Comănac Dragoș-Mihail

Public transport access detection

AMS Mathematical Subject Classification - 68T45 Machine vision and scene understanding

ACM Computing Reviews Categories and Subject Descriptors - I.4 IMAGE PROCESSING AND COMPUTER VISION
- I.4.6. Segmentation - Edge and feature detection

Contents

1. You Only Look Once: Unified, Real-Time Object Detection

- https://arxiv.org/abs/1506.02640

- This article has a total of 39 references, listed at the end of the article and numerated.

- First the author/authors are mentioned, then the title and optionally the conference or journal and the pages of the reference inside the article or book. At the end the year is mentioned.

- Each article is referenced throughout the article by the number assigned at the end of the article in the following way “[number]”

- Also, there are around 76 citations throughout the article.

- This article is relevant to the proposed theme because it presents the YOLO detection system, which can perform object detection (object localization and classification) in real time.

Abstract

- short description of the system and how it compares to other methods, together with performance metrics

1 Introduction

- illustrate the downsides of other existing detection systems and presents the advantages that make YOLO run faster.

- performance metrics are presented.

- disadvantages of YOLO are also presented.

2 Unified Detection

- provides explanations of the output of the network and how the input (image) is processed.

2.1 Network Design

- the structure and type of the network (inspired by the GoogleLeNet) is presented.

- a fast version is presented

- a figure explaining the architecture is present

2.2 Training

- process of training is described, which starts with pretraining

2.3 Inference

- describes one pass through the network

2.4 Limitations of YOLO

- explains on how small and nearby objects can be missed

- another limitation is that bounding box error is treated the same in small boxes and large boxes

3 Comparison to Other Detection Systems

- comparison to: Deformable parts models, R-CNN, Fast and Faster R-CNN, Deep MultiBox, OverFeat, MultiGrasp

4 Experiments

- different experiments are conducted on Pascal VOC 2007 dataset

4.1 Comparison to Other Real-Time Systems

- Deformable parts models, R-CNN, Fast and Faster R-CNN and different versions of YOLO are compared by mean average precision and frames per second

4.2 VOC 2007 Error Analysis

- deeper analysis of the errors of YOLO and R-CNN

4.3 Combining Fast R-CNN and YOLO

- combination of R-CNN and YOLO is analyzed

4.4 VOC 2012 Results

- performance on VOC 2012 dataset is presented

4.5 Generalizability: Person Detection in Artwork

- YOLO is compared to other systems on Picasso Dataset and the People-Art Dataset (person detection in artwork)

- YOLO has a better performance on these datasets

5 Real-Time Detection In The Wild

- it is shown that YOLO performs well also on a live camera

6 Conclusion

- short recap of YOLO advantages

2. SSD: Single Shot MultiBox Detector

- https://arxiv.org/abs/1512.0232

- This article has a total of 26 references, listed at the end of the article and numerated.

- First the author/authors are mentioned, then the title and optionally the conference or journal and the pages of the reference inside the article or book. At the end the year is mentioned and the page in the article where the cited article is referenced.

- Each article is referenced throughout the article by the number assigned at the end of the article in the following way "[number]".

- Also, there are around 62 citations throughout the article.

- This article is relevant to the proposed theme because it presents the SSD detection system, which can perform object detection (object localization and classification) in real time. Also, it as an alternative to YOLO.

Abstract

- description of how SSD eliminates object proposal generation, and all computation is encapsulated in a single pass through the network

- performance metrics are also mentioned

1 Introduction

- short description of the state of the art and why it is not suited for real time object detection

- speed/accuracy tradeoff

- description of how the system is better than YOLO

- summary of contributions

2 The Single Shot Detector (SSD)

2.1 Model

- description of SSD framework for detection

2.2 Training

- training process is described: Matching strategy, Training objective, Choosing scales and aspect rations for default boxes, Hard negative mining, Data augmentation

3 Experimental Results

- the base network is VGG16 and additional parameter details are provided

3.1 PASCAL VOC2007

- on this dataset, SSD is compared to Fast and Faster R-CNN

3.2 Model analysis

- experiments to assess the impact on performance

- data augmentation is essential, more default box shapes are better, multiple output layers at different resolutions is better

