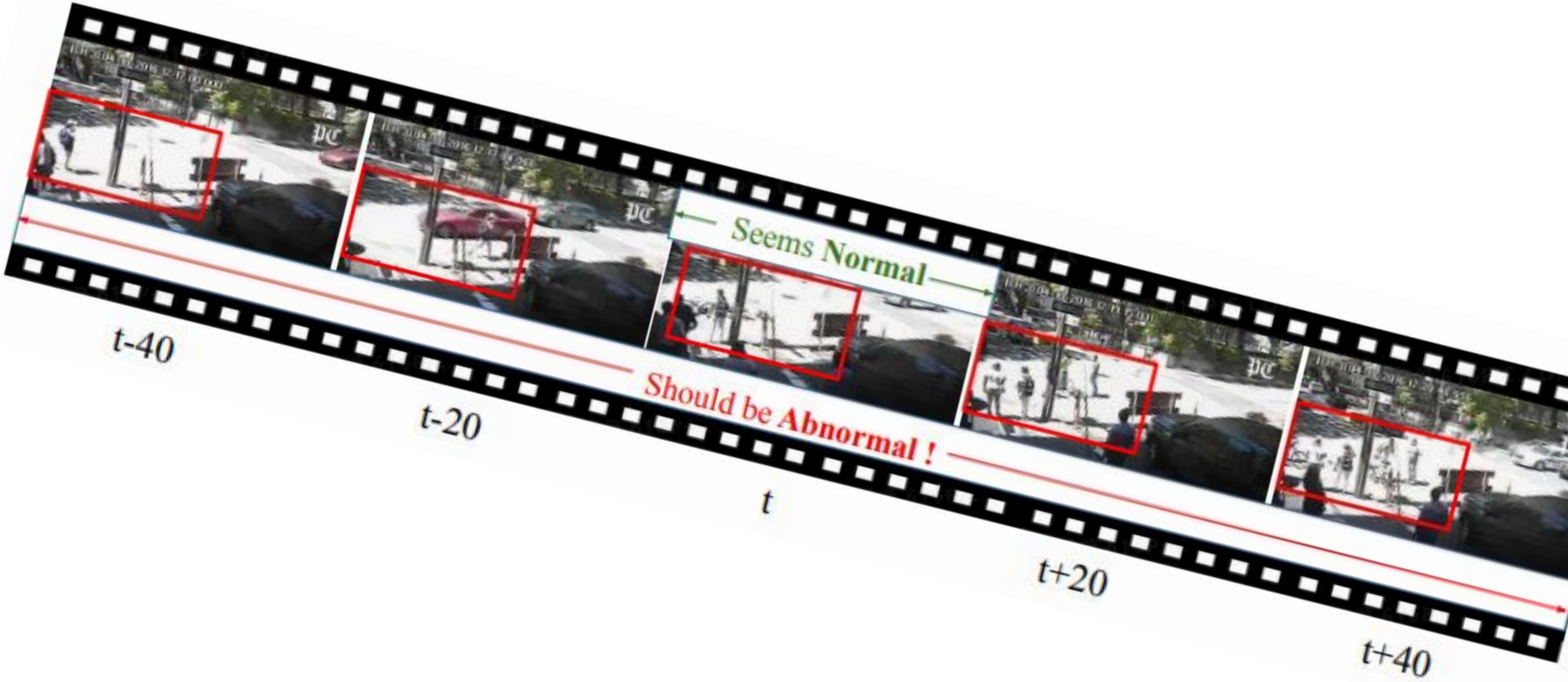
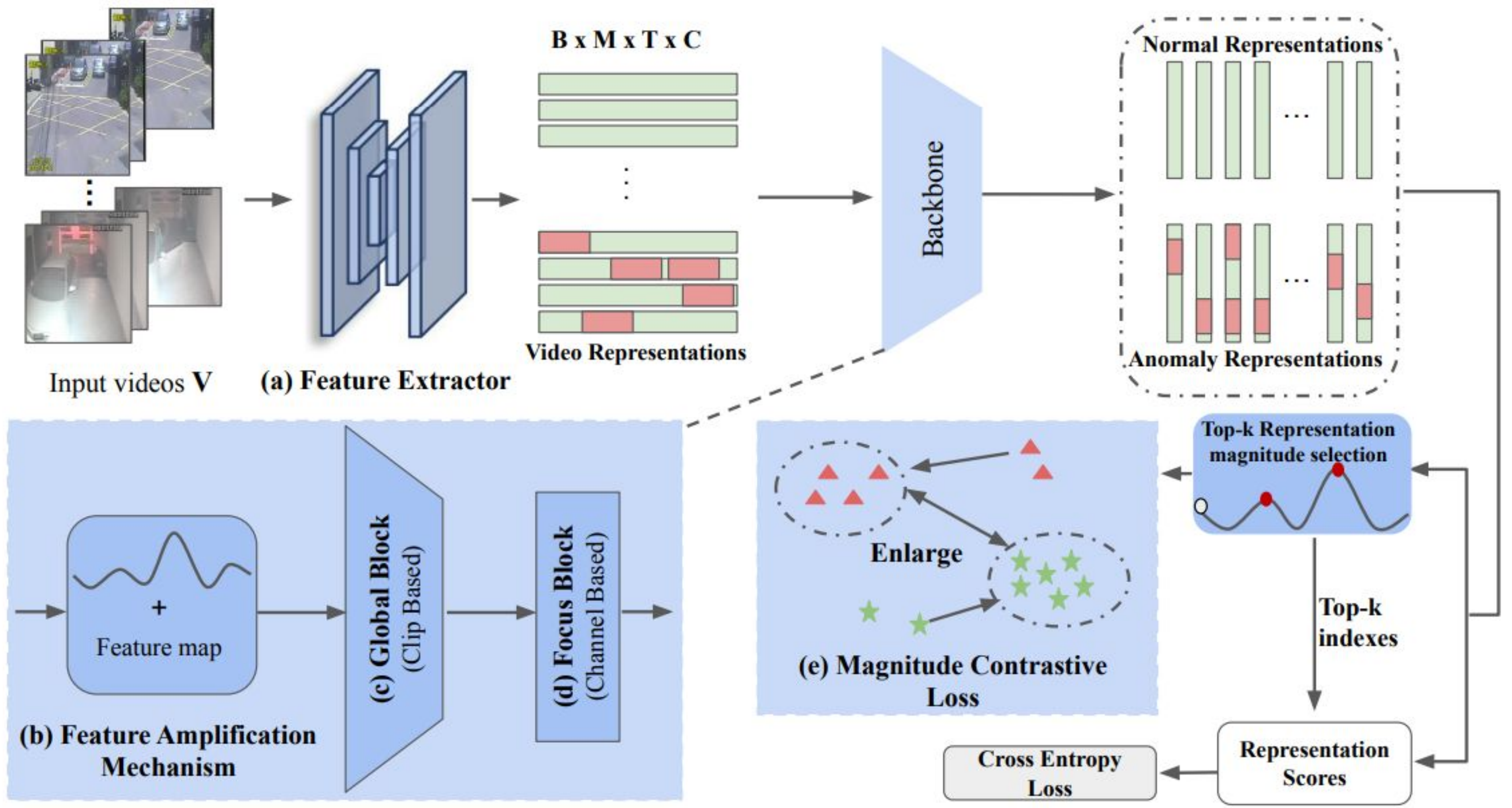
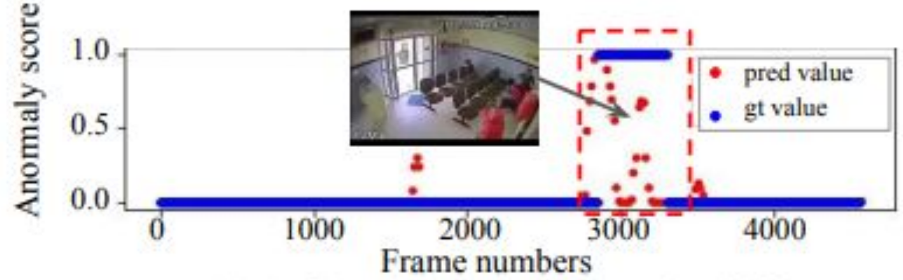
MGFN : Magnitude-Contrastive Glance-and-Focus Network for Weakly-Supervised **Video Anomaly Detection**

