Fishing Trawler Event Detection: An Important Step Towards Digitization of Sustainable Fishing

Tor-Arne S. Nordmo
Department of Computer Science
UiT: The Arctic University of Norway
Tromsø, Norway
tor-arne.s.nordmo@uit.no

Aril B. Ovesen
Department of Computer Science
UiT: The Arctic University of Norway
Tromsø, Norway

Håvard D. Johansen
Department of Computer Science
UiT: The Arctic University of Norway
Tromsø, Norway

Pål Halvorsen
Department of Computer Science
SimulaMet, OsloMet
Oslo, Norway

Michael A. Riegler
Department of Computer Science
SimulaMet, OsloMet
Oslo, Norway

Dag Johansen
Department of Computer Science
UiT: The Arctic University of Norway
Tromsø, Norway

***Abstract*—Detection of anomalies within data streams is an important task that is useful for different important societal challenges such as in traffic control and fraud detection. To be able to perform anomaly detection, unsupervised analysis of data is an important key factor, especially in domains where obtaining labelled data is difficult or where the anomalies that should be detected are often changing or are not clearly definable at all. In this article, we present a complete machine learning based pipeline for real-time unsupervised anomaly detection that can handle different input data streams simultaneously. We evaluate the usefulness of the proposed method using three well-known datasets (fall detection, crime detection, and sport event detection) and a completely new and unlabelled dataset within the domain of commercial fishing. For all datasets, our method outperforms the baselines significantly and is able to detect relevant anomalies while simultaneously having low numbers of false positives. In addition to the good detection performance, the presented system can operate in real-time and is also very flexible and easy to expand.**

***Index Terms*—Event detection, Unsupervised, Sustainable Fishing**

I. Introduction

Data streams containing video, images and sensor data are being generated in multiple areas of society. Humans and machines are constantly observed at different frequencies and qualities. Examples include traffic or security video surveillance, monetary fraud transaction monitoring, or detection of errors in industry production facilities. To be able to build useful and efficient analysis applications in such domains, one usually requires data streams with labelled data like, for example, bounding boxes around people in surveillance videos or annotations of events in sensor data streams. For some applications and data steams, this is feasible, but for others, it is not. This often depends on the amount of data (too much data to annotate, requiring large amounts of manual labor), or the simple fact that one does not know what to annotate. Proper datasets are key to development of machine learning applications. Problems in many existing datasets include completeness, quantity, validity, and correctness of data, and correct labelling of data for supervised learning approaches. For some application domains, proper datasets are not available at all, or have very limited value. For instance, finding outlier values in a series of data, that is to temporally or spatially localize the anomaly events in a time-series sequence, can be challenging.

These challenges are specifically relevant for anomaly detection, because it is often not know what anomalies can happen, and it would require a huge number of annotations. In addition, feature extraction comes with a computational cost that might be a bottleneck in, for instance, a real-time surveillance context. The current trend in research is to find alternative solutions often incorporating semi-, self- or unsupervised [1]–[3] learning. Most of these solutions are focused on one specific type of data stream, i.e., video or sensor data only, although some application scenarios provide several data streams simultaneously that can be analysed to produce a better result (e.g., multiple video streams from different angles, video and sensor data from traffic, or sensors measuring different bio signals from an intensive care unit patient).

To address the challenge of anomaly detection when multiple data streams are provided, we propose an unsupervised anomaly detection system that is able to handle several, multimodal data streams simultaneously. The system is designed based on the specific use-case given in the Dutkat project targeting sustainable harvesting of marine resources off-shore [4]. This is a multi-billion dollar industry worldwide, but one that also comes with serious problems according to, for instance, the United Nations and their sustainability focus [5]. The main goal of Dutkat is to detect potential criminal activity on large off-shore fishing vessels by introducing robust, privacy-preserving edge-computing systems [6], and multimodal data streams on each vessel. Although the Dutkat use-case was one of the main motivators for developing the system, we also show that it is very flexible and can be applied to totally different use-cases involving data streams and anomaly detection. We demonstrate that our system is relatively application domain agnostic by using datasets from the surveillance, elderly care (unexpected falls), and sport domains.

The main contributions of this paper are as following: (i) we propose an unsupervised anomaly detection system that can handle several, multimodal data streams simultaneously in the environment of a fishing trawler that come with specific requirements and limitations, (ii) we evaluate our system on three different datasets with temporal annotations; and (iii) we apply it on unlabeled data from a fishing trawler's surveillance system, where we manually validate the results.

Our experiments show that the proposed system is able to detect anomalies completely unsupervised in an efficient manner, and utilizes information from different data streams if available. In addition, we show that it outperforms several core baselines for the labeled datasets. Finally, and most important we show that it can be used for the specific use case of anomaly detection on fishing trawlers that comes with requirements.

II. Related Work

Anomaly detection has been researched extensively in the past years. Most novel methods rely on deep-learning based approaches [7] that require a lot of training data. Less research has been performed in the direction of unsupervised machine learning for anomaly detection. This is probably due to several factors such as difficulty to verify and evaluate the output and the general lower performance of unsupervised methods compared to supervised ones [8]. Some methods also rely on a combination of supervised and unsupervised learning [9].

