Detection of cats in images

ANNA MARIA WALACH (s121540)
Technical University of Denmark
02238 Biometric Systems

The LaTeX `acmtrans` document style formats articles in the style of the ACM transactions. Users who have prepared their document with LaTeX can, with very little effort, produce camera-ready copy for these journals.

Categories and Subject Descriptors: D.2.7 [**Software Engineering**]: Distribution and Maintenance—*documentation*; H.4.0 [**Information Systems Applications**]: General

General Terms: Algorithms, Experimentation

Additional Key Words and Phrases: cat, detection, images

1. INTRODUCTION

There are more and more projects that uses live streaming and social media power to help and safe feral cats. Some of them are meant to control the population in the area [LangleyAnimalProtectionSociety 2015], other focus on raising awareness about feral cats, spaying and neutering importance [TinyKittens 2015] or just to increase changes of finding a new home for homeless cats and kittens [TheCritterRoom 2015].

People are also more interested in health and safety of their own pets. Lower prices of video technologies and smarthone popularity helped to create various apps for controlling your pet lifestyle. [ZilliansInc 2015] created an automatic feeder with cat identification system and connected it to the smartphone app (product can be seen in fig. 1. It allows owner to supervise the amount of dry food and water his cats are consuming everyday, to receive alarms in case of abnormalities and even spy on them while eating.

Fig. 1. CatFi product used by a cat.

The PiP [ThePetRecognitionCompany 2015] app is designated to help people find their lost cats and dogs. According to American Human Society, almost 3.5 million pets are lost each year and about 80% of dogs and 98% of cats are never reunited with their families. The authors of the smarthone app allows the owners to made profiles for their pets (including pictures) and in case the pet is missing, you launch the "alarm" on that pet, together with last seen location. If anyone finds a homeless animal, he may send the photo of it to the PiP app, which automatically try to match one of the lost pets with found pet.

Increasing number of pets' photographies also encouraged camera producers to

implement appropriate support for it in their devices. Right now, Fujifilm company introduced Auto Dog / Cat Detection function [FujiFilm 2015] in some of their cameras that allows to automatically detect pet's face on the image and put auto-focus on it.

1.1 Problem description

Almost all of the applications and initiatives described in previous section already use or would benefit from the cats detection and/or identification. In this research I would like to focus on topic of cat detection in images. The main interest is on the use case of detecting cats presence in shots from live cameras and streams. Such images are characterized by diversified, often bad lighting settings, with possible presence of another animals or human beings. I will perform a state-of-the-art survey and then perform a comparison of existing, available algorithms.

2. THE STATE OF THE ART

Reliable comparison of existing software for detecting cats in pictures is hard to performed. It is caused by a few factors:

— **black-box algorithms** - many algorithms, that claim to have high accuracy, are part of the hardware product [ZilliansInc 2015] or closed-source app [ThePetRecognitionCompany 2015]. This way it is impossible to make a theoretical comparison of the algorithms and usually significantly hampers possibility of performing experiments with test data, as the products' interfaces are not adjusted to bulk operations.

— **high diversity of use cases** - although there are a lot of studies focusing on cat face detection, usually the classification or testing environmental does not check robustness of the algorithm in typical real-time scenarios. The classification is e.g. made as cat vs. still life [Fleuret and Geman 2007; Akihiko Yamada et al. 2011] or as cat/dog breed detection [Omkar M. Parkhi et al. 2012].

— **different test sets** - even for algorithms with similar use cases, e.g. cat vs. still life, different test set are in use to estimate the accuracy of algorithm, which makes it significantly harder to perform direct comparisons. Usually articles use their own data sets [Omkar M. Parkhi et al. 2012; Fleuret and Geman 2007; Akihiko Yamada et al. 2011] and only some of them compare to standard data bases, like Pascal 2007 cats database [Omkar M. Parkhi et al. 2012].

As the main idea of this research is to find an algorithm that will allow for highly accurate detection of cat faces in the pictures, only articles presenting "cat vs. still life" classification will be taken under consideration.

2.1 Unsupervised image categorization

One of the more interesting approaches to this problem was presented in [Le et al. 2012]. Researchers from Google and Stanford university constructed large scale unsupervised learning tool. It was a 9-layered neural network build on a cluster that consisted of 1 000 machines with total of 16 000 cores. The network was trained with 10 million unlabelled images, all of them had the same size and local contrast normalization was applied on them. The model consisted of 1 bilion connections and it took 3 days to train the classifiers. Final results are very promising for

overall image categorization issue, because authors claims to have 15.8% accuracy for ImageNet dataset (22 000 object categories), which is a 70% improvement in comparison to the previous state of the art.

