Boosting the Performance of Object Detection CNNs with Context-Based Anomaly Detection

Jan Blaha(✉), George Broughton, and Tomáš Krajník

Artificial Intelligence Center, FEE CTU, Prague, Czech Republic
{jan.blaha,george.broughton,tomas.krajnik}@fel.cvut.cz

Abstract. In this paper, we employ anomaly detection methods to enhance the ability of object detectors by using the context of their detections. This has numerous potential applications from boosting the performance of standard object detectors, to the preliminary validation of annotation quality, and even for robotic exploration and object search. We build our method on autoencoder networks for detecting anomalies, where we do not try to filter incoming data based on anomality score as is usual, but instead, we focus on the individual features of the data representing an actual scene. We show that one can teach autoencoders about the contextual relationship of objects in images, i.e. the likelihood of co-detecting classes in the same scene. This can then be used to identify detections that do and do not fit with the rest of the current observations in the scene. We show that the use of this information yields better results than using traditional thresholding when deciding if weaker detections are actually classed as observed or not. The experiments performed not only show that our method significantly improves the performance of CNN object detectors, but that it can be used as an efficient tool to discover incorrectly-annotated images.

Keywords: Anomaly detection · Object detection · Context-aware neural networks · Explainable neural networks · Autoencoders

1 Introduction

The development of deep-learning (DL)—as a subset of the machine learning field (ML)—in the last few years has brought great advances in the field of artificial intelligence (AI), and with it the spread of such deep neural networks into life-critical domains. That places a great deal of pressure onto the reliability, efficiency, and also on the careful preparation of datasets for a large number of individual applications. Moreover, these requirements are even stronger when the implementation is deployed in systems which interact with the physical world, since they can cause injuries to people around them as well as cause damage to property and the robot itself.

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 349, pp. 159–176, 2021.
https://doi.org/10.1007/978-3-030-67537-0_11

We believe that addressing the problem of contextual understanding can be useful when considering the above-stated concerns when dealing with perception. A system capable of taking into account context can often be much more effective in its tasks. This has some biological foundations, with strong evidence implying that humans similarly depend on this kind of contextual reasoning when performing tasks such as object recognition[3]. Greater reliability comes with greater accuracy, but it is also connected to context modelling by research [9,11], and also with the explainability of decisions that the system makes.

The complexity of deep neural networks has meant that the question of explainability is still open to research, but explainable context modelling would necessarily help the system to present its decisions in a human understandable way. In this paper, we present the idea that an important part of context modelling is the ability to spot anomalies, which is a well studied problem, and deep learning has been applied to it in the past [32].

Apart from not being able to reason about the decisions of the employed system, the training of large deep architectures is tedious and time-consuming. This is mainly because of the preparation required for the diversity, relevance and size of the training datasets, and most importantly, the quality of their annotation. We show that the use of anomaly detection addresses both of the aforementioned problems: it can spot potential detection errors of the networks, as well as identify incorrectly annotated images. If the system is able of spotting anomalies in the incoming data, it can flag wrongly annotated data in automatic dataset annotations, which then puts less pressure on human annotators or people that manually check the results. Even datasets completely annotated by humans are prone to errors, so double-checking of suspicious results can be useful as well. Better datasets inevitably lead to better models, and possibly to positive impacts on the length of the learning process, which can lead to money-saving due to the high energy consumption of datacenters used for the training of such huge models.

The ability to detect potential mistakes in one's output is a necessary precursor to automatically correcting one's mistakes, which, in turn, leads to higher accuracy and efficiency in the task performed. In another scenario, we might assume that our detection systems output is reliable, and the knowledge of anomalous outputs can be used for different tasks like robotic spatial or spatio-temporal exploration as indicated in [24,25]. Here, a high anomaly score indicates the possibility of object presence at a given location even if it is not currently visible, or can indicate that the detected object is at an unusual location. The work presented at [25] shows that the ability to detect these situations can be used to trigger robot's exploratory behaviours, which results in efficient spatio-temporal exploration [25].

This is in contrast to other currently existing DL-based models, which incorporate context modelling into their object detecting architectures, like [15]. Our approach is based on two standalone methods, one of which detects objects and the other is handling the context modelling through anomaly detection. This brings not only advantages in terms of being able to reason about the decisions

of such a combined model, but also minimal overhead to the training and deployment of the context-aware detection system.