3.3 PASCAL VOC2012

- similar tests are conducted for PASCAL VOC2012

3.4 COCO

- similar tests are conducted on COCO

3.5 Preliminary ILSVRC results

- similar tests are conducted on ILSVRC2014 DET

3.6 Data Augmentation for small Object Accuracy

- SSD struggles with small objects, therefore data augmentation improves performance

3.7 Inference Time

- time statistics are shown for one pass through the network

4 Related Work

- evolution of object detection

5 Conclusions

- recap of key features of SSD

6 Acknowledgment

- details on how to project started

3. Diagnosing Error in Object Detectors

- https://dhoiem.web.engr.illinois.edu/publications/eccv2012_detanalysis_derek.pdf

- This article has a total of 30 references, listed at the end of the article and numerated.

- First the author/authors are mentioned, then the title and optionally the conference or journal and the pages of the reference inside the article or book. At the end the year is mentioned and the page in the article where the cited article is referenced.

- Each article is referenced throughout the article by the number assigned at the end of the article in the following way "[number] references "[number]", "[number, number,…]", "[number-number]""

- Also, there are around 25 citations throughout the article.

- This article is relevant to the proposed theme because it analyzes different aspects that can impact object detection performance.

Abstract

- different aspects which influence error are enumerated

1 Introduction

- many papers use accuracy or average precision in comparing different methods, but they do not explain why one method is better than another

- the main contribution are analysis tools, which perform investigation of object detectors

- experiments are based on the PASCAL VOC 2007 dataset

2 Analysis of False Positives

- false positives are a major error

- one type of false positive is localization error, and it occurs when the overlap (IOU) between the predicted box and the ground truth box is less than 0.5

- other types are duplicate detections, confusion with similar or dissimilar objects or with background

3 False Negatives and Impact of Object Characteristics

- object characteristics are added to the PASCAL VOC dataset images

3.1 Definitions of Objects Characteristics

- different object characteristics are described such as: object size, aspect ratio, occlusion, truncation, visibility of parts and viewpoint

3.2 Normalized Precision Measure

- in calculating the precision, instead of the number of objects in a class, a normalized number is used N=0.15*total number of images, so that the comparison between classes is relevant

3.3 Analysis

- an in-depth analysis is performed on how the detectors behave on the dataset

4 Conclusion

4.1 Diagnosis

- recap of the causes of errors

4.2 Recommendations

- very small or very large objects are hard to detect

- localization or unusual views also impacts precision

4. Speed/accuracy trade-offs for modern convolutional object detectors

- https://arxiv.org/pdf/1611.10012.pdf

- This article has a total of 47 references, listed at the end of the article and numerated.

- First the author/authors are mentioned, then the title and optionally the conference or journal and the pages of the reference inside the article or book. At the end the year is mentioned and the page in the article where the cited article is referenced. Other references are github links.

- Also, there are around 119 citations throughout the article.

- Each article is referenced throughout the article by the number assigned at the end of the article in the following way "[number] references "[number]", "[number, number,…]", "[number-number]""

- This article is relevant to the proposed theme because it provides a guide for choosing a detection architecture that fits the speed, memory, and accuracy requirements.

Abstract

- description of the goals of this article and the systems that are to be compared in achieving those goals

1 Introduction

- in-depth analysis of the resources trade-offs is missing from different articles such as YOLO and SSD

- contributions are enumerated such as: survey of modern detection systems, own implementation of those systems with different variations

2 Meta-architectures

- the use of anchors in Fast R-CNN and MultiBox are described

2.1 Meta-architectures

2.1.1 Single Shot Detector (SSD)

- architectures that use a single feed-forward convolutional network to directly predict classes and anchor offsets

2.1.2 Faster R-CNN

- detection happens in two stages: region proposals and classification

2.2 R-FCN

- two stage detector

3 Experimental setup

- detection platform is created using Tensorflow so that swapping feature extractors is easier

