Learning Memory-guided Normality for Anomaly Detection

Hyunjong Park* Jongyoun Noh* Bumsub Ham†
School of Electrical and Electronic Engineering, Yonsei University

Abstract

We address the problem of anomaly detection, that is, detecting anomalous events in a video sequence. Anomaly detection methods based on convolutional neural networks (CNNs) typically leverage proxy tasks, such as reconstructing input video frames, to learn models describing normality without seeing anomalous samples at training time, and quantify the extent of abnormalities using the reconstruction error at test time. The main drawbacks of these approaches are that they do not consider the diversity of normal patterns explicitly, and the powerful representation capacity of CNNs allows to reconstruct abnormal video frames. To address this problem, we present an unsupervised learning approach to anomaly detection that considers the diversity of normal patterns explicitly, while lessening the representation capacity of CNNs. To this end, we propose to use a memory module with a new update scheme where items in the memory record prototypical patterns of normal data. We also present novel feature compactness and separateness losses to train the memory, boosting the discriminative power of both memory items and deeply learned features from normal data. Experimental results on standard benchmarks demonstrate the effectiveness and efficiency of our approach, which outperforms the state of the art.

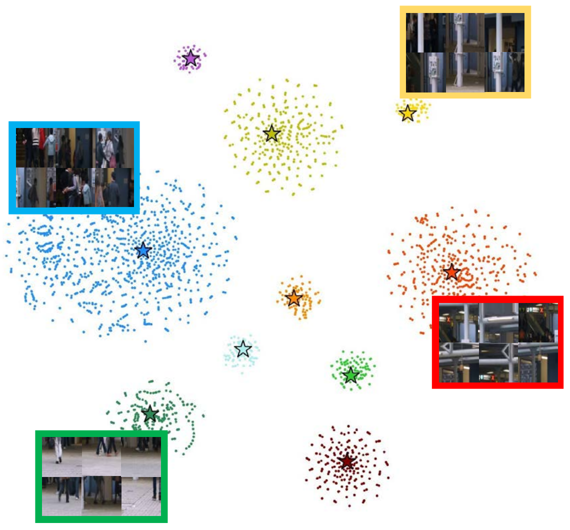
Figure 1: Distributions of features and memory items of our model on CUHK Avenue [24]. The features and items are shown in points and stars, respectively. The points with the same color are mapped to the same item. The items in the memory capture diverse and prototypical patterns of normal data. The features are highly discriminative and similar image patches are clustered well. (Best viewed in color.)

1. Introduction

The problem of detecting abnormal events in video sequences, *e.g.*, vehicles on sidewalks, has attracted significant attention over the last decade, which is particularly important for surveillance and fault detection systems. It is extremely challenging for a number of reasons: First, anomalous events are determined differently according to circumstances. Namely, the same activity could be normal or abnormal (*e.g.*, holding a knife in the kitchen or in the park). Manually annotating anomalous events is in this context labor intensive. Second, collecting anomalous datasets requires a lot of effort, as anomalous events rarely happen in real-life situations. Anomaly detection is thus typically deemed to be an unsupervised learning problem, aiming at learning a model describing normality without anomalous samples. At test time, events and activities not described by the model are then considered as anomalies.

*Equal contribution. †Corresponding author.

There are many attempts to model normality in video sequences using unsupervised learning approaches. At training time, given normal video frames as inputs, they typically extract feature representations and try to reconstruct the inputs again. The video frames of large reconstruction errors are then treated as anomalies at test time. This assumes that abnormal samples are not reconstructed well, as the models have never seen them during training. Recent methods based on convolutional neural networks (CNNs) exploit an autoencoder (AE) [1, 17]. The powerful representation capacity of CNNs allows to extract better feature representations. The CNN features from abnormal frames, on the other hand, are likely to be reconstructed by combining those of normal ones [22, 8]. In this case, abnormal frames have low reconstruction errors, often occurring when a majority of the abnormal frames are normal (*e.g*., pedestrians in a park). In order to lessen the capacity of CNNs, a video prediction framework [22] is introduced that minimizes the difference between a predicted future frame and its ground truth. The drawback of these methods [1, 17, 22] is that they do not detect anomalies directly [35]. They instead leverage proxy tasks for anomaly detection, *e.g*., reconstructing input frames [1, 17] or predicting future frames [22], to extract general feature representations rather than normal patterns. To overcome this problem, Deep SVDD [35] exploits the one-class classification objective to map normal data into a hypersphere. Specifically, it minimizes the volume of the hypersphere such that normal samples are mapped closely to the center of the sphere. Although a single center of the sphere represents a universal characteristic of normal data, this does not consider various patterns of normal samples.

We present in this paper an unsupervised learning approach to anomaly detection in video sequences considering the diversity of normal patterns. We assume that a single prototypical feature is not enough to represent various patterns of normal data. That is, multiple prototypes (*i.e*., modes or centroids of features) exist in the feature space of normal video frames (Fig. 1). To implement this idea, we propose a memory module for anomaly detection, where individual items in the memory correspond to prototypical features of normal patterns. We represent video frames using the prototypical features in the memory items, lessening the capacity of CNNs. To reduce the intra-class variations of CNN features, we propose a feature compactness loss, mapping the features of a normal video frame to the nearest item in the memory and encouraging them to be close. Simply updating memory items and extracting CNN features alternatively in turn give a degenerate solution, where all items are similar and thus all features are mapped closely in the embedding space. To address this problem, we propose a feature separateness loss. It minimizes the distance between each feature and its nearest item, while maximizing the discrepancy between the feature and the second nearest one, separating individual items in the memory and enhancing the discriminative power of the features and memory items. We also introduce an update strategy to prevent the memory from recording features of anomalous samples at test time. To this end, we propose a weighted regular score measuring how many anomalies exist within a video frame, such that the items are updated only when the frame is determined as a normal one. Experimental results on standard benchmarks, including UCSD Ped2 [21], CUHK Avenue [24] and ShanghaiTech [26], demonstrate the effectiveness and efficiency of our approach, outperforming the state of the art.

