Video Anomaly Detection

Duo Jiang, Ademola Oladosu, Yunya Wang

1 Introduction

Video surveillance has had growing importance over the past few decades, as individuals, corporations and governments seek to protect themselves. As a result of this growth, large amounts of video footage have been generated, the quantity of which can be unmanageable for just a handful of individuals or organizations. Intelligent video surveillance addresses this problem by taking advantage of automated systems for anomaly detection. By taking as input a video and generating a metric for measuring the likelihood of a video or a section of a video containing an an anomaly, an interested party can prioritize which sections of a video to review. This area of research has been growing in recent years, particularly among computer vision researchers, who have experimented with various representations of video files.

Our approach in this project first implements some standard classification techniques where we assume there are no time-series based relationships between our video frames. After this, we utilize the time-series dependencies in our data set and use two methodologies to identify the anomalies in our data set. The first is to model our data as a Gaussian process; the latter methodology is latent space auto-regression, which develops a novelty score for each new input data point after obtaining a probabilistic model to represent data points that are known not to be anomalies. One basic assumption with Gaussian processes is that we know the probabilistic distribution of normal video frames. The latent space auto-regression approach makes no such assumption about the distribution of normal events and abnormalities. Since these distributions are learned, we generally expect it to perform better than the first model when identifying anomalies.

When asking the question of whether or not a given video frame is or isn't an anomaly, the kNN model proved to be the best basic approach explored. The AUC on the the test set was shown to be 0.717. When the data was modeled as a Gaussian process, we saw an improvement in AUC to 0.857. When we make no assumptions about the probabilistic distribution of our data set, we saw that the latent space auto-regression approach provides the highest AUC of 0.954. This suggests that the probability distribution assumed by the Gaussian process is valid but not perfect.

2 Related Work

With respect to reconstruction-based methods, prior work have tried to develop a parametric projection and reconstruction of normal data. For example, earlier sparse-coding algorithms represented normal behaviors as linear combinations of a small number of basis components. With this methodology, the underlying assumption is that that novelties would show a non-sparse representation in the subspace. Improvements to this approach have been made by using deep auto-encoders for the projection step. Another alternative to this has been to impose a sparsity regularization over the learned representation and then using a recurrent neural network (RNN) to ensure smoothness over time.

With respect to probabilistic methods, various strategies have been tried to approximate the density function of what is considered to be normal motion. A major challenge has been how to generate such estimates in highly dimensional feature spaces. One solution is to generate features using the histogram of optical flow technique, which we use in our Gaussian process algorithm. More recently, deep representations such as Gaussian classifiers and Gaussian mixtures have been used. One of the most recent developments is the used of Generative Adversarial Networks (GANs). However, since GANs use an implicit density function, they cannot generate a likelihood value for a new data point. Consequently, GAN-based models evaluate novelty differently. Two options that have been tried include using a guided latent space search and querying the discriminator to obtain a normality score.

3 Problem Definition and Algorithm

3.1 Task

We wish to pass a series of frames from a given video into a model and have the model determine whether or not the events taking place in that frame are normal or contain an anomaly. The models we construct will take as inputs a 158 X 238 image file, process the data appropriately, and will provide a probability distribution of the frame belonging to a given class. After a threshold is selected in each model, a class label is assigned, one which corresponds to normal events and the other which indicates that there are abnormal events in the image file.

3.2 Algorithm 0: Baseline Models

Although we evaluated several traditional non-time series based classification techniques, we present only the logistic regression and k-nearest neighbors (kNN) solutions to our task. As these models do not represent the emphasis of our project, we briefly outline our rationale and expectations here. We attempted logistic regression as it is easily computed and interpreted. In logistic regression, we simply try to calculate the probability of class membership given a set of observed features.

We attempted kNN for similar reasons but also because we hypothesized that normal frames would contain very similar information in its pixel array while abnormal frames would contain different information (but similar) in its image file. In each of these models, the frames that contained anomalies were labeled one class while the rest were assigned to the other class. The similarities we expected would give rise to a smaller Euclidean distance between items of the same class. However, unless we expect abnormalities we observe in the future to mimic abnormalities on which we have trained, it is unlikely that these results can be generalized to other data sets. However, the results can perform a target baseline to which we can compare other model results.