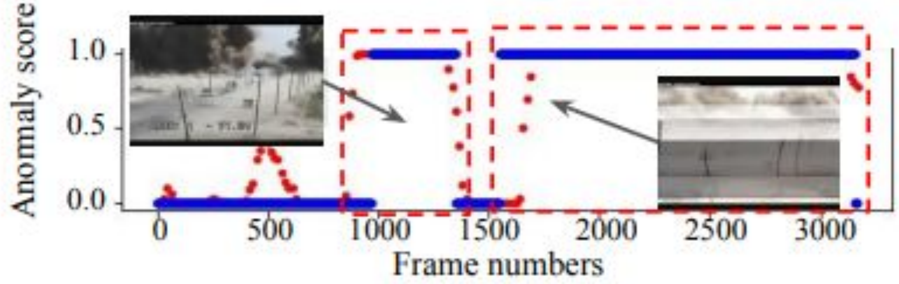

Seems Normal
Should be Abnormal !
t-40
t-20
t
t+20
t+40



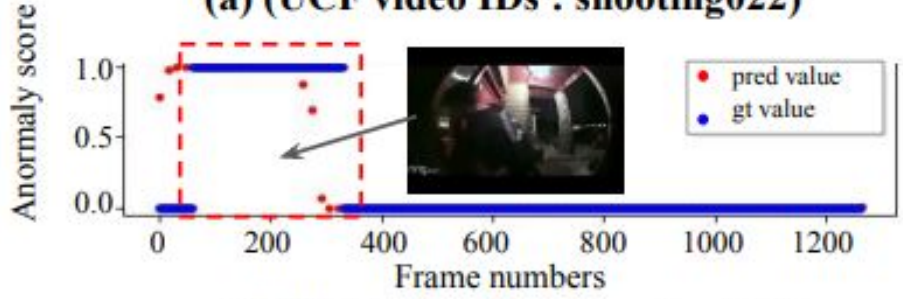
🟥 Anomaly Representation 🟩 Normal Representation ● Top-k anomaly magnitudes ▲ Top-k abnormal features ★ Top-k normal features



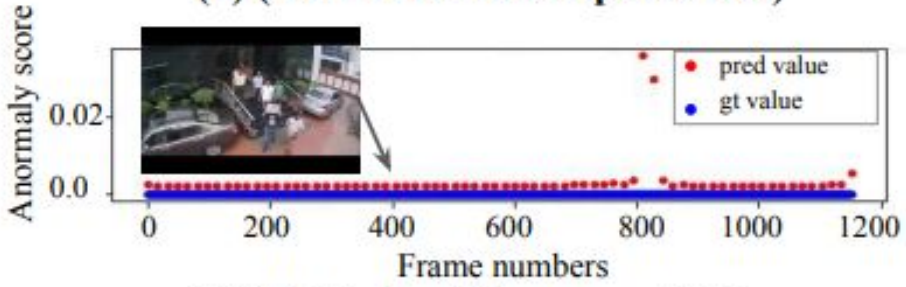
(a) (UCF video IDs : shooting022)



(b) (UCF video IDs :explosion008)



(c) (UCF video IDs : burglary033)



(d) (UCF video IDs : normal248)

1. "…/MGFN./option.py" Code

- **Purpose**: Defines a function parse_args() to parse command-line arguments using argparse.
- **Argument Definitions:** Specifies various arguments for the program such as feature extractor type, feature size, modality, paths to data lists and ground truth files, and other hyperparameters like dropout rate and learning rate.
- **GPU Settings:** Sets the environment variable CUDA_VISIBLE_DEVICES based on the --gpus argument, allowing the specification of GPUs to be used.
- **Argument Parsing:** Uses argparse.ArgumentParser to handle command-line arguments and returns the parsed arguments.
- **Flexibility:** Allows easy modification and experimentation with different settings by changing command-line arguments.

2. "../MGFN./train.py" Code

- **Loss Functions:** Defines custom loss functions, including contrastive loss, sparsity, and smoothness loss, for training the model.
- **Training Function:** Contains the train function which performs the forward pass and backpropagation on the training data.
- **Model Update:** Updates the model parameters using the computed loss and optimizer.
- **Loss Computation:** Combines multiple loss functions to compute the total loss for each training step.
- **Utility Functions:** Provides functions to compute specific types of losses based on the model's outputs and training data.