The main contributions of this paper can be summarized as follows:

- We propose to use multiple prototypes to represent the diverse patterns of normal video frames for unsupervised anomaly detection. To this end, we introduce a memory module recording prototypical patterns of normal data on the items in the memory.
- We propose feature compactness and separateness losses to train the memory, ensuring the diversity and discriminative power of the memory items. We also present a new update scheme of the memory, when both normal and abnormal samples exist at test time.
- We achieve a new state of the art on standard benchmarks for unsupervised anomaly detection in video sequences. We also provide an extensive experimental analysis with ablation studies.

Our code and models are available online: `https://cvlab.yonsei.ac.kr/projects/MNAD`.

2. Related work

Anomaly detection. Many works formulate anomaly detection as an unsupervised learning problem, where anomalous data are not available at training time. They typically adopt reconstructive or discriminative approaches to learn models describing normality. Reconstructive models encode normal patterns using representation learning methods such as an AE [48, 36], a sparse dictionary learning [6, 49, 24], and a generative model [43]. Discriminative models characterize the statistical distributions of normal samples and obtain decision boundaries around the normal instances *e.g*., using Markov random field (MRF) [15], a mixture of dynamic textures (MDT) [28], Gaussian regression [4], and one-class classification [39, 27, 14]. These approaches, however, often fail to capture the complex distributions of high-dimensional data such as images and videos [3].

CNNs have allowed remarkable advances in anomaly detection over the last decade. Many anomaly detection methods leverage reconstructive models [9, 26, 5, 33] exploiting feature representations from *e.g*., a convolutional AE (Conv-AE) [9], a 3D Conv-AE [50], a recurrent neural network (RNN) [29, 26, 25], and a generative adversarial network (GAN) [33]. Although CNN-based methods outperform classical approaches by large margins, they even reconstruct anomalous samples with a combination of normal ones, mainly due to the representation capacity of CNNs. This problem can be alleviated by using predictive or discriminative models [22, 35]. The work of [22] assumes that anomalous frames in video sequences are unpredictable, and trains a network for predicting future frames rather than the input itself [22]. It achieves a remarkable performance gain over reconstructive models, but at the cost of runtime for estimating optical flow between video frames. It also requires ground-truth optical flow to train a sub-network for computing flow fields. Deep SVDD [35]

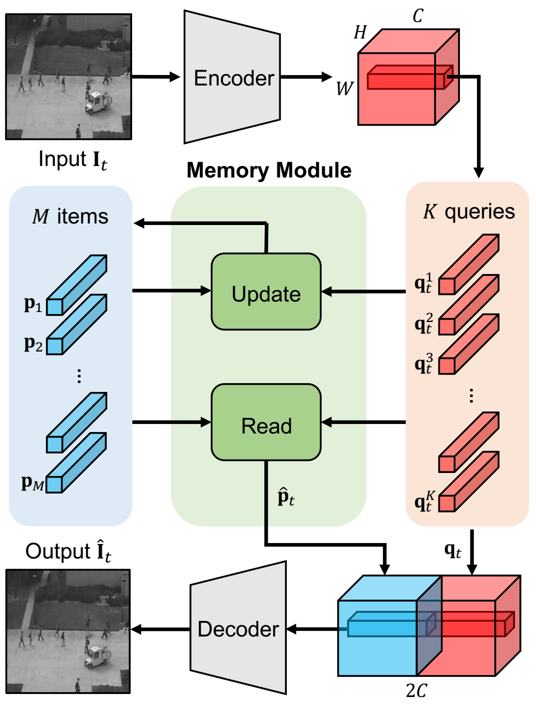

Figure 2: Overview of our framework for reconstructing a video frame. Our model mainly consists of three parts: an encoder, a memory module, and a decoder. The encoder extracts a query map $\mathbf{q}_t$ of size $H \times W \times C$ from an input video frame $\mathbf{I}_t$ at time t. The memory module performs reading and updating items $\mathbf{p}_m$ of size $1 \times 1 \times C$ using queries $\mathbf{q}_t^k$ of size $1 \times 1 \times C$, where the numbers of items and queries are M and K, respectively, and $K = H \times W$. The query map $\mathbf{q}_t$ is concatenated with the aggregated (*i.e.*, read) items $\hat{\mathbf{p}}_t$. The decoder then inputs them to reconstruct the video frame $\hat{\mathbf{I}}_t$. For the prediction task, we input four successive video frames to predict the fifth one. (Best viewed in color.)

leverages CNNs as mapping functions that transform normal data into the center of the hypersphere, whereas forcing anomalous samples to fall outside the sphere, using the one-class classification objective. Our method also lessens the representation capacity of CNNs but using a different way. We reconstruct or predict a video frame with a combination of items in the memory, rather than using CNN features directly from an encoder, while considering various patterns of normal data. In case of future frame prediction, our model does not require computing optical flow, and thus it is much faster than the current method [22]. Deep-Cascade [37] detects various normal patches using cascaded deep networks. In contrast, our method leverages memory items to record the normal pattern explicitly even in test sequences. Concurrent to our method, Gong *et al.* introduce a memory-augmented autoencoder (MemAE) for anomaly detection [8]. It also uses CNN features but using a 3D Conv-AE to retrieve relevant memory items that record normal patterns, where the items are updated during training only. Unlike this approach, our model better records *diverse and discriminative* normal patterns by separating memory items explicitly using feature compactness and separateness losses, enabling using a small number of items compared to MemAE (10 vs 2,000 for MemAE). We also update the memory at test time, while discriminating anomalies simultaneously, suggesting that our model also memorizes normal patterns of test data.

