Learning Temporal Regularity in Video Sequences

Mahmudul Hasan Jonghyun Choi† Jan Neumann† Amit K. Roy-Chowdhury Larry S. Davis‡

UC Riverside Comcast Labs, DC† University of Maryland, College Park‡

{mhasa004@,amitrc@ece.}ucr.edu {jonghyun_choi,jan_neumann}@cable.comcast.com† lsd@umiacs.umd.edu‡

Abstract

Perceiving meaningful activities in a long video sequence is a challenging problem due to ambiguous definition of 'meaningfulness' as well as clutters in the scene. We approach this problem by learning a generative model for regular motion patterns (termed as regularity) using multiple sources with very limited supervision. Specifically, we propose two methods that are built upon the autoencoders for their ability to work with little to no supervision. We first leverage upon the conventional handcrafted spatio-temporal local features and learn a fully connected autoencoder on them. Second, we build a fully convolutional feed-forward autoencoder to learn both the local features and the classifiers as an end-to-end learning framework. The advantage of our approach over other methods is that it can capture the regularities from multiple visually distinctive sources in a single model. We evaluate our methods in both qualitative and quantitative ways - showing the learned regularity of videos in various aspects and demonstrating competitive performance on anomaly detection datasets as an application.

1. Introduction

The availability of large numbers of uncontrolled videos gives rise to the problem of watching long hours of meaningless scenes [1]. Automatic segmentation of 'meaningful' moments in such videos without supervision or with very limited supervision is a fundamental problem for various computer vision applications such as video annotation [2], summarization [3, 4], indexing or temporal segmentation [5], anomaly detection [6], and activity recognition [7]. We address this problem by modeling temporal regularity of videos with limited supervision, rather than modeling the sparse irregular or meaningful moments in a supervised manner.

Learning temporal visual characteristics of meaningful or salient moments is very challenging as the definition of such moments is ill-defined and is visually unbounded. On the other hand, learning temporal visual characteristics of ordinary moments is relatively easier as they often exhibit

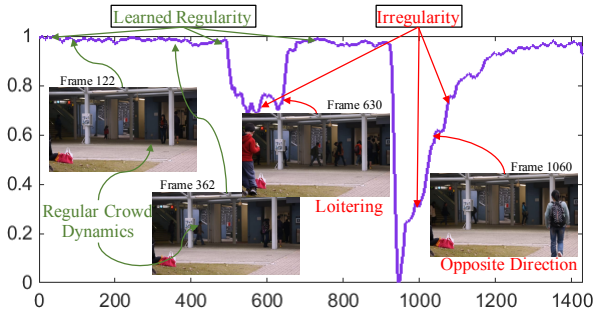


Figure 1. Learned regularity of a video sequence. Y-axis refers to regularity score and X-axis refers to frame number. When there are irregular motions, the regularity score drops significantly (from CUHK-Avenue dataset [8]).

temporally regular dynamics such as periodic crowd motions. We focus on learning the characteristics of regular temporal patterns with a very limited form of labeling - we assume that all events in the training videos are part of the regular patterns. Especially, we use multiple video sources, e.g., different datasets, to learn the regular temporal appearance changing pattern of videos in a single model that can then be used for multiple videos.

Given the training data of regular videos only, learning the temporal dynamics of regular scenes is an unsupervised learning problem. A state-of-the-art approach for such unsupervised modeling involves a combination of sparse coding and bag-of-words [8–10]. However, bag-of-words does not preserve spatio-temporal structure of the words and requires prior information about the number of words. Additionally, optimization involved in sparse coding for both training and testing is computationally expensive, especially with large data such as videos.

We present an approach based on autoencoders. Its objective function is computationally more efficient than sparse coding and it preserves spatio-temporal information while encoding dynamics. The learned autoencoder reconstructs regular motion with low error but incurs higher reconstruction error for irregular motions. Reconstruction error has been widely used for abnormal event detection [6], since it is a function of frame visual statistics and abnormalities manifest themselves as deviations from normal visual patterns. Figure 1 shows an example of learned regular-

ity, which is computed from the reconstruction error by a learned model (Eq.3 and Eq.4).

We propose to learn an autoencoder for temporal regularity based on two types of features as follows. First, we use state-of-the-art handcrafted motion features and learn a neural network based deep autoencoder consisting of seven fully connected layers. The state-of-the-art motion features, however, may be suboptimal for learning temporal regularity as they are not designed or optimized for this problem. Subsequently, we directly learn both the motion features and the discriminative regular patterns using a fully convolutional neural network based autoencoder.

We train our models using multiple datasets including CUHK Avenue [8], Subway (Enter and Exit) [11], and UCSD Pedestrian datasets (Ped1 and Ped2) [12], without compensating the dataset bias [13]. Therefore, the learned model is generalizable across the datasets. We show that our methods discover temporally regular appearance-changing patterns of videos with various applications - synthesizing the most regular frame from a video, delineating objects involved in irregular motions, and predicting the past and the future motions from a single frame. Our model also performs comparably to the state-of-the-art methods on anomaly detection task evaluated on multiple datasets including recently released public ones.

Our **main contributions** are summarized as follows:

- Showing that an autoencoder effectively learns the regular dynamics in long-duration videos and can be applied to identify irregularity in the videos.
- Learning the low level motion features for our proposed method using a fully convolutional autoencoder.
- Applying the model to various applications including learning temporal regularity, detecting objects associated with irregular motions, past and future frame prediction, and abnormal event detection.

2. Related Work

Learning Motion Patterns Without Supervision. Learning motion patterns without supervision has received much attention in recent years [14–16]. Goroshin *et al.* [17] trained a regularized high capacity (*i.e.*, deep) neural network based autoencoder using a temporal coherency prior on adjacent frames. Ramanathan *et al.* [18] trained a network to learn the motion signature of the same temporal coherency prior and used it for event retrieval.

