Distributed Embedded Deep Learning based Real-time Video Processing

Weishan Zhang[1] Dehai Zhao[1] Liang Xu[1] Zhongwei Li[1] Wenjuan Gong[1] Jiehan Zhou[2]
[1]Department of Software Engineering, China University of Petroleum
No. 66 Changjiang West Road, Qingdao, China. 266580
[2] University of Oulu, Department of Information Processing Science, Finland
{zhangws, wenjuangong}@upc.edu.cn {zhaodh.upc}@gmail.com {jiehan}@ee.oulu.fi

***Abstract*—There arises the needs for fast processing of continuous video data using embedded devices, for example the one needed for UAV aerial photography. In this paper, we proposed a distributed embedded platform built with NVIDIA Jetson TX1 using deep learning techniques for real time video processing, mainly for object detection. We design a Storm based distributed real-time computation platform and ran object detection algorithm based on convolutional neural networks. We have evaluated the performance of our platform by conducting real-time object detection on surveillance video. Compared with the high end GPU processing of NVIDIA TITAN X, our platform achieves the same processing speed but a much lower power consumption when doing the same work. At the same time, our platform had a good scalability and fault tolerance, which is suitable for intelligent mobile devices such as unmanned aerial vehicles or self-driving cars.**

***Keywords*-Distributed embedded platform, video processing, low power consumption, Stream processing**

I. Introduction

With the rapid development of hardware technology and embedded devices, a credit-card-sized component can take computer's place to do computing efficiently. Although these advanced embedded devices have strong computing capabilities, a single embedded device is limited after all for some complex computing tasks, for example real time video processing. Naturally these arises the needed to built distributed networked embedded platform to achieve faster and stronger processing capabilities.

Since Alex and Hinton won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012 using Deep Convolutional Neural Network (DCNN) [1], deep learning based techniques have refreshed a lot of record in computer vision field, making it one of the best choices for video analysis. AlexNet contained six billion float parameters and it needed one billion floating point operation to solve a color image with a resolution of 224×224. Because of the limitation of hardware at that time, AlexNet had to run on two GPUs. Bigger deep learning network may lead to high performance. For example, Google DeepMind confirmed that the distributed version of AlphaGo was significantly stronger, winning 77% of games against single machine AlphaGo [4].

However, the high performance of large-scale deep neural networks is achieved with the cost of high power consumption. For example, the distributed version of AlphaGo had 280 GPUs and 77KW power consumption, which is equal to 100 adults. On the one hand, big networks need high-performance devices. On the other hand, mobile devices' compute capability is limited by weight and battery capacity. Though mobile cloud computing can transfer a part of calculation to back end cloud servers, the bandwidth, latency and availability are challenging when solving high traffic and real-time stream data. Therefore it's necessary to design a distributed platform to achieve efficient embedded computing using deep learning.

In this paper, we propose a distributed embedded deep learning platform using the latest NVIDIA Jetson TX1 [1], supported by Apache Storm real-time stream processing framework, which is called DEDLuS (Distributed Embedded Deep Learning platform using Storm). An object detection algorithm based on DCNN is used to evaluate the performance of proposed platform, including scalability, fault tolerance and power consumption. The deep learning network is trained on NVIDIA TITAN X and is applied to detect object from real-time video in the proposed platform.

The rest of the paper is organized as follows. Section II describes the platform we built, including the hardware, the object detection algorithm and the overall architecture. Section III is the evaluation of our platform. Section IV introduces some related work and compares them with our work. Conclusion and future work end the paper.

II. Design and Architecture of the DEDLuS platform

The idea behind of the DEDLuS is that it should be usable for lightweight deep learning based video processing. Video stream captured from cameras is preprocessed on a normal PC, including compression and resizing. Then the data are transfered to each TX1 board, on which only one Storm worker is running. The DCNN network model we used to test is pre-trained on 20 types of natural images. After one cycle of detection, the results which contains time stamp,

[1]https://developer.nvidia.com/embedded-computing

coordinates and classes are merged together and stored on disk. Figure 1 shows the work flow of the whole video processing process of the DEDLuS platform.

A. Hardware

As we know, the training of a neural network as well as the evaluation of a neural network after training involves a large amount of parallelizable float point computation from matrix multiplications. This type of computation can be accelerated by GPUs. Therefore the hardware we used to build the DEDLuS platform is the latest NVIDIA Jetson TX1, as shown in Figure 2. It is built around NVIDIA Maxwell^{TM} architecture with 256 CUDA cores delivering over 1 teraflops of performance, along with 64-bit ARM CPUs, 4K video encode and decode capabilities at 60 fps, and a camera interface capable of 1400MPix/s. The board combines CPU and GPU functionality in a single unit as Tegra APU, which provides high floating point throughput while consuming very low power (10W).

B. Object Detection Algorithm

Object detection pipelines generally start by extracting a set of robust features from input images. Then classifiers or localizers are used to identify objects in the feature space. These classifiers or localizers are run either in sliding window fashion over the whole image or on some subset of regions in the image [5]. Comparing several top detection frameworks, we chose YOLO [6] as our object detection method.

As shown in figure 3, YOLO is simple. We chose YOLO because it has many benefits over conventional object detection method.

- YOLO is extremely fast. The base network runs at 45fps with no batch processing on a TITAN X GPU and tiny network runs at more than 150 fps. This means YOLO can process real-time streaming video smoothly.
- YOLO sees the entire image during training and test time so that it can encode contextual information about objects, which makes YOLO makes less background errors than Fast R-CNN [7].
- YOLO learns generalizable representations of objects. A network model trained on natural images can preforms well when tested on artwork.

C. Architecture of the DEDLuS platform

The architecture of the platform described with a UML deployment diagram is shown in Figure 5. It mainly consists of three parts: Video capture module, data management nodes (supervisor1,2) and data processing nodes (supervisor3,4,5,6). The data management node is a common desktop with i7 CPU and 16G RAM. It is receiving and preprocessing video data from video capture module, which is a Webcam. It also merges and stores the results achieved by data processing node. Each data processing node correspond to one Jetson TX1 board, on which the object detection algorithm runs. In order to reduce network throughput, video stream is encoded on data management node and decode on data processing node.

The Storm topology is shown in Figure 4. Spout is a message producer, which transmit video frames to Strom bolts. We added time stamp to every frame so that the result can be reconstructed correctly. Bolts in the second layer receive video frame data and conduct object detection. Then the results which contain time, location and class information were merged on the bolt in the third layer.

III. Evaluation

For evaluation purpose, we use four NVIDIA Jetson TX1 development boards and a PC to build a prototype of the DEDLuS platform. Considering the computing capability and RAM of the TX1 board, we chose the tiny YOLO model, which has only 9 layers. In addition, we only started one worker on each board. The prototype of the DEDLuS platform is designed only for object detection, and the model was pre-trained on 20 class natural images, which has a mean Average Precision of 70%. The test data was traffic surveillance video and we conduct real-time object detection on it.

A. Evaluation of performance

At the beginning of experiment, the whole cluster is in standby model. We can see from table I, the standby power consumption of Jetson TX1 is very low, which is about 2W each board. And the 1.06M/s network throughput is the heartbeat of Storm cluster. When starting one worker to run object detection algorithm, the power consumption is increased to 18.8W. And if use three boards, the detection speed reaches to 35fps, which means we can process streaming video in real-time with less than 28 milliseconds of latency. The fourth board is not necessary for real-time video analysis and it can be used to make sure that the DEDLuS platform is fault tolerant.

A single board's mean power consumption is 9.2W, which is near to the peak(10W). At the same time, each board's GUP utilization is over 90%, which means the board is full load to run the algorithm. As a comparison with the presented cluster, we also ran the same object detection algorithm on GTX TITAN X GPU, and the results are shown in table II. We can see the TITAN's power consumption is higher than the proposed platform when achieving the same detection speed.

The histogram figure 6 shows the tendency of the performance. We can see that with the increasing of stared worker, all the measured values increase correspondingly, which means our platform has a good scalability. If there are more video to be processed, we can satisfy this demand conveniently by adding more boards to the DEDLuS platform.

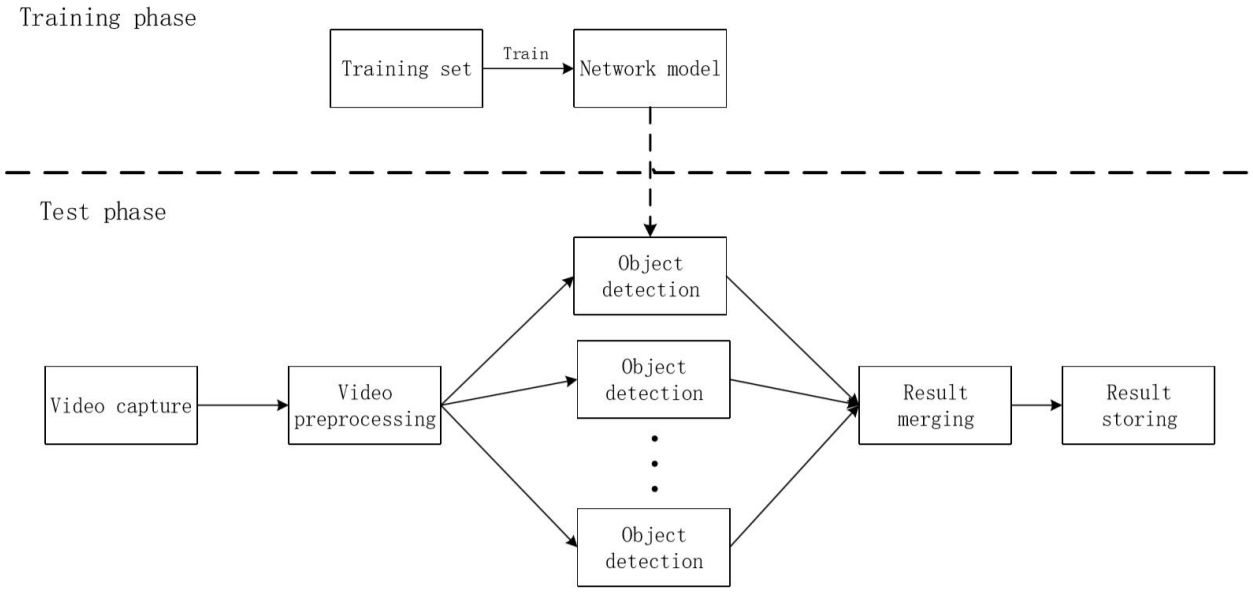

Figure 1. Work flow of the DEDLuS platform for video process

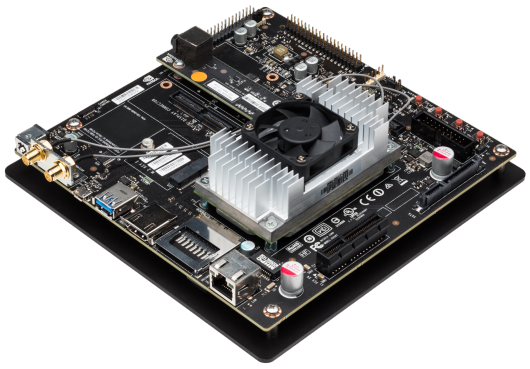

Figure 2. NVIDIA Jetson TX1 development board

Table I
PERFORMANCE OF THE DEDLUS PLATFORM

Number of board	Speed	Power consumption	network throughput
0	0fps	9W	1.06M/s
1	12fps	18.8W	8.53M/s
2	23fps	28W	14.93M/s
3	35fps	37.5W	21.48M/s
4	46fps	45.8W	26.89M/s

The first column represents the number of boards that are running object detection algorithm and 0 represents standby. The power consumption was only four board's measurement without any other devices.

Because of the time stamp on each frame, we can ignore the time sequence of processing frame. The detection result of each frame contains location, class and time stamp. They are merged together and reconstructed. Then the results are stored in text form. The display can be seen in figure 7 by loading video and text result simultaneously.

Table II
PERFORMANCE OF TITAN X GPU

Speed	Memory usage	GPU usage	Power consumption
40fps	630M	22%	90W

Our platform has continuously run for over one week in the experiment without any error. The above evaluations show that the DEDLuS platform have good scalability.

IV. RELATED WORK

There are many successful applications of distributed computing for deep learning or video processing, which provide us valuable reference. Weishan Zhang et al. [8] proposed a deep-intelligence frame work for on-line video processing. They used Hadoop for off-line processing and Storm for on-line processing. The framework has good availability, scalability and fault tolerance. However, the the DEDLuS platform is focusing on embedded devices based distributed real time video processing.

Qian Ning et al. [9] proposed mobile Storm, which was the first distributed stream processing platform for mobile. It can process real-time stream data using a cluster of mobile devices in a local network without offloading computation or remote servers. However, the algorithm running on the platform is face detector, which is far simpler than the object detection algorithm we used. And the computing capability of mobile phone is weak than the board we used.

Yucheng Li et al. [10] designed a real-time dynamic video target tracking and display system based on S5PV210

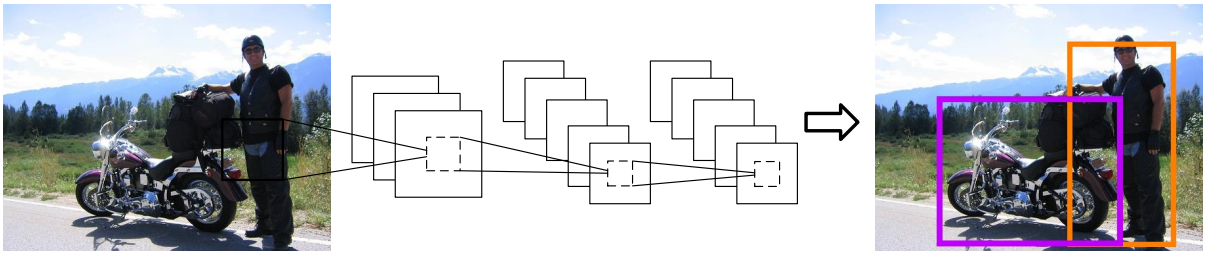

Figure 3. YOLO object detection method
There are mainly three step to analyze an image.(1)Resize the input image to 448×448, (2)input the image to a convolutional neural network, (3)localize and classify the object by the model's confidence.

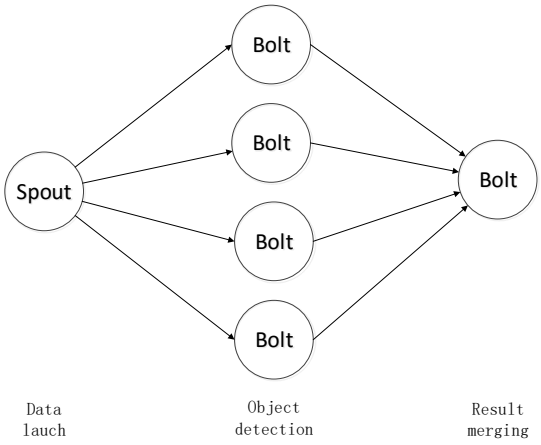

Figure 4. Storm Topology

microprocessor and embedded Linux operating system. The algorithm extracts moving vehicle foreground by Gaussian mixture model. Comparing with their work, we use deep learning method to conduct object detection on distributed embedded platform, which performs better.

Yahia et al. [11] implemented image processing in a Xilinx FPGA using System Generator for DSP, with a focus on achieving overall high performance, low cost and short development time. Our goal of low cost is similar with theirs, but applying distributed computing, our platform has higher computing capabilities.

V. Conclusion and Future Work

In this work, we introduced the distributed embedded deep learning platform and evaluated the performance of it by conducting real-time object detection in surveillance video. The results show that the DEDLuS platform has good availability, scalability and performance. We achieved the goal of conducting complex deep learning algorithm on small devices with low power consumption.

In the future, we will extend the current evaluations by adding more boards to find the bottleneck of the DEDLuS platform. Additionally, we will try more kinds of object detection algorithms and find the trade-off between speed and accuracy. Further more, we will connect the DEDLuS platform to cloud server. When facing a new class of objects, the DEDLuS platform will request cloud server to update model, which will achieve continuous learning.

Acknowledgment

The research is supported by the National Natural Science Foundation of China (Grant No. 61402533) and also Natural Science Foundation of Shandong Province (Grant No. ZR2014FM038), "Key Technologies Development Plan of Qingdao Technical Economic Development Area". Weishan Zhang has been supported by the start-up funds for "Academic Top-Notch Professors in China University of Petroleum."

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[5] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *arXiv preprint arXiv:1506.02640*, 2015.

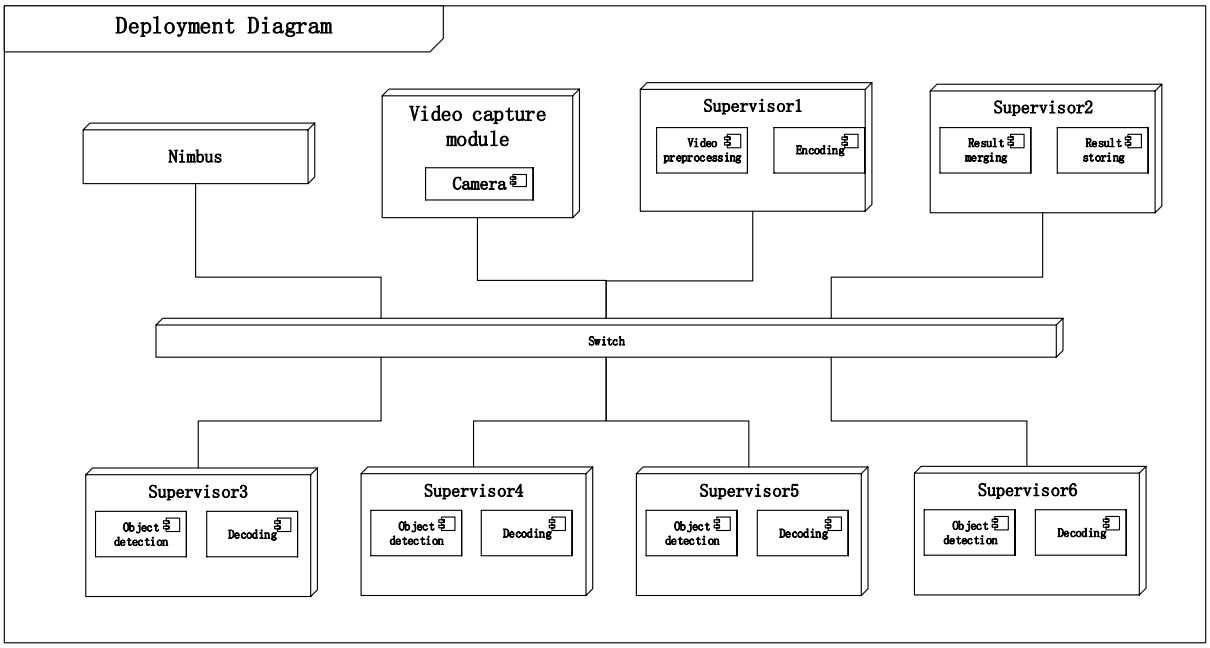

Figure 5. Deployment diagram of the DEDLuS platform

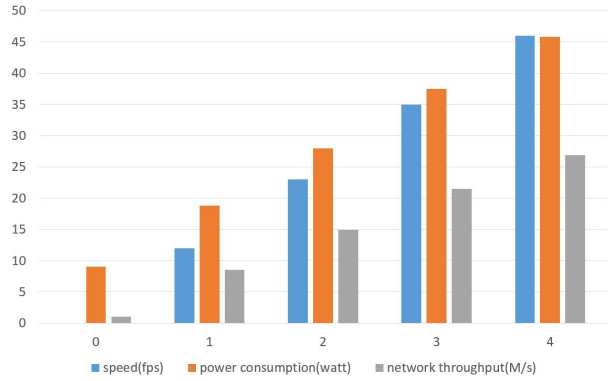

Figure 6. Performance of the DEDLuS platform

[7] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[8] W. Zhang, L. Xu, Z. Li, Q. Lu, and Y. Liu, "A deep-intelligence framework for online video processing," *IEEE Software*, vol. 33, no. 2, pp. 44–51, 2016.

[9] Q. Ning, C.-A. Chen, R. Stoleru, and C. Chen, "Mobile storm: Distributed real-time stream processing for mobile clouds," in *Cloud Networking (CloudNet), 2015 IEEE 4th International Conference on*. IEEE, 2015, pp. 139–145.

[10] Y.-C. Li and P. Sun, "The design of embedded video-based vehicle tracking system," in *Digital Manufacturing and Automation (ICDMA), 2013 Fourth International Conference on*. IEEE, 2013, pp. 1437–1440.

[11] Y. Said, T. Saidani, F. Smach, and M. Atri, "Real time hardware co-simulation of edge detection for video processing system," in *Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean*. IEEE, 2012, pp. 852–855.

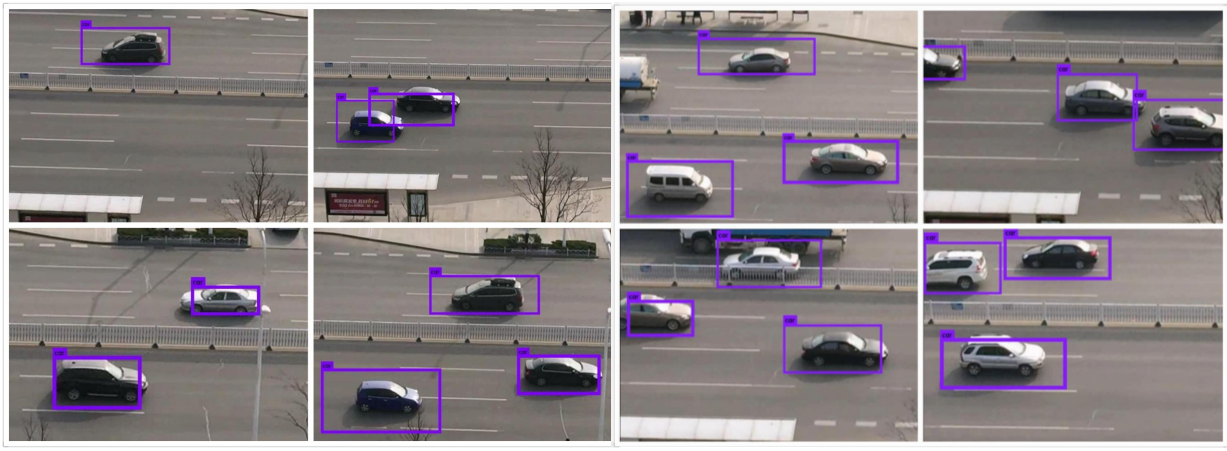

Figure 7. Result display