3.3 Algorithm 1: Anomaly Detection in Video Surveillance via Gaussian Process

Given that we have two states (normal and abnormal events), our task can be rewritten as a one-class classification (OCC) problem. Modeling our data as a Gaussian process can be used to solve OCC problems so this is the first time-series based algorithm we use to address our task. Here, we explain what Nannan(1) did on this paper. A summary of the feature construction, which is critical part of the model application, is provided in the Data section. Here, however, we focus on how to use the Gaussian process model to compute anomaly scores on the video frames that comprise our testing data.

After finding the centers of the clusters (centroids), we can refer to these as basic vectors. The number of basic vectors needs to be sufficiently large to capture information about the video while remaining small enough to not become computationally expensive. In our experiment, there are about one hundred thousand histograms of optical flow for each of the 200 frames, depending on the size of cells in each frame. After using the online clustering algorithm defined in the paper, we can convert all histograms of optical flows to about 100 basic vectors and, based on our trial run and subsequent results, this appears to be a reasonable balance of size.

Once we have all the basic vectors, our algorithm feeds all the basic vectors with y=1 into a Gaussian process with mean 0 and uses the RBF kernel. For each histogram of optical flow in the testing data x*, there are three values that can be used as anomaly scores.

- Predictive mean of y*
- inverse of variance y*
- $p(y* = 1|X, y, x*)$.

The reason these three metrics can be used as anomaly scores is that they all decrease if x* is further away from training data. Also, it is reasonable to assume that the further away x* is from the training data, the more likely it is an anomaly. Therefore, there are two consequences of x* being away from training data. One is that the metric we are evaluating is lower; the other is that x* is more likely to be an anomaly in this instance. This is illustrated in Figure 1 and enables this model to be easily interpreted.

Another basic assumption behind this logic is smoothness. In this case, this means that if two Xs are close in

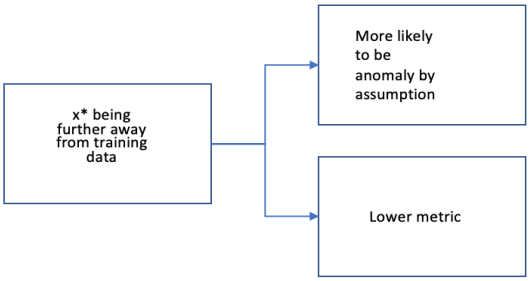

Figure 1: Logic behind why that works

that high-dimensional space, the two Ys are also close. This is a reasonable assumption that allows us to use RBF as our kernel.

Although optical flow helps capture the motion of pixels, there is still auto-correlation between the histogram of optical flows in the same cell, which needs to be captured. In other words, if there is an anomaly in one cell in this frame, there is greater probability that there would also be an anomaly in the same cell of next several frames. Therefore, instead of using the metric for that cell as an anomaly score, a better approach is to use a weighted average anomaly score that incorporates the score of cells in adjacent frames. In the paper, the procedure is as follows:

$$\tau_0 = 1$$
$$\tau_\alpha = exp(-\frac{\alpha+1}{2}), \alpha = 1, 2, 3, ..., N$$
$$S_t = \sum_{n=0}^{N} \tau_n S_{t-n}$$

where S_t is the metric of the cell at frame t, and N is the number of frames in the past that have an impact in the current cell. This paper only uses the frames in the past as inputs to calculate anomaly score. However, at the cost of response time, the performance should be further improved if we can also take anomaly scores in the same cell in the next a few frames as inputs since these are also very informative. So an alternative is :

$$\tau_0 = 1$$
$$\tau_\alpha = exp(-\frac{\alpha+1}{2}), \alpha = 1, 2, 3, ..., N$$
$$S_t = \sum_{n=0}^{N} \tau_n S_{t-n} + \sum_{m=0}^{M} \tau_m S_{t+m}$$

where M is the number of frames used in the future. This should be smaller than N because we don't want to sacrifice too much response time.