To analyze temporal information, recurrent neural networks (RNN) have been widely used for analyzing speech and audio data [19]. For video analysis, Donahue *et al.* take advantage of long short term memory (LSTM) based RNN for visual recognition with the large scale labeled data [20]. Du *et al.* built an RNN in a hierarchical way to recognize actions [21]. The supervised action recognition setup requires human supervision to train the models. Ranzato *et al.* used the RNN for motion prediction [22], while we model the temporal regularity in a video sequence.

Anomaly Detection. One of the applications of our model is abnormal or anomalous event detection. The survey paper [6] contains a comprehensive review of this topic. Most video based anomaly detection approaches involve a local feature extraction step followed by learning a model on training video. Any event that is an outlier with respect to the learned model is regarded as the anomaly. These models include mixtures of probabilistic principal components based on optical flow [23], sparse dictionary [8, 9], Gaussian regression based probabilistic framework [24], spatio-temporal context [25, 26], sparse autoencoder [27], codebook based spatio-temporal volumes analysis [28], and shape [29]. Xu *et al.* [30] propose a deep model for anomalous event detection that uses a stacked autoencoder for feature learning and a linear classifier for event classification. In contrast, our method is an end-to-end trainable generative model that is generalizable across multiple datasets.

Convolutional Neural Network (CNN). Since Krizhevsky *et al.*'s work on image classification [31], CNN has been widely applied to various computer vision tasks such as feature extraction [32], image classification [33], object detection [34, 35], face verification [36], semantic embedding [37, 38], video analysis [16, 22], and *etc*. Particularly in video, Karpathy *et al.* and Ng *et al.* recently proposed a supervised CNN to classify actions in videos [7, 39]. Xu *et al.* train a CNN to detect events in videos [40]. Wang *et al.* learn a CNN to pool the trajectory information for recognizing actions [41]. These methods, however, require human supervision as they are supervised classification tasks.

Convolutional Autoencoder. For an end-to-end learning system for regularity in videos, we employ the convolutional autoencoder. Zhao *et al.* proposed a unified loss function to train a convolutional autoencoder for classification purposes [42]. Noh *et al.* [43] use convolutional autoencoders for semantic segmentation.

3. Approach

We use an autoencoder to learn regularity in video sequences. The intuition is that the learned autoencoder will reconstruct the motion signatures present in regular videos with low error but will not accurately reconstruct motions in irregular videos. In other words, the autoencoder can model the complex distribution of the regular dynamics of appearance changes.

As an input to the autoencoder, initially, we use state-of-the-art handcrafted motion features that consist of HOG and HOF with improved trajectory features [44]. Then we learn the regular motion signatures by a (fully-connected) neural network based autoencoder. However, even the state-of-the-art motion features may not be optimal for learning regularity as they are not specifically designed for this purpose. Thus, we use the video as an input and learn both local motion features and the autoencoder by an end-to-end learning

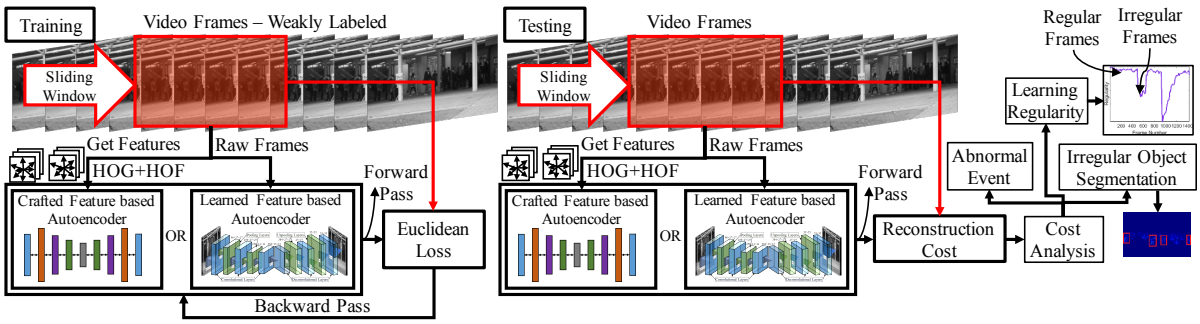


Figure 2. Overview of our approach. It utilizes either state-of-the-art motion features or learned features combined with autoencoder to reconstruct the scene. The reconstruction error is used to measure the regularity score that can be further analyzed for different applications.

model based on a fully convolutional neural network. We illustrate the overview of our the approach in Fig. 2.

3.1. Learning Motions on Handcrafted Features

We first extract handcrafted appearance and motion features from the video frames. We then use the extracted features as input to a fully connected neural network based autoencoder to learn the temporal regularity in the videos, similar to [45, 46].

Low-Level Motion Information in a Small Temporal Cuboid. We use Histograms of Oriented Gradients (HOG) [5, 47] and Histograms of Optical Flows (HOF) [48] in a temporal cuboid as a spatio-temporal appearance feature descriptor for their efficiency in encoding appearance and motion information respectively.

Trajectory Encoding. In order to extract HOG and HOF features along with the trajectory information, we use the improved trajectory (IT) features from Wang *et al.* [44]. It is based on the trajectory of local features, which has shown impressive performance in many human activity recognition benchmarks [44, 49].

As a first step of feature extraction, interest points are densely sampled at dense grid locations of every five pixels. Eight spatial scales are used for scale invariance. Interest points located in the homogeneous texture areas are excluded based on the eigenvalues of the auto-correlation matrix. Then, the interest points in the current frame are tracked to the next frame by median filtering a dense optical flow field [50]. This tracking is normally carried out up to a fixed number of frames (L) in order to avoid drifting. Finally, trajectories with sudden displacement are removed from the set [44].

Final Motion Feature. Local appearance and motion features around the trajectories are encoded with the HOG and HOF descriptors. We finally concatenate them to form a 204 dimensional feature as an input to the autoencoder.