Memory networks. There are a number of attempts to capture long-term dependencies in sequential data. Long short-term memory (LSTM) [11] addresses this problem using local memory cells, where hidden states of the network record information in the past partially. The memorization performance is, however, limited, as the size of the cell is typically small and the knowledge in the hidden state is compressed. To overcome the limitation, memory networks [45] have recently been introduced. It uses a global memory that can be read and written to, and performs a memorization task better than classical approaches. The memory networks, however, require layer-wise supervision to learn models, making it hard to train them using standard backpropagation. More recent works use continuous memory representations [40] or key-value pairs [30] to read/write memories, allowing to train the memory networks end-to-end. Several works adopt the memory networks for computer vision tasks including visual question answering [19, 7], one-shot learning [38, 13, 2], image generation [51], and video summarization [20]. Our work also exploits a memory module but for anomaly detection with a different memory updating strategy. We record various patterns of normal data to individual items in the memory, and consider each item as a prototypical feature.

3. Approach

We show in Fig. 2 an overview of our framework. We reconstruct input frames or predict future ones for unsupervised anomaly detection. Following [22], we input four successive video frames to predict the fifth one for the prediction task. As the prediction can be considered as a reconstruction of the future frame using previous ones, we use almost the same network architecture with the same losses for both tasks. We describe hereafter our approach for the reconstruction task in detail.

Our model mainly consists of three components: an encoder, a memory module, and a decoder. The encoder inputs a normal video frame and extracts query features. The features are then used to retrieve prototypical normal patterns in the memory items and to update the memory. We feed the query features and memory items aggregated (*i.e.*, read) to the decoder for reconstructing the input video frame. We train our model using reconstruction, feature compactness, and feature separateness losses end-to-end. At test time, we use a weighted regular score in order to prevent the memory from being updated by abnormal video frames. We com-

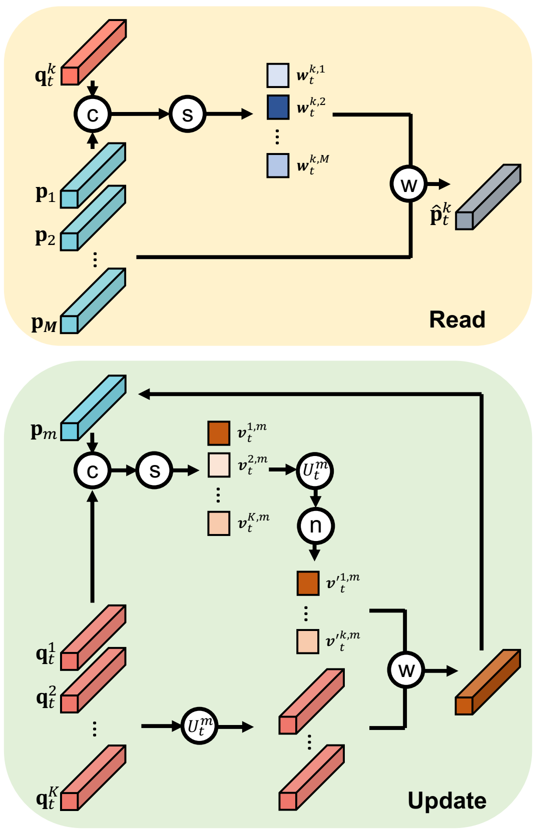

Figure 3: Illustration of reading and updating the memory. To read items in the memory, we compute matching probabilities $w_t^{k,m}$ in (1) between the query $\mathbf{q}_t^k$ and items $(\mathbf{p}_1, \ldots \mathbf{p}_M)$, and apply a weighted average of the items with the probabilities to obtain the feature $\hat{\mathbf{p}}_t^k$. To update the items, we compute another matching probabilities $v_t^{k,m}$ in (4) between the item $\mathbf{p}_m$ and the queries $(\mathbf{q}_t^1, \ldots \mathbf{q}_t^K)$. We then compute a weighted average of the queries in the set U_t^m with the corresponding probabilities, and add it to the initial item $\mathbf{p}_m$ in (3). c: cosine similarities; s: a softmax function; w: a weighted average; n: max normalization; U_t^m: a set of indices for the m-th memory item. See text for details. (Best viewed in color.)

pute the discrepancies between the input frame and its reconstruction and the distances between the query feature and the nearest item in the memory to quantify the extent of abnormalities in a video frame.

3.1. Network architecture

3.1.1 Encoder and decoder

We exploit the U-Net architecture [34], widely used for the tasks of reconstruction and future frame prediction [22], to extract feature representations from input video frames and to reconstruct the frames from the features. Differently, we remove the last batch normalization [12] and ReLU layers [18] in the encoder, as the ReLU cuts off negative values, restricting diverse feature representations. We instead add an L2 normalization layer to make the features have a common scale. Skip connections in the U-Net architecture may not be able to extract useful features from the video frames especially for the reconstruction task, and our model may learn to copy the inputs for the reconstruction. We thus remove the skip connections for the reconstruction task, while retaining them for predicting future frames. We denote by $\mathbf{I}_t$ and $\mathbf{q}_t$ a video frame and a corresponding feature (*i.e.*, a query) from the encoder at time t, respectively. The encoder inputs the video frame $\mathbf{I}_t$ and gives the query map $\mathbf{q}_t$ of size $H \times W \times C$, where H, W, C are height, width, and the number of channels, respectively. We denote by $\mathbf{q}_t^k \in \mathbb{R}^C$ $(k = 1, \ldots K)$, where $K = H \times W$, individual queries of size $1 \times 1 \times C$ in the query map $\mathbf{q}_t$. The queries are then inputted to the memory module to read the items in the memory or to update the items, such that they record prototypical normal patterns. The detailed descriptions of the memory module are presented in the following section. The decoder inputs the queries and retrieved memory items and reconstructs the video frame $\hat{\mathbf{I}}_t$.

3.1.2 Memory

The memory module contains M items recording various prototypical patterns of normal data. We denote by $\mathbf{p}_m \in \mathbb{R}^C$ $(m = 1, \ldots, M)$ the item in the memory. The memory performs reading and updating the items (Fig. 3).