[10] present an unsupervised anomaly detection algorithm for traffic video data that achieves an F1 score of 0.5926 on the NVIDIA AI CITY 2020 challenge test dataset [11]. [12] proposed another unsupervised method using autoencoders to detect anomalies in high-performance computing systems. The challenge in this specific area is that the available datasets are rather small and supervised methods cannot easily be used. In addition to the challenges of lacking enough training data, detecting unknown abnormalities in real-time has attracted focused research. This is often an important requirement for systems that intend to be used in real world scenarios that are time-critical [13], [14].

Applying anomaly detection in real-world scenarios comes with several challenges that we are tackling with the presented work here. First, a real-world capable system needs to handle the input data in real-time and optimally should also be able to deal with new data or additional data streams. Furthermore, depending on the application, one might not know what different types of anomalies can happen and how these anomalies might change over time. Thus, a complete system needs to be agile and able to react to changes and new anomalies. This comes with a trade-off in terms of recall and precision, that is whether all relevant events are captured versus how many of the captured events reported actually are relevant ones. In scenarios like capturing potential fraudulent activities on fishing vessels, one rather would like to detect all suspicious events and accept a larger number of potentially false positives [4]. Additionally, the fishing trawler use case also comes with limitations such as very limited bandwidth due to satellite connections, computational power limitations and that the detected anomalies will change over time due to different reasons such as change of crew or behaviour on the boat. This rules out most existing related work since we cannot rely on a model that is trained on annotated data or methods that are computational heavy. Thus, we present in this paper a method that is light weight and specifically designed for the application on fishing trawlers.

III. System

Our goal is to provide real-time anomaly detection from multiple data streams using an analytics engine running on the edge nodes onboard fishing vessels. For this, we propose an efficient and highly modular pipeline system consisting of three steps: (1) feature extraction, (2) an optional embedding layer, and (3) a moving average-based anomaly detection method. The pipeline allows for analysis of multidimensional input, such as multicamera-based datasets, or combinations of anomaly detection methods. We emphasize that our design is modular and can be configured with multiple inputs, feature extraction methods, and embedding processes, depending on the target context. An example configuration of our use-case can be seen in Figure 1, with multiple inputs, different pretrained feature extraction approaches, with the result being combined at the end. The combination of the anomaly detection outputs can also be performed in numerous ways, for example by using simple boolean OR/AND operations, or more clever weighting strategies. Below, we describe the specific components used for our evaluation.

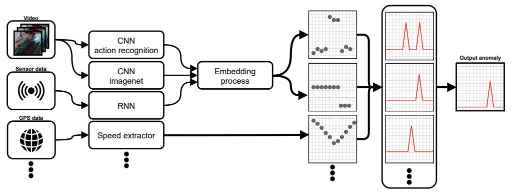

Fig. 1. Example of pipeline configuration. Notice how multiple, multimodal data streams can be processed in different ways and run through the anomaly detection algorithm, before being combined. The embedding process is also dependent on the feature extractor, and thus is not always necessary.

A. Feature Extraction

The feature extraction step of our pipeline can be based on a general machine learning model like a pretrained neural network. Such a model does not have to be trained on the use-case. For example, in this work we use an action recognition network trained on tasks not relevant for anomaly detection on a fishing trawler. We chose this model due to the fact that it is not well know which actions happen on a fishing boat and a more general action recognition dataset might the best to generalize to our case. The specific predictions do not matter in isolation, only the change in predictions over time. To perform action recognition, we utilize an 18-layer R(2+1)D network, introduced by [15]. The network was pretrained on the Kinetics-400 dataset [16], which consists of 400 action classes. For the used architecture, the pooling layer outputs a 512-dimensional feature vector that is fed to the fully-connected layer with softmax. The output from this network functions as feature extraction, where the predicted class of a subsequence of the video, is a data point in the transformed time-series.

For alternative data sources, feature extraction can take different forms, such as neural network based classification of the time series data, or direct use of the raw data without any feature extraction steps.

An important thing to note is that we chose this particular pretrained network due to our Dutkat use-case where we are attempting to identify unexpected and potentially illicit actions of the fishermen. However, due to the lack of data of such actions, we are particularly interested in changes of actions which the pretrained network predicts. These actions will often not be correct, but changes between different actions will most likely reflect a change in the actual actions as well.

B. *Embedding of labels*

The output vector from the pretrained neural network is ordered alphabetically. That is, the indices of the output vector are ordered by the alphabetical order of the corresponding labels. Thus, values that are close together in the output vector will not represent any realistic ordinal relationship. In our experiments we observed that this can cause the anomaly detection method to misclassify certain data points due to the output value changing dramatically, while the underlying labels are semantically similar. For example, the output value might change from 112 to 353, which corresponds to class indices that are sorted alphabetically, but the underlying labels corresponding to these values could be, for instance, "eating chips" and "tasting food", which are semantically close. To address this problem, we applied an embedding on the labels to investigate if it can lead to better results. This embedding would place semantically similar labels close together and thus should lead to better input for the anomaly detection.