3.1.1 Model Architecture

Next, we learn a model for regular motion patterns on the motion features in an unsupervised manner. We propose to use a deep autoencoder with an architecture similar to Hinton *et al.* [45] as shown in Figure 3.

Our autoencoder takes the 204 dimensional HOG+HOF feature as the input to an encoder and a decoder sequentially. The encoder has four hidden layers with 2,000, 1,000, 500, and 30 neurons respectively, whereas the decoder has three hidden layers with 500, 1,000 and 2,000 neurons respectively. The small-sized middle layers are for learning compact semantics as well as reducing noisy information.

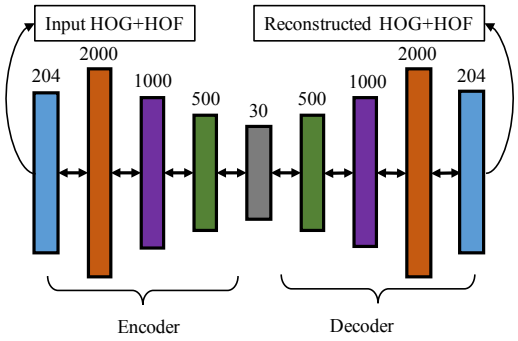


Figure 3. Structure of our autoencoder taking the HOG+HOF feature as input.

Since both the input and the reconstructed signals of the autoencoder are HOG+HOF histograms, their magnitude of them should be bounded in the range from 0 to 1. Thus, we use either sigmoid or hyperbolic tangent ($\tanh$) as the activation function instead of the rectified linear unit (ReLU). ReLU is not suitable for a network that has large receptive fields for each neuron as the sum of the inputs to a neuron can become very large.

In addition, we use the sparse weight initialization technique described in [51] for the large receptive field. In the initialization step, each neuron is connected to k randomly chosen units in the previous layer, whose weights are drawn from a unit Gaussian with zero bias. As a result, the total number of inputs to each neuron is a constant, which prevents the large input problem.

We define the objective function of the autoencoder by an Euclidean loss of input feature ($\mathbf{x}_i$) and the reconstructed feature ($f_W(\mathbf{x}_i)$) with an L_2 regularization term as shown in Eq.1. Intuitively, we want to learn a non-linear classifier

so that the overall reconstruction cost for the i^{th} training features $\mathbf{x}_i$ is minimized.

$$\hat{f}_W = \arg\min_W \frac{1}{2N} \sum_i \|\mathbf{x}_i - f_W(\mathbf{x}_i)\|_2^2 + \gamma \|W\|_2^2, \quad (1)$$

where N is the size of mini batch, γ is a hyper-parameter to balance the loss and the regularization and $f_W(\cdot)$ is a non-linear classifier such as a neural network associated with its weights W.

3.2. Learning Features and Motions

Even though we use the state-of-the-art motion feature descriptors, they may not be optimal for learning regular patterns in videos. To learn highly tuned low level features that best learn temporal regularity, we propose to learn a fully convolutional autoencoder that takes short video clips in a temporal sliding window as the input. We use fully convolutional network because it does not contain fully connected layers. Fully connected layers loses spatial information [52]. For our model, the spatial information needs to be preserved for reconstructing the input frames. We present the details of training data, network architecture, and the loss function of our fully convolutional autoencoder model, the training procedure, and parameters in the following subsections.

3.2.1 Model Architecture

Figure 4 illustrates the architecture of our fully convolutional autoencoder. The encoder consists of convolutional layers [31] and the decoder consists of deconvolutional layers that are the reverse of the encoder with padding removal at the boundary of images.

We use three convolutional layers and two pooling layers on the encoder side and three deconvolutional layers and two unpooling layers on the decoder side by considering the size of input cuboid and training data.

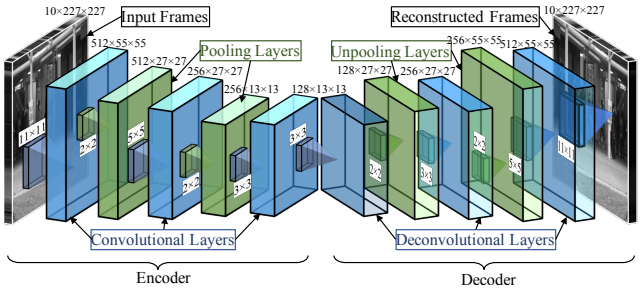


Figure 4. Structure of our fully convolutional autoencoder.

The first convolutional layer has 512 filters with a stride of 4. It produces 512 feature maps with a resolution of 55×55 pixels. Both of the pooling layers have kernel of size 2×2 pixels and perform max poling. The first pooling layer produces 512 feature maps of size 27×27 pixels. The second and third convolutional layers have 256 and 128 filters respectively. Finally, the encoder produces 128 feature maps of size 13×13 pixels. Then, the decoder reconstructs the input by deconvolving and unpooling the input in reverse order of size. The output of final deconvolutional layer is the reconstructed version of the input.

Input Data Layer. Most of convolutional neural networks are for classifying images and take an input of three channels (for R,G, and B color channel). Our input, however, is a video, which consists of an arbitrary number of channels. Recent works [7, 20] extract features for each video frame, then use several feature fusion schemes to construct the input features to the network, similar to our first approach described in Sec. 3.1.

We, however, construct the input by a temporal cuboid using a sliding window technique without any feature transform. Specifically, we stack T frames together and use them as the input to the autoencoder, where T is the length of the sliding window. Our experiment shows that increasing T results in a more discriminative regularity score as it incorporates longer motions or temporal information as shown in Fig. 5.

