Deep Convolutional Neural Networks for Forest Fire Detection

Qingjie Zhang, Jiaolong Xu, Liang Xu and Haifeng Guo

Aviation University of Air Force, Changchun, China

nudtzhang@hotmail.com

Keywords: Fire detection, Convolutional Neural Networks, UAVs

Abstract. We proposed a deep learning method for forest fire detection. We train both a full image and fine grained patch fire classifier in a joined deep convolutional neural networks (CNN). The fire detection is operated in a cascaded fashion, ie the full image is first tested by the global image-level classifier, if fire is detected, the fine grained patch classifier is followed to detect the precise location of fire patches. Our fire patch detector obtains 97% and 90% detection accuracy on training and testing datasets respectively. To facilitate the evaluation of various fire detectors in the community, we build a fire detection benchmark. According to our best knowledge, this is the first one with patch-level annotations.

Introduction

Detecting fire at early stage before they turn to catastrophic event is crucial to prevent fire disastrous and save people's life and properties. Though fire and smoke detection sensors are widely installed in indoor environments, they generally require the fire to burn for a while to produce large amount of smoke and then trigger the alarm. Besides, these devices can not be deployed in large scale outdoor environments, eg forest, wild area. In contrary, vision-based fire detection system capture images from cameras and detect fire immediately, thus suitable for early fire detection. Such system is also cheap and easy to install. In this work, we proposed a vision-based fire detection method which can work with non-stationary camera. Our fire detection system can be mounted on unmanned aerial vehicles (UAVs) for large scale forest fire detection.

There has been several methods in the literatures for vision-based fire detection. Colour model, motion, spatial and temporal features are primarily used as fire has very specific characteristics compare to other object. Almost all of the proposed methods follow similar detection pipeline, ie, first find moving pixels using background subtraction then apply color model to find fire color regions. These regions are further analysed spatially and temporally to detect the irregular and flickering characteristics of fire. Since motion is the dominant feature, these methods work only with fixed camera, ie in surveillance scenarios. In this work, our method is not limited by such constraints. We employ the modern powerful deep learning method to learn feature representations from the data and train discriminative classifiers for fire region detection. In particular, we use deep convolution neural networks (CNN) as the learning machine.

One of our contributions in this work is that we proposed a fire detection benchmark. To the best of our knowledge, there is no standard dataset and evaluation protocol in the previous literatures which makes it difficult to compare various fire detection methods. We collected the fire sequence from some previous literatures as well as our own dataset and made patch-wise annotations. We expect this benchmark can facilitate the future researches in this area.

The rest of the paper is organized as follows: in Sec.2 we review the vision-based fire detection methods, in Sec. 3, we introduce our dire detection benchmark. Our CNN-based fire detection method is presented in Sec.4 and the experimental results are shown in Sec.5. Finally, Sec.6 summarizes the paper and discusses several future research lines.

Related Work

Compare to the general object detection, literatures about fire detection in the computer vision are very few. As fire is a non-rigid object with dynamic shapes, the vast majority of the literatures use

motion and color features [1, 2] or spatial temporal features [3]. Beside, most of the work assume fire detection at a surveillance scenario, i.e. a camera s mounted in a fixed place to detect flames and/or smokes.
Background subtraction thus is widely be applied to improve the accuracy and robustness [4]. In this work, we are not assuming a surveillance scenario, our method can be applied to more challenging dynamic environments, e.g. mounting cameras on UAVs. This makes background subtraction and motion analysis difficult.
From the previous literatures, the classic fire detection pipeline can be summarized as following steps: (1) Moving pixels and regions exaction [5, 4]. (2) Flame and/or smoke region candidates extraction using color model, e.g., HIS color model is used in [5, 4]. (3) Further analysis to the candidate regions, e.g. foreground region analysis [5], fire dynamic behaviour analysis [6][2].
Instead of following traditional vision-based fire detection pipeline, we use CNN for learning feature representation-s and fire classifiers. Deep convolutional neural networks has shown state-of-the-art performances for many comput-er vision tasks, e.g., object recognition [7], detection [8], semantic segmentation [9]. In fact, learning-based methods are rarely investigated in the previous work. The recent work of [10] proposed a covariance descriptor based on color, spatial and temporal information, and train a SVM as classifier. Similar to our method, the proposed method also works with non-stationary cameras. SVM classifier is also employed in [3]. In contrary to these method which requires carefully designed hand-crafted features, we use CNN to learn feature representation from raw pixels, thus does not require color model or spatial temporal informations, though such information can be complementary to our method. Our experiments show that using pure learning-based method, we obtain surprisingly good results, e.g., fire patch recognition accuracy is as high as 90% in our testing data set and training accuracy is 97%.