The embedding process is shown in Figure 2. The textual labels that correspond to the output vector indices are first embedded via a *sent2vec* model, devised and implemented by [17], that was trained on a Wikipedia corpus. Then, ten high-intensity and low-intensity activities were chosen from the labels to function as "support-vectors", analogous to how a support vector machine works. A line is then drawn between the means of the low-intensity support-vectors and the high-intensity support-vectors. This line represents an intensity axis. Every point is then projected onto this line, and they are then max-min scaled to values between 0 and 1.

The axis used is based on the assumption that anomalies in videos containing people only arise when a drastic change in intensity occurs. We conjecture that such changes are context dependent and might change depending on the dataset on which the method is to be used.

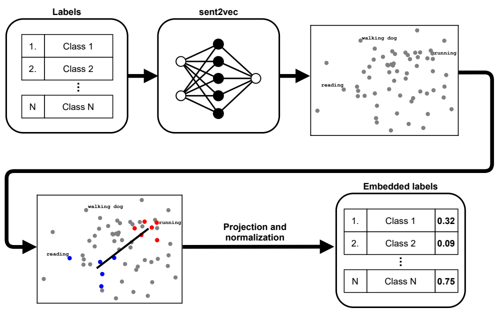

Fig. 2. Illustration of embedding process. Note how the classes are distributed when they are embedded by sent2vec, and that in the embeddings are in a different order than the index provided in the beginning.

C. *Anomaly detection method*

Most anomaly detection methods evaluate if a data point is part of the dataset distribution, and if not, labels it as an anomaly. We, however, are interested in an unexpected transition in the values of a time-series from single or multiple data streams, where the values are still part of the distribution. These are known as changing points [18].

Since we are interested in changing points, i.e., points in a time-series where the sequence changes for an interval of time, we chose a moving average approach. This approach compares the median of the previous n points to the current point in a time-series. If the current point is outside of the historical interquartile range, it is deemed an anomaly.

For features extracted from video data (e.g. which class does a given subsequence belong to), the anomaly detection method looks for drastic changes in the features.

IV. Datasets

The system was applied on three different labeled datasets for evaluation, and an additional unlabeled dataset for testing. The datasets depict varied situations and actions, with different video durations and potential artifacts. The datasets also consist of very different camera positions and movements, with static, moving, and multi-positional cameras depending on the dataset. The action recognition model was pretrained on the Kinetics-400 dataset [16]. The reason for choosing the Kinetics-400 dataset for pretraining is twofold: (i) models trained on this dataset were most easily accessible, (ii) it contains many classes, which makes it more likely that the model can react on a potentially similar event in the other datasets.

The *Multiple cameras fall* (Falling) dataset by [19] consists of 24 videos with eight cameras filming the scenarios. Of these, 23 depict a person performing several activities, before falling onto a mattress or chair. The remaining video does not contain a fall. The videos have a resolution of 720×480 and a framerate of 30 frames per second (FPS).

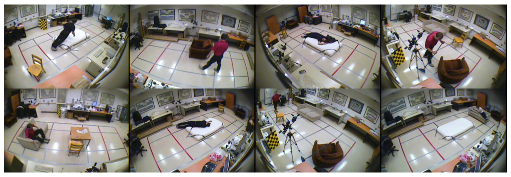

Fig. 3. Frames from different videos of the Falling dataset [19]. Notice the different angles and objects in the scene. (The images are from publicly available videos.)

The Falling dataset is different from the other datasets that are explored in this paper, because it is based on eight cameras capturing each fall from different angles. The different angles can be seen in Figure 3. This allows us to detect anomalies that might otherwise be obscured by the orientation of the person in the video and test the system for its multiple streams capabilities.

The *Real-world Anomaly Detection in Surveillance Videos* (AV) dataset, created by [20], consists of 1,900 videos containing for example actions like abuse, arrests, arson, assaults, and accidents. The videos are from static surveillance cameras and contain varying backgrounds and numbers of people. The videos have a resolution of 320×240 and a framerate of 25 FPS. Despite the videos having the same resolution they might have different letterboxing, as can be seen in Figure 4.

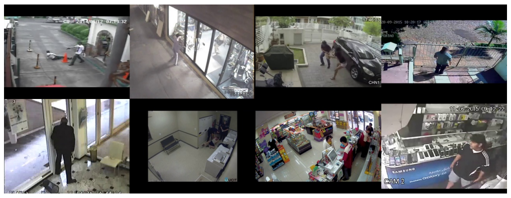

Fig. 4. Frames from different classes within the AV dataset [20], e.g., image three is depicting a robbery and image six is depicting an arrest. (The images are from publicly available videos.)

We will only utilize the videos that have corresponding temporal annotations, as this is needed for the evaluation. This reduces the dataset size to 304 videos. Moreover, the majority of these videos are the "Normal" videos (i.e., they do not contain any anomalies), so these are also ignored, which reduces the dataset further to 154 videos on which we can evaluate our system.

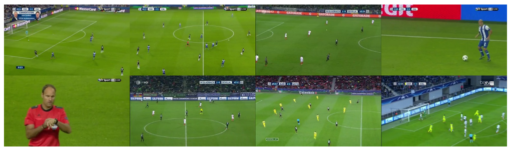

Fig. 5. Frames illustrating the variety in the Soccernet-v2 dataset [21], with close-ups, wide-angle shots, and the camera following a specific player/referee. (The images are from publicly available videos.)