Our method for anomaly detection builds upon existing research of using autoencoders for this task. We present a proof-of-concept for how these could be used as an extension of existing capabilities of state-of-the-art neural network models for object detection. In our experiments, we look at how the autoencoder can be attached to the output of the YOLOv3 object detection neural network [21]. Unlike the anomaly detection methods [1,22,33], who focus solely on the ability to detect anomalous images, our method is designed to indicate which actual feature causes the anomaly. We also provide a link to a repository containing all the code used for reproduction of the described methods and experiments [13].

To test the abilities of our method, we perform two experiments for finding positive anomalies, that is objects that are present but anomalous in the context, and negative anomalies, i.e. objects that are expected to be present but are not. These experiments are performed over annotations present in the Microsoft COCO dataset [18], and they resulted in flagging up several potentially wrong annotations, such as those shown in Fig. 1. Finally, we combine our method for anomaly detection with the YOLOv3 network and test its accuracy against the standalone YOLOv3, with our system determining whether low probability detections should be boosted up in confidence given the context of the surrounding detections. This experiment is based on comparing the mentioned methods by their accuracy on images in Microsoft COCO dataset, and we take the included annotations as ground truth.

2 Related Work

During the last few years, there has been much research conducted in the field of deep learning, particularly for the task of computer vision, making it impossible to cover the entire field in a short paper. In this section, we focus on two main topics relevant to this work—object detection networks and autoencoders with their use for anomaly detection.

2.1 Object Detection

Computer vision has advanced rapidly in the last decades due to the introduction of deep-learning into the field. Starting with AlexNet in 2012 [17], the progress quickly lead to the introduction of the exponentially bigger VGG architecture [27], GoogLeNet [28], and ResNet [14] networks. Once whole image classification was achieved, research efforts focused at the task of multiple object detection and localisation, such as You Only Look Once (YOLO) [21].

Of course, the task of training these ever-growing networks with more and more training parameters placed a huge requirement for an ever greater number of training examples. With this, a number of public image datasets were made available, providing annotations for many millions of images [8].

Fig. 1. Our system can be used for the cleaning or correcting of DL/ML image datasets, by flagging unlikely objects. This Figure presents a selection of images from the Microsoft COCO dataset [18] with objects that our system identified as being anomalous given the context of the other objects detected. In the top-left the presence of a surfboard on the wall given that it is surrounded by kitchen objects was flagged to be anomalous. On the right, we have an image from the output of the YOLO neural network, where a horse was detected (without the understanding of it being a painting) when all the rest of the objects are typically found indoors, and according to the context, this was strange. Finally, in the bottom left, we have the annotations provided by the COCO dataset. Our system suggested both that a bicycle annotated outside the window was strange given the indoors nature of the rest of the image, and that there should probably be some chairs in this scene, where instead it was annotated using a sofa.

These datasets have powered the training of these huge networks, but it is not without its problems. Annotating datasets using object detection networks, trained from other datasets, leads to the amplification of errors. Therefore, the image datasets must be hand-annotated, and with the current architectures datasets must be tailored for a specific domain, as for example the prior on the distribution of objects in different environments can not be easily changed for the already-trained network. A tragic example of how a wrong choice of prior dataset distribution can cause a system failure is the Uber autonomous car fatality, where the dataset used to train the pedestrian detector was biased towards humans at crosswalks [5].

Even with software aimed at streamlining the task of annotating datasets [10], the task is still intensely tedious, repetitive, and time-consuming. This inevitably leads to errors creeping into the dataset, even when the annotations are also checked by other humans. This can be in the form of inconsistent annotating across different people, such as edge cases where different people label the object as different classes, to whether they choose to annotate very small objects in the background of an image or not. These errors are then manifested in the trained networks that learn from bad data. They may then be then be biased to make the same kind of mistake the annotators made, whether by rejecting a good detection, or misclassifying.

Therefore tools that are better able to understand the images, for example, that can look for bad annotations in a dataset are of immense importance. The authors of [31] have shown that understanding beyond the individual parts of an image, i.e. awareness of the global context, is highly beneficial. However, the most popular state-of-the-art detectors, such as YOLO, do not make use of this. Instead, they work by looking for regions of interest, and then classifying them separately. Thus, there is no interplay between these classification results within the same image. For human vision, context is a strong cue – people can detect objects that are expected to be seen in a particular context even if they are largely occluded, poorly illuminated or anomalously-shaped, see Fig. 1. However, a context-unaware method might reject a detection simply because poor illumination or occlusion causes the detection probability to fall below a fixed threshold. As shown in [31] this has a potentially negative effect on the overall accuracy.

