IMAGE RECOGNITION METHOD USING MODULAR SYSTEMS

Sungkwan Je[1], HongHanh Nguyen[2] and Junhyuk Lee[3]
KPST(Korea Platform Service Technology) Co., LTD.
Daejeon, Korea
jimmy[1], honghanh[2], jun[3]@kpst.co.kr

***Abstract*—In this paper, we implemented an open source based IDE (Integrated Development Environment) that modularizes various deep-learning techniques in different environments. This modularization system can schematize a complex structure deep-learning analysis into a simple form that help user to understand and use very difficult deep learning techniques conveniently.**

Keywords—deepLearning, Moduler, Caffe, IDE(integrated development environment)

I. INTRODUCTION

As digital contents are growing up, an effective and natural way to search and retrieval information among huge of data has been getting an attention. Deep learning is part of a broader family of machine learning methods based on learning representations of data. Research in this area attempts to make better representations from big-data, computer vision, automatic speech recognition, natural language processing, audio recognition and bioinformatics. It has been shown to produce state-of-the-art results on various tasks. Various deep learning architectures such as deep neural networks, convolutional deep neural networks, deep belief networks and recurrent neural networks have been developed [1].

But it is difficultly applied to application like face recognition, classification, retrieval information, and so on. It is too complex algorithm and required high performance hardware. So, some open source deep learning libraries such as Torch [2], Caffe [3], and Theano [4], have been provided. Caffe is a well-known and widely used deep learning framework among the deep learning library.

The framework is a BSD-licensed C++ library, with CUDA used for GPU computation, for training and deploying general-purpose convolutional neural networks. Caffe also provides well-supported bindings to Python/Numpy and MATLAB. Caffe is developed by the Berkeley Vision and Learning Center (BVLC). It was first designed for visual deep learning applications. But now, with the help of an active community of contributors on Github, it has been adopted and improved for other deep learning applications such as text, sound or time series data. Fast CUDA code and GPU computation makes Caffe perfect for research experiments and industry deployment [3].

II. CAFFE

A. Advantages

Caffe has following advantages that make it widely used in visual deep learning [5]:

- Caffe is most suited for convolution neural networks and is one of the fastest available implementations of convnets. With Caffe, researchers can build a state of the art model for their tasks through provided training, testing, finetuning and deploying tools. Each of these task come with detail explained examples making it easy to use and quickly applied.
- Open and modular design of Caffe allowing easy extension to new data formats, network layers, and loss functions. Lots of layers and loss functions have already implemented, and many examples that show how to apply them into various recognition tasks are available.
- Caffe model definitions are written as configuration files using the Protocol Buffer language. Caffe supports network architectures in the form of arbitrary directed acyclic graphs. Upon instantiation, Caffe reserves exactly as much memory as needed for the network, and abstracts from its underlying location in host or GPU. Switching between a CPU and GPU implementation is exactly one function call.
- Python and MATLAB bindings of Caffe are useful for rapid prototyping and interfacing with existing research code. Both languages can be used to construct networks and classify inputs.
- Caffe provides a model zoo with pre-trained reference models for visual tasks (AlexNet, GoogleNet, etc). This is something that we do not find in other framework.
- Besides, large community is also an advantage of Caffe. Many of the recently developed techniques (fully CNN, semantic segmentation…) are come from community of contributors. Caffe users have enthusiastic supports thanks to this community.

978-1-4673-9795-7/15 $31.00 © 2015 IEEE
DOI 10.1109/CSCI.2015.71

TABLE I. COMPARISON OF POPULAR DEEP LEARNING FRAMEWORKS

Framework	License	Core language	Binding(s)	CPU	GPU	Open source	Training	Pretrained models	Development
Caffe	BSD	C++	Python, Matlab	√	√	√	√	√	distributed
Cuda-convnet	Unspecified	C++	Python		√	√	√		discontinued
Decaf	BSD	Python		√		√	√	√	discontinued
OverFeat	Unspecified	Lua	C++, Python	√				√	centralized
Theano/Pylearn2	BSD	Python		√	√	√	√		distributed
Torch	BSD	Lua		√	√	√	√		distributed

B. Comparison to other deep learning frameworks

In comparison to related software, following table 1. is a summarization the landscape of a convolutional neural network software used in recent publications provided on Caffe's official page [5].

- Core language is the main library language
- Bindings have an officially supported library interface for feature extraction, training, etc.
- CPU indicates availability of host-only computation, no GPU usage.
- GPU indicates the GPU computation capability essential for training modern CNNs.

Caffe differs from other contemporary CNN framework in two major ways:

1) The implementation is completely C++ based, which eases integration into existing C++ systems and interfaces common in industry. Caffe supports both CPU and GPU modes which removes the barrier of specialized hardware for deployment and experiments on a trained model.

2) Reference models are provided for quick experimentation, without the need for costly re-learning. By finetuning for related tasks, these models provide a warm start to new research and applications. Caffe provides not only the trained models but also the recipes and code to reproduce them.

We can use Convnet-benchmarks', an easy benchmarking of all public open-source implementations of convnets, to see the comparison in speed between Caffe and other deep learning tools for convnets [6]. A benchmarking with AlexNet on ImageNet dataset shows that Caffe is pretty fast framework.

C. DIGITS

Caffe deep learning frameworks designed to let researchers create and explore Convolution Neural Networks (CNNs) and other Deep Neural Networks (DNNs) easily, while delivering high speed needed for both experiments and industrial deployment [7]. The researchers and data scientists, who want to experiment a new network, need to provide a network definition, and optimization parameters for training process in a plaintext followed the Caffe proto buffer format, fetch it to Caffe through command line tools. However, the command line tool of Caffe is not a friendly interface for all the users. This disadvantage causes some inconveniences for users when working with Caffe.

In the Caffe user community, there are so many ideas about developing a GUI version of Caffe tools. At the same time, Nvidia introduced DIGITS – an interactive deep learning development tool integrates Caffe framework with a friendly web based graphical user interface. DIGITS is new system for developing, training and visualizing deep neural networks. It puts the power of deep learning into an intuitive browser-based interface, so that researchers can quickly design the best Deep Learning Network for their data using real time network behavior visualization [7].

Advantages of DIGITS:

- Provides a user-friendly web based interface for training and classification images. After simple setup and launch steps, user can use it to train Deep Neural Networks with a few clicks. With DIGITS, users can create databases from images provided in various format and sources. Once having a database, users can configure network model and begin training.
- Allows user monitor the network training in real time during training process. Users can see the network's output such as loss, accuracy, and learning rate change after each step of training.
- Keeps track of existing databases and previously trained network models available on the machine, as well as the training activities in progress. Users can maximize accuracy by changing parameters such as bias, neural activation functions, pooling windows, and layers.
- Makes it easy to visualize networks and quickly compare their accuracies. When selecting a model, DIGITS shows the status of the training exercise and its accuracy, and provides the option to load and classify images while the network is training or after training completes.

III. PROPOSAL MODULAR SYSTEMS

Many deep learning techniques are difficult to learn to apply like image recognition, classification, retrieval information, and so on. They are too complex algorithms and required the skill to the neural network. Some currently provided deep learning frameworks are not easy for beginners.

In this paper, we implemented to the deep learning modular systems based on the IDE a set of standardized parts (or independent units) of Caffe framework like the homepage wizard described in Fig .1. In DIGITS, there are some stages provided such as select a dataset, select a Standard Networks, choose solver options in "Solver Options" and select GPU to training under web site. Users just choose the options provided in combo box style. It is quite simple for users to customize by user on their own.

As said, the most important part in building a convolution neural network is defining the network: how many inputs and outputs needed, how many layers and what kind of layers should be added, what parameter should be defined…Users have to provide all these information in Caffe required format using Google Proto Buffers. DIGITS does not allow users to define the network visually, only to visualize some configurations for training process. Although Caffe provides quite clearly examples and format for defining the network in a plaintext, this is still not a simple task. Users have to memorize the key work format to define a layer, this kind of layer with be defined what kind of parameter, and so on. Only a small typo (a mistake in typed text) in the definition can drive the network performance miles away from expectation of the designer. Rechecking typos in a long document is a painful and time consuming. Hence, a visual network editor is a necessity in a Caffe GUI tool. This visual editor will remove some headaches caused by trivial errors in writing a plaintext, and is useful for users who do not want to struggle with the complicated technical details. If we have a visual editor, by few clicks, or drag and drop actions, we can choose the algorithms used to build the network, experiment, and adjust parameters for better network performance.

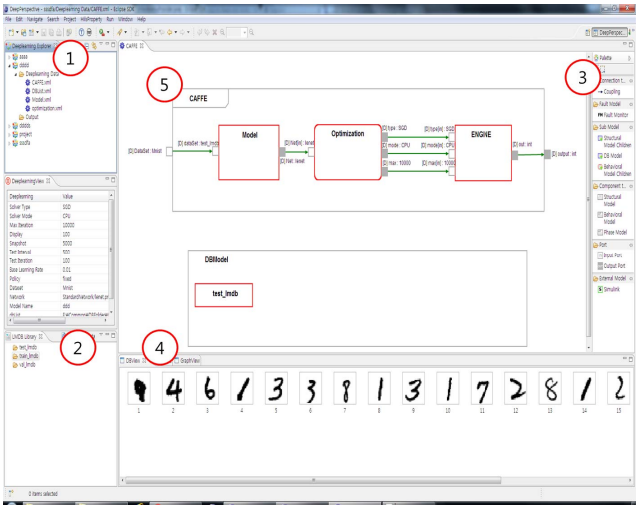

Fig. 1. Designed MODULAR SYSTEMS

With this modular deep learning system, a user can simply draw the structure using pieces of independent units of Caffe framework. Various convolutional neural networks such as Alexnet, Lenet, and GoogLenet can be loaded and customized by user on their own.

A. *Dataset*

In this framework, we provide most commonly used databases as MNIST handwritten digit database and dataset provided by Imagenet. These databases are converted and provided in Lightning Memory-Mapped Database (LMDB) format. LMDB is a tiny database with following great capabilities [8].

Ordered-map interface (keys are always sorted, supports range lookups)

Fully transactional, full ACID semantics with MVCC.

Reader/writer transactions: readers don't block writers and writers don't block readers. Writers are fully serialized, so writes are always deadlock-free.

Read transactions are extremely cheap, and can be performed using no mallocs or any other blocking calls.

Supports multi-thread and multi-process concurrency, environments may be opened by multiple processes on the same host.

Multiple sub-databases may be created with transactions covering all sub-databases.

Memory-mapped, allowing for zero-copy lookup and iteration.

Licensed under the OpenLDAP Public License

In Fig .1, database view ② displays LMDB module including datasets in LMDB format. When users choose a LMDB, images belong to chosen LMDB will be displayed in image view ④. User can check not only provided LMDB but also retrieved images.

B. *Model*

Convolutional Neural Networks (CNNs) are a particular type of deep models responsible for many exciting recent results in computer vision.

Deep learning training GUI tool system provides three standard and most commonly used deep learning models: LeNet, AlexNet, and GoogleNet.

- LeNet: LeNet is a convolution network designed for handwritten and machine printed character recognition. CNNs gained fame through the success of LeNet on the challenging task of handwritten digit recognition in 1989. For easily use proposed algorithm, master in deep learning Yann Lecun has introduced Torch – a deep learning framework. Facebook also uses CNN as main algorithm for its "deepface" project [9].
- AlexNet: After LeNet, it took a couple of decades for CNNs to generate another breakthrough in computer vision, beginning with AlexNet, which won the world wide ImageNet Large Scale Visual Recognition Challenge (ILSVRC).The researchers from University of Toronto proposed AlexNet algorithm in NIPS 2012. AlexNet has similar architecture with LeNet but with nine layers of Convolution [10].

- GoogLeNet: GoogLeNet algorithm was introduced in 2014 by a Google team lead by Szegedy of Google. GoogLeNet is a 22 layers deep network, it won the classification and object recognition challenges in Imagenet competition 2014 [11].

In Fig .1, the area marked with ① is model view, which provides three standard network models: Lenet, GoogLeNet, and AlexNet. User can choose these models by Drag and Drop actions. The area marked with ⑤ is architecture design area. The architecture consists of Deep learning component and Dataset component. Deep learning component includes Model – a network definition, Optimization (or Solver) - parameters used for monitoring and coordinating training process, and a training engine (in this case is Caffe). If user clicks "Model", detailed topology of selected network model will be displayed as seen in Fig. 2. Corresponding model definition will be saving in a Prototxt file using Google Proto Buffers. After that, this prototxt file will be provided to Caffe framework engine for executing algorithm. Moreover, user can change or create new convolutional network architecture in Fig. 2. by using layers provided in layer view marked with ③ in Fig .1. The layer view includes all the layers defined in Caffe framework. Users simply add, delete these layers, or change value of parameters in each layer to create their own network architecture. This visual network editor tool is an advantage of this modular system.

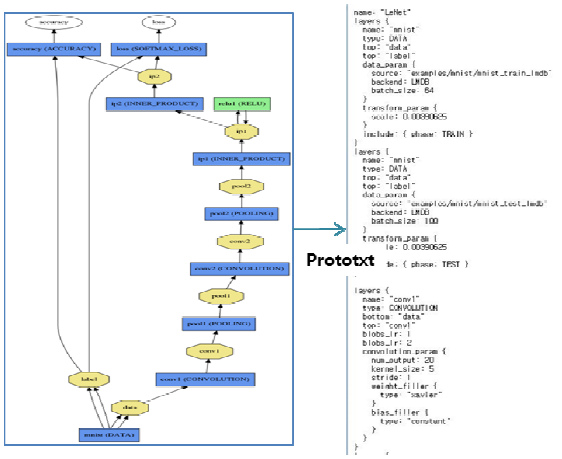

Fig. 2. LeNet architecture convert to prototxt.

C. Optimazation

Solver optimizes the network weights to minimize the loss over the input data by coordinating forward/ backward, weight updates and scoring. The solver takes the weights all the way from initialization to learned model through solver methods and corresponding provided solver parameters. In Caffe, there are three algorithms are used for address the general optimization problem of loss minimization. They are Stochastic Gradient Descent (SGD), Adaptive Gradient (ADAGRAD), and Nesterov's Accelerated Gradient (NESTEROV). During training process, network's weights and gradients will be updated according to chosen algorithm. For defining a solver, besides choosing solver algorithm, users also have to provide other solver parameters such as learning rate, learning rate policy and related parameters, weight decay, number of iterations on dataset, and so on. In original Caffe framework, these parameters a defined in a prototxt file followed Google Proto Buffer format. But with our deep learning GUI tool framework, user can easily choose value for each parameter by the mouse right-click.

Besides the lack of visual network editor, a browser-based interface of DIGITS is not appropriate for a long term and private researching. For developing a convolution neural network with high accuracy, researchers have to adjust the network parameters and train it repeatedly. They might want to keep it private until the research is finished. Because DIGITS runs a web server, all data and working results are stored on the server and shared with other users. Sometimes, this causes some inconvenience, so a desktop application might be more appropriate in this case. With a desktop app, users can have control over their data, network configuration and experiment results on their own desktop machine offline and privately.

D. Wizard

For beginner, we implemented deep learning wizard. A software wizard is a user interface type that presents to a user a sequence of dialog boxes that lead the user through a series of well-defined deep learning steps describes in Fig .3. Tasks that are complex, infrequently performed, or unfamiliar may be easier to perform using a wizard.

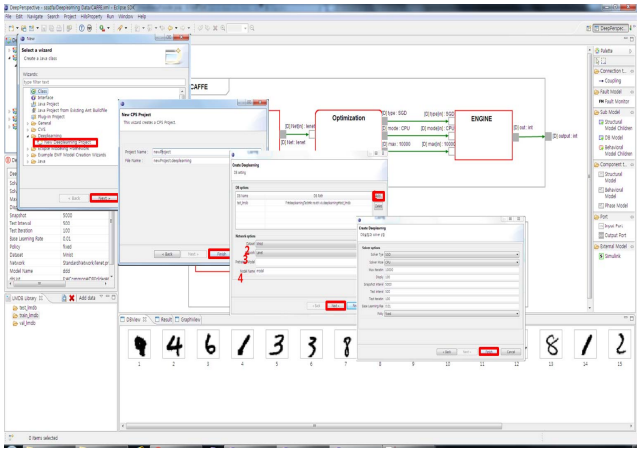

Fig. 3. Deep learning wizard for beginner.

Following this wizard, deep-learning system will use input DB to start training and testing processes. If an organization or a company has any business plan related to item recognition, it needs to start from collecting DB and then develop a new algorithm. There will be difficulties in algorithm development process. Using similar simple deep-learning algorithm GUI wizard several mouse-clicks, the user can analyze performance of the plan and then figure out its commercialization possibility. Moreover, with visual network editor, a network architecture can be designed and modified easily even for a user without deep knowledge in deep learning algorithm.

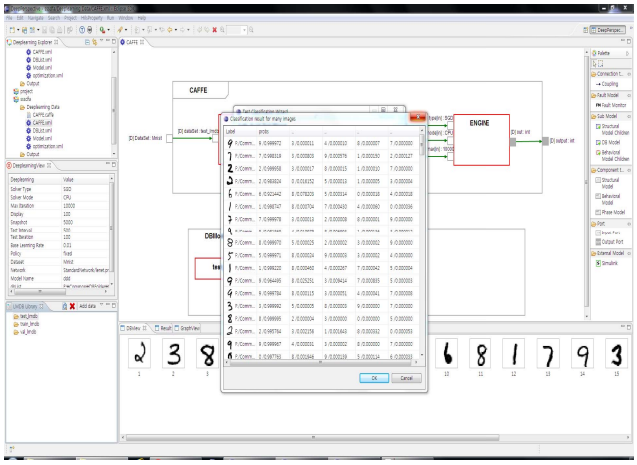

Fig. 4. Results of test processing MNIST.

Fig. 4 display the classification test result of a Caffe model trained on MNIST handwritten digit database using LeNet network. And Fig. 5 display the each layer trained result of a Caffe model trained.

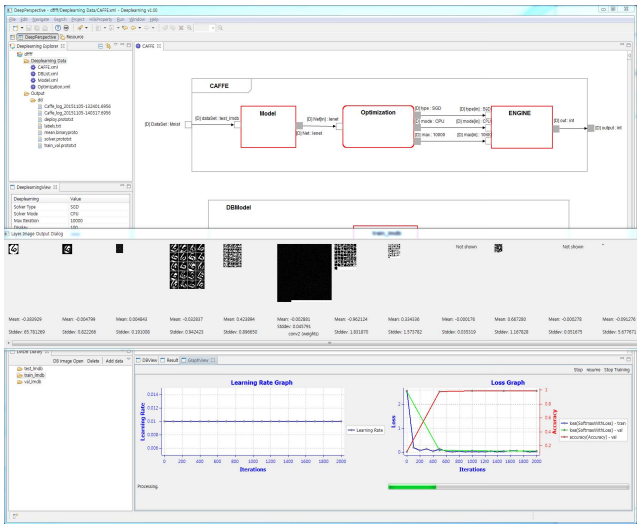

Fig. 5. Results of each layer trained data

Developed open-source based IDE is to help users to understand and use complex deep-learning algorithms more easily. Using this deep-learning modularization system and image recognition method will help modularize developing techniques and quickly check applicability of related services as well as increase performance of services that use collected DB.

ACKNOWLEDGMENT

This work was supported by the ICT R&D program of MSIP/NIPA [S0142-15-1017, Deep Learning Analysis tool and integrated system for Visual contents retrieval]

REFERENCES

[1] CVPR 2014 TUTORIAL ON DEEP LEARNING FOR VISION: [online]. https://sites.google.com/site/deeplearningcvpr2014/

[2] Torch: [online]. http://torch.ch/

[3] Caffe: [online]. http://caffe.berkeleyvision.org/

[4] Theano: [online]. http://deeplearning.net/software/theano/

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding" Report for ACM Multimedia Open Source competition, 2014.

[6] convnet-benchmarks: [online]. https://github.com/soumith/convnet-benchmarks

[7] DIGITS deep learning GPU training system: [online]. http://devblogs.nvidia.com/parallelforall/digits-deep-learning-gpu-training-system/

[8] Symas Lightning Memory-Mapped Database: [online]. http://symas.com/mdb/

[9] Koray Kavukcuoglu, Pierre Sermanet, Y-Lan Boureau, Karol Gregor, Michaël Mathieu and Yann LeCun, "Learning Convolutional Feature Hierachies for Visual Recognition", Advances in Neural Information Processing Systems, 23, 2010.

[10] A. Krizhevsky, I. Sutskever, and G. E Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", Advances in Neural Information Processing Systems, 2012.

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, "Going Deeper with Convolutions",2015 Conference on Computer Vision and Pattern Recognition, June, 2015.

[12] R. Collobert, S. Bengio, and J. Mariéthoz, "Torch: a modular machine learning software library". IDIAP Research Report 02-46, 30 October 2002, Retrieved 24 April 2