The Soccernet-v2 dataset ([21], [22]) consists of 500 broadcast soccer games with 300,000 temporal annotations including action and camera labels. We randomly chose a subset of all the games from 2015-2016, which totals 45 games. Each video lasts for approximately 45 minutes, which is much longer than the videos of the two previous datasets discussed above. The videos have a resolution of 398×224 and a framerate of 25 FPS.

Two groups of events from the videos are labeled; the camera movements and actions. The camera movements consist of events like close-ups on players/referee or behind the goal, and when switching between different cameras. The action labels consist of soccer events like goals, free-kicks, and fouls.

A major difference with this dataset compared to the two above is that the camera often is far away from the players to capture the coordinated flow of the game involving multiple players and their opponents. Therefore, detecting actions can be relatively easy in certain cases, such as in the fourth picture in Figure 5, but more difficult in other cases.

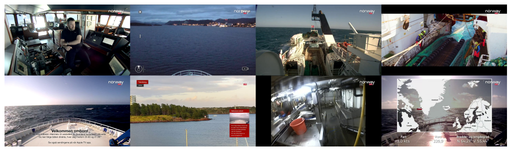

Fig. 6. Frames from the FT dataset. Since the videos are from a live-stream, several unnatural artifacts and subsequences are included. Notice the eighth image containing an overlay explaining where the boat is at the time, but it obscures the background which we are interested in. (The images are from publicly available videos.)

The fishing trawler dataset consists of a collection of 30 videos of approximately one hour each. They are a series of live-streams from 2019 filmed aboard the Hermes[1] fishing trawler during operation in the Arctic Sea. The videos contain ten different camera positions, including a camera on front of the trawler, another angled on the deck, and a camera on the factory level of the ship. These are alternated between at semi-regular intervals. The videos have a resolution of $1,280 \times 720$ and a framerate of 25 FPS.

The FT dataset contains many artifacts since it is from a live-stream, This can be, for instance, as overlay map explaining where they are or information about what they are catching, or even external sequences which are not part of the video stream captured on board. This can be seen in Figure 6.

V. Evaluation

Evaluating an unsupervised anomaly detection system is difficult. Comparing against labeled datasets can be potentially misleading, due to temporal labels only reflecting when a human detects a change in the time-series. However, the change in the flow of the time-series might have occurred earlier. Therefore, it is important to consider the interval around a temporal label that decides whether a prediction is classified correctly or not. In addition an unsupervised system might detect anomalies that are not detected by the human or are not normal from a data perspective but from a human perspective depending on the use case nothing special (e.g., a bird flying trough the scene).

[1]https://www.hermesas.no

A. Experimental Setup

We implemented the system in Python using PyTorch ([23]) and the Anomaly Detection Toolkit (ADTK)[2]. The pretrained neural network model was imported from the `torchvision` module. The frames are extracted from the videos and are resized and combined into a tensor in the batch generator. The feature extraction was performed on each dataset, and the results were saved in comma-separated files. Then, the anomaly detection was applied to the extracted features. Each data point corresponds with an 8-frame clip, which is one of the accepted inputs to the action recognition network used. Due to the specific use case the system is designed for and the requirements coming with it, it would not provide any insights if we compare the system with other existing anomaly detection methods. But to compensate for this we tested the system on three datasets not coming from the fishing domain to show that it is able to detect anomalies at a acceptable level.

All datasets except the Falling dataset were analyzed using a simple linear pipeline which consists of the input, a pretrained action recognition network, an embedding layer, and the anomaly detection algorithm. The Falling dataset contains multiple data streams capturing the same event, therefore the configuration is slightly different. Multiple data sources, corresponding with the different camera angles, are fed into the same pretrained network and embedding layer, before being the processed data is run through the anomaly detection algorithm separately. These detection streams are then combined naively using a simple boolean-OR combination, i.e., at the current timestep, if an anomaly is detected in any of the streams, it is regarded as an anomaly in the combined output.

The evaluation was prohibitively expensive on the Soccernet-v2 dataset due to the length of the videos, and required optimization. The evaluation of each predicted point, which consists of a comparison against temporal annotations, was divided across multiple nodes. This allowed us to evaluate detected anomalies in a parallelized fashion.

All the experiments were run on a Linux machine with Ubuntu 20.04 LTS distribution having a 3.70GHz Intel Xeon CPU E5-1620, 64GB DDR3 RAM, and an OCZ-VERTEX 4 512GB SSD. The feature extraction was run on an NVIDIA RTX 2080Ti GPU. We assume this system will run on large trawlers, which are equipped with similar computational capabilities, sufficient for running inference on pretrained CNNs or similar architectures.

B. Evaluation Methodology

For evaluation purposes, we compare our system to multiple baseline classifiers, namely a uniform random baseline and two constant baselines (i.e., classifying every data point as an anomaly or non-anomaly). We compare accuracy, recall, precision, F1-score, and Matthew's correlation coefficient (MCC)[3]. The metrics are calculated as macro-averages over the two classes (anomaly/non-anomaly). Due to the imbalanced nature of the datasets, the accuracy will generally be quite high and misleading, but we include it for completeness. For the AV dataset, we further evaluate the performance per class, to determine whether our approach is better at detecting specific types of anomalies. Finally, we evaluate how the interval range of whether a point is deemed as a true positive or a false positive affects the results.