Fire Detection Benchmark

For general object detection, pedestrian or face detection, there has been benchmarks available for comparison of state-of-the-art algorithms. As far as we know, this is no public available fire detection dataset available.

Dataset

We collect a fire detection dataset from various online resources. Video data are mainly acquired from [11] which are commonly used in recent fire detection literatures. The current dataset contains 25 videos acquired in the forest environment, including 21 positive (fire) sequences and 4 negative (non-fire) sequences. To train image-based fire classifier, we further extract images from the videos with a sample rate of 5, i.e. sample one image every five frames.
These images are resized in canonical size of 240 × 320. In order to support evaluation of fire patch localization, We manually annotated the exiting fire patches with 32 × 32 bounding boxes. As a proof of concept, in this paper, we use a small subset for developing and evaluate our algorithms. The statistics of the used train and test images and the number of annotated patches are shown in Table 1. Some sample images with annotations can be seen in Fig. 1. Depends on the acceptance, we consider to make this benchmark public available to facilitate the future researcher on evaluation of their vision-based fire detectors.

Table 1: Statistics of the proposed fire detection dataset

	# images	# patches	# positive patches	#negative patches
Train set	178	12460	1307	11153
Test set	59	4130	539	3591

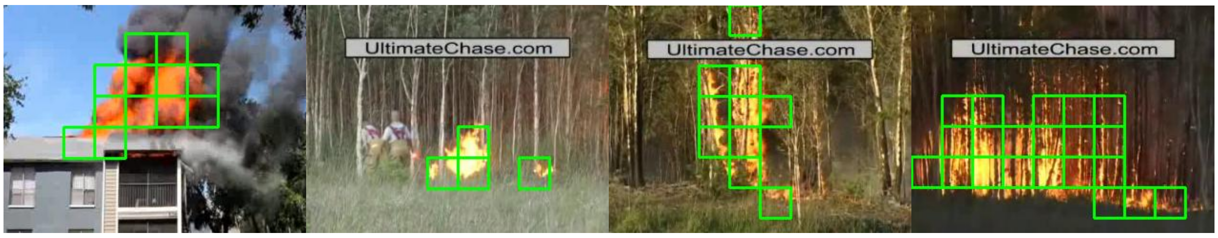

Figure 1: Sample images from the proposed benchmark and their patch-wise ground truth

The current dataset used in our experiments is relatively small, due to limitation of the time consuming manual annotation work. In the future, we consider to increase the number of images and adding more precise annotations, e.g. 16×16 patches. Since the manual annotation is time consuming, expensive and prone to errors, we also consider to use computer graphic engine to create synthetic dataset with automatically labelled ground truth, similar to the work in [12, 13] for pedestrian detection. Such annotations can be done at pixel-wise level.

Evaluation criterion

Most of the literatures report image-level evaluation, i.e., fire and non-fire image classification accuracy. Some of them also report patch-level detection accuracy, but ground truth are not public available. Based on our patch-level annotations, we consider an evaluation criterion of both image-level and patch-level classification accuracy. We use following criterion for evaluate a fire detector:

$$\begin{aligned} accuracy &= \frac{TP+TN}{N_{pos}+N_{neg}} \\ detection\,rate &= \frac{TP}{N_{pos}} \\ false\,alarm\,rate &= \frac{FN}{N_{neg}} \end{aligned} \tag{1}$$

where TP is the number of truth positives, i.e. the number of fire patches which are classified as fire, TN is the number of truth negatives, i.e. the number of non-fire patches which are classified as non-fire, FN is the number of false negatives, i.e., the number of non-fire patches which are classified as fire. N_{pos} and N_{neg} are the number of positives and the number of negatives in the ground truth respectively.