In last few years there has been quite some research into how to incorporate context inherently into the architecture of these object detectors [2,6,15, 20,26,34]. All these new architectures present a unique approach to context modelling ranging from applying spatial Recurrent Neural Networks (RNNs) over the whole image [2] to modelling objects and relationships between them as graphs [20]. The context is then learned from the training dataset together with the object detection and further augments the learning process. We present a simpler approach, which is easier to train. It presents minimal overhead to the standard object detection pipelines, and its separation from the detection engine results in the ability to indicate the cause of the anomaly, i.e. the result is explainable. In this paper, we will address how contextual information can be added to traditional detectors through anomaly detection, and assess to what extent it is or is not beneficial.

2.2 Autoencoders and Their Use for Anomaly Detection

One of the first efforts to use a network with a layer smaller than its input (bottleneck) to learn compact representations of input data goes to 1991 [16]. This approach is based on creating a network capable of learning a mapping for nonlinear component analysis, which reduces the dimensionality of presented data while preserving their most significant features. The result is a general, versatile method that learns a generalised representation of the inputs.

It has been shown that two-layer bottleneck architectures are equivalent to standard Principal Component Analysis (PCA) when trained with a squared error loss criteria [4]. Nowadays applications of this architecture range from denoising by extraction of important features from images [29] or recorded speech [12] to anomaly detection through dimensionality reduction of general data [23].

Autoencoders (AE), as they became known, are nowadays an interesting but niche form of neural networks that can learn by themselves how to perform lossy compression on the inputs, no matter the type of data they are given. During the training phase, they would learn this compression-by-generalisation by training through comparison of the network inputs against its outputs generated by the hidden 'bottleneck' layer. When the network had been sufficiently trained, it could be split in half at the bottleneck, with the first half becoming the encoder, responsible for data compression, and the second half becoming the decoder, used for recreating the original data from the encoded form. While being interesting for learning how to encode any kind of data, 'vanilla' autoencoders did not reach widespread use for a number of reasons; the compression ratio is poor, they require time to learn for every data kind used, and the relationship between datasets and the optimal configuration of the network is not obvious.

However, one of their popular applications is anomaly detection. For example, [22] showed how the network architecture can be used for anomaly detection in telemetry data. Furthermore, others [33,35] have extended the technique to deep-autoencoders. Their research has shown how deep autoencoders can further boost the performance of autoencoder anomaly detection by capturing the characteristics of the underlying processes that generate the data. Another related architecture is the variational autoencoder (VAE) [1], which looks at the use of reconstruction probabilities rather than simply minimising loss.

The way these networks work is that essentially they learn a generalised distribution of the data. Once trained, the inputs to the network can be compared to its outputs to locate the parts of the input data that the network failed to recreate. The inability to reproduce a particular input implies that its presence in the training data is rare, i.e. that the input is anomalous.

In this paper, we will look at combining the problems of the context in deep object detectors with the use of autoencoders to spot anomalies. We hypothesise that through anomaly detection, the autoencoders can quantify the consistency of scene context, which can then be used to detect wrong classifications as well as to refine the output of the object detectors.

3 Method

This section concerns the use of autoencoders for anomaly detection itself. After the problem definition, we describe the training process and the design choices influencing the parameters used in the autoencoders.

The core idea we are elaborating upon is that the autoencoders are trained to reproduce their inputs with a constrained intermediate representation, which

forces them to model the underlying processes that generate the input data. We argue that in the case of object detection, the underlying process that generates the visible classes relates to the scene context. We hypothesise that training dataset classes exhibit strong correlation in their occurence within similar scenes represented by the individual images. Therefore, one would expect that the images would often contain similar combinations of objects typical for common scenes, and that the autoencoders would be able to learn these frequent combinations. Subsequently, the autoencoders should be able to learn if an object is missing or redundant in an observed combination. To evaluate this hypothesis, we utilise the Microsoft COCO dataset, which covers images of common objects in their natural context [18].

3.1 Conventions

The following convention will be used for the rest of the method description:

1. Input vector—Given the architecture of simple linear autoencoders, their input is a list of numbers which we call the input vector. We use each position of this fixed-length vector to encode the presence of an object class. The class-position map can be arbitrary, we use the ordering induced by COCO dataset class ids.
2. Output vector—The output of the autoencoder is of the same dimension and ordering as the input.
3. Annotation—The COCO dataset we used for both training and evaluation of our method comes in the form of images and segmentation masks of annotated objects visible in those images. For our purposes, we only use the class labels attached to individual boxes, which we refer to as the annotation. We use the term annotations to indicate the set of annotations present in one image, as well as the set of all annotations present in a dataset grouped by the image they belong to.