Furthermore, we benchmark our system in terms of processing performance to evaluate if it can be applied in a real-time scenario, i.e., processing the videos at the speed of the framerate. However, the ADTK framework is not designed for real-time analysis, i.e., it expects a complete Pandas dataframe before performing any analysis. Thus, we have chosen to first generate a dataframe that is the length of the comparison window of the anomaly detection method, then actively append new data points and remove the oldest data points to maintain a fixed size. The average framerate is calculated based on ten random videos within each dataset.

Finally, we manually inspect and analyze the results of applying our system on the FT dataset, to gain some insight as to how it handles this specific scenario. We particularly focus on what type of events are classified as anomalies, and also carefully investigate what the system misses.

[2]https://arundo-adtk.readthedocs-hosted.com/en/stable/

[3]Note: MCC cannot be calculated for the constant baselines due to division by zero.

C. Results and Discussion

In this section and the respective subsections we present the results and discussions around them. We start with a more general perspective followed by deeper analysis of different interesting aspects of our evaluation.

Falling Dataset					
Method	Recall	Precision	F1-score	Accuracy	MCC
Uniform	0.49	0.49	0.46	0.50	0.009
Constant 1	0.50	0.38	0.43	0.23	N/A
Constant 0	0.50	0.12	0.19	0.77	N/A
Ours	**0.76**	**0.82**	**0.79**	**0.86**	**0.57**
AV Dataset					
Method	Recall	Precision	F1-score	Accuracy	MCC
Uniform	0.50	0.50	0.34	0.50	-0.003
Constant 1	0.50	0.01	0.01	0.01	N/A
Constant 0	0.50	0.49	0.50	**0.99**	N/A
Ours	**0.58**	**0.57**	**0.57**	**0.99**	**0.15**
Soccernet-v2 Dataset					
Method	Recall	Precision	F1-score	Accuracy	MCC
Uniform	**0.50**	0.50	0.43	0.50	-0.001
Constant 1	**0.50**	0.07	0.12	0.14	N/A
Constant 0	**0.50**	0.43	0.45	0.86	N/A
Ours	**0.50**	**0.82**	**0.48**	**0.87**	**0.10**

TABLE I

COMPARISON OF OUR SYSTEM AGAINST BASELINES. EACH TABLE CORRESPONDS TO THE RESULTS FROM A SPECIFIC DATASET. BOLD INDICATES THE BEST PERFORMANCE.

1) Compared to baseline: **Falling Dataset.** As documented in Table I, our system achieves the best results on the Falling dataset. We conjecture this is due to multiple reasons; first, this dataset consists of multiple cameras filming the scene from different angles, thus erroneous changes in predicted actions will be filtered out due to the boolean-OR combination approach in the anomaly detection algorithm, which is different from the other datasets which do not require combinati multiple data sources. Secondly, each video only conta single person throughout the majority of the duration v makes it easier for the algorithm to focus on the specific a

AV Dataset. The results on the AV dataset are a bit c to the baselines compared to the Falling dataset, but w get better results for all metrics which show that the pro method is able to detect anomalies even in difficult scen The reason for the results being worse than for the F dataset is most likely due to the variety of the conten quality of the videos. Some actions are visually eas detect than others. A more detailed discussion on this specific challenge is provided in subsection V-C2.

Soccernet-v2. Our system was closest to the baseline when applied on the Soccernet-v2 dataset. We assume this is due to the large distance between the camera and the players in the majority of the duration of the videos. The greatest difference is for the precision, which implies that the system is performing well at detecting events/anomalies, while detecting few false positives.

With regard to the MCC, as stated previously, we can only compare the uniform classifier to our system. The MCC is a metric that handles imbalanced data well, taking all elements of the confusion matrix into account. A value of zero would indicate a classifier predicting correctly 50% of the time. A value of one would indicate a perfect classifier. We see that the uniform classifier has an MCC of approximately zero. The largest difference in the MCC is for the Falling dataset, with 0.57 for our system. Though MCC might not be a reliable indicator of how good a classifier is [24], it clearly correlates to a certain degree and gives a better understanding of the performance when the datasets are biased. For all three datasets, the MCC was above random predictions for our system.

From this first analysis, we can see that the system is performing well on the task of unsupervised anomaly detection. Specifically, we can see that multiple data streams appear to lead to better results (Falling vs the other two). It is apparent that the type of classifier that produces the input for the anomaly detector is important. The basic action recognition model used was best suited for the falling detection whereas the event in the other two datasets where not specifically part of the actions. Using more specific models most probably would lead to better results, for example, a soccer or crime event-specific model for the respective dataset (or even combination of models capturing different aspects of the stream).