Proposed Method

Our goal is to detect fire at patch level. Using machine learning method, the straight forward way is to learn a fire patch classifier using annotated positive and negative samples, i.e. a binary classification model. Following this basic idea, we have two proposals, (1) learn binary classifier using annotated patches from scratch, and (2) first learn a full image fire classifier and then apply fine-grained patch classifier if the image is classified as contains fire. We give details of these two proposals in the following sections.

Proposal 1: learning fire patch classifier from scratch

Learning a binary classifier using our annotated patches is straightforward. However, training such classifier is nontrivial because the positive and negative samples are unbalanced, as we have large amount of negative patches and only a few positive patches (see Table 1). In this work, we investigated two types of classifiers: linear classifier and a non-linear one.

For linear classifier, we use linear SVM from [14] and for the non-linear classifier, we use CNN from Caffe frame-work [15]. Due to the size of our annotated dataset is small, we adopt the CIFAR 10 network [16]. The structure of the CIFAR 10 network is illustrated in the Fig 2, we changed the number of output in last fully connected layer into 2 for our binary classification problem. In this small network, we also included a drop out layer to reduce over fitting.

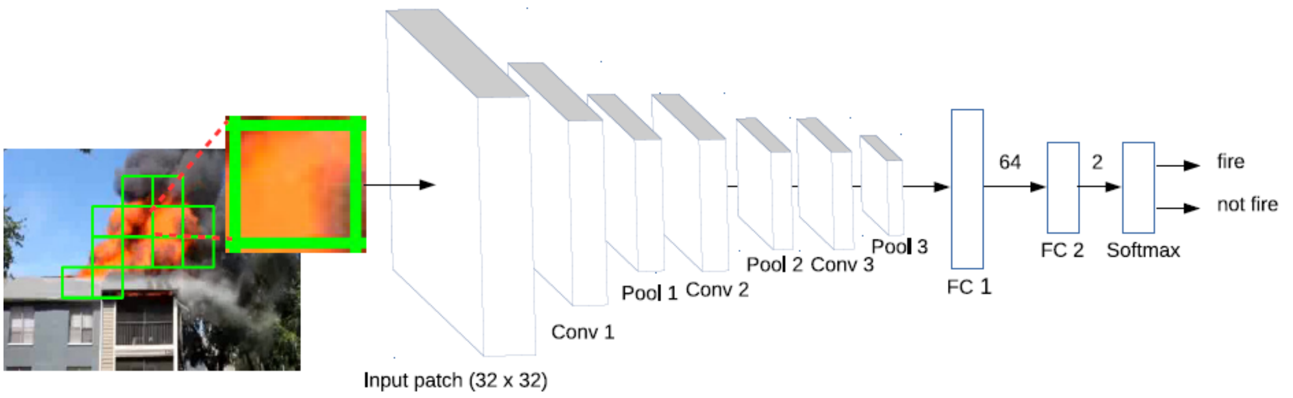

Figure 2: The architecture of the patch CNN classifier. The network is trained from the raw pixels features, i.e. the 32×32 patches.

Proposal 2: cascade CNN fire detector

One shortcoming of the above patch classifier is that the classifier is trained with local patches which do not contain global information. Also, the computation of these patch classifiers is high due to it has to test multiple patches for every frame.

To address the problem of lack of global information, we can train a full image classifier based on the positive and negative images. An image is called negative image if there is no fire patch in it and a positive image should contain at least one fire patch. Since we have a very small training dataset, training a full image classifier from scratch may suffer from over fitting. One efficient way to train a CNN classifier on small dataset is the so called fine-tuning technique, which can transfer the previously learned network parameters to the new model for the new dataset. We use the 8 layers AlexNet available in Caffe framework as our pre-trained model. The network architecture is shown in Fig 3.