3.2 Design of Autoencoder

We created a model with one hidden layer fully connected to both the input and output layers. The size of the hidden layer was a tested parameter with regards to anomaly detection performance, and the size of the input-output layers was equal to the number of dataset classes. To investigate the method properly we tried several different sizes of the encoding layer, which effectively determines the relationship between how simplified and therefore how generalised the inputs become in the inner, encoded representation.

The autoencoders were trained to reproduce a set of input vectors, where for every set of annotations A from the set of all sets of annotations the input vector was created as

$$i[class] = \begin{cases} 1 & class \in A \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

As the autoencoder size is linked to the number of classes in the output vector of the object detector, it is extremely small in comparison to the object detector. For example, when paired with YOLO using the COCO image dataset, the autoencoder is comprised of an input layer of 80 neurons, then a hidden layer of in the region of 10 neurons, then an output layer of 80 again. Therefore with less than 200 neurons, the run-time performance is extremely fast on low-end modern hardware, in the sub-millisecond range to perform inference on a GPU.

3.3 Anomaly Detection

For the task of detecting anomalies using such a trained network, multiple approaches can be employed. The universal autoencoder-based anomaly detecting methods usually use a certain measure of anomality of a data sample. This measure is then computed for the data at hand, and based on its value the sample is either flagged as an anomaly or not. Such a measure can be the mean squared sum of errors between the input and output vector—the reconstruction error—or for example the reconstruction probability, employed by Variational AutoEncoders (VAEs) [1]. The reason why we can't adopt the same approach is due to a different formalisation of the task. These methods usually consider anomalies to be generated by a completely different process than the normal data, so their use case is then to filter anomalous inputs out of the incoming data stream.

In contrast, we know that all of our data was created by the same process and it is only the context that makes part of the input vector anomalous or not. We therefore emphasise the ability to detect anomalous components of the input vector, which not only provide us with the information in the form of an anomality score, but also allows for an explanation of the results. In certain scenarios, it can enable us to correct mistakes in the data, if that is the expected cause of anomalies.

The way we trained our autoencoders to reproduce vectors representing object class presence in some plausible scene with limited resources, leads to a behaviour where presence or absence of a given feature in the input vector affects with varying strength all of the features of the output vector. In other words when the context—the present features—strongly suggests the presence of another, currently absent object, the output of the network will be higher for that missing object. Furthermore, this also applies the other way round, when the context disagrees with the presence of some object, its value will be lowered in the output vector.

Although this gives us the general idea about how to read the output of the AE, it is yet to be determined how large must be the difference between the value for a given feature in the input and the output vector, so that it is reasonable to flag the feature value as anomalous. We do not investigate this problem much as we think it is a large problem worthy of its own investigation. Instead we limit ourselves to consider only a static thresholding value, which we set accordingly to the task at hand, as can be seen in the section about experiments.

We consider something as an anomaly, if the output of the autoencoder for this class differed from the input by more than 0.5.

4 Experiments

We designed a set of experiments to test the performance of our method in several scenarios, corresponding to two of the presented potential applications—the verification of potentialy false annotations and boosting of standard object detection networks.

First of all, we test the ability of the network to detect anomalies—both positive and negative. Then we present an experiment to prove the applicability of our method to the problem of adding contextual awareness to deep-learning object detectors to boost their performance.

One of the problems with quantifying anomalous objects in a scene is that standard clusters of objects often seen together may only semi-regularly include certain classes. For example, in a kitchen scene, the presence of cutlery is not always expected as it may be tidied away and out of view. Therefore, the system should not identify the positive or negative identifications of these as anomalous. When coming up with a method for evaluation, we had to take this into account.

4.1 Training of the Autoencoders

Before the actual experiments we must discus how the anomaly detecting autoencoders were trained.

As a training set for the autoencoder, we processed the annotations from the COCO dataset in a way where we created an input vector for every image annotation, set A, according to Eq. 1. We excluded the object class *person* from all annotations, because as we found out during preliminary experiments, it is vastly over-represented in the dataset and thus is not reasonably represented for anomaly analysis. For the breakdown of the dataset, we used the COCO validation set to determine how good our methods were performing, and we broke the training set for autoencoder into two sets for training and validation, in a ratio of 10:1. We trained all models for 120 epochs with the binary cross entropy loss function [7] and using the Adadelta optimizer [30].