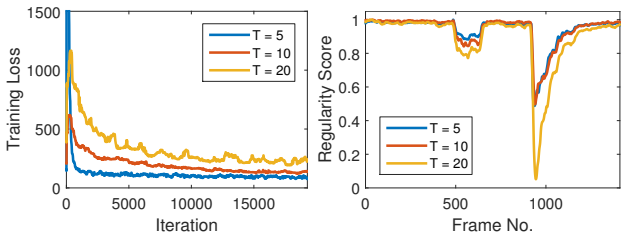


Figure 5. Effect of temporal length (T) of input video cuboid. (Left) X-axis is the increasing number of iterations, Y-axis is the training loss, and three plots correspond to three different values of T. (Right) X-axis is the increasing number of video frames and Y-axis is the regularity score. As T increases, the training loss takes more iterations to converge as it is more likely that the inputs with more channels have more irregularity to hamper learning regularity. On the other hand, once the model is learned, the regularity score is more distinguishable for higher values of T between regular and irregular regions (note that there are irregular motions in the frame from 480 to 680, and 950 to 1250).

Data Augmentation In the Temporal Dimension. As the number of parameters in the autoencoder is large, we need large amounts of training data. The size of a given training datasets, however, may not be large enough to train the network. Thus, we increase the size of the input data by generating more input cuboids with possible transformations to the given data. To this end, we concatenate frames with various skipping strides to construct T-sized input cuboid. We sample three types of cuboids from the video sequences - stride-1, stride-2, and stride-3. In stride-1 cuboids, all T frames are consecutive, whereas in stride-2 and stride-3 cuboids, we skip one and two frames, respectively. The stride used for sampling cuboids is two frames.

We also performed experiments with precomputed optical flows. Given the gradients and the magnitudes of op-

tical flows between two frames, we compute a single gray scale frame by linearly combining the gradients and magnitudes. It increases the temporal dimension of the input cuboid from T to $2T$. Channels $1, \ldots, T$ contain gray scale video frames, whereas channels $T+1, \ldots, 2T$ contain gray scale flow information. This information fusion scheme was used in [39]. Our experiments reveal that the overall improvement is insignificant.

Convolutional and Deconvolutional Layer. A convolutional layers connects multiple input activations within the fixed receptive field of a filter to a single activation output. It abstracts the information of a filter cuboid into a scalar value.

On the other hand, deconvolution layers densify the sparse signal by convolution-like operations with multiple learned filters; thus they associate a single input activation with patch outputs by an inverse operation of convolution. Thus, the output of deconvolution is larger than the original input due to the superposition of the filters multiplied by the input activation at the boundaries. To keep the size of the output mapping identical to the preceding layer, we crop out the boundary of the output that is larger than the input.

The learned filters in the deconvolutional layers serve as bases to reconstruct the shape of an input motion cuboid. As we stack the convolutional layers at the beginning of the network, we stack the deconvolutional layers to capture different levels of shape details for building an autoencoder. The filters in early layers of convolutional and the later layers of deconvolutional layers tend to capture specific motion signature of input video frames while high level motion abstractions are encoded in the filters in later layers.

Pooling and Unpooling Layer. Combined with a convolutional layer, the pooling layer further abstracts the activations for various purposes such as translation invariance after the convolutional layer. Types of pooling operations include 'max' and 'average.' We use 'max' for translation invariance. It is known to help classifying images by making convolutional filter output to be spatially invariant [31].

By using 'max' pooling, however, spatial information is lost, which is important for location specific regularity. Thus, we employ the unpooling layers in the deconvolution network, which perform the reverse operation of pooling and reconstruct the original size of activations [43, 53, 54].

We implement the unpooling layer in the same way as [53, 54] which records the locations of maximum activations selected during a pooling operation in switch variables and use them to place each activation back to the originally pooled location.

Optimization Objective. Similar to Eq.1, we use Euclidean loss with L_2 regularization as an objective function on the temporal cuboids:

$$\hat{f}_W = \arg\min_W \frac{1}{2N} \sum_i \|\mathbf{X}_i - f_W(\mathbf{X}_i)\|_2^2 + \gamma \|W\|_2^2, \quad (2)$$

where $\mathbf{X}_i$ is i^{th} cuboid, N is the size of mini batch, γ is a hyper-parameter to balance the loss and the regularization and $f_W(\cdot)$ is a non-linear classifier - a fully convolutional-deconvolutional neural network with its weights W.

3.3. Optimization and Initialization

To optimize the autoencoders of Eq.1 and Eq.2, we use a stochastic gradient descent with an adaptive subgradient method called AdaGrad [55]. AdaGrad computes a dimension-wise learning rate that adapts the rate of gradients by a function of all previous updates on each dimension. It is widely used for its strong theoretical guarantee of convergence and empirical successes. We also tested Adam [56] and RMSProp [57] but empirically chose to use AdaGrad.

We train the network using multiple datasets. Fig. 6 shows the learning curves trained with different datasets as a function of iterations. We start with a learning rate of 0.001 and reduce it when the training loss stops decreasing. For the autoencoder on the improved trajectory features, we use mini-batches of size $1,024$ and weight decay of 0.0005. For the fully convolutional autoencoder, we use mini batch size of 32 and start training the network with learning rate 0.01.

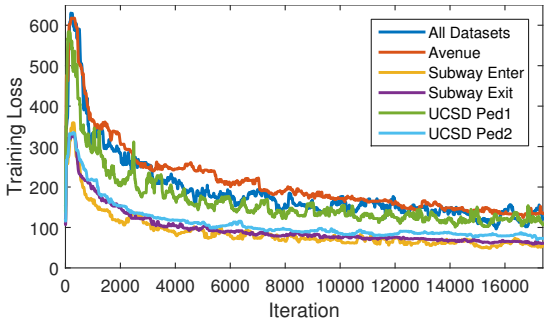


Figure 6. Loss value of models trained on each dataset and all datasets as a function of optimization iterations.