Read. To read the items, we compute the cosine similarity between each query $\mathbf{q}_t^k$ and all memory items $\mathbf{p}_m$, resulting in a 2-dimensional correlation map of size $M \times K$. We then apply a softmax function along a vertical direction, and obtain matching probabilities $w_t^{k,m}$ as follows:

$$w_t^{k,m} = \frac{\exp((\mathbf{p}_m)^T \mathbf{q}_t^k)}{\sum_{m'=1}^{M} \exp((\mathbf{p}_{m'})^T \mathbf{q}_t^k)}. \quad (1)$$

For each query $\mathbf{q}_t^k$, we read the memory by a weighted average of the items $\mathbf{p}_m$ with the corresponding weights $w_t^{k,m}$, and obtain the feature $\hat{\mathbf{p}}_t^k \in \mathbb{R}^C$ as follows:

$$\hat{\mathbf{p}}_t^k = \sum_{m'=1}^{M} w_t^{k,m'} \mathbf{p}_{m'}. \quad (2)$$

Using all items instead of the closest item allows our model to understand diverse normal patterns, taking into account the overall normal characteristics. That is, we represent the query $\mathbf{q}_t^k$ with a combination of the items $\mathbf{p}_m$ in the memory. We apply the reading operator to individual queries, and obtain a transformed feature map $\hat{\mathbf{p}}_t \in \mathbb{R}^{H \times W \times C}$ (*i.e.*, aggregated items). We concatenate it with the query map $\mathbf{q}_t$ along the channel dimension, and input them to the decoder. This enables the decoder to reconstruct the input frame using normal patterns in the items, lessening the representation capacity of CNNs, while understanding the normality.

Update. For each memory item, we select all queries declared that the item is the nearest one, using the matching probabilities in (1). Note that multiple queries can be assigned to a single item in the memory. See, for example, Fig. 5 in Sec. 4.3. We denote by U_t^m the set of indices for the corresponding queries for the m-th item in the memory. We update the item using the queries indexed by the set U_t^m only as follows:

$$\mathbf{p}^m \leftarrow f(\mathbf{p}^m + \sum_{k \in U_t^m} {v'}_t^{k,m} \mathbf{q}_t^k), \tag{3}$$

where $f(\cdot)$ is the L2 norm. By using a weighted average of the queries, rather than summing them up, we can concentrate more on the queries near the item. To this end, we compute matching probabilities $v_t^{k,m}$ similar to (1) but by applying the softmax function to the correlation map of size $M \times K$ along a horizontal direction as

$$v_t^{k,m} = \frac{\exp((\mathbf{p}_m)^T \mathbf{q}_t^k)}{\sum_{k'=1}^{K} \exp((\mathbf{p}_m)^T \mathbf{q}_t^{k'})}, \tag{4}$$

and renormalize it to consider the queries indexed by the set U_t^m as follows:

$${v'}_t^{k,m} = \frac{v_t^{k,m}}{\max_{k' \in U_t^m} v_t^{k',m}}. \tag{5}$$

We update memory items recording prototypical features at both training and test time, since normal patterns in training and test sets may be different and they could vary with various factors, *e.g.*, illumination and occlusion. As both normal and abnormal frames are available at test time, we propose to use a weighted regular score to prevent the memory items from recording patterns in the abnormal frames. Given a video frame $\mathbf{I}_t$, we use the weighted reconstruction error between $\mathbf{I}_t$ and $\hat{\mathbf{I}}_t$ as the regular score $\mathcal{E}_t$:

$$\mathcal{E}_t = \sum_{i,j} W_{ij}(\hat{\mathbf{I}}_t, \mathbf{I}_t) \|\hat{\mathbf{I}}_t^{ij} - \mathbf{I}_t^{ij}\|_2, \tag{6}$$

where the weight function $W_{ij}(\cdot)$ is

$$W_{ij}(\hat{\mathbf{I}}_t, \mathbf{I}_t) = \frac{1 - \exp(-\|\hat{\mathbf{I}}_t^{ij} - \mathbf{I}_t^{ij}\|_2)}{\sum_{i,j} 1 - \exp(-\|\hat{\mathbf{I}}_t^{ij} - \mathbf{I}_t^{ij}\|_2)}, \tag{7}$$

and i and j are spatial indices. When the score $\mathcal{E}_t$ is higher than a threshold γ, we regard the frame $\mathbf{I}_t$ as an abnormal sample, and do not use it for updating memory items. Note that we use this score only when updating the memory. The weight function allows to focus more on the regions of large reconstruction errors, as abnormal activities typically appear within small parts of the video frame.

3.2. Training loss

We exploit the video frames as a supervisory signal to discriminate normal and abnormal samples. To train our model, we use reconstruction, feature compactness, and feature separateness losses ($\mathcal{L}_{\text{rec}}$, $\mathcal{L}_{\text{compact}}$ and $\mathcal{L}_{\text{separate}}$, respectively), balanced by the parameters λ_{c} and λ_{s} as follows:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \lambda_{\text{c}} \mathcal{L}_{\text{compact}} + \lambda_{\text{s}} \mathcal{L}_{\text{separate}}. \tag{8}$$

Reconstruction loss. The reconstruction loss makes the video frame reconstructed from the decoder similar to its ground truth by penalizing the intensity differences. Specifically, we minimize the L2 distance between the decoder output $\hat{\mathbf{I}}_t$ and the ground truth $\mathbf{I}_t$:

$$\mathcal{L}_{\text{rec}} = \sum_{t}^{T} \|\hat{\mathbf{I}}_t - \mathbf{I}_t\|_2, \tag{9}$$

where we denote T by the total length of a video sequence. We set the first time step to 1 and 5 for reconstruction and prediction tasks, respectively.