3. "…/MGFN./main.py" Code

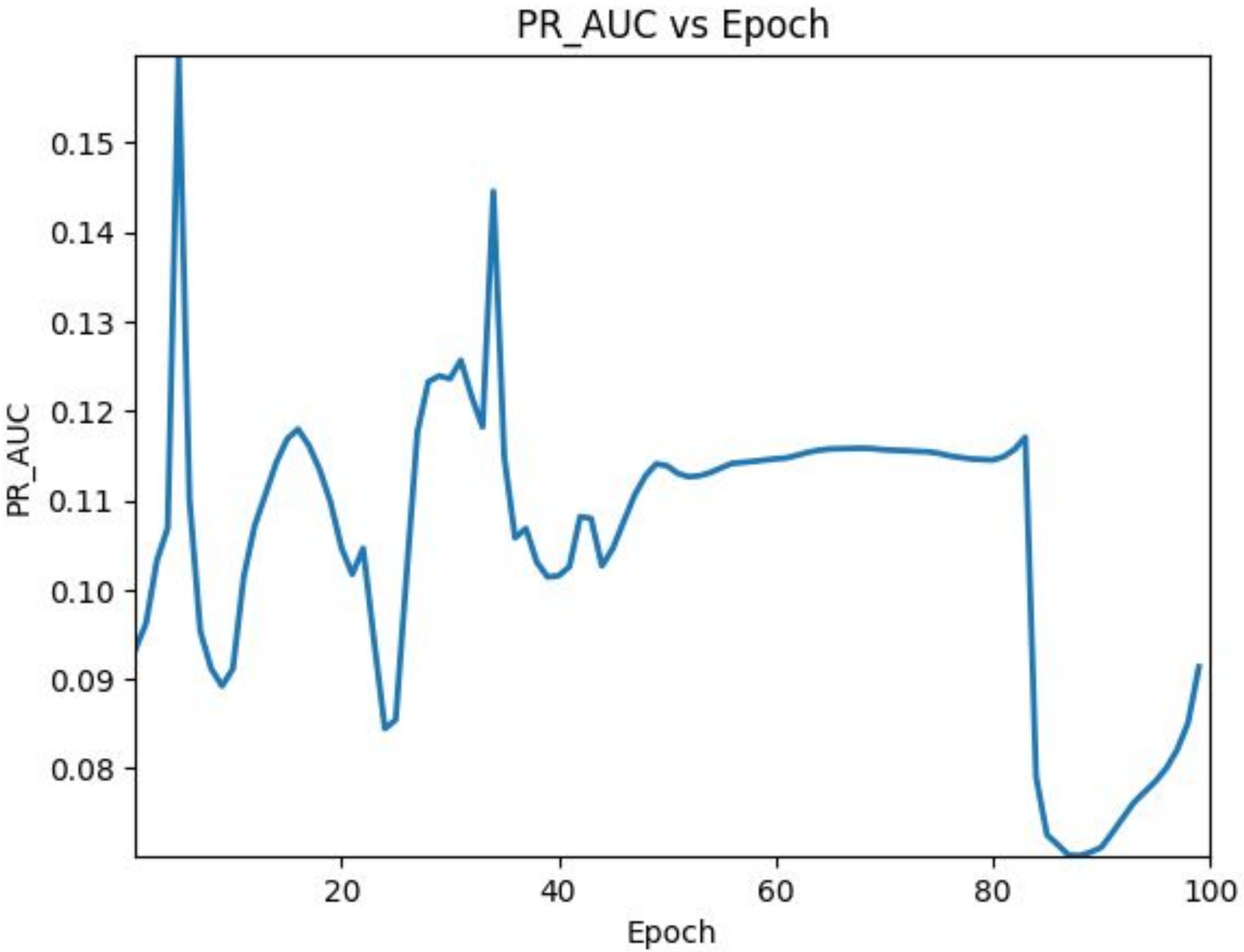
- **Training and Testing Setup:** Initializes configurations, data loaders, and model for training and testing using a deep learning model.