We initialized the weights using the Xavier algorithm [58] since Gaussian initialization for various network structure has the following problems. First, if the weights in a network are initialized with too small values, then the signal shrinks as it passes through each layer until it becomes too small in value to be useful. Second, if the weights in a network are initialized with too large values, then the signal grows as it passes through each layer until it becomes too large to be useful. The Xavier initialization automatically determines the scale of initialization based on the number of input and output neurons, keeping the signal in a reasonable range of values through many layers. We empirically observed that the Xavier initialization is noticeably more stable than Gaussian.

3.4. Regularity Score

Once we trained the model, we compute the reconstruction error of a pixel's intensity value I at location (x, y) in

frame t of the video sequence as following:

$$e(x, y, t) = \|I(x, y, t) - f_W(I(x, y, t))\|_2, \quad (3)$$

where f_W is the learned model by the fully convolutional autoencoder. Given the reconstruction errors of the pixels of a frame t, we compute the reconstruction error of a frame by summing up all the pixel-wise errors: $e(t) = \sum_{(x,y)} e(x, y, t)$. We compute the regularity score $s(t)$ of a frame t as follows:

$$s(t) = 1 - \frac{e(t) - \min_t e(t)}{\max_t e(t)}. \quad (4)$$

For the autoencoder on the improved trajectory feature, we can simply replace $I(x, y)$ with $p(x, y)$ where $p(\cdot)$ is an improved trajectory feature descriptor of a patch that covers the location of (x, y).

4. Experiments

We learn the model using multiple video datasets, totaling 1 hour 50 minutes, and evaluate our method both qualitatively and quantitatively. We modify[1] and use Caffe [59] for all of our experiments on NVIDIA Tesla K80 GPUs.

For qualitative analysis, we generate the most regular image from a video and visualize the pixel-level irregularity. In addition, we show that the learned model based on the convolutional autoencoder can be used to forecast future frames and estimate past frames.

For quantitative analysis, we temporally segment the anomalous events in video and compare performance against the state of the arts. Note that our model is not fine-tuned to one dataset. It is general enough to capture regularities across multiple datasets.

4.1. Datasets

We use three datasets to train and demonstrate our models. They are curated for anomaly or abnormal event detection and are referred to as Avenue [8], UCSD pedestrian [12], and Subway [11] datasets. We describe the details of datasets in the supplementary material.

4.2. Learning a General Model Across Datasets

We compare the generalizability of the trained model using various training setups in terms of regularity scores obtained by each model in Fig. 7. Blue (conventional) represents the score obtained by a model trained on the *specific target* dataset. Red (generalized) represents the score obtained by a model trained on *all* datasets, which is the model we use for all other experiments. Yellow (transfer) represents the score obtained by a model trained on *all datasets except that specific target* dataset. Red shaded regions represent ground truth temporal segments of the abnormal events.

[1] https://github.com/mhasa004/caffe

By comparing 'conventional' and 'generalized', we observe that the model is not degraded by other datasets. At the same time, by comparing 'transfer' and either 'generalized' or 'conventional', we observe that the model is not too much overfitted to the given dataset as it can generalize to *unseen* videos in spite of potential dataset biases. Consequently, we believe that the proposed network structure is well balanced between overfitting and underfitting.

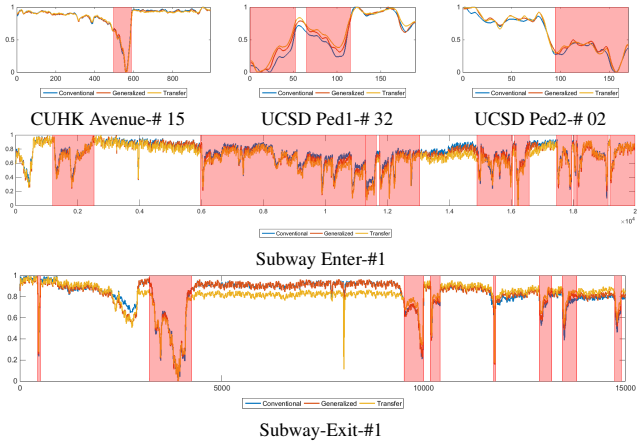


Figure 7. Generalizability of Models by Obtained Regularity Scores. 'Conventional' is by a model trained on the *specific target* dataset. 'Generalized' is by a model trained on *all* datasets. 'Transfer' is by a model trained on *all datasets except that specific target* datasets. Best viewed in zoom.

4.3. Visualizing Temporal Regularity

The learned model measures the intensity of regularity up to pixel precision. We synthesize the most regular frame from the test video by collecting the pixels that have the highest regularity score by our convolutional autoencoder (conv-autoencoder) and autoencoder on improved trajectories (IT-autoencoder).

The first column of Fig. 8 shows sample images that contain irregular motions. The second column shows the synthesized regular frame. Each pixel of the synthesized image corresponds to the pixel for which reconstruction cost is minimum along the temporal dimension. The right most column shows the corresponding regularity score. Blue represents high score, red represents low.

Fig. 9 shows the results using IT-autoencoder. The left column shows the sample irregular frame of a video sequences, and the right column is the pixel-wise regularity score for that video sequence. It captures irregularity to patch precision; thus the spatial location is not pixel-precise as obtained by conv-autoencoder.