3.1 Architectural configuration

3.1.1 Feature extractors

- six feature extractors are considered: VGG-16, Resnet-101, Inception v2, v3, Inception Resnet v2, MobileNet

3.1.2 Number of proposals

- number of box proposal varies between 10 and 300

3.1.3 Output stride settings for Resnet and Inception Resnet

- stride of 8 improves the mAP but increases the running time

3.2 Loss function configuration

3.2.1 Matching

- bipartite or argmax

3.2.2 Box encoding

- φ(ba; a) = [10 · xc wa , 10· yc ha , 5·log w, 5·log h]

3.2.3 Location loss

- smooth L1 (or Huber) loss is used in all experiments

3.3 Input size configuration

- Faster R-CNN and R-FCN scale the input images to M pixels on the shorter edge and SSD resizes the images to MxM pixels

- M=600 and M=300 are tested

3.4 Training and hyperparameter tunning

- models are trained end to end

3.5 Benchmarking procedure

- a machine with 32GB RAM, Intel Xeon E5-1650 v2 processor and an Nvidia GeForce GTX Titan X GPU card is used

3.6 Model details

- details of each model using different feature extractors are described

3.6.1 Faster R-CNN

3.6.2 R-FCN

3.6.3 SSD

4 Results

- data resulted in the experiments is analyzed

4.1 Analyses

4.1.1 Accuracy vs time

- R-FCN and SSD models are faster but Faster R-CNN is more accurate and can be just as fast is the number of regions is smaller

4.1.2 Critical points on the optimality frontier

- SSD models with Inception v2 and Mobilenet feature extractors are most accurate of the fastest models, with Mobilenet being twice as fast but slightly worse in accuracy

- the “sweet spot” is R-FCN with Resnet or R-CNN with Resnet and 50 proposals

4.1.3 The effect of the feature extractor

- Faster R-CNN and R-FCN rely more on the feature extractor’s classification accuracy than SSD

4.1.4 The effect of object size

- performance is better on larger objects

- SSD performs worse on small objects

4.1.5 The effect of image size

- a decrease in resolution reduces accuracy and inference time

4.1.6 The effect of the number of proposals

- number of proposed boxes can be reduced without affecting mAP too much

4.1.7 FLOPs analysis

- each model has a different average ration of flops to observed running time in milliseconds

4.1.8 Memory analysis

- total usage is measured

4.1.9 Good localization at .75 IOU means good localization at all IOU thresholds

4.2 State-of-the-art detection on COCO

- five models of Faster R-CNN are selected, based on Resnet and Inception Resnet

4.3 Example detections

- visualization of detections on images from the COCO dataset

5 Results

- recap of what is presented in the article

Acknowledgements

5. EDLines: A real-time line segment detector with a false detection control

- https://www.researchgate.net/publication/220644982_EDLines_A_real-time_line_segment_detector_with_a_false_detection_control

- There are around 44 references are mentioned at the end, and they are not enumerated.

- First the author/authors are mentioned, then the title and optionally the conference or journal and the pages of the reference inside the article or book. Publication year and links also appear.

- Throughout the article the references are highlighted between round brackets. The authors or the title are mentioned.

- This article is relevant to the proposed theme because it presents a line detector that needs no parameter tuning. It could be used as a final stage for detecting the bus doors.

Abstract

- short description of the algorithm and how much faster it is compared with the fastest line detector at the time

1 Introduction

- examples of applications that use line detection
- different approaches are discussed
- summary of the three stages that comprise EDLines

2 Edge detection by Edge Drawing

- the output is a set of connected edge segments, compared to the binary edge image generated by other edge detectors

3 Line segment extraction

- the Least Squares Line Fitting Method is used to fit pixels in lines

4 Line validation

- the line validation methos is based on the Helmholtz principle, meaning that object are seen as outliers to the background
- long lines are better detected
- faster approach is to choose lines that are longer than a certain threshold

5 Internal parameters

- gradient magnitude and direction computation
- line validation parameters
- gradient threshold, anchor threshold and scan interval
- line fit parameters

6 Experiments

- anchor threshold is an empirical parameter
- different images are analyzed

7 Conclusion

- short recap and future work