Feature compactness loss. The feature compactness loss encourages the queries to be close to the nearest item in the memory, reducing intra-class variations. It penalizes the discrepancies between them in terms of the L2 norm as:

$$\mathcal{L}_{\text{compact}} = \sum_{t}^{T} \sum_{k}^{K} \|\mathbf{q}_t^k - \mathbf{p}_p\|_2, \tag{10}$$

where p is an index of the nearest item for the query $\mathbf{q}_t^k$ defined as,

$$p = \underset{m \in M}{\operatorname{argmax}}\, w_t^{k,m}. \tag{11}$$

Note that the feature compactness loss and the center loss [44] are similar, as the memory item $\mathbf{p}_p$ corresponds the center of deep features in the center loss. They are different in that the item in (10) is retrieved from the memory, and it is updated without any supervisory signals, while the cluster center in the center loss is computed directly using the features learned from ground-truth class labels. Note also that our method can be considered as an unsupervised learning of joint clustering and feature representations. In this task, degenerate solutions are likely to be obtained [44, 47]. As will be seen in our experiments, training our model using the feature compactness loss only makes all items similar, and thus all queries are mapped closely in the embedding space, losing the capability of recording diverse normal patterns.

Feature separateness loss. Similar queries should be allocated to the same item in order to reduce the number of items and the memory size. The feature compactness loss in (10) makes all queries and memory items close to each other, as we extract the features (*i.e.*, queries) and update the items alternatively, resulting that all items are similar. The items in the memory, however, should be far enough apart from each other to consider various patterns of normal data. To prevent this problem while obtaining compact feature representations, we propose a feature separateness loss, defined with a margin of α as follows:

$$\mathcal{L}_{\text{separate}} = \sum_{t}^{T} \sum_{k}^{K} [\|\mathbf{q}_t^k - \mathbf{p}_p\|_2 - \|\mathbf{q}_t^k - \mathbf{p}_n\|_2 + \alpha]_+, \tag{12}$$

where we set the query $\mathbf{q}_t^k$, its nearest item $\mathbf{p}_p$ and the second nearest item $\mathbf{p}_n$ as an anchor, and positive and hard negative samples, respectively. We denote by n an index of the second nearest item for the query $\mathbf{q}_t^k$:

$$n = \underset{m \in M, m \neq p}{\operatorname{argmax}} \; w_t^{k,m}. \tag{13}$$

Note that this is different from the typical use of the triplet loss that requires ground-truth positive and negative samples for the anchor. Our loss encourages the query and the second nearest item to be distant, while the query and the nearest one to be nearby. This has the effect of placing the items far away. As a result, the feature separateness loss allows to update the item nearest to the query, whereas discarding the influence of the second nearest item, separating all items in the memory and enhancing the discriminative power.

3.3. Abnormality score

We quantify the extent of normalities or abnormalities in a video frame at test time. We assume that the queries obtained from a normal video frame are similar to the memory items, as they record prototypical patterns of normal data. We compute the L2 distance between each query and the nearest item as follows:

$$D(\mathbf{q}_t, \mathbf{p}) = \frac{1}{K} \sum_{k}^{K} \|\mathbf{q}_t^k - \mathbf{p}_p\|_2. \tag{14}$$

We also exploit the memory items implicitly to compute the abnormality score. We measure how well the video frame is reconstructed using the memory items. This assumes that anomalous patterns in the video frame are not reconstructed by the memory items. Following [22], we compute the PSNR between the input video frame and its reonstruction:

$$P(\hat{\mathbf{I}}_t, \mathbf{I}_t) = 10 \log_{10} \frac{\max(\hat{\mathbf{I}}_t)}{\|\hat{\mathbf{I}}_t - \mathbf{I}_t\|_2^2 / N}. \tag{15}$$

where N is the number of pixels in the video frame. When the frame $\mathbf{I}_t$ is abnormal, we obtain a low value of PSNR and vice versa. Following [22, 8, 26], we normalize each error in (14) and (15) in the range of [0, 1] by a min-max normalization [22]. We define the final abnormality score $\mathcal{S}_t$ for each video frame as the sum of two metrics, balanced by the parameter λ, as follows:

$$\mathcal{S}_t = \lambda(1 - g(P(\hat{\mathbf{I}}_t, \mathbf{I}_t))) + (1 - \lambda) g(D(\mathbf{q}_t, \mathbf{p})), \tag{16}$$

where we denote by $g(\cdot)$ the min-max normalization [22] over whole video frames, *e.g.*,

$$g(D(\mathbf{q}_t, \mathbf{p})) = \frac{D(\mathbf{q}_t, \mathbf{p}) - \min_t(D(\mathbf{q}_t, \mathbf{p})}{\max_t(D(\mathbf{q}_t, \mathbf{p})) - \min_t(D(\mathbf{q}_t, \mathbf{p}))}. \tag{17}$$

4. Experiments

4.1. Implementation details

Dataset. We evaluate our method on three benchmark datasets and compare the performance with the state of the art. 1) The UCSD Ped2 dataset [21] contains 16 training and 12 test videos with 12 irregular events, including riding a bike and driving a vehicle. 2) The CUHK Avenue dataset [24] consists of 16 training and 21 test videos with 47 abnormal events such as running and throwing stuff. 3) The ShanghaiTech dataset [26] contains 330 training and 107 test videos of 13 scenes. It is the largest dataset among existing benchmarks for anomaly detection.

Training. We resize each video frame to the size of 256 $\times$ 256 and normalize it to the range of [-1, 1]. We set the height H and the width W of the query feature map, and the numbers of feature channels C and memory items M to 32, 32, 512 and 10, respectively. We use the Adam optimizer [16] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, with a batch size of 4 for 60, 60, and 10 epochs on UCSD Ped2 [21], CUHK Avenue [24], and ShanghaiTech [26], respectively. We set initial learning rates to 2e-5 and 2e-4, respectively, for reconstruction and prediction tasks, and decay them using a cosine annealing method [23]. For the reconstruction task, we use a grid search to set hyper-parameters on the test split of UCSD Ped1 [21]: $\lambda_\mathrm{c} = 0.01$, $\lambda_\mathrm{s} = 0.01$, $\lambda = 0.7$, $\alpha = 1$ and $\gamma = 0.015$. We use different parameters for the prediction task similarly chosen using a grid search: $\lambda_\mathrm{c} = 0.1$, $\lambda_\mathrm{s} = 0.1$, $\lambda = 0.6$, $\alpha = 1$ and $\gamma = 0.01$. All models are trained end-to-end using `PyTorch` [32], taking about 1, 15 and 36 hours for UCSD Ped2, CUHK Avenue, and ShanghaiTech, respectively, with an Nvidia GTX TITAN Xp.