However, the most interesting part of [Le et al. 2012] for this research is that after performing face-detection experiment, authors tried to used trained classifiers to detect cats. The results are very promising - the random guess had 64.8% chance, while this algorithm was able to detect cats with 74.8% accuracy.

This algorithm and use of deep learning networks seems to be a promising technique for classification of images categories, however seemed to be lacking needed accuracy for binary classification (cat vs. not-cat).

2.2 Pet detection method for digital camera

Another interesting project was performed by [Akihiko Yamada et al. 2011]. They tried to construct a cat and dogs face detection mechanism for use in camera auto-focus and auto-exposure. They were not satisfied with current method, because camera has a very high constraints about calculation time and power consumption, were most of the studies focus on high recall and small false-detection. Authors decided to use four direction features as an input for the classifier. In order to speed-up the process, classification of an image window was hierarchical - if coarse classifier did not detect a sign of face in the window, it was automatically rejected and not send to more detailed and feature-rich fine classifier.

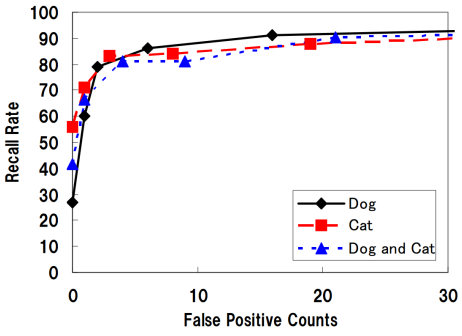

Fig. 2. ROC curves achieved as a result of [Akihiko Yamada et al. 2011].

The results are very promising. As described in fig. 2, with the average recall on 90%, the false positive count is around 20. Although the FPC may seem high (the sample size was 400, with 200 of non-animal pictures), it is still on user-acceptable level. What is more important, the processing time was only 245ms on ARM9 device with 166MHz core. It is impressive result, especially because this solution turned out to be also very robust to illumination changes. Authors measured how color and intensity fluctuation caused by illumination changes influence the recall rate. Intensity changes didn't cause the recall rate to drop under 80% even on very high intensity values. With 50% color fluctuation recall rate still oscillate between 80%-90%.

Although the results are not perfect (usually recall rate do not exceed 90%), short processing time allows to use this algorithm in real-time cases, with may be important case with live streaming observation. This solution can be used as a part of pet-identification algorithm, that informs appropriate person when the cat appears in the camera (e.g. if it is known it is pregnant and needs to be caught and checked) and in such cases time is always a crucial value. It could be an interesting experiment to see if a little bit of power efficiency (i.e. better processing unit) could significantly improve the recall rate.

2.3 Shape and texture features for cat face detection

The solution presented in [Zhang et al. 2008] is the only one with existing available implementation. Also, as part of it, the on-line interface was provide for testing [Heather 2015]. Authors of [Zhang et al. 2008] decided, that both texture and shape are important features of cat face detection. Their solution has two steps - in first, they create separately trained shape and texture detector, both based on Haar-like features. As a second step they are jointly training fusion classifier (joint shape and texture). Both shape and texture classifier were trained on differently normalized data. Their features are based on oriented gradients.

Important part of that study was creating a database of 10 000 annotated cat pictures [Zhang et al. 2015] (annotations of 9 points in the cat face, like ears triangles, nose, mouth). This pictures will be the part of this research dataset.

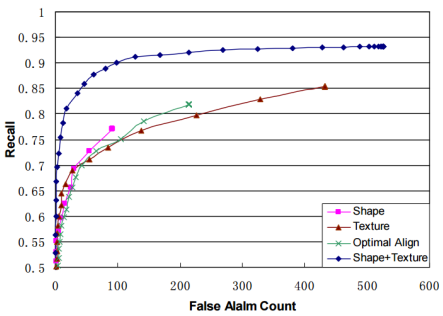

Fig. 3. ROC curves achieved as a result of [Zhang et al. 2008].

The results of that study are presented in fig. 3. The recall rate oscillate around 90% with 100 FPC for the joint texture and shape detector. Because huge focus of [Zhang et al. 2008] was on feature engineering and comparing their features with different standard approaches, authors also made a direct comparison, using Pascal 2007 data set. They claim that Average Precision (AP) of their approach is 0.364, when best reported method has AP equalled to 0.24.

The study seemed to find interesting and quite straight forward method of detecting cats face. Coming from human face detection, they adjusted their feature choice with respect to differences appearing in cat face and then tested different type of gradient features to picked the most suitable one for this aim. However, it should be noted that this article is 7 years old and probably some of the studies presented in previous sections may have better results, but it is hard to estimate, as everyone are using different databases for testing.