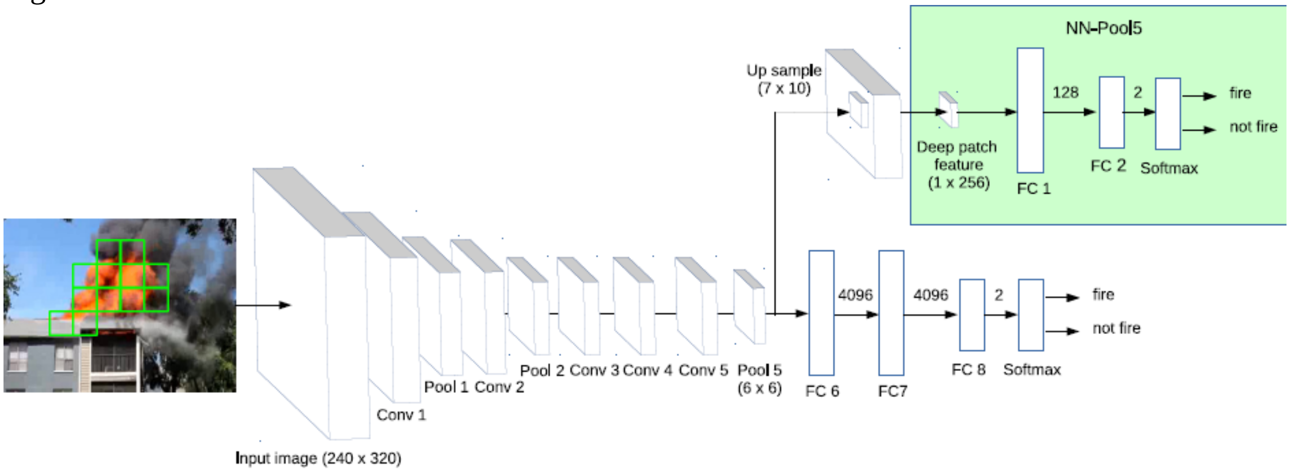

Figure 3: The architecture of proposed cascade CNN fire classifiers. The bottom deep CNN is fine tuned from AlexNet which is trained on ImageNet dataset. The fine-grained patch classifier is trained with the up sampled Pool-5 features, which is a 2 fully connected layers neural networks.

Once the full image classifier is trained, we can classify whether an image contains fire patches or not. If fire patches exist, we apply a fine-grained fire patch classifier to find the precise location. One way is to directly apply the patch classifier we proposed in 4.1. However, simply apply a patch classifier trained from scratch is not optimal. The reason is that, first, the global (full image) and local (patch) classifiers are independent, there is no interaction or information sharing between them; second, the computation is the sum of both full image and local patch classifiers, thus is redundant. A better solution is to share the features between full image and local patch classifiers.

We proposal to train patch classifier directly from the full image CNN, i.e. extract the patch features from the full image CNN. The idea is shown in Fig 3. In practice, we extract local patch features from the Pool-5 layer. The features from the Pool-5 layer has dimension of 6 × 6 × 256, which can be regarded as a 6×6 grid with 256 dimensional feature for each grid. Note that our patch size is 32 × 32, which means we have 7 × 10 grids for our 240 × 320 size image. So the extract Pool-5 features is not matched the grid size of our ground truth. There are several strategies to handle this problem, e.g., converting the fully connected layer into convolution layer [9]. However, we found a simple linear interpolation works just well. So we simply up sample a 7 × 10 grid features from the 6 × 6 Pool-5 grid features. In this way, we obtained 256 dimensional features for each patch. With these extracted deep learned features, we are able to train a patch classifier. In practice, we found a non-linear small neural network works better than linear SVM. The architecture of the small non-linear network is show in Fig 3, inside the top right green box.

The advantage of this method is that: (1) it is a well-known fact that the high level feature representations from the deep network is more discriminative than the raw pixel features or other hand-crafted features. Thus, we can train a patch classifier with higher accuracy. (2) there is no redundant feature computation since both global and local classifiers share the same network for computing features. (3) the computation of the deep feature patch classifier is even lower due to the low dimension of the high level features in CNN. For example, using Pool-5 feature, the dimension for each patch is 256 while using raw pixel features, it is 32 × 32 × 3 = 3072.

Experiments

In this section, we evaluate the proposed methods on our dataset. We first investigate the performance of the patch classifier trained from scratch using raw pixel features. We compare two different classifiers: (1) linear SVM classifier and (2) CNN classifier. The linear SVM classifier is trained using liblinear [14], the hyper-parameter of C is tuned on our training dataset and set to 1.0. For the CNN classifier, we use the architecture shown in Fig 2, i.e. the CIFAR 10 net. We call these two baselines SVM-Raw and CNN-Raw respectively. Our CNN networks is trained in Caffe framework using RMSprop [17] solver for optimization, we use patch size of 70 and the parameters for the RMSprop solver is shown in Table 2 CNN-Raw. The results of different patch classifiers on training and testing set are shown in Table 3.