3.4 Algorithm 2: Latent Space Auto-Regression

In this section, we implemented another time-series algorithm, which is a combination of auto-encoder and the generic auto-regressive process invented by Davide Abati.

In this methodology, we created a metric called novelty score to identify anomalies in the video. The higher the score, the more likely we are to have an anomaly. The novelty score is composed of two major components; the first is reconstruction error, the latter is the suprisal term. Here, we elaborate on the formulation and consequences of these two terms.

The reconstruction error is calculated from the auto-encoding process. Below, we can find the flow chart for the auto-encoder process, as it relates to the video anomaly context. The auto-encoder is comprised

of an encoder and a decoder process. The encoder process transforms x to a compressed latent variance z:$z = f(x, \theta_f)$, whereas the decoder provides a reconstructed version of the input: $\tilde{x} = g(z, \theta_g)$ (2). The encoder process is trained with the normal video frames where there are no anomalies; when the anomaly comes in, the reconstructed input will deviate largely from the input and thus the reconstruction error will increase drastically.

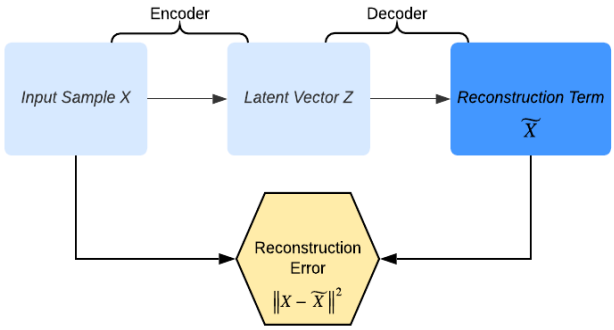

Figure 2: Encoder Process

The second part of the novelty score is surprisal term, and this is where auto-regression comes into place. Auto-regressive models provide a general formulation for tasks involving sequential predictions.(2) The surprisal score is calculated as $-log(P(z))$; the lower the probability, the less likely the event is going to happen and more likely it is going to be an anomaly.

Here, we try to estimate $p(z)$ through factorization and through estimation of each conditional probability density, as expressed by the below formula.

$$p(z) = \prod_{i=1}^{i=d} P(z_i | z < i)$$

The latent variable is conceptually similar to the latent variables used in LSTM. Whereas LSTM sequentially observes latent symbols $z < z_i$ and outputs the CPD of z_i as the hidden state of the last layer, in the this method, we sequentially observe $z < z_i$ and estimate the conditional probability $P(z_i | z < z_i)$ at each input i.

We construct a weighted average of the reconstruction error and surprisal term to obtain the final novelty score and identify the anomaly in each video clip.

4 Experimental Evaluation

4.1 Data

4.1.1 Data Set Description

This project worked with the UCSD Anomaly detection data set. A stationary camera was mounted at a fixed elevation above the ground and was positioned to overlook and monitor pedestrian walkways. This camera recorded a total of 98 videos that were available for analysis. 70 of these videos were mounted with the camera positioned to detect motion perpendicular to the plane. These constituted the primary focus of our analysis. The latter 28 videos were positioned to detect motion parallel to the plane. While these were not used as part of this project, they represent additional data that can be used to verify if the model results obtained can be generalized to a similar data set.

Of the 70 videos, 34 contained no abnormal activities. This means that each video consisted of pedestrians walking on the walkway. The other 36 videos contained an anomaly somewhere within the sequence. These anomalies typically represented about 1/3 of the length of the video, when present. An anomaly can be defined as either the situation where a non-pedestrian is on the walkway e.g. a golf cart or when a pedestrian displays

(a) Left: original video frame

(b) Right: Optical Flow of the left frame

Figure 3: Visualizations of Optical Flows. Left: frame 1 from UCSD anomaly detection dataset. Right: Color Coded Optical Flows of the left frame in which different colors represent different directions. We used a different way of calculating Optical Flows than the original paper but the main idea stays the same

anomalous motion patterns e.g. a skateboarder. Each video that was explored lasted around 8 seconds and was split into 200 frames. Each frame was an 158 X 238 image file whose color profile could be represented as a 158 X 238 array.