4.4. Predicting the Regular Past and the Future

Our convolutional autoencoder captures temporal appearance changes since it takes a short video clip as input. Using a clip that is blank except for the center frame, we can

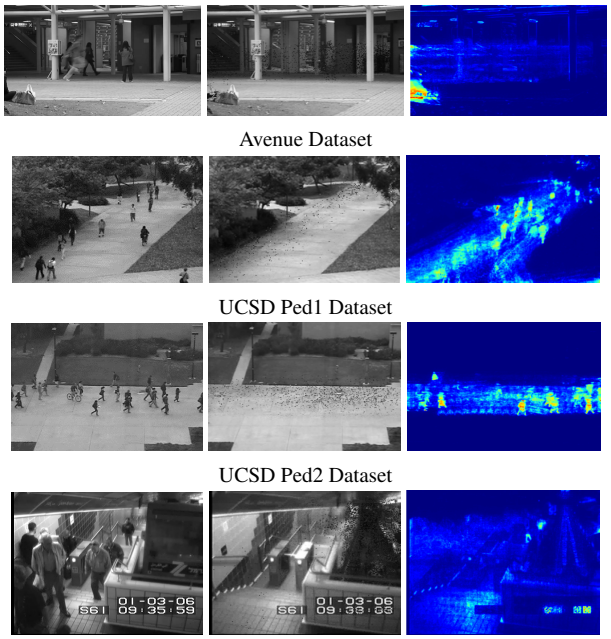


Figure 8. (Left) A sample irregular frame. (Middle) Synthesized regular frame. (Right) Regularity Scores of the frame. Blue represents regular pixel. Red represents irregular pixel.

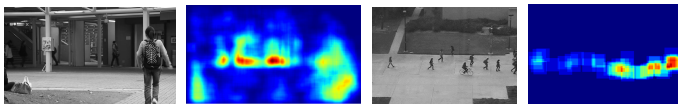


Figure 9. Learned regularity by improved trajectory features. (Left) Frames with irregular motion. (Right) Learned regularity on the entire video sequence. Blue represents regular region. Red represents irregular region.

predict both near past and future frames of a regular video clip for the given center frame.

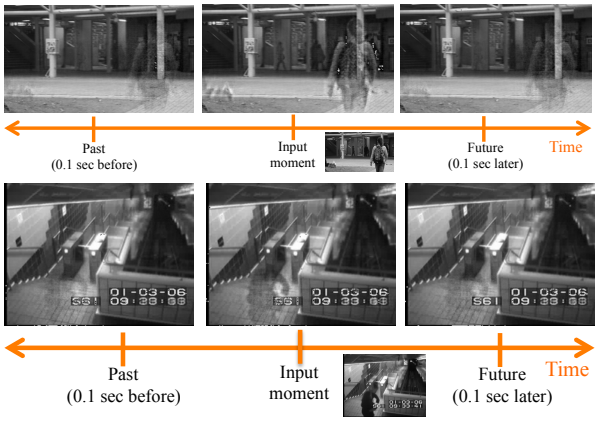


Figure 10. Synthesizing a video of regular motion from a single seed image (at the center). Upper: CUHK-Avenue. Bottom: Subway-Exit.

Given a single irregular image, we construct a temporal cube as the input to our network by padding other frames with all zero values. Then we pass the cube through our learned model to extrapolate the past and the future of that center frame. Fig. 10 shows some examples of generated videos. The objects in an irregular motion start appearing from the past and gradually disappearing in the future.

Since the network is trained with regular videos, it learns the regular motion patterns. With this experiment, we showed that the network can predict the regular motion of the objects in a given frame up to a few number of past and future frames.

4.5. Anomalous Event Detection

As our model learns the temporal regularity, it can be used for detecting anomalous events in a weakly supervised manner. Fig. 11 shows the regularity scores as a function of frame number. Table 1 compares the anomaly detection accuracies of our autoencoders against state-of-the-art methods. To the best of our knowledge, there are no correct detection or false alarm results reported for UCSD Ped1 and Ped2 datasets in the literature. We provide the EER and AUC measures from [60] for reference. Additionally, the state-of-the-art results for the avenue dataset from [8] are not directly comparable as it is reported on the old version of the Avenue dataset that is smaller than the current version.

We find the local minimas in the time series of regularity scores to detect abnormal events. However, these local minima are very noisy and not all of them are meaningful local minima. We use the persistence1D [61] algorithm to identify meaningful local minima and span the region with a fixed temporal window (50 frames) and group nearby expanded local minimal regions when they overlap to obtain the final abnormal temporal regions. Specifically, if two local minima are within fifty frames of one another, they are considered to be a part of same abnormal event. We consider a detected abnormal region as a correct detection if it has at least fifty percent overlap with the ground truth.

Our model outperforms or performs comparably to the state-of-the-art abnormal event detection methods but with a few more false alarms. It is because our method identifies any deviations from regularity, many of which have not been annotated as abnormal events in those datasets while competing approaches focused on the identification of abnormal events. For example, in the top figure of Fig. 11, the 'running' event is detected as an irregularity due to its unusual motion pattern by our model, but in the ground truth it is a normal event and considered as a false alarm during evaluation.

4.6. Filter Responses

We visualize some of the learned filter responses of our model on Avenue datasets in Fig. 12. The first row visualizes one channel of the input data and two filter responses of the conv1 layer. These two filters show completely op-