4.2. Results

Comparison with the state of the art. We compare in Table 1 our models with the state of the art for anomaly detection on UCSD Ped2 [21], CUHK Avenue [24], and ShanghaiTech [26]. Following the experimental protocol in [22, 8, 26], we measure the average area under curve (AUC) by computing the area under the receiver operation characteristics (ROC) with varying threshold values for abnormality scores. We report the AUC performance of our models using memory modules for the tasks of frame reconstruction and future frame prediction. For comparison, we also provide the performance without the memory module. The suffices '-R' and '-P' indicate the reconstruction and prediction tasks, respectively.

From the table, we observe three things: (1) Our model with the prediction task (Ours-P w/ Mem.) gives the best results on UCSD Ped2 and CUHK Avenue, achieving the average AUC of 97.0% and 88.5%, respectively. This demonstrates the effectiveness of our approach to exploiting a memory module for anomaly detection. Although our method is outperformed by Frame-Pred [22] on Shang-

Table 1: Quantitative comparison with the state of the art for anomaly detection. We measure the average AUC (%) on UCSD Ped2 [21], CUHK Avenue [24], and ShanghaiTech [26]. Numbers in bold indicate the best performance and underscored ones are the second best.

	Methods	Ped2 [21]	Avenue [24]	Shanghai [26]
	MPPCA [15]	69.3	-	-
	MPPC+SFA [15]	61.3	-	-
	MDT [28]	82.9	-	-
	AMDN [46]	90.8	-	-
	Unmasking [41]	82.2	80.6	-
	MT-FRCN [10]	92.2	-	-
	AMC [31]	<u>96.2</u>	<u>86.9</u>	-
Recon.	ConvAE [9]	85.0	80.0	60.9
	TSC [26]	91.0	80.6	67.9
	StackRNN [26]	92.2	81.7	68.0
	AbnormalGAN [33]	93.5	-	-
	MemAE w/o Mem. [8]	91.7	81.0	69.7
	MemAE w/ Mem. [8]	94.1	83.3	<u>71.2</u>
	Ours-R w/o Mem.	86.4	80.6	65.8
	Ours-R w/ Mem.	90.2	82.8	69.8
Pred.	Frame-Pred [22]	95.4	85.1	**72.8**
	Ours-P w/o Mem.	94.3	84.5	66.8
	Ours-P w/ Mem.	**97.0**	**88.5**	70.5

haiTech, it uses additional modules for estimating optical flow, which requires more network parameters and ground-truth flow fields. Moreover, Frame-Pred leverages an adversarial learning framework, taking lots of effort to train the network. On the contrary, our model uses a simple AE for extracting features and predicting the future frame, and thus it is much faster than Frame-Pred (67 fps vs. 25 fps). This suggests that our model offers a good compromise in terms of AUC and runtime; (2) Our model with the reconstruction task (Ours-R w/ Mem.) shows the competitive performance compared to other reconstructive methods on UCSD Ped2, and outperforms them on other datasets, except MemAE [8]. Note that MemAE exploits 3D convolutions with 2,000 memory items of size 256. On the contrary, our model uses 2D convolutions and it requires 10 items of size 512. It is thus computationally much cheaper than MemAE: 67 fps for our model vs. 45 fps for MemAE; (3) Our memory module boosts the AUC performance significantly regardless of the tasks on all datasets. For example, the AUC gains are 2.7%, 4.0%, and 3.7% on UCSD Ped2, CUHK Avenue, and ShanghaiTech, respectively, for the prediction task. This indicates that the memory module is generic and it can be added to other anomaly detection methods.

Runtime. With an Nvidia GTX TITAN Xp, our current implementation takes on average 0.015 seconds to determine abnormality for an image of size 256 $\times$ 256 on UCSD Ped2 [21]. Namely, we achieve 67 fps for anomaly detection, which is much faster than other state-of-the-art methods based on CNNs, *e.g.*, 20 fps for Unmasking [41], 50 fps for StackRNN [26], 25 fps for Frame-Pred [22], and 45 fps for MemAE [8] with the same setting as ours.

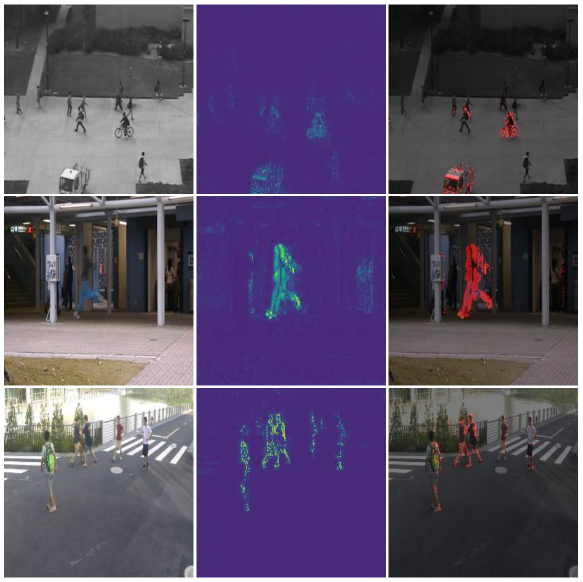
Figure 4: Qualitative results for future frame prediction on (top to bottom) UCSD Ped2 [21], CUHK Avenue [24], and ShanghaiTech [26]: input frames (left); prediction error (middle); abnormal regions (right). We can see that our model localizes the regions of abnormal events. Best viewed in color.