2.4 Stationary features and cat detection

In the study [Fleuret and Geman 2007] authors presents a novel approach based on pose-indexed features. Firstly, the image is processed using custom edge detectors and grey scale histogram. Then, the second level feature is the head-belly pose, described as a reference frame, rotated accordingly to the pose vector. The expected result is a frame containing the cat.

The classifiers used in that project are boosted trees and asymmetric weighting by sampling. The classifiers are are using hierarchical search strategy, to avoid very costly scene processing. By using this methods, authors managed to avoid fragmentation of the data for separate training of pose classifiers.

The results of that study for the head-belly detection task are presented in fig. 4. The recall is around 80% and 85% (depending on method), with FAC equalled to respectively 17 and 11 on average with variation 3.5. Authors also performed head detection experiment with a little bit better results (recall 95% with FAC ranging from 6.65 to 12.31 depending on method).

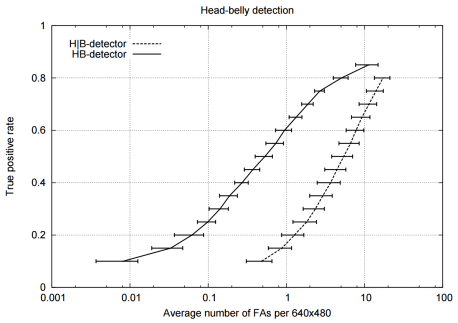

Fig. 4. ROC curves achieved as a result of [Fleuret and Geman 2007].

Summing up, that novel approach seems to be very effective, but the images were mostly focusing on cat and still life cases, so it is hard to estimate the efficiency of this study in e.g. forest live streaming. Also, that study, similar to previous one described, is quite old if we take under consideration how fast machine learning is developing in last years, so it would probably be an interesting project to revise those ideas and try to apply them to modern approaches.

2.5 Detecting and tracking lions in videos

Very promising results and approaches are presented in study [Burghardt et al. 2006]. Authors are trying to create a tool for detecting the lion face in video footage, track it and based on the behaviour, add simple description (e.g. *"lioness walking towards camera, lies down on ground"*).

They are using Haag-like features, first presented in [Viola and Jones 2001]. During the studies, they tested that broader Haag-like feature set, like the one introduced in [Lienhart and Maydt 2002], may strongly increase accuracy of the algorithm. Then AdaBoost was used for finding the best classifier.

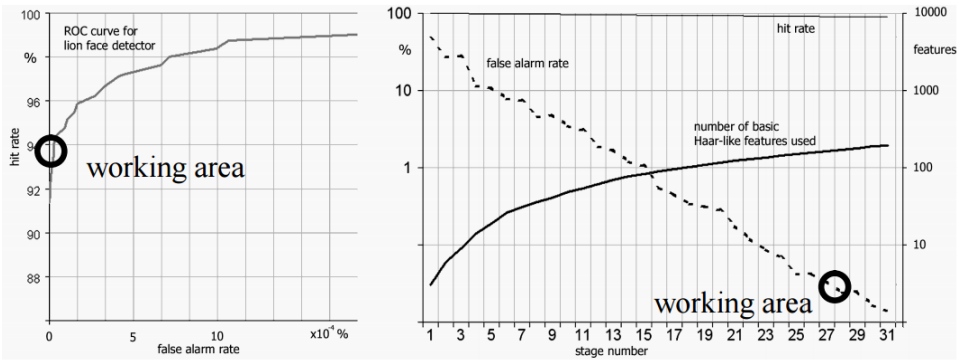

Fig. 5. (left) ROC curve for lion face training using 680 positive and 1000 negative images; (right) Outline of the learning process focused on the decrease of false alarms compared to the number of necessary single Haar-like features used to achieve this reduction, as described in [Burghardt et al. 2006].

The result is tree classifier, trained to recognize frontal lion face. Then authors are using additional techniques to track the lion face movement, even when the pose is different (not frontal), but this is unfortunately out of scope of this project. However, what is interesting, is the final ROC curve, presented in fig. 5. The recall is equalled to 93% with false acceptance rate at the level of 10^{-4}%.

That study is offering effective solution to the problem described in this research. With satisfying level of accuracy and implementation based on standard solution ([Viola and Jones 2001]), even though the article is fairly old (9 years), it is probably the most promising technique.