4.1.2 Feature Extraction Construction

Several techniques are available for expressing each video frame as a singular entity. One means of aggregation of the 158 X 238 array was to concatenate the array and generate a 37604 X 1 array for each frame. This method was used when generating the baseline models and also in the pre-processing of data for latent space auto-regression. In the case of latent space auto-regression, the data processing was done by using TensorFlow to transform all videos into tensors. A similar approach to this was to use a vertical or horizontal summation to generate a 158 X 1 or a 238 X 1 array for each frame. While this is computationally more efficient especially as we assemble multiple videos to train, preliminary analysis showed that these summation models performed poorer than the concatenation counterparts. This is explained by the loss of information experienced when a row or column of information is compressed into one value.

The Gaussian process algorithm used a different methodology for feature construction. Firstly, for each video frame of size w $\times$ h, we divide the frame of size L $\times$ L with overlapping elements. Second, for each cell, we calculate optical flows at every pixel in the cell. In Figure 3, the right subplot is a visualization of our optical flows of the left subplot. Lastly, divide all pixels in the cell into eight intervals by angles and accumulate all pixels' magnitude in each interval. In that way, for each interval a histogram of magnitude is obtained and for each interval a 8 dimension histogram is obtained. After that, we normalized all histogram of optical flows by dividing all values by the maximum value in that interval. This histogram of optical flows captures both direction and speed and thus is a good representation for further analysis.

However, we cannot simply input all histograms of optical flow into a Gaussian Process model; due to the sheer volume, this would be computationally expensive. Therefore, the paper used an online clustering algorithm which clusters all the feature vectors and use the center of the clusters as the final inputs to the Gaussian Process model. The reason traditional clustering methodologies like k-means do not apply here is that we don't know the number of clusters ahead of time. We replicated the the online clustering algorithm-the interested reader can read the paper by Nannan(1) as we do not present further details on it here.

4.2 Methodology

In the data set we focused on, we were provided with 70 videos. Although each method used a slightly different split of training, test, and validation, the universal evaluation metric was the AUC. This was selected because this is a base-rate invariant method that shows our model's ability to balance false positives and false negatives.

For the baseline and Gaussian process models, all 70 videos (14,000 frames) were split into a 70:20:10 balance where 70% of the frames were used to train, 20% were used to validate the results for the hyperparameter tuning, and 10% of the data was used for the final prediction.

For latent space auto-regression, our methodology was to train on all the videos without abnormalities e.g. 34 videos. Testing was only conducted on 8 videos because of computational limitations.

4.3 Results

For our baseline modeling, we experimented with data representations by trying various aggregation techniques across several models. The first question (Question 1) we tried to answer was whether or not we could predict if a video contained an anomaly. Although this line was not pursued in our additional time-series models, this exploration gave a sense of which representations would perform best. THe second question (Question 2), to which we can compare the time-series models asked if we could identify the frames that were anomalous in a given video.

In the table below, we present the logistic regression results to answer the first question the first question (logistic regression) and the kNN results to answer the second question.

Question	Video Frame Aggregation	Video Sequence Aggregation	Modeling Technique	AUC
1	Concatenation	Concatenation	Logistic Regression	0.675
1	Concatenation	Summation	Logistic Regression	0.658
1	Summation	Concatenation	Logistic Regression	0.647
1	Summation	Summation	Logistic Regression	0.592
2	Concatenation	N/A	kNN	0.717
2	Summation	N/A	kNN	0.688

Table 1: AUC for Standard Classification Methods

We saw consistently that concatenation provided better results than summation since there was no loss of information. Although the data size became quite large, the simplicity of the baseline techniques did not lead to computational difficulties.

In the table below, we list AUCs for the Gaussian process algorithm using different metrics. GP-M uses mean as anomaly score, GP-V uses the inverse of variance and GP-P uses $p(y* = 1|X, y, x*)$. One thing to note is that although in this dataset, GP-M has a better performance than GP-V and GP-P, more comparisons on more data sets are needed to determine which metric is generally better. For our experiment, however, we present the anomaly score metric's results to show our model's overall performance.