Table 2: Solver parameters for CNN patch classifier

Classifier	Solver type	Learning rate	Weight decay	Rms decay	Batch size
CNN-Raw	RM-Sprop	5e-5	0.1	0.99	70
Fine tune	RM-Sprop	5e-5	0.05	0.9	50
NN-Pool5	RM-Sprop	5e-5	0.05	0.99	500

Table 3: Accuracies of different patch classifiers

		SVM-Raw	CNN-Raw
Train set 1	Accuracy	92.2%	93.1%
	Detection rate	56.2%	84.5%
	False alarm rate	3.6%	3.9%
Train set 2	Accuracy	74%	88.6%
	Detection rate	23.9%	59.6%
	False alarm rate	18.5%	6.5%

Next, we evaluate our full image CNN classifier. Using fine tuning, our full image classifier can be trained in just several hundred iterations and reaches both training and testing accuracy as high as 100% which is surprisingly good.

Table 4: Accuracies of different patch classifiers using Pool-5 feature

		SVM-Pool5	CNN-Pool5
Train set 1	Accuracy	95.6%	97.3%
	Detection rate	76.2%	84.8%
	False alarm rate	2.1%	1.2%
Train set 2	Accuracy	89%	90.1%
	Detection rate	40.1%	39.2%
	False alarm rate	3.7%	2.3%

Table 5: Detection speed of different patch classifiers

Classifier	SVM-Raw	CNN-Raw	NN-Pool5
Time per image (s)	0.16	2.1	1.4

This might attribute to the pre-train AlexNet which has been trained on large scale ImageNet database with millions of images. With this fine-tuned model we extract features from Pool-5 layer. Now, we evaluate our patch classifier trained with Pool-5 layer features. Still, we compare two different patch classifiers, i.e. linear SVM and non-linear neural network (NN) classifier. For linear SVM, we set C = 1.0. For the NN classifier, batch size is set 500, and the solver parameters can be found in Table 2 NN-Pool5. Table 4 shows the accuracies on both training and testing datasets. Finally, we compare running time of different classifiers. We evaluate the per image processing time. The results are shown in Table 5. Note that when use NN-POol5 classifier, we included the process time of the full image CNN classifier. In fact, the full image classification is the dominant time consumption, which is around 1.42 seconds per images while the following NN-Pool5 consumes less than 0.01 second in our practice. Also note that, we do not use GPU for CNN computing. As AlexNet takes only 1.2 ms on a typical GPU, so when with GPU mode, our cascade CNN fire detection system is expected to run at 1000 frames per second, which is more than sufficient for real-time fire detection systems. Some qualitative results are shown in Fig. 4, where we can see that our final model NN-Pool5 consistently output performs SVM-Raw and CNN-Raw, which demonstrated the effectiveness of our proposal.