With a careful study of the training graphs, we determined what shape of autoencoder would work best for us. Counterintuitively, the goal of training isn't to minimise loss; that would defeat the point of generalisation and would always favour a larger intermediate layer. Instead, some loss is what is helping us generalise. In validation some loss will always be present as there will be less common groups of objects in the evaluation dataset than in the training dataset. In fact if one considers the total number of possible subsets, it is a certainty. The important point here though is that the less likely a cluster is, the more likely it is less common because it contains some anomaly.

After taking this into consideration, as can be seen in Fig. 2, we decided that the best point that simplifies the model enough to generalise, but also still has relatively low loss, lies somewhere around the range of 15–25 neurons. Below this number, the loss increases massively. Above this number, each additional neuron has little affect on the end model. Therefore we ended up using a model for our experiments with 20 neurons, in the centre of the sweet-spot.

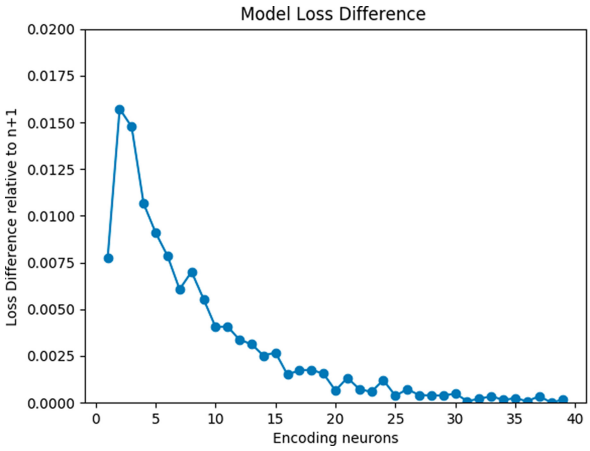

Fig. 2. Here we can see how much loss each model gained relative when it's final neuron was removed. On the X-axis is the number of neurons in the model, and on the Y-axis how much additional loss the model gained compared to a model one neuron larger. We can see that removing neurons from very large intermediate layers has little effect on the loss. This is relatively steady down to about the range of 15–25 neurons, where the removal of each neuron begins to introduce more and more loss. Below 15 neurons, removing each extra neuron adds significant extra loss to the model, before dropping for one neuron as the loss is already very high.

4.2 Anomaly Detection

Our first two experiments focus on assessing the capabilities of the autoencoder networks to detect positive and negative anomalies. The motivation for this experiment is the application to verification of automatically or otherwise annotated datasets. If one has an idea about the spatial distribution of objects that are being annotated in some images, then our method would be able to detect these annotations that violate such a distribution. The actual performance is presented as a result of the two following experiments, which consist of artificially adding and removing object classes to a set of manually annotated images

and seeing how well our system can detect these. We trained several autoencoders that differed in the number of neurons in the encoding layer, effectively setting the compression ratio.

For these experiments, we used the images and annotations from the Microsoft COCO dataset [19], which are already split into training and validation sets. As mentioned before, this data was preprocessed to remove annotations of 'people', and we also imposed some limits regarding the minimum number of annotations required in an image, which we discuss later, as looking for anomalies in images with single classes makes little sense.

Positive Anomalies. In this experiment, we wanted to investigate how well autoencoders can detect anomalous classes artificially added to images. We took all the images with at least three different annotations present, and then added an extra class that was not already present. We then looked at how well our system was able to flag this added class as being anomalous.

Out of all 80 possible classes, we would select randomly 10, which we would then add one by one to all the images, where this class was not already present in the annotations. Adding an object consists of setting to 1 the respective value in the autoencoders input vector. Then, if the output of the autoencoder for this class differed from the input by more than 0.5, we would classify it as a correct anomaly detection. Because of the random nature of the experiment, we re-ran it for 20 times to get more statistically significant results.

Negative Anomalies. Regarding the negative anomalies, we designed a similar experiment, where instead of adding random object classes we instead removed classes randomly from those that appeared in a given image.

We wanted to show the autoencoders detecting the objects we removed from the scene, so as a part of the preprocessing, we filtered out the images the autoencoders considered anomalous in the dataset itself. The images with less than five annotations were filtered as well, so that after removal of an object class, there would be at least four classes left for the autoencoder to understand the context.

For every remaining image, randomly five of its annotations were selected and one at a time was removed from the set, so their respective position set to 0 in the input vector. If the output of the autoencoder differed at the corresponding position of this class from the input by more than 0.25, we would classify it as a correct negative anomaly detection. The reason for a threshold lower for negative anomalies is that a presence of single object is highly fluctuating in real-world environments, so we settle with a lower confidence—the group of objects, after we removed one class, might actually be perfectly normal.