GP-M	85.7
GP-V	83.5
GP-P	79.8

Table 2: AUCs for Gaussian Process

VIDEO-ID	AUROC-LLK	AUROC-REC	AUROC-NS
Test001	1.000	0.629	0.985
Test002	0.992	1.000	1.000
Test003	0.807	1.000	0.950
Test004	1.000	1.000	1.000
Test005	0.954	1.000	1.000
Test006	1.000	0.816	0.934
Test007	1.000	1.000	1.000
Test012	0.962	0.991	0.990
avg	0.933	0.909	0.954

Figure 4: AUC Comparison for Latent Regression

The second column listed out the AUC score based on Surprisal term only (log-likeliood), the third column

listed out the AUC score based on reconstruction error only and the last column is the AUC score based on novelty score. On the overall basis, the algorithm based on novelty score performed better than the other two methods.

4.4 Discussion

Our initial hypotheses were two-fold: assuming a time-series dependency of the information in the video frames would provide better results than treating the data set as independent. Secondly, we rationalized that the latent space auto-regression technique would provide superior prediction results as it makes no assumptions about the distribution of the normal data points or of the nature of the novelty points. Both of these were shown to be correct in light of the AUC results provided.

For the Gaussian process approach, we use a weighted-average approach that only considers the anomaly score of N previous frames. There is validity to this approach if we expect to determine live whether or not a given frame contains an anomaly. Since we have no knowledge of future events to use in our prediction, using this information in our model could represent a leak. However, in the use case where we have an entire video sequence and we wish to analyze the entire sequence retroactively to identify the anomaly, the inclusion of subsequent frames' anomaly score may be valid.

When we equip a deep auto-encoder with a density estimator that learns the probability distribution auto-regressively, this confers an advantage to novelty detection. The performance of this method suggests that while the Gaussian process approximation is valid, it does not capture the true probability distribution of normal events.

A summary of our baseline and model results is shown below, signifying latent space auto-regression as the superior technique:

Modeling Technique	**AUC**
kNN (Baseline)	0.717
Gaussian Process	0.857
Latent Space Auto-Regression	0.954

Table 3: Summary of Model Results

5 Conclusions

At the onset of this project, we sought to implement a model that could determine whether a video frame contained abnormal events or not. When using the baseline models that assumed no time-series dependency, an AUC of 0.717 was obtained using the kNN method. An improvement on this metric was obtained when the data was modeled as a Gaussian process, resulting in an AUC of 0.857. The latent space auto-regression technique provided an even further improvement to this, providing an AUC of 0.954.

Over the course of the project, it was apparent that the methodology selected for constructing features had significant impact on the quality of the model result. For example, we saw that concatenating the image frames provided better results than summation techniques. We also believe that the loss of data when dividing each video frame into a smaller grid for the histogram of optical flow method contributed to why its performance was inferior to latent space auto-regression.

Next steps for this project should center on exploring data representations that better balance the loss of information with computational efficiency. In addition to this, we referenced 28 additional videos where the camera was mounted to capture pedestrian movement parallel to the plane. Additional research could investigate whether each technique used for the original videos could be transported without manipulation to the new videos or if retraining is required.

6 Student Contributions

Duo Jiang- implemented the Gaussian process approach
Ademola Oladosu- developed baseline models, compared results, consolidated report
Yunya Wang- implemented the Latent Space Autoregression approach

Github Repository, containing relevant code: https://github.com/ReginaMayhem/Video-Anomaly-Detection

References

[1] Nannan Li, Xinyu Wu, Huiwen Guo, Dan Wu, Yongsheng Ou, Yen-lun Chen, *Anomaly Detection in Video Surveillance via Gaussian Process*. April 2015 International Journal of Pattern Recognition and Artificial Intelligence.

[2] Abati, Davide. Porrello, Angelo. Calderara, Simone. Cucchiara, Rita. "Latent Space Autoregression for Novelty Detection." Latent Space Autoregression for Novelty Detection, 4 July 2018.