2.6 Discussion

Described articles are presenting different approaches to the same problem, with focus on different factors (accuracy, efficiency, performance). In regards to the main problem in this article (cat detection in live streams), probably the studies described in section 2.2 and 2.5 are the best approaches. Those are the only studies that perform actual experiments focused on illumination invariance and/or live video sapmles. It is hard to estimate robustness of algorithm depending on pose variation, because none of the study actually focus on that, usually omitting this problem in some way (i.e. tracking techniques in [Burghardt et al. 2006]. Although everyone are mentioning that cat detection is harder than human detection because of much greater pose possibilities, not many tests are performed to check the robustness of algorithm to it, usually assuming that the classifier is trained only for the frontal view.

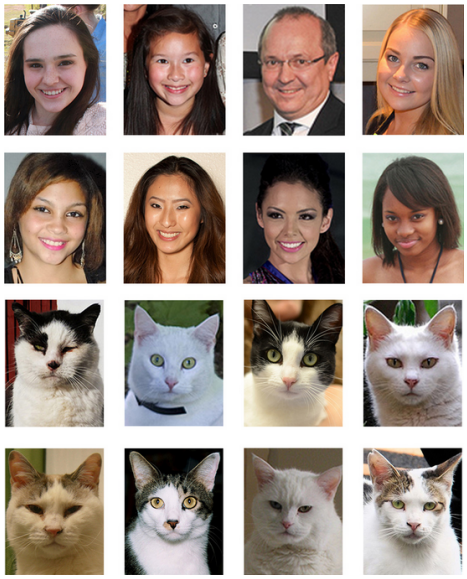

Fig. 6. First two rows - human recognized as cats by [Heather 2015]. Last two rows - cats recognized as human by OpenCV [Itseez 2015] face detection algorithm

The main difficulty in comparing various methods of cat face detection is using different databases with different sizes (ranging from 400 to a million) and different non-cat data. From all the studies, only the one described in sec. 2.1 did have a database containing human faces or other animals. Interesting test was performed by an artists(!) group Shinseungback Kimyonghun [ShinseungbackKimyonghun 2015]. They used Kittydar [Heather 2015], the application based on algorithm described in sec. 2.3 and OpenCV face detection algorithm to parse pictures of humans and cats. They ended up with 200 images, half of them with humans recognized by Kittydar as cats and half of them with cats recognized by OpenCV as humans. Part of their results is shown in figure 6.

3. PROBLEM ANALYSIS

This section is meant to describe the properties of algorithm suitable for cat detection. It will be started with recalling of expected algorithm applications:

— **be first part of cat identification system** - as such, it should have low false rejection rate. High false acceptance is allowed, as it would be filtered by identification subsystem.

— **work in poor light settings** - The live stream takes place in different locations - house, forests, gardens, often 24/7, and there may be situation when no artificial light is provided. The algorithm should be as robust as possible to light variance.

— **distinguish between cats and cat-like, humans, other animals** - a lot of current studies does not take under consideration that human faces or other animals may be the part of background. It is important to prepare features so that not too many false acceptance are caused by another beings (although, as mentioned above, false acceptance ratio is not the most important statistic).

3.1 Algorithm properties

With these applications in mind, one can summarize it with list of algorithm properties:

— robust to light/pose variation

— as low as possible false rejection count

— false acceptance count may be high

— additional focus on distinguishing between cats and other living creatures

Following considerations from [Zhang et al. 2008], it is good idea to discuss different methods that can be used in the solution:

— **sliding window detection vs. part based detection** - there are two different ways of detecting objects in the images. Sliding window scan all possible sub-windows in the image and try to classify each of them (does it have a face in it?). Part-based detects simple part of the objects (e.g. for human face recognition it could be eyes and mouth) and try to assemble them, following geometric constraints. Both of those methods are used, with successes, in human face detections. Probably sliding window would be a better method for this use case, more robust to illumination changes.

— **image features vs. gradient features** - image features are for example intensity values or PCA/wavelet coefficients. On the other side, gradient features are extracted from oriented gradients or the edge map. Both of this approaches demonstrated satisfying results in previous studies. Although image features are said to be less robust to illumination changes, they're successfully used in detection from video [Burghardt et al. 2006], so it is hard to decide one way or another before further experiments.

4. EXPERIMENTS

The experiments will be performed using Python interface of OpenCV library.

OpenCV [Itseez 2015] is an open source computer vision framework, developed from 1999, right now maintained by Itseez company. The library is cross-platform

and has C, C++, Java, Python and even Matlab/Octave interfaces. It is realease under BSD licence, which means it is free for both academic and commercial use. OpenCV is written in C++ and put strong focus on computational efficiency, supporting multi-core and GPU processing. It offers broad range of applications from computer vision, like facial recognition system, motion tracking, 2D/3D feature toolkits and also some machine learning techniques: different type of classifiers (SVM, Bayes, Random Forest, ANN) or boosting methods (boosting, gradient boosting). It has large user community (around 50 thousand) and is considered to be the best free tool for computer vision.