2) Deeper analysis on AV and Soccernet-v2 datasets: We assumed that certain classes in the AV dataset would be difficult to detect, such as the shoplifting class. Shoplifters try to be inconspicuous while performing their illicit activity, and as such, the detected actions might not change very much. Results documented in Figure 7 demonstrate that our system performs closest to the baseline on the Arson and Vandalism classes of videos. We can also observe a difference between the Constant 1 and Constant 0 baseline showing that Constant 1 is performing worse. This indicates that the dataset is containing

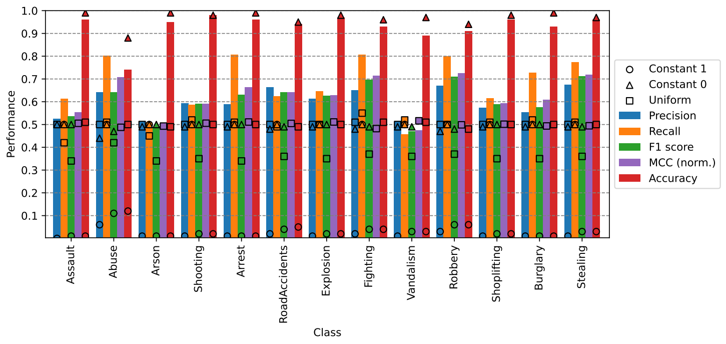

Fig. 7. Per class performance on the AV dataset. The circle, triangle and square markers indicate the results of the baseline classifiers for each metric.

a lot of normal frames and not so many positive frames which makes it again a difficult dataset. Most classifiers would easy start focusing on the normal frames and basically become a Constant 0 classifier. Our method avoided this and was able to detect some of the anomalies correct which is shown in the MCC with 0.15. Visually inspecting videos in these classes, we saw that, for the Arson videos, the actions of the arsonist do not change significantly over the course of the videos except for when they flee the scene. The system usually detects when the fire has reached a certain size, but not when it is ignited in the beginning. With the Vandalism videos, the system's performance highly depends on the type of vandalism and the distance from the camera to the perpetrator. Videos of tagging or videos where the camera is far away are more difficult for the system.

Soccernet-v2: Camera vs Actions					
Labels	Recall	Precision	F1-score	Accuracy	MCC
Camera labels	0.51	0.78	0.49	0.90	0.11
Action labels	0.51	0.67	0.51	0.95	0.08

TABLE II
PERFORMANCE OF OUR SYSTEM ON THE SOCCERNET-V2 DATASET USING THE CAMERA LABELS VS. THE ACTION LABELS.

Regarding the Soccernet-v2 dataset, there are a total of 14,969 camera labels and 10,008 action labels. Due to the distance between the camera and the players, our hypothesis was that it might be difficult for our system to detect the actions described by the action labels. Looking at Table II, we can see the the recall is the same for both sets of labels, which implies that the system that the same relative number of relevant anomalies/events are detected. However, the precision is quite different, with our system achieving better precision using the camera labels. This implies that fewer false positives are generated. This might align with our hypothesis, because the camera events are easier to detect.

3) Interval range for evaluation: In Figure 8, we are evaluating how the interval range around a true label affects the performance on the different datasets. As we remarked in the beginning of this section, evaluating time-series detection methods is difficult. Using only the true labels timestamp as a correct classification would be equal to results when range size

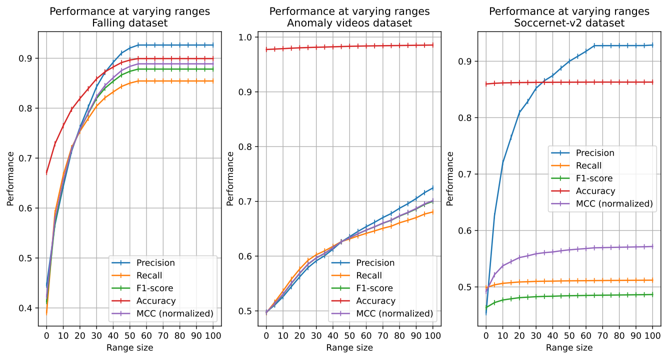

Fig. 8. How the interval range around a temporal label affects the results of our system. Due to a data point representing 8-frame clips, an interval range of 100 $(x-100, x+100)$ corresponds with ± 30 seconds around a temporal label for a 25 FPS video.

is set to zero. This would not be a realistic goal to optimize. As long as the system manages to detect an event/anomaly within a certain interval, this should be classified as a true positive. The range size in Figure 8 ranges from 0 to 100 data points in either direction around a true label. 100 data points correspond to approximately 32 seconds, so that would allow for any anomaly detected within a minute around a true label to be classified as a true positive.

In the Falling dataset, we can see that the results plateau at a range size of about 55, which corresponds with 20 seconds. As the longest video in this dataset is 44.6 seconds (average 11.1 seconds), it makes sense that the results plateau around this value, because at this point every predicted anomaly will correspond with one of the labels. We see a similar plateauing in the soccernet-v2 dataset, however the duration of these videos are always approximately 45 minutes. We assume this might instead be due to the camera movement, and a player rarely being in focus for a long time.

For both the AV and the soccernet-v2 dataset, the accuracy is consistently high regardless of range size. This is probably due to the imbalance of anomalies/events versus non-events being relatively greater than for the Falling dataset. This is because the videos in the Falling datasets are usually very short, thus the section of the videos containing the fall makes up a significant portion of the video.