Qualitative results. We show in Fig. 4 qualitative results of our model for future frame prediction on UCSD Ped2 [21], CUHK Avenue [24], and ShanghaiTech [26]. It shows input frames, prediction error, and abnormal regions overlaid to the frame. For visualizing the anomalies, we compute pixel-wise abnormality scores similar to (16). We then mark the regions whose abnormality scores are larger than the average value within the frame. We can see that 1) normal regions are predicted well, while abnormal regions are not, and 2) abnormal events, such as the appearance of vehicle, jumping and fight on UCSD Ped2, CUHK Avenue, and ShanghaiTech, respectively, are highlighted.

4.3. Discussions

Ablation study. We show an ablation analysis on different components of our models in Table 2. We report the AUC performance for the variants of our models for reconstruction and prediction tasks on UCSD Ped2 [21]. As the AUC performance of both tasks shows a similar trend, we describe the results for the frame reconstruction in detail.

We train the baseline model in the first row with the reconstruction loss, and use PSNR only to compute abnormality scores. From the second row, we can see that our model with the memory module gives better results. The third row shows that the AUC performance even drops when the feature compactness loss is additionally used, as the memory

Table 2: Quantitative comparison for variants of our model. We measure the average AUC (%) on UCSD Ped2 [21].

Task	Memory module	$\mathcal{L}_{\text{compact}}$	$\mathcal{L}_{\text{separate}}$	$\mathcal{E}_t$	$\mathcal{S}_t$	Ped2 [21]
Recon.	✗	-	-	-	-	86.4
	✓	✗	✗	✓	✓	86.9
	✓	✓	✗	✓	✓	86.4
	✓	✗	✓	✓	✓	<u>89.3</u>
	✓	✓	✓	✗	✓	87.1
	✓	✓	✓	✓	✗	89.0
	✓	✓	✓	✓	✓	**90.2**
Pred.	✗	-	-	-	-	94.3
	✓	✗	✗	✓	✓	95.0
	✓	✓	✗	✓	✓	94.8
	✓	✗	✓	✓	✓	<u>96.5</u>
	✓	✓	✓	✗	✓	96.0
	✓	✓	✓	✓	✗	95.7
	✓	✓	✓	✓	✓	**97.0**

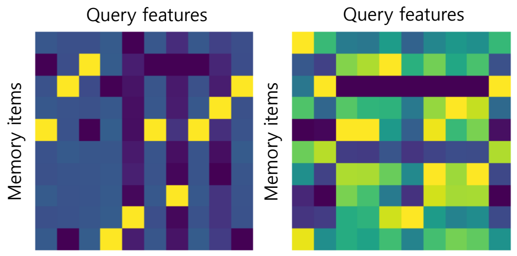

Figure 5: Visualization of matching probabilities in (1) learned with (left) and without (right) the feature separateness loss (blue: low, yellow: high). We randomly select 10 query features for the purpose of visualization. Best viewed in color.

items are not discriminative. The last row demonstrates that the feature separateness loss boosts the performance drastically. It provides the AUC gain of 3.8%, which is quite significant. The last four rows indicate that 1) feature compactness and separateness losses are complementary, 2) updating the memory item using $\mathcal{E}_t$ with normal frames only at test time largely boosts the AUC performance, and 3) our abnormality score $\mathcal{S}_t$, using both PSNR and memory items, quantifies the extent of anomalies better than the one based on PSNR only.

Memory items. We visualize in Fig. 5 matching probabilities in (1) from the model trained with/without the feature separateness loss for the reconstruction task on UCSD Ped2 [21]. We observe that each query is highly activated on a few items with the separateness loss, demonstrating that the items and queries are highly discriminative, allowing the sparse access of the memory. This also indicates that abnormal samples are not likely to be reconstructed with a combination of memory items.

Feature distribution. We visualize in Fig. 6 the distribution of query features for the reconstruction task, randomly chosen from UCSD Ped2 [21], learned with and without the feature separateness loss. We can see that our model trained

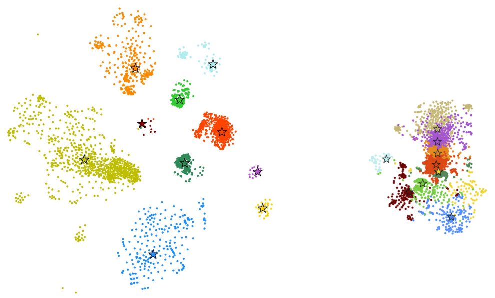
Figure 6: t-SNE [42] visualization for query features and memory items. We randomly sample 10K query features, learned with (left) and without (right) the feature separateness loss, from UCSD Ped2 [21]. The features and memory items are shown in points and stars, respectively. The points with the same color are mapped to the same item. The feature separateness loss enables separating the items, recording the diverse prototypes of normal data. Best viewed in color.

without the separateness loss loses the discriminability of memory items, and thus all features are mapped closely in the embedding space. The separateness loss allows to separate individual items in the memory, suggesting that it enhances the discriminative power of query features and memory items significantly. We can also see that our model gives compact feature representations.

Reconstruction with motion cues. Following [8], we use multiple frames for the reconstruction task. Specifically, we input sixteen successive video frames to reconstruct the ninth one. This achieves AUC of 91.0% for UCSD Ped2, providing the AUC gain of 0.8% but requiring more network parameters ($\sim$4MB).

5. Conclusion

We have introduced an unsupervised learning approach to anomaly detection in video sequences that exploits multiple prototypes to consider the various patterns of normal data. To this end, we have suggested to use a memory module to record the prototypical patterns to the items in the memory. We have shown that training the memory using feature compactness and separateness losses separates the items, enabling the sparse access of the memory. We have also presented a new memory update scheme when both normal and abnormal samples exist, which boosts the performance of anomaly detection significantly. Extensive experimental evaluations on standard benchmarks demonstrate the our model outperforms the state of the art.

Acknowledgments. This research was partly supported by Samsung Electronics Company, Ltd., Device Solutions under Grant, Deep Learning based Anomaly Detection, 2018–2020, and R&D program for Advanced Integrated-intelligence for Identification (AIID) through the National Research Foundation of KOREA(NRF) funded by Ministry of Science and ICT (NRF-2018M3E3A1057289).