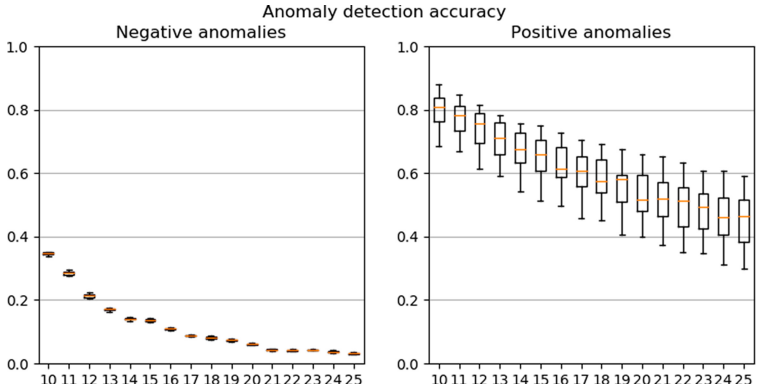

Fig. 3. Results of the first two experiments to test the ability of the autoencoder to spot anomalies, artificially introduced to a scene. On the left we removed object classes from the scene, on the right we added them. Every box represents a model with a different number of neurons in the encoding layer.

Results. All the results are depicted in Fig. 3.

Detecting negative anomalies is clearly a much harder problem than detecting the positive ones, as discussed already in the section about the experiment. These results are predictably worse, because removing objects from the scene might not result in an anomalous scene, but only another valid setting, by the nature of the negative anomalies.

The best accuracy in the proposed experiment was achieved by the autoencoder with the smallest number of neurons and was around 35%. That the more complex models are able to better capture the fluctuating nature of single objects and removing one from a valid set might have only created another valid set, probably led to decrease in accuracy with the number of encoding neurons. However, we expect the negative anomaly detection to perform better in real applications with more complex models, because they should exhibit a lower false-positive rate.

For positive anomalies, we show that our method is very good at detecting these, with an accuracy rate mostly between 75 and 85% for the best model. Also, as we can see from the graphs, the lower the number of neurons in the encoding layer, the better the results.

This last result is due to the fact that more complex models with more neurons are then likely to pick up anomalies during their training from the training dataset. That in turn is caused by anomality not being an absolute property, but rather a context-related measure. And last, since we added random classes, some of them had to be generally less common objects as well, which small encoders might not have reflected.

4.3 Boosting Object Classification by Context-Awareness

In this experiment, we wanted to show how well our method is able to dynamically asses the output of a neural network regarding it's confidence in individual detections. We used the pretrained YOLOv3 network for object detections, which we then run on the images in the COCO dataset, against which it was trained.

Using this network one normally has to set a thresholding parameter, which determines at what level of confidence one will accept the output of the network as a valid object detection. Our intention was to use the negative anomaly detection to strengthen the confidence in objects that were detected below this threshold, but given the context of the scene, they are likely to be observed.

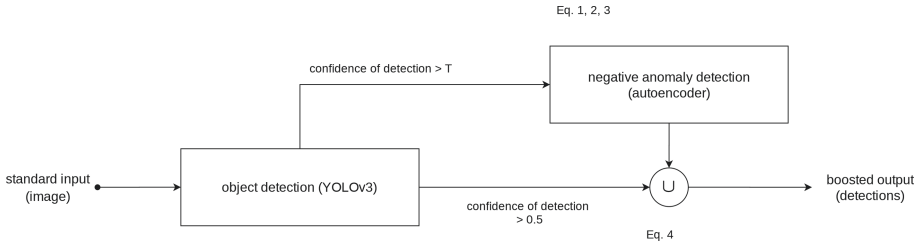

Fig. 4. Diagram describing the application of anomaly detection to boosting of the object detection. First, the input image is analysed by an object detector. The negative anomalies, that are detected are then combined with the standard output of the detector, for boosting the accuracy of detections.

The whole experiment was set up to examine the difference in accuracy between a standard object detecting network and one enhanced with an autoencoder. We needed to tailor the experiment to account for two basic properties of the situation. We know that with a high threshold the network performs reasonably well [21], but the threshold also determines how many objects will be detected, so it can be effectively used for setting the accuracy versus extracted information ratio. Second, we are limited by the ground truth available, basically the annotated image dataset.

