Creating training datasets for OCR in mobile device video stream

Dmitry Ilin
Smart Engines
Victoria, Seychelles
Email: dmitry.ilin@phystech.edu

Valeriy Krivtsov
Moscow Institute of Physics and Technology
141700, Moscow, Russia
Email: krivtsov@phystech.edu

KEYWORDS

Monospaced font recognition, training datasets creation, neural networks, video stream, mobile device.

ABSTRACT

This paper studies methods of data sampling for training of convolutional neural networks for character recognition. These methods are considered for optical character recognition of machine readable zone (MRZ) of documents captured by a mobile phone camera. Advantages and disadvantages of training on natural and artificial datasets are discussed. In this paper we describe some set of image transformations and give examples of their practical implementation. At the end we show how adding artificial examples (to the training database), generated according to the analysis of recognition error, improves the quality of recognition.

INTRODUCTION

Optical character recognition in a video stream has its own peculiarity. If the optical axis of the camera is fixed orthogonal to the object plane, the model of geometric distortion consists of orthogonal transformations only. But for mobile camera projective transformations appear naturally, that complexify the model. Furthermore, there are other types of distortions: blinks, blurring, unfocused images, digital noise in low light conditions (distortions caused by recording conditions), and compression artifacts and camera filters, e.g. Bayesian filter (distortions caused by image processing). Special attention should be paid to careful use of complex algorithms because of medium CPU performance and low RAM capacity of mobile devices (Hudelist et al. 2014).

The paper is structured as follows. In Section 1 we discuss some existing methods of creation of training datasets, their advantages and limitations. In Section 2, we consider methods of datasets creation. We distinguish three methods of datasets creation. In Section 3 we describe the recognition algorithm and training and testing methods used in our experiments. In Section 4 we present the results of these methods and their impact on the error structure.

1. EXISTING METHODS OF DATASET CREATION

There exist numerous publicly available image databases: MNIST, EVVE (Revaud et al. 2013), faces (Cinbis et al. 2011), human actions (Laptev et al. 2008), buildings, cars, plants, etc. There are a few methods for creation of datasets using Google image search engine which give an opportunity to get many examples of frequent image classes (Cheng et al. 2015). One can use 3D modeling to create images with different projective distortions. There is also a range of instruments for image labeling. (Russell et al. 2008). The problem is that most of these methods and instruments are applicable to images of frequent classes only. To get enough examples of rare or even not existing in public datasets images one needs to collect them, which might turn to a rather long and expensive process. There are some ways to improve this collecting and marking process (e.g. Google reCAPTCHA) but they are expensive too.

2. NATURAL AND ARTIFICIAL DATASETS

In this section we describe three methods of image dataset designing. These methods can be classified by the way the images are obtained to natural, partially natural and artificial methods.
Each dataset should satisfy a number of requirements.
First, the number of examples should be sufficient for the teaching method and the complexity of the recognition method chosen. For example, in our experiments we use a convolutional neural network with 900 neurons and about 5000 weights. Thus, according to (LeCun et al. 1998), the minimal number of image examples is more than 50000. According to experiments with high quality recognition systems based on a neural network described in (Krizhevsky et al. 2012), even 10^6 examples might not be enough. Image distortions of different types increase the task complexity and the required number of images dramatically.

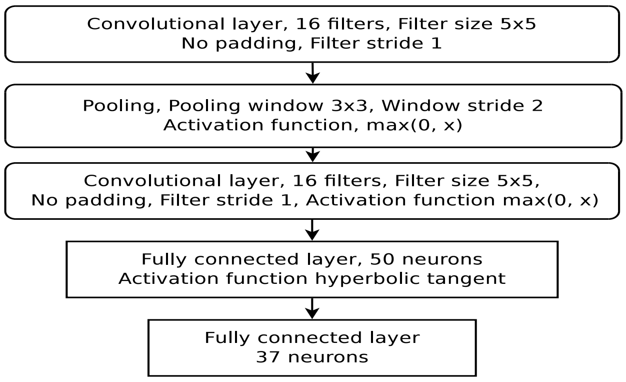
Second, datasets (especially, test datasets) must contain real examples with real image distortions and deformations.
Second, datasets (especially test datasets) must contain real examples with real image distortions. Otherwise one cannot predict the recognition quality in real life.
Third, the numbers of images of different classes should be nearly equal. Otherwise certain statistical information

Proceedings 29th European Conference on Modelling and Simulation ©ECMS Valeri M. Mladenov, Petia Georgieva, Grisha Spasov, Galidiya Petrova (Editors)
ISBN: 978-0-9932440-0-1 / ISBN: 978-0-9932440-1-8 (CD)

about the class frequencies should accompany the dataset. Such an approach goes beyond this paper.
Three methods of dataset development can be distiguished according to the dataset nature:
1) Natural datasets (closest to reality, but too expensive)
Data for natural datasets are obtained in real conditions and therefore contain a number of different distortions, geometric transformations and a correct ratio between them. Varying image capturing conditions and devices, one can get excellent data for training of recognition systems. Some distorted images can be even unrecognizable by human eye, but they still are presented in the dataset. However, obtaining natural datasets is rather expensive and takes a lot of human time. There is a number of useful techniques (e.g. Google reCAPTCHA) accelerating this process, but they are expensive too. Moreover, some images for datasets can be obtained in the same conditions and could be pretty much the same (therefore the quality of the datasets is rather low).
2) Natural datasets + generated images (a cheaper method, but not so precise)
More complex but more useful method to obtain an image dataset is to add to a natural dataset a number of different distortions of its images. One can generate a huge number of examples of good quality from a small number of natural examples. Using mathematical models of distortion one simulates the corresponding real distortions. As a result, acquired images replace naturally obtained images fairly well. This method simplifies the process of datasets collecting and reduces its price. On the other hand, this method has some important limitations. No one knows the real ratio between different distortions — it can only be estimated indirectly. As a result, adding too much examples of images with distortion (or set of distortions) of the same type can complicate training of a neural network and decrease the quality of its recognition. Another problem relates to additional random distortions that could be added to training datasets occasionally, during distortion generation process. For example, inaccurate work with translation distortion of an image can lead to non-equal amount of images with different translation vectors. Finally, the distortion generation process may take huge computational time and powers. On the other hand, it can save man-hours.
3) Artificial datasets (easy to generate, but possibly far from reality)
This method is much cheaper than the first two methods proposed because it saves human time. There is no need in image marking and data collecting — the whole dataset generation process is controlled by choice of an algorithm and a set of its parameters. One can generate a huge number of images with all types of distortions. However, weak connection between such a dataset and reality might cause bad recognition of real data.
In our work we consider training of the neural network on datasets designed using all of these methods.

3. TESTING METHODS

In this section, we describe the technology of learning and testing and the datasets used. We used convolutional neural network as a recognition method. This method has shown good results in different tasks and situations. The configuration of the neural network is shown on Fig. 1. We collected about 800000 unique symbols from


Convolutional layer, 16 filters, Filter size 5x5
No padding, Filter stride 1
Pooling, Pooling window 3x3, Window stride 2
Activation function, max(0, x)
Convolutional layer, 16 filters, Filter size 5x5,
No padding, Filter stride 1, Activation function max(0, x)
Fully connected layer, 50 neurons
Activation function hyperbolic tangent
Fully connected layer
37 neurons
Figure 1: Network Structure.

MRZ (machine readable zone) of various documents. The quality of these symbols differs, as they has been obtained in different conditions (e.g., brightness, outdoor/indoor, time of the day, etc.) and with different mobile devices (e.g., iPhones of 4, 5, 6 generations and Samsung Galaxy of several models). We made a number of short video movies (5-15 seconds) from different angles to the light source and to the document, which eventually were turned into storyboards. Thus we knew the conditions each symbol image was obtained under. Using this information we partitioned the whole set of examples of symbol images into training, validation and test sets. We did not use images from the same source (i.e. video) in different sets. Moreover, we tried not to use images taken under similar conditions in training, validation and test sets. Symbols for training datasets were centered manually.
We used the back propagation training method of training with decreasing of learning speed during training, dropout and regularization. We trained the neural network in batch mode with 10 epochs and 200 iterations per each epoch. By “one experiment” we mean “training and testing of one network”. No pre-training or post-processing methods were used. The set of examples was splitted 250000 examples for training, 300000 examples for validation and 350000 for test. Such a proportion was chosen in order to view various types of mistakes from validation data and not to use a huge amount of images for training. The test and validation sets were completely natural anf fixed for all experiments. The frequency of symbols of different classes in the test set was close to their frequency in reality. The frequency of symbols of different classes in the validation and trainig sets was equalize. The test dataset contained examples of the following classes in the following quantities:

0: 22416	1: 17602	2: 13746	3: 8115
4: 8587	5: 9383	6: 8697	7: 8082
8: 9734	9: 8847	A: 12022	B: 1834
C: 3891	D: 2952	E: 7349	F: 3282
G: 2169	H: 3309	I: 6737	J: 93
K: 2702	L: 4989	M: 6244	N: 7897
O: 4515	P: 4944	Q: 109	R: 7717
S: 5499	T: 3730	U: 4224	V: 3117
W: 744	X: 331	Y: 1834	Z: 1246
<: 110438			

Table 1: Test Results of Networks Trained on Natural Dataset.

Experiment	MRZ quality (correctly recognized zones in percents)	Correctly recognized symbols (in percents)
1	84.90	99.628
2	84.06	99.606
3	83.86	99.600
4	84.96	99.630
5	84.76	99.624
6	87.00	99.683
7	85.10	99.633
8	84.26	99.611
9	85.30	99.639
10	85.06	99.632
11	85.33	99.640
12	81.16	99.526
13	84.53	99.618
14	84.76	99.624
15	85.36	99.640
16	86.63	99.674
17	85.10	99.633
18	85.20	99.636
19	84.80	99.625
20	84.40	99.615
21	84.80	99.625
22	85.20	99.636
23	84.00	99.604
24	84.20	99.609
25	84.77	99.625
Mean values		
	84.78	**99.625**

We say that MRZ is recognized correctly if all symbols from the zone are recognized correctly. Sometimes we give results not only in percents of correctly recognized symbols, but also in percents of correctly recognized zones.

4. RESULTS

In this section we discuss the results testing of the networks trained on different datasets and decribe some useful techniches improving training.

1) Training on natural datasets.
Generation of training datasets from real images has low computational complexity. We took all images for training, equalized their amount (just repeat rare images) and divide them into train batches. Then we made 25 experiments. We trained 25 networks with different initial conditions. The results of testing are shown in Table 1. Careful study of misrecognitions made by the networks on validation dataset could help us to change the train dataset and improve the recognition. The most frequent misrecognitions are shown in Table 2 As shown in Table 2, the majority of errors is related to pairs of symbols (”0”,”O”), (”<”,”2”), and (”8”,”B”). The predominant

Table 2: The most frequent mistakes on natur images.

symbol	errors total	was misrecognized as
’0’	437	’O’:419, ’U’:5, ’J’:4, ’2’:2, ’1’:1, ’G’:1
’<’	71	’2’:29, ’K’:6, ’P’:6, ’4’:4, ’6’:4, ’1’:3
’8’	35	’B’:10, ’6’:10, ’D’:4, ’E’:2, ’M’:2, ’S’:2
’O’	20	’0’:19, ’Q’:1
’4’	19	’6’:5, ’N’:3, ’<’:2, ’A’:1, ’D’:1, ’F’:1
’6’	18	’G’:4, ’S’:4, ’D’:3, ’O’:2, ’4’:2, ’8’:2
’1’	17	’T’:6, ’Y’:5, ’7’:2, ’3’:1, ’6’:1, ’8’:1
’L’	14	’I’:9, ’4’:4, ’C’:1
’M’	14	’H’:7, ’P’:5, ’3’:1, ’N’:1
’E’	14	’C’:5, ’I’:3, ’B’:2, ’F’:2, ’A’:1, ’G’:1
’A’	14	’J’:7, ’P’:3, ’E’:1, ’M’:1, ’4’:1, ’6’:1

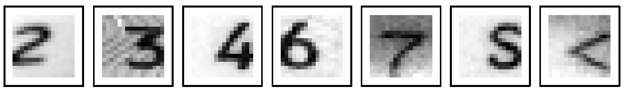


Figure 2: Wrongly Recognized Translated Images.

misrecognitions of ”0” and ”O” can easily be corrected by the statistical post processing (in MRZ recognition task). Other misrecognition are not so simple for context correction. Looking at these misrecognitions closer, one can see several types of images, whose recognition is difficult for a neural network trained on natural images. Namely, these are noisy images, projectively distorted images (e.g. translated images and especially rotated ones), images overlapped by random lines, and blurred images.

2) Training on natural + partially generated datasets.
The information about the most frequent types of mistakes allowed us to improve recognition adding some distorted images to the original training dataset. Wrongly recognized translated images are shown on Figure 2. First, we added randomly translated images. We made two shifted images per each natural. Each component of the shift vector $(\delta x, \delta y)$ (in pixels) was taken as rounded to integer random value φ from a centered Gaussian distribution

$$P(\varphi) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\varphi^2}{2\sigma^2}}$$

Changing σ for x and y axes separately, one can get various distribution of translated images. For example the distribution of shifts for $\sigma = 2$ is shown in Figure 5.

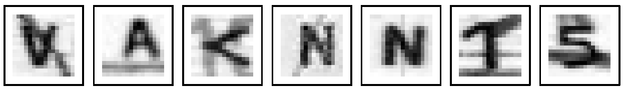


Figure 3: Wrongly Recognized Images With Lines

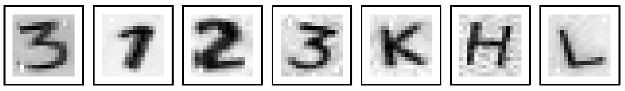


Figure 4: Wrongly Recognized Rotated Images

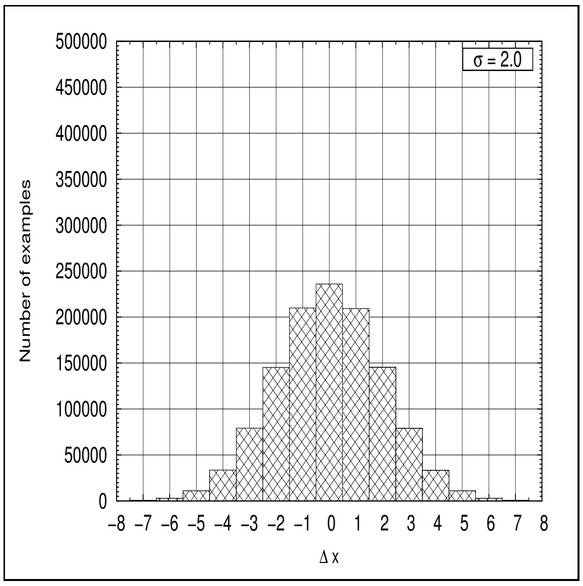


Figure 5: Number of Examples with Different Shifts Along X Axis in a Training Dataset.

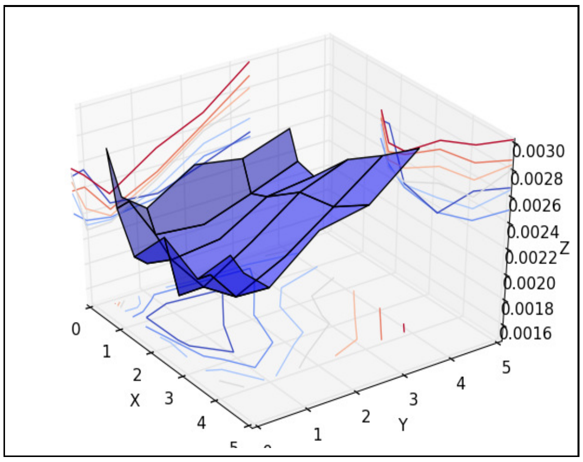


Figure 6: Validation Errors for Training with Translations Generated with Different X,Y Sigma Pairs

We used $\sigma = 0.25, 0.5, 1.0, 2.0, 3.0, 4.0$ for each axis. For each pair of σ values we made 10 experiments and calculated the average misclassification error. The results are presented in Figure 6. The minimal number of error was reached for $\sigma_x = 0.25$ and $\sigma_y = 0.5$. With respect to training without translated images, the MRZ quality increased from 84.78 to 89.68 and the number of correctly recognized symbols — from 99.62 to 99.75.

After this improvement the most frequent remaining errors are misclassifications of images with lines overlapped. Wrongly recognized images with lines are shown on Figure 3. Again, one can generate images with such lines and add them to the train dataset. We generated images with 1, 2, or 3 lines of different directions added to the random place inside the image. Lines are added to the random place inside the picture, by the different angles to the botders of image. We have added images with lines to the training dataset, so that total amount of images increased by $1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20, 25, 30, 35, 40, 45$ percents. The results are shown on Figure 7. We obtain the best result with 3 percents of images with lines added.

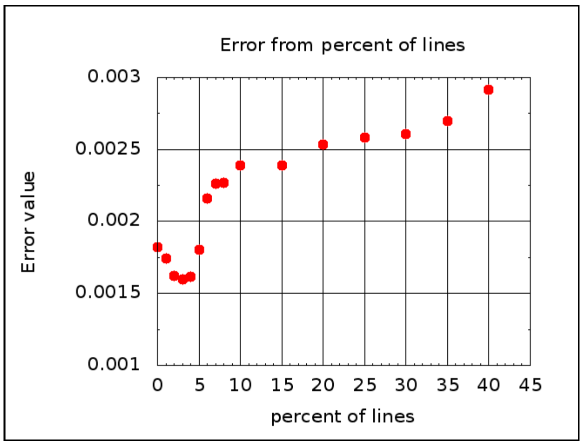


Figure 7: Validation Errors for Training with Different Percentages of Line-Overlapping Added.

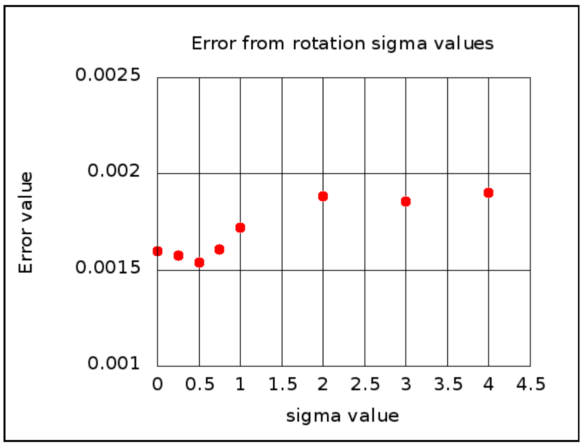


Figure 8: Validation Errors For Training With Rotations of Images Added.

Next, the most frequent remaining error became misclassifications of rotated images. Wrongly recognized rotated images are shown on Figure 4. We have added random 10 percents of rotated by a random angle images from the centered Gaussian distribution

$$P(\delta\alpha) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{\varphi^2}{2\sigma^2}}$$

We used $\sigma = 0, 0.25, 0.5, 0.75, 1.0, 2.0, 3.0, 4.0$ The results are shown on Figure 8. We obtain the best result with $\sigma = 0.5$

3) Training on artificial datasets.
We took font samples, background samples and generated training datasets from them. As mentioned in the previous section, we added artificial distortions to the training dataset. Comparing to real images, artificially generated datasets are too clean, which makes a training process complicated. After training on artificial datasets without distortions we got low quality on validation and test

Table 3: Summarised Results.

Training dataset type	MRZ quality	Symbols quality
Natural images	84.78	99.625
Natural + translated images.	89.68	99.750
Natural + translated images + line overlapping	93.19	99.839
Natural + translated images + line overlapping + rotated images	95.50	99.871
Artificial datasets	81.72	99.542

datasets. There were lots of misrecognitions of rather simple blurred images, images with small lightspots, and images with combinations of small distortions. We could try to add different distortions with many parameters but optimization of these parameters is rather difficult and might take a lot of time. In our work, we used nomad optimization package for optimization of distortion parameters. As a result we achieved 81.72 percent of correctly recognized MRZ strings.

The best results achieved by our neural network trained on datasets of the three types are summarised in Table 3.

CONCLUSION

In this paper, we describe three methods of training datasets development. The best results were obtained on datasets created from natural images with distortions generated from natural datasets. Using information about neural network (trained on natural datasets) recognition errors, we have improved our results and designed a network more stable to distortions, compared to network trained only on natural images.. As a result, using some of the distortions for generating additional examples in training datasets, we achieved quality 94.5% of correctly recognized MRZ strings (0.9987% per letter). Using the context postprocessing, which solves ”0” - ”O”, ”M”-”N”-”H” problems, we achieve quality 99.3% of correctly recognized MRZ strings.

ACKNOWLEDGEMENTS

This work is supported by Russian Foundation for Basic Research (projects RFBR 13-07-12170, 13-07-12178).

REFERENCES

Cheng, Dong Seon; Francesco Setti; Nicola Zeni; Roberta Ferrario; and Marco Cristani. 2015. “Semantically-driven automatic creation of training sets for object recognition.” *Computer Vision and Image Understanding*, 131(0):56 – 71. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2014.07.005. Special section: Large Scale Data-Driven Evaluation in Computer Vision.

Cinbis, Ramazan Gokberk; Jakob Verbeek; and Cordelia Schmid, “Unsupervised Metric Learning for Face Identification in TV Video.” In *ICCV 2011 - International Conference on Computer Vision*, pages 1559–1566 (IEEE, Barcelona, Spain, 2011). doi: 10.1109/ICCV.2011.6126415.

Hudelist, Marco A.; Claudiu Cobârzan; and Klaus Schoeffmann, “Opencv performance measurements on mobile devices.” In *Proceedings of International Conference on Multimedia Retrieval*, ICMR ’14, pages 479:479–479:482 (ACM, New York, NY, USA, 2014). ISBN 978-1-4503-2782-4. doi: 10.1145/2578726.2578798.

Krizhevsky, Alex; Ilya Sutskever; and Geoffrey E. Hinton, “Imagenet classification with deep convolutional neural networks.” In F. Pereira; C.J.C. Burges; L. Bottou; and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105 (Curran Associates, Inc., 2012).

Laptev, Ivan; Marcin Marszałek; Cordelia Schmid; and Benjamin Rozenfeld, “Learning realistic human actions from movies.” In *Conference on Computer Vision & Pattern Recognition* (2008).

LeCun, Y.; L. Bottou; G. Orr; and K. Muller, “Efficient backprop.” In G. Orr and Muller K., editors, *Neural Networks: Tricks of the trade* (Springer, 1998).

Revaud, Jérôme; Matthijs Douze; Cordelia Schmid; and Hervé Jégou, “Event retrieval in large video collections with circulant temporal encoding.” In *CVPR 2013 - International Conference on Computer Vision and Pattern Recognition*, pages 2459–2466 (IEEE, Portland, United States, 2013). doi:10.1109/CVPR.2013.318.

Russell, Bryan C.; Antonio Torralba; Kevin P. Murphy; and William T. Freeman. 2008. “Labelme: A database and web-based tool for image annotation.” *Int. J. Comput. Vision*, 77(1-3):157–173. ISSN 0920-5691. doi:10.1007/s11263-007-0090-8.

AUTHOR BIOGRAPHIES

DMITRY A. ILIN was born in Yaroslavl, Russia. He studied applied physics at Moscow Institute of Physics and Technology (MIPT). Dmitry has graduated from MIPT with a Master degree in 2014. At this moment, he is a PhD student at Institute of Systems Analysis. His research interests are in the areas of computer vision and machine learning. His e-mail address is dmitry.ilin@phystech.edu.

VALERIY E. KRIVTSOV In 1971 graduated from MIPT, Candidate of Physical and Mathematical Sciences. 1971-1981 - engineer, junior researcher, senior researcher at the Institute of Control Sciences of the Russian Academy of Sciences. From 1981 - senior researcher, leading researcher, head of the laboratory at the Institute of Systems Analysis of the Russian Academy of Sciences. From 1976 - assistant, docent, head of specialization at MIPT concurrently. Author of 3 monographes and over 100 articles. His research interests are in the areas of information technologies and computer sciences, mathematical programming. His e-mail address is krivtsov@phystech.edu.