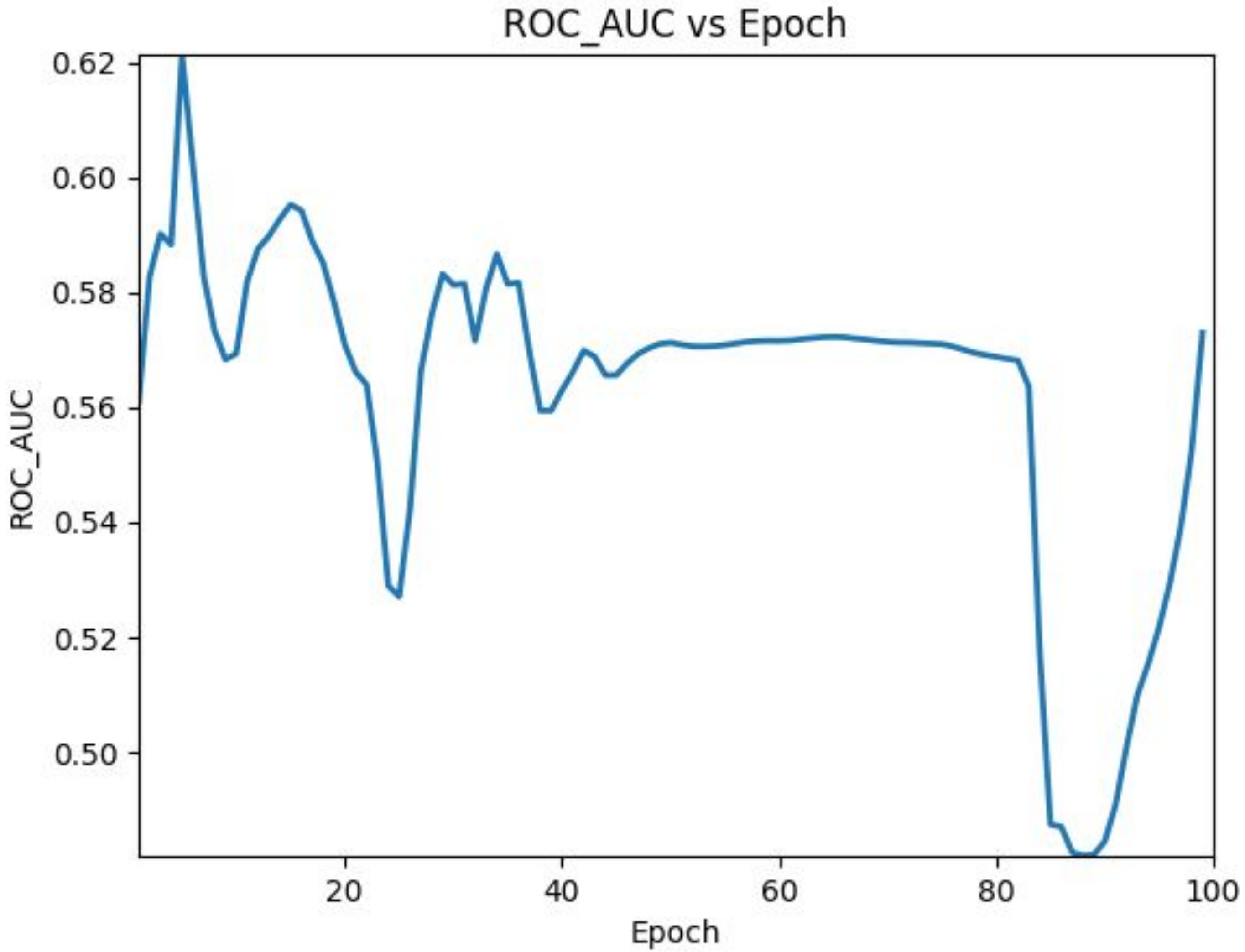
- **Model Initialization:** Loads a pre-trained model if available, sets up the device (GPU/CPU) for training, and initializes the optimizer.

- **Training Loop:** Iterates over epochs, trains the model on the training data, and computes metrics like AUC and PR AUC on the test data.

- **Logging and Saving:** Writes training and test results to a CSV file and uses tensorboardX for logging. Saves the best model based on performance metrics.

- **Configuration Management:** Saves the configuration and model parameters to a file for reproducibility.


PR_AUC vs Epoch
PR_AUC
0.15
0.14
0.13
0.12
0.11
0.10
0.09
0.08
20
40
60
80
100
Epoch


ROC_AUC vs Epoch
ROC_AUC
Epoch
0.62
0.60
0.58
0.56
0.54
0.52
0.50
20
40
60
80
100