The general idea of the experiment is to count how many times the network was wrong on the validation dataset against which it was trained. Denoting the decision threshold T, we compare how well the YOLOv3 model performs with respect to T, to a case where we set the threshold to 0.5 as a reasonable default and let the autoencoder detect other objects in the image, that should be included as well, but their confidence lies in the interval $<T, 0.5>$. The process is depicted in Fig. 4.

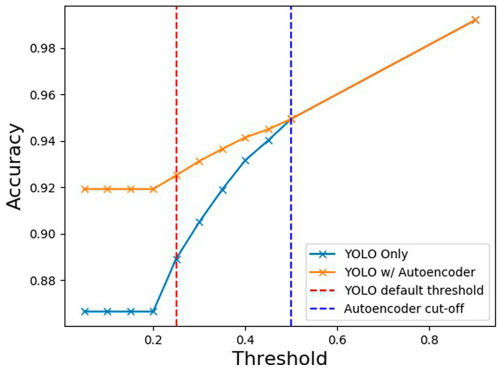

Fig. 5. Results of the second experiment. Detections above the right threshold are classed as absolute, and therefore not changed. Similarly, below the left threshold are classed as so weak not to warrant investigation. However, between these ranges, we use the autoencoder to identify if the detection, despite just missing out on the detection threshold, makes a lot of contextual sense. If so, we count it as a detection. One can see that this results in a significant boost to the accuracy of YOLO object detection when this method is employed

Results. Specifically, the procedure is as follows:

1. We analyse the image in the dataset with YOLOv3 network, which results in a set of detections Det and we create a set
$$D = \{d \,|\, conf(d) > T, d \in Det\}, \tag{2}$$
where $conf(d)$ is the level of confidence the network has in particular detection.
2. We calculate the accuracy, comparing the set D with the set of annotations we have from COCO dataset. This is the accuracy of standard YOLO itself. We only compare whether the class was present in annotations or not - we don't compare the number of detections or bounding box precision.
3. Next, we create an input vector i for the autoencoder:
$$i[c] = max\{conf(d) \,|\, class(d) = c, d \in D\}, \tag{3}$$
where c is the particular object class, $class(d)$ is the class of given detection d and in the case that no object of a given class was detected, we set the input to 0.
4. The output vector o of the autoencoder is then transformed into a set
$$O = \{c \,|\, o[c] > 0.1, c \in C\} \cup \{class(d) \,|\, conf(d) > 0.5, d \in Det\}, \tag{4}$$
where C is the set of all classes and 0.1 was selected as a reasonable threshold for autoencoder output from our experience with experiments for anomaly detections. We then compare the set O with the set of annotations of given image the same way we did with the simple YOLO output.

Results of this experiment are depicted in Fig. 5. We show that on the interval $<0.2, 0.5>$ our method clearly outperforms the standard YOLOv3 network. At the level of 0.2 by almost 5%. The reason both curves converge to the same accuracy value beyond a threshold of 0.5 is that we always include everything with detection confidence above this value as a given, a certain detection used for gathering the context.

5 Conclusion

We investigated the use of autoencoder-based anomaly detections for the task of image-based object detection. The core idea of our approach is to use autoencoders to capture the contextual information present in images, which affects the joint occurrence of the detected object classes. Simply put, we use the autoencoders to answer two questions: 'What does not belong here?' and 'What is missing here?'. We demonstrate that autoencoders can quantify the aforementioned questions and use this quantification to detect potentially incorrect detections and annotations in the training data. We also show that knowledge of the context can boost the precision of state-of-the-art object detection frameworks.

Our work shows that decoupling the context-aware autoencoder from the context-unaware object detector results in a system that can be re-trained for a particular object distribution in short time while being computationally inexpensive. In the experiments performed, we demonstrate that using a ∼200-neuron-sized autoencoder can not only significantly improve performance of the YOLOv3 detector, but also detect errors artificially injected into the training set. Thus, the method presented addresses two prominent issues troubling practical deployments of machine learning methods – deployment in scenarios with different distributions of the detected classes and dataset annotation quality.

In the future we would like to extend our work in several directions. We would like to investigate different, more complicated linear or deep autoencoders capable of capturing not only co-occurrence, but also the spatial and temporal relations between the detected classes. We will compare the decoupled approach to the newest context-aware deep neural networks, that incorporate different levels of contextual information to achieve better performance. Finally, we believe that the ability to spot potential errors in the detector's output is an important step towards introspective, explainable learning methods and we will pursue their deployment in systems capable of long-term autonomous operation.