References

[1] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *NIPS*, 2007. 1, 2

[2] Qi Cai, Yingwei Pan, Ting Yao, Chenggang Yan, and Tao Mei. Memory matching networks for one-shot image recognition. In *CVPR*, 2018. 3

[3] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*, 2019. 2

[4] Kai-Wen Cheng, Yie-Tarng Chen, and Wen-Hsien Fang. Video anomaly detection and localization using hierarchical feature representation and gaussian process regression. In *CVPR*, 2015. 2

[5] Yong Shean Chong and Yong Haur Tay. Abnormal event detection in videos using spatiotemporal autoencoder. In *ISNN*, 2017. 2

[6] Yang Cong, Junsong Yuan, and Ji Liu. Sparse reconstruction cost for abnormal event detection. In *CVPR*, 2011. 2

[7] Chenyou Fan, Xiaofan Zhang, Shu Zhang, Wensheng Wang, Chi Zhang, and Heng Huang. Heterogeneous memory enhanced multimodal attention model for video question answering. In *CVPR*, 2019. 3

[8] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *ICCV*, 2019. 1, 3, 6, 7, 8

[9] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K Roy-Chowdhury, and Larry S Davis. Learning temporal regularity in video sequences. In *CVPR*, 2016. 2, 7

[10] Ryota Hinami, Tao Mei, and Shin'ichi Satoh. Joint detection and recounting of abnormal events by learning deep generic knowledge. In *ICCV*, 2017. 7

[11] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997. 3

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4

[13] Łukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. Learning to remember rare events. In *ICLR*, 2017. 3

[14] Vagia Kaltsa, Alexia Briassouli, Ioannis Kompatsiaris, Leontios J Hadjileontiadis, and Michael Gerasimos Strintzis. Swarm intelligence for detecting interesting events in crowded environments. *IEEE TIP*, 2015. 2

[15] Jaechul Kim and Kristen Grauman. Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates. In *CVPR*, 2009. 2, 7

[16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[17] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1, 2

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012. 4

[19] Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. Ask me anything: Dynamic memory networks for natural language processing. In *ICML*, 2016. 3

[20] Sangho Lee, Jinyoung Sung, Youngjae Yu, and Gunhee Kim. A memory network approach for story-based temporal summarization of 360 videos. In *CVPR*, 2018. 3

[21] Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. Anomaly detection and localization in crowded scenes. *IEEE TPAMI*, 2013. 2, 6, 7, 8

[22] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection–a new baseline. In *CVPR*, 2018. 1, 2, 3, 4, 6, 7

[23] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6

[24] Cewu Lu, Jianping Shi, and Jiaya Jia. Abnormal event detection at 150 FPS in MATLAB. In *ICCV*, 2013. 1, 2, 6, 7

[25] Weixin Luo, Wen Liu, and Shenghua Gao. Remembering history with convolutional lstm for anomaly detection. In *ICME*, 2017. 2

[26] Weixin Luo, Wen Liu, and Shenghua Gao. A revisit of sparse coding based anomaly detection in stacked RNN framework. In *ICCV*, 2017. 2, 6, 7

[27] Junshui Ma and Simon Perkins. Time-series novelty detection using one-class support vector machines. In *IJCNN*, 2003. 2

[28] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos. Anomaly detection in crowded scenes. In *CVPR*, 2010. 2, 7

[29] Jefferson Ryan Medel and Andreas Savakis. Anomaly detection in video using predictive convolutional long short-term memory networks. *arXiv preprint arXiv:1612.00390*, 2016. 2

[30] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-value memory networks for directly reading documents. In *EMNLP*, 2016. 3

[31] Trong-Nguyen Nguyen and Jean Meunier. Anomaly detection in video sequence with appearance-motion correspondence. In *ICCV*, 2019. 7

[32] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. 2017. 6

[33] Mahdyar Ravanbakhsh, Moin Nabi, Enver Sangineto, Lucio Marcenaro, Carlo Regazzoni, and Nicu Sebe. Abnormal event detection in videos using generative adversarial nets. In *ICIP*, 2017. 2, 7

[34] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, 2015. 4

[35] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *ICML*, 2018. 2

[36] Mohammad Sabokrou, Mahmood Fathy, Mojtaba Hoseini, and Reinhard Klette. Real-time anomaly detection and localization in crowded scenes. In *CVPRW*, 2015. 2

[37] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, and Reinhard Klette. Deep-Cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes. *IEEE TIP*, 2017. 3

[38] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *ICML*, 2016. 3

[39] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 2001. 2

[40] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. In *NIPS*, 2015. 3

[41] Radu Tudor Ionescu, Sorina Smeureanu, Bogdan Alexe, and Marius Popescu. Unmasking the abnormal events in video. In *ICCV*, 2017. 7

[42] Laurens Van Der Maaten. Accelerating t-SNE using tree-based algorithms. *JMLR*, 2014. 8

[43] Namrata Vaswani, Amit K Roy-Chowdhury, and Rama Chellappa. "Shape Activity": A continuous-state hmm for moving/deforming shapes with application to abnormal activity detection. *IEEE TIP*, 2005. 2

[44] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 5

[45] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. In *ICLR*, 2015. 3

[46] Dan Xu, Yan Yan, Elisa Ricci, and Nicu Sebe. Detecting anomalous events in videos by learning deep representations of appearance and motion. *CVIU*, 2017. 7

[47] Jianwei Yang, Devi Parikh, and Dhruv Batra. Joint unsupervised learning of deep representations and image clusters. In *CVPR*, 2016. 5

[48] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. Deep structured energy based models for anomaly detection. *arXiv preprint arXiv:1605.07717*, 2016. 2

[49] Bin Zhao, Li Fei-Fei, and Eric P Xing. Online detection of unusual events in videos via dynamic sparse coding. In *CVPR*, 2011. 2

[50] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *ACM MM*, 2017. 2

[51] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. DM-GAN: Dynamic memory generative adversarial networks for text-to-image synthesis. In *CVPR*, 2019. 3