Python is general-purpose, high-level programming language. It support object oriented programming and has elements of imperative, functional and procedural style. Python has dynamic typing and garbage collector. There are many different implementations of Python. The most popular and reliable one are also free and open-source. It can be packaged to be used on almost all the platforms. It's characteristic is simplicity and high reliability. Python has a lot of frameworks and libraries that allows it to be useful in many different fields, e.g. Django (web development), SciPy, SciKit (scientific and machine learning), OpenStack (system administration). Also, many existing framework are ported or have interfaces in Python (like OpenCV).

4.1 Tested algorithms

Algorithms, that will be tested in the experiments, are classifiers based on Haar-like features. Two approaches will be tested:

— *predefined cascade classifier* - OpenCV offers an existing, pre-trained frontal cat face classifier.

— *trained cascade classifier* - OpenCV offers possibility of self-training the classifier with the data.

4.2 Classifier training

Following properties of classifier (and it's influence on accuracy) will be tested during the experiment:

— feature *size* (how many Haar feature are taken under consideration) - tested values will be: basic (only upright features), core, all (upright + 45 degree rotated feature set)

— *maximal false acceptance rate* (per stage).

— *number of stages* (tested values: 1, 5, 10).

Feature type (Haar or LBP - local binary patterns [Liao et al. 2007]) won't be considered as a changing property, because LBP usually has worse results (although is also faster to learn). Number of positive and negative samples is constrained by out of memory exception on values only around 300-500 images, depending on rest of the factors, so also cannot be accuracy influencing factor (as it is already small). Boosting classifier is Gentle AdaBoost (used in some of the articles in previous section).

4.3 Data set

The data set fujifilm - photo restrictions

5. BIBLIOGRAPHY

REFERENCES

Akihiko Yamada, Kazuhiro Kojima, J. K., Okamoto, M., and Murata, H. 2011. Directional edge-based dog and cat face detection method for digital camera. *IEEE International Conference on Consumer Electronics*.

Burghardt, T., acl, J., and Thomas, B. T. 2006. Tracking animals in wildlife videos using face detection.

Fleuret, F. and Geman, D. 2007. Stationary features and cat detection. *Idiap Research Report*.

FujiFilm. 2009 (accessed June 22, 2015). *Auto Dogs & Cats Detection*. `http://www.fujifilm.com/products/digital_cameras/pet/`.

Heather, A. 2013 (accessed June 22, 2015). *Kittydar - face detection for cat*. `http://harthur.github.io/kittydar/`.

Itseez. 2015 (accessed June 22, 2015). *OpenCV*. `http://opencv.org/`.

LangleyAnimalProtectionSociety. 2015 (accessed June 22, 2015). *Trap-Neuter-Return (TNR) Pilot*. `http://www.tinykittens.com/cats`.

Le, Q. V., Ranzato, M. A., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. 2012. Building high-level features using large scale unsupervised learning.

Liao, S., Zhu, X., Lei, Z., Zhang, L., and Li, S. Z. 2007. Learning multi-scale block local binary patterns for face recognition.

Lienhart, R. and Maydt, J. 2002. An extended set of haar-like features for rapid object detection.

Omkar M. Parkhi, A. V., Zisserman, A., and Jawahar, C. V. 2012. Cats and dogs.

ShinseungbackKimyonghun. 2013 (accessed June 22, 2015). *Cat or Human*. `http://ssbkyh.com/works/cat_human/`.

TheCritterRoom. 2015 (accessed June 22, 2015). *Foster Kitten Cam*. `https://livestream.com/FosterKittenCam`.

ThePetRecognitionCompany. 2013 (accessed June 22, 2015). *PiP - reuniting lost pets with their families*. `http://www.petrecognition.com/`.

TinyKittens. 2015 (accessed June 22, 2015). *Rescue Kitten Reality TV*. `http://www.tinykittens.com/`.

Viola, P. and Jones, M. 2001. Robust real-time object detection.

Zhang, W., Sun, J., and Tang, X. 2008. Cat head detection - how to effectively exploit shape and texture features. *Proc. of European Conf. Computer Vision 4*, 802–816.

Zhang, W., Sun, J., and Tang, X. 2008 (accessed June 22, 2015). *Cat database*. `http://137.189.35.203/WebUI/CatDatabase/catData.html`.

ZilliansInc. 2015 (accessed June 22, 2015). *CatFi - Smart Feeder*. `http://catfi.com/`.

Received June 2015;