Acknoledgement. The authors acknowledge the support of the Czech Science Foundation project "Towards long-term autonomy through introduction of the temporal domain into spatial representations used in robotics "20-27034J". The calculations were performed using computational resources provided by the OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

References

1. An, J., Cho, S.: Variational autoencoder based anomaly detection using reconstruction probability. Spec. Lect. IE **2**(1), 1–18 (2015)
2. Bell, S., Lawrence Zitnick, C., Bala, K., Girshick, R.: Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2874–2883 (2016)
3. Biederman, I.: Recognition-by-components: a theory of human image understanding. Psychol. Rev. **94**(2), 115 (1987)
4. Bishop, C.M.: Pattern Recognition and Machine Learning. Information Science and Statistics. Springer, New York (2006)
5. Board, T.N.T.S.: Collision between vehicle controlled by developmental automated driving system and pedestrian, Tempe, Arizona, p. 78, 18 March 2018
6. Chen, X., Gupta, A.: Spatial memory for context reasoning in object detection. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4086–4096 (2017)
7. Creswell, A., Arulkumaran, K., Bharath, A.A.: On denoising autoencoders trained to minimise binary cross-entropy. arXiv:1708.08487 [cs, stat] October 2017
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
9. Divvala, S.K., Hoiem, D., Hays, J.H., Efros, A.A., Hebert, M.: An empirical study of context in object detection. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1271–1278. IEEE (2009)
10. Dutta, A., Zisserman, A.: The VGG image annotator (via). arXiv preprint arXiv:1904.10699 (2019)
11. Galleguillos, C., Belongie, S.: Context based object categorization: a critical survey. Comput. Vis. Image Understand. **114**(6), 712–722 (2010)
12. Gehring, J., Miao, Y., Metze, F., Waibel, A.: Extracting deep bottleneck features using stacked auto-encoders. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 3377–3381, May 2013. DOIurl10.1109/ICASSP.2013.6638284. ISSN: 2379-190X
13. George Broughton, J.B.: Auto image anomaly (2020). https://github.com/broughtong/Auto-Image-Anomaly
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3588–3597 (2018)
16. Kramer, M.A.: Nonlinear principal component analysis using autoassociative neural networks. AIChE J. **37**(2), 233–243 (1991). https://doi.org/10.1002/aic.690370209. https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690370209
17. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
18. Lin, T.Y., et al.: Microsoft COCO: common objects in context. arXiv:1405.0312 [cs] February 2015

19. Lin, T.-Y.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
20. Liu, Y., Wang, R., Shan, S., Chen, X.: Structure inference net: object detection using scene-level context and instance-level relationships. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6985–6994 (2018)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
22. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, p. 4. ACM (2014)
23. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis - MLSDA 2014, Gold Coast, Australia QLD, Australia, pp. 4–11. ACM Press (2014). https://doi.org/10.1145/2689746.2689747. http://dl.acm.org/citation.cfm?doid=2689746.2689747
24. Santos, J.M., Krajník, T., Fentanes, J.P., Duckett, T.: Lifelong information-driven exploration to complete and refine 4-d spatio-temporal maps. IEEE Robot. Autom. Lett. **1**(2), 684–691 (2016). https://doi.org/10.1109/LRA.2016.2516594
25. Santos, J.M., Krajník, T., Duckett, T.: Spatio-temporal exploration strategies for long-term autonomy of mobile robots. Robot. Auton. Syst. **88**(C), 116–126 (2017)
26. Shrivastava, A., Gupta, A.: Contextual priming and feedback for faster R-CNN. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 330–348. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_20
27. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
28. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
29. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: Proceedings of the 25th International Conference on Machine Learning - ICML 2008, Helsinki, Finland, pp. 1096–1103. ACM Press (2008). https://doi.org/10.1145/1390156.1390294. http://portal.acm.org/citation.cfm?doid=1390156.1390294
30. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv:1212.5701 [cs] December 2012
31. Zhao, R., Ouyang, W., Li, H., Wang, X.: Saliency detection by multi-context deep learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1265–1274 (2015)
32. Zhou, C., Paffenroth, R.C.: Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD 2017, Halifax, NS, Canada, pp. 665–674. ACM Press (2017). https://doi.org/10.1145/3097983.3098052. http://dl.acm.org/citation.cfm?doid=3097983.3098052
33. Zhou, C., Paffenroth, R.C.: Anomaly detection with robust deep autoencoders. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 665–674. ACM (2017)

34. Zhu, Y., Urtasun, R., Salakhutdinov, R., Fidler, S.: segDeepM: exploiting segmentation and context in deep neural networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4703–4711 (2015)
35. Zong, B., et al.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection (2018)