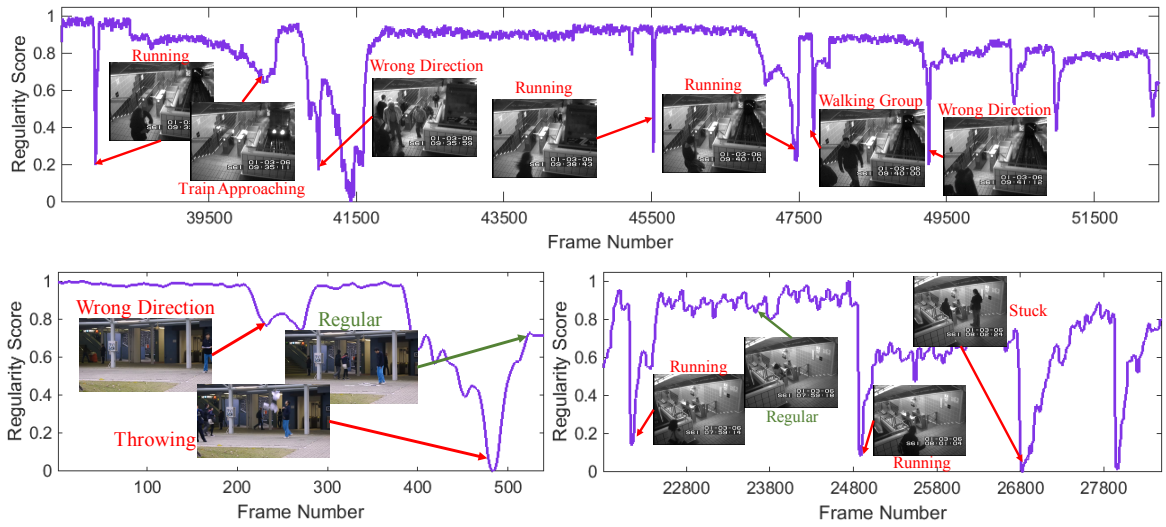


Figure 11. Regularity score (Eq.3) of each frame of three video sequences. (Top) Subway Exit, (Bottom-Left) Avenue, and (Bottom-Right) Subway Enter datasets. Green and red colors represent regular and irregular frames respectively.

Dataset			Regularity	Anomaly Detection					
					Correct Detection / False Alarm			AUC/EER	
Name	# Frames	# Regular Frames	Conv-AE Correct Detect / FA	# Anomalous Event	Conv-AE	IT-AE	State of the art	Conv-AE	State of the art
CUHK Avenue	15,324	11,504	11,419/355	47	45/4	43/8	12/1 (Old Dataset) [8]	70.2/25.1	N/A
UCSD Ped1	7,200	3,195	3,135/310	40	38/6	36/11	N/A	81.0/27.9	92.7/16.0 [60]
UCSD Ped2	2,010	374	374/50	12	12/1	12/3	N/A	90.0/21.7	90.8/16.0 [30]
Subway Entrance	121,749	119,349	112,188/4,154	66	61/15	55/17	57/4 [8]	94.3/26.0	N/A
Subway Exit	64,901	64,181	62,871/1,125	19	17/5	17/9	19/2 [8]	80.7/9.9	N/A

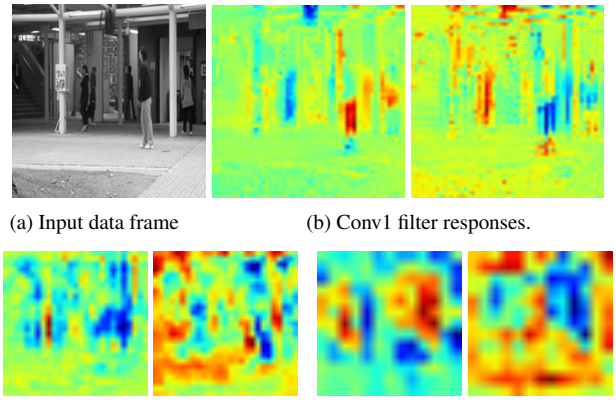
Table 1. Comparing abnormal event detection performance. AE refers to auto-encoder. IT refers to improved trajectory.

posite responses to the irregular object - the bag in the top of the frame. The first filter provides very low response (blue color) to it, whereas the second filter provides very high response (red color). The first filter can be described as the filter that detects regularity, whereas the second filter detects irregularity. All other filters show similar characteristics. The second row of Fig. 12 shows the responses of the filters from conv2 and conv3 layers respectively.

Additional results can be found in the supplementary material. Data, codes, and videos are available online[2].

5. Conclusion

We present a method to learn regular patterns using autoencoders with limited supervision. We first take advantage of the conventional spatio-temporal local features and learn a fully connected autoencoder. Then, we build a fully convolutional autoencoder to learn both the local features and the classifiers in a single learning framework. Our model is generalizable across multiple datasets even with potential dataset biases. We analyze our learned models in a number of ways such as visualizing the regularity in frames and pixels and predicting a regular video of past and future given only a single image. For quantitative analysis, we show that our method performs competitively to the state-of-the-art anomaly detection methods.



(a) Input data frame (b) Conv1 filter responses.

(c) Conv2 filter responses. (d) Conv3 filter responses.

Figure 12. Filter responses of the convolutional autoencoder trained on the Avenue dataset. Early layers (conv1) captures fine grained regular motion pattern whereas the deeper layers (conv3) captures higher level information.

Acknowledgement

M. Hasan and A. K. Roy-Chowdhury are partially supported by NSF grant IIS-1316934. L.S. Davis were partially supported by MURI from the Office of Naval Research under the Grant N00014- 10-1-0934.

[2] http://www.ee.ucr.edu/~mhasan/regularity.html

References

[1] M. Sun, A. Farhadi, and S. Seitz, "Ranking Domain-specific Highlights by Analyzing Edited Videos," in *ECCV*, 2014. 1

[2] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, vol. 101, no. 1, pp. 184–204, 2013. 1

[3] Y. Song, J. Vallmitjana, A. Stent, and A. Jaimes, "TVSum: Summarizing Web Videos Using Titles," in *CVPR*, 2015. 1

[4] W.-S. Chu, Y. Song, and A. Jaimes, "Video Co-summarization: Video Summarization by Visual Co-occurrence," in *CVPR*, 2015. 1

[5] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *CVPR*, 2008. 1, 3

[6] O. Popoola and K. Wang, "Video-based abnormal human behavior recognition; a review," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, pp. 865–878, 2012. 1, 2

[7] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale Video Classification with Convolutional Neural Networks," in *CVPR*, 2014. 1, 2, 4

[8] C. Lu, J. Shi, and J. Jia, "Abnormal event detection at 150 fps in matlab," in *ICCV*, 2013, pp. 2720–2727. 1, 2, 6, 7, 8

[9] B. Zhao, L. Fei-Fei, and E. P. Xing, "Online detection of unusual events in videos via dynamic sparse coding," in *CVPR*, 2011, pp. 3313–3320. 1, 2

