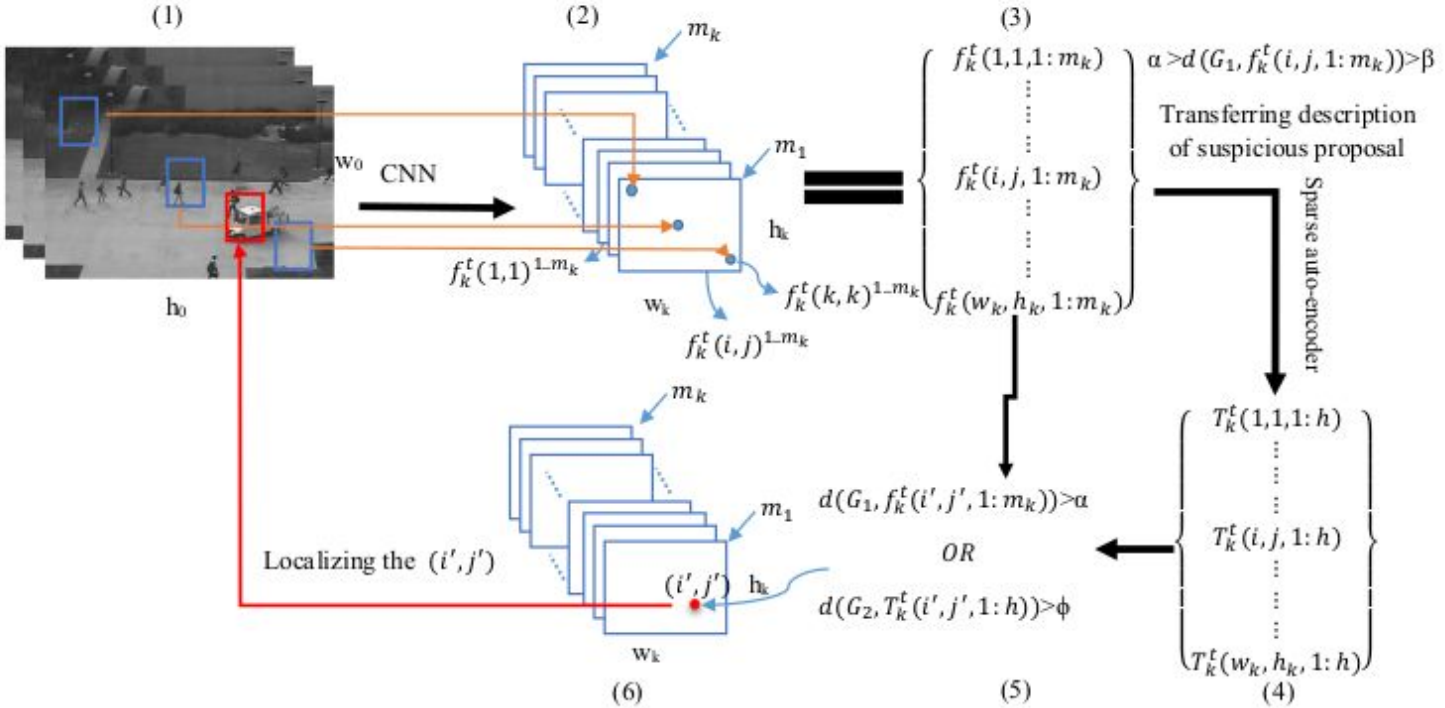


Figure 1: Effect of representing receptive fields with an added convolutional layer. *Left:* Input frame. *Middle:* Heat-map visualisation of the 2^{nd} layer of a pre-trained FCN. *Right:* Heat-map visualisation of the 3^{nd} layer of a pre-trained FCN with added convolutional layer.

About Deep Anomaly

Network Architecture



Pros of this implementation

- Real time application (300+fps)
- Accuracy at par or better than the state of the art
  Probabilities may be relative, or worse, wrong
- Only Fine Tuning required, uses existing networks to improve its performance
- Largely unsupervised
- First time that an FCN is used for anomaly detection]
- FIRST PUBLIC IMPLEMENTATION

Current Progress

- Understood the paper, previous work and network components
- Procured the datasets, discarded some
- Prepared preprocessing functions
- Procured Alexnet
- Built the trainable layers- Sparse Auto-encoders, etc.
- Built the distance function metrics and the Gaussian Classifiers.
- Built a method to implement Transfer Learning.

Let us walk you through our work...

Datasets and preprocessing scripts

Part of Alexnet that would be integrated into our code

Final layers of the network that we built

Thank you!

What Next?

- Integrating the networks
- Checking if everything works the way it should
- Fine Tuning the final layers
- Testing the results
- Prepare a report