FPS Benchmark

Dataset	#Input streams (res.)	Preprocessing	Avg. Framerate
Falling	8 (720 × 480)	Act. Recog.	108.2
AV	1 (320 × 240)	Act. Recog.	219.3
Soccernet-v2	1 (398 × 224)	Act. Recog.	200.9
FT	1 (1280 × 720)	Act. Recog.	157.6

TABLE III
RESULTS FROM FPS BENCHMARK. AS WE CAN SEE, THE SYSTEM CAN BE APPLIED ON REAL-TIME LIVE-STREAMED DATA.

4) Real-time benchmark: Based on the results in Table III, we observe that the system can be utilized when analyzing live data streams. Due to differences in FPS and resolution between the different datasets, the preprocessing causes the results to differ. Due to the Falling dataset consisting of multiple data streams per video, this obviously causes the framerate to be lower. The data streams are processed sequentially through the pretrained network and embedding layer, but the anomaly detection algorithm is applied in parallel. The results on the FT dataset are also quite lower than the AV and Soccernet-v2 datasets. This is most likely due to the resolution of the videos being higher and the duration being longer. However, despite the different results, we conclude that the system can be applied in a real-time context and deployment.

Anomalies detected in FT dataset

Event Type	Percentage
Scene Change	34.4
Human Activity	25.6
Lighting Change	7.6
Overlay	3.9
High Seas	5.7
Camera	2.8
Other	22.0

TABLE IV
MANUAL INSPECTION RESULTS FROM FT DATASET.

5) False Positives: It is important to note that events that are classified as false positives are not necessarily wrong. The system might detect events that were not deemed important by the annotators of their dataset. Nevertheless, these points might still be interesting, as they might indicate event such as the change in flow of a soccer match, or new people entering the frame in a surveillance video. With unsupervised approaches it is therefore impossible to be completely sure what is a false positive versus a true positive, but for the evaluations in the previous sections we have assumed the temporal annotations to be correct. However, based on our system's precision in Table I, we can see that, generally, the system does not introduce too many false positives.

6) FT dataset: Finally, we look at the manual inspection of the results on the FT dataset trying to detect anomalies on the fishing boat. We have applied the system on the entire dataset, and manually gone through and evaluated the predicted anomalies. The majority of events that are detected are changes between different cameras/scenes. After that, human activity is the next largest group, which contains actions such as "talking", "talking on the phone", "Leaning forward", "working", etc. Of the ten different camera positions, the angles within the bridge are the ones that depict most scenes with people on-screen. Then, we have lighting changes, which are either due to electric lights being turned on/off, or the sun lighting up different areas of the vessel due to rocking waves. The "High Seas" tag is also dependent on large waves rocking the vessel, which for example causes the horizon to cover more/less of the screen, or the sun to move in and out of frame. As such, there may be some overlap between these two classes. There are two types of overlays that are added over the videos; small notes containing factoids related to fishing and the sea, and larger overlays that cover most of the screen and show a map of where the vessel is at the moment. Finally, camera movement refers to a specific angle where the camera can be manually controlled by the skipper to show interesting events such as whale sightings or other vessels. Regarding the predicted events under "Other", these include all events that are relatively rare, like movement of cranes/heavy machinery, whale spottings, compression artifacts, and "nothing", which are events where we do not observe any obvious change and constitute 14.3% of the detected events. Some of these anomalies can be seen in Figure 9.

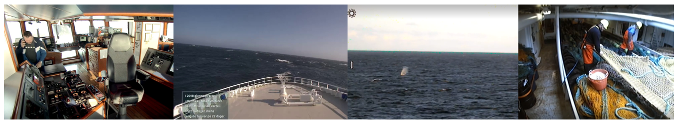
Fig. 9. Example frames of anomalies found in the dataset, containing (from left to right) "talking on the phone", "high seas", "whale spotting", and "working". (The images are from publicly available videos.)

17.2% of the scene changes were not detected. False negatives other than scene changes are difficult to evaluate, but certain events that are detected previously, like overlays, but not later are relatively easy to spot during manual inspection (2.1% of these are missed).

Overall, we can observe that the proposed system has capabilities to detect interesting and relevant anomalies. This makes it a possible alternative to check all content or not checking at all for the specific use-case. In addition, we only tested it with one data stream and a general action recognition model. If we combine different streams from trawlers and build more specific models, the performance will increase most certainly. Furthermore, it can be used to build datasets since it can be used to determine which possible events can occur, label them and make them usable for building supervised models that again can be used in the system.

VI. Conclusion

In this paper, we presented an unsupervised machine learning approach for anomaly event detection of multimodal data streams. The system is highly modular and can combine the results from multiple data streams, using different pretrained feature extraction methods that are not necessarily trained for the use-case. We apply our system on three labeled datasets, and one unreleased, unlabeled dataset. The system outperformed the baselines on all labeled datasets examined in this paper, however on the Soccernet-v2 dataset, which consists of far-away video shots of multiple players, performed the worst and was slightly above baseline.

In the future, we would like to try a more complex configuration of the system pipeline. The system pipeline can be expanded to utilize more pretrained networks for feature extraction, e.g., using a network trained on the Imagenet dataset [25] could make it easier to detect road accidents. It would also be interesting to use a detection network to detect and crop around individuals within the videos to see whether that could result in better performance, particularly on the videos where the camera is filming multiple people from a distance.

We aim to apply our system when annotating the FT dataset, and we will release this dataset in the near future. We will also apply our system when we deploy the Dutkat system on fishing vessels in the near future. In the actual use-case, we will utilize surveillance video data, sensor data from scales and other tools, etc. to try to incorporate as many sources as possible for the anomaly detection. Based on our preliminary analysis of the problem of detecting criminal/anomalous activity on fishing vessels based on video data, we have concluded that acquiring data of such specific events is infeasible. Therefore, we will in the future use multiple different data streams and perform anomaly detection on these combined, using the system described in this paper. These data streams would potentially consist of surveillance video data, sensor data estimating catch biomass, AIS (automatic identification system) data, etc. The resulting events and anomalies we detect from this data will be used to incrementally build labelled datasets that can be used in the system to improving performance and detect other anomalies.

Acknowledgments

We particularly would like to thank Signor Antonsen of Hermes AS, Kim H. Andreassen, Jon F. Mikalsen, and Kjetil Robertsen.

References

[1] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "Ganomaly: Semi-supervised anomaly detection via adversarial training," in *Asian conference on computer vision*. Springer, 2018, pp. 622–637.

[2] C.-L. Li, K. Sohn, J. Yoon, and T. Pfister, "Cutpaste: Self-supervised learning for anomaly detection and localization," in *IEEE/CVF CVPR*, 2021, pp. 9664–9674.

[3] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data," *PloS one*, vol. 11, no. 4, p. e0152173, 2016.

[4] T.-A. S. Nordmo, A. B. Ovesen, H. D. Johansen, M. A. Riegler, P. Halvorsen, and D. Johansen, "Dutkat: A multimedia system for catching illegal catchers in a privacy-preserving manner," in *Proceedings of the 2021 Workshop on Intelligent Cross-Data Analysis and Retrieval*, ser. ICDAR '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 57–61.

[5] United Nations Office on Drugs and Crime, "Fisheries crime," 2016.

[6] A. B. Ovesen, T.-A. S. Nordmo, H. D. Johansen, M. A. Riegler, P. Halvorsen, and D. Johansen, "File system support for privacy-preserving analysis and forensics in low-bandwidth edge environments," *Information*, vol. 12, no. 10, 2021.

[7] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[8] B. R. Kiran, D. M. Thomas, and R. Parakkal, "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos," *Journal of Imaging*, vol. 4, no. 2, p. 36, 2018.

[9] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," *arXiv preprint arXiv:1906.02694*, 2019.

[10] K. Doshi and Y. Yilmaz, "Fast unsupervised anomaly detection in traffic videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020.

[11] M.-C. Chang, C.-K. Chiang, C.-M. Tsai, Y.-K. Chang, H.-L. Chiang, Y.-A. Wang, S.-Y. Chang, Y.-L. Li, M.-S. Tsai, and H.-Y. Tseng, "Ai city challenge 2020-computer vision for smart transportation applications," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 620–621.

[12] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano, and L. Benini, "Anomaly detection using autoencoders in high performance computing systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 9428–9433.

[13] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, “Unsupervised real-time anomaly detection for streaming data,” *Neurocomputing*, vol. 262, pp. 134–147, 2017.

[14] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli, “Adversarially learned one-class classifier for novelty detection,” in *IEEE CVPR*, 2018, pp. 3379–3388.

[15] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, “A closer look at spatiotemporal convolutions for action recognition,” in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.

[16] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, “The kinetics human action video dataset,” 2017.

[17] M. Pagliardini, P. Gupta, and M. Jaggi, “Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features,” in *NAACL 2018*, 2018.

[18] A. Blázquez-García, A. Conde, U. Mori, and J. A. Lozano, “A review on outlier/anomaly detection in time series data,” *CoRR*, vol. abs/2002.04236, 2020.

[19] E. Auvinet, C. Rougier, J. Meunier, A. St-arnaud, and J. Rousseau, “Multiple cameras fall dataset,” 2010. [Online]. Available: http://www.iro.umontreal.ca/~labimage/Dataset/

[20] W. Sultani, C. Chen, and M. Shah, “Real-world anomaly detection in surveillance videos,” pp. 6479–6488, 2018.

[21] A. Deliege, A. Cioppa, S. Giancola, M. J. Seikavandi, J. V. Dueholm, K. Nasrollahi, B. Ghanem, T. B. Moeslund, and M. Van Droogenbroeck, “Soccernet-v2,” 2021. [Online]. Available: https://soccer-net.org

[22] ——, “Soccernet-v2: A dataset and benchmarks for holistic understanding of broadcast soccer videos,” in *IEEE/CVF CVPR*, June 2021, pp. 4508–4519.

[23] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[24] D. Chicco, N. Tötsch, and G. Jurman, “The matthews correlation coefficient (mcc) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation,” *BioData mining*, vol. 14, no. 1, pp. 1–22, 2021.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.