[10] Y. Cong, J. Yuan, and J. Liu, "Sparse Reconstruction Cost for Abnormal Event Detection," in *CVPR*, 2011. 1

[11] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 3, pp. 555–560, 2008. 2, 6

[12] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *CVPR*, 2010, pp. 1975–1981. 2, 6

[13] A. Torralba and A. A. Efros, "Unbiased Look at Dataset Bias," in *CVPR*, 2011. 2

[14] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *CVPR*, 2011. 2

[15] H. Jhuang, T. Serre, L. Wolf, and T. Poggio, "A biologically inspired system for action recognition," in *ICCV*, 2007. 2

[16] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatiotemporal features," in *ECCV*, 2010. 2

[17] R. Goroshin, J. Bruna, J. Tompson, D. Eigen, and Y. Lecun, "Unsupervised Feature Learning From Temporal Data," in *ICLR Workshop*, 2015. 2

[18] V. Ramanathan, K. Tang, G. Mori, and L. Fei-Fei, "Learning Temporal Embeddings for Complex Video Analysis," *arXiv preprint arXiv:1505.00315*, 2015. 2

[19] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP*, 2013. 2

[20] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015. 2, 4

[21] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *CVPR*, 2015. 2

[22] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra, "Video (language) modeling: a baseline for generative models of natural videos," *Arxiv*, 2014. 2

[23] J. Kim and K. Grauman, "Observe locally, infer globally: a space-time mrf for detecting abnormal activities with incremental updates," in *CVPR*, 2009, pp. 2921–2928. 2

[24] K.-W. Cheng, Y.-T. Chen, and W.-H. Fang, "Video anomaly detection and localization using hierarchical feature representation and gaussian process regression," in *CVPR*, 2015, pp. 2909–2917. 2

[25] T. Xiao, C. Zhang, and H. Zha, "Learning to detect anomalies in surveillance video," *Signal Processing Letters, IEEE*, vol. 22, no. 9, pp. 1477–1481, 2015. 2

[26] Y. Zhu, N. M. Nayak, and A. K. Roy-Chowdhury, "Context-aware activity recognition and anomaly detection in video," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 7, no. 1, pp. 91–101, 2013. 2

[27] M. Sabokrou, M. Fathy, M. Hoseini, and R. Klette, "Real-time anomaly detection and localization in crowded scenes," in *CVPRW*, 2015. 2

[28] M. J. Roshtkhari and M. D. Levine, "Online dominant and anomalous behavior detection in videos," in *CVPR*, 2013. 2

[29] N. Vaswani, A. K. Roy-Chowdhury, and R. Chellappa, "" shape activity": a continuous-state hmm for moving/deforming shapes with application to abnormal activity detection," *Image Processing, IEEE Transactions on*, vol. 14, no. 10, pp. 1603–1616, 2005. 2

[30] D. Xu, E. Ricci, Y. Yan, J. Song, and N. Sebe, "Learning deep representations of appearance and motion for anomalous event detection," in *BMVC*, 2015. 2, 8

[31] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012. 2, 4, 5

[32] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *CVPR*, 2014. 2

[33] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2014. 2

[34] T. Dean, M. Ruzon, M. Segal, J. Shleps, S. Vijayanarasimhan, and J. Yagnik, "Fast, accurate detection of 100,000 object classes on a single machine," in *CVPR*, 2013. 2

[35] S. Ren, A. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *NIPS*, 2015. 2

[36] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *CVPR*, 2014. 2

[37] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov, "DeViSE: A Deep Visual-Semantic Embedding Model," in *NIPS*, 2013. 2

[38] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *CVPR*, 2014. 2

[39] J. Y. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond Short Snippets: Deep Networks for Video Classification," in *CVPR*, 2015. 2, 5

[40] Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative cnn video representation for event detection," in *CVPR*, 2015. 2

[41] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *CVPR*, 2015. 2

[42] J. Zhao, M. Mathieu, R. Goroshin, and Y. Lecun, "Stacked What-Where Auto-encoders," *arXiv preprint arXiv:1506.02351*, 2015. 2

[43] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation," in *ICCV*, 2015. 2, 5

[44] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013. 2, 3

[45] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006. 3

[46] M. Hasan and A. K. Roy-Chowdhury, "Continuous Learning of Human Activity Models Using Deep Nets," in *ECCV*, 2014. 3

[47] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005. 3

[48] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *ECCV*, 2006. 3

[49] U. Gaur, Y. Zhu, B. Song, and A. Roy-Chowdhury, "A "String of Feature Graphs" Model for Recognition of Complex Activities in Natural Videos," in *ICCV*, 2011. 3

[50] H. Wang, A. Klaser, C. Schmid, and L. Cheng-Lin, "Action Recognition by Dense Trajectories," in *IEEE CVPR*, 2011. 3

[51] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *ICML*, 2013. 3

[52] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015. 4

[53] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014. 5

[54] M. D. Zeiler, G. W. Taylor, and R. Fergus, "Adaptive deconvolutional networks for mid and high level feature learning," in *ICCV*, 2011. 5

[55] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *The Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011. 5

[56] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980*, 2015. 5

[57] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," in *COURSERA: Neural Networks for Machine Learning*, 2012. 5

[58] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *AISTAT*, 2010. 5

[59] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: An open source convolutional architecture for fast feature embedding," http://caffe.berkeleyvision.org/, 2014. 6

[60] V. Saligrama and Z. Chen, "Video Anomaly Detection Based on Local Statistical Aggregates," in *CVPR*, 2012. 7, 8

[61] Y. Kozlov and T. Weinkauf, "Persistence1D: Extracting and filtering minima and maxima of 1d functions," http://people.mpi-inf.mpg.de/~weinkauf/notes/persistence1d.html, accessed: 2015-11-01. 7