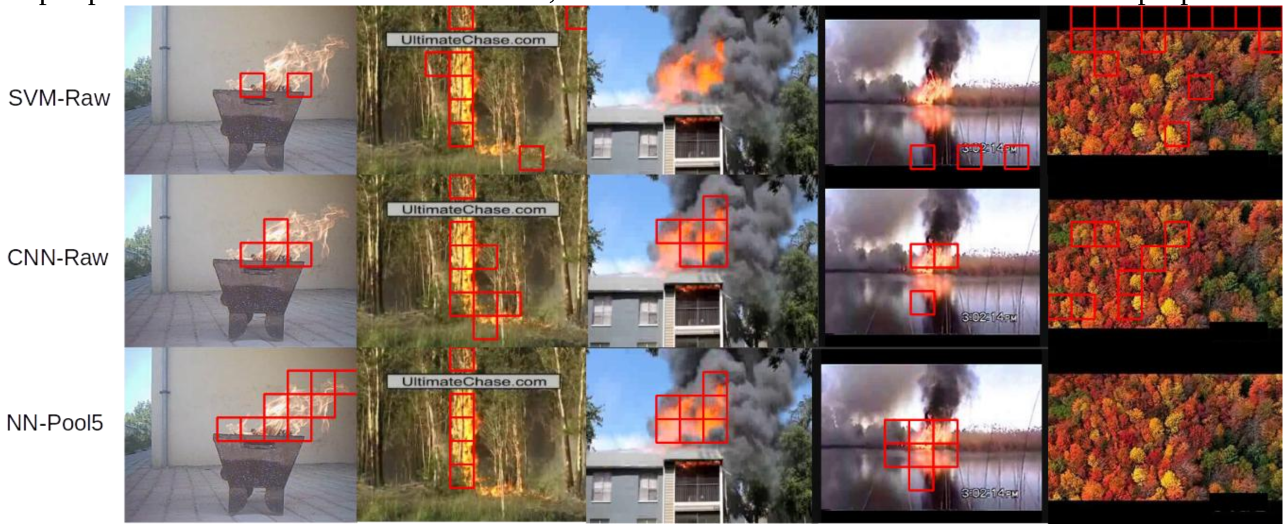

Figure 4: Detections results of different classifiers, from top to bottom, SVM-Raw, CNN-Raw and NN-Pool5.

Conclusions

In this work, we proposed to use a deep learning approach for training forest fire detector. The proposed model consists of a full image CNN and local patch NN classifier, both classifiers share the same deep neural networks. Besides, we proposed a fire detection benchmark due to the lack of standard dataset in the computer vision community. For the future work, we would like to increase the size of the benchmark and make finer grained annotations, e.g., using compute graphic engine to create synthetic data and generate pixel-wise annotations for free. We also plan to mount the proposed fire detection system on UAVs for real world forest fire detection.

Acknowledgements

This work was financially supported by the Projects of Jilin Science & Techonology(20130522108JH) and National Natural Science Foundation (61203355).

References

[1] H. O. H. U. M. Celik, T.; Demirel, “Fire detection in video sequences using statistical color model,” in IEEE InternationalConferenceonAcoustics, SpeechandSignalProcessing, 2006.

[2] I. K. Martin Mueller, Peter Karasev and A. Tannenbaum, “Optical flow estimation for flame detection in videos,” IEEE Trans. on Image Processing, vol. 22, no. 7, 2013.

[3] N. A. Che-Bin Liu, “Vision based fire detection,” in Int. Conf. in Pattern Recognition, 2004.

[4] P. Gomes, P. Santana, and J. Barata, “A vision-based approach to fire detection,” International Journal of Advanced Robotic Systems, 2014.

[5] X. Z. Chunyu Yu, Zhibin Mei, “A real-time video fire flame and smoke detection algorithm,” in Asia-Oceania Symposium on Fire Science and Technology, 2013.

[6] A. E. C. B. Ugur Toreyin, “Online detection of fire in video,” in IEEE Conf. on Computer Vision and Pattern Recognition, 2007.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks.” In Advances in Neural Information Processing Systems, Lake Tahoe, Nevada, USA, 2012.

[8] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in IEEE Conf. on Computer Vision and Pattern Recognition, Columbus, Ohio, USA, 2014.

[9] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in IEEE Conf. on Computer Vision and Pattern Recognition, 2015.

[10] Y. Habibolu, O. Gnay, and A. etin, “Covariance matrix-based fire and flame detection method in video,” Machine Vision and Applications, vol. 23, no. 6, pp. 1103–1113,2011.

[11] B.U.Toreyin, “Fire detection dataset,” http://signal.ee.bilkent.edu.tr/VisiFire/.

[12] D. Vázquez, A. López, J. Mar´ın, D. Ponsa, and D. Gerónimo, “Virtual and real world adaptation for pedestrian detection,” IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 36, no. 4, pp. 797–809, 2014.

[13] J. Xu, D. Vazquez, A. Lopez, J. Marin, and D. Ponsa, “Learning a part-based pedestrian detector in a virtual world,” IEEE Trans. on Intelligent Transportation Systems, vol. 15, no. 5, pp. 2121–2131, 2014.

[14] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: a library for large linear lassification,”Journal of Machine Learning Research, vol. 9, pp. 1871–1874, 2008.

[15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long,R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” arXivpreprint arXiv:1408.5093, 2014.

[16] A. Krizhevsky, “Learning multiple layers of features from tiny images,” http://www.cs.toronto.edu/ kriz/cifar.html, Tech. Rep., 2009.

[17] T. Tieleman and G. Hinton, “Rmsprop: Divide the gradient by a running average of its recent magnitude.” COURSERA: Neural Networks for Machine Learning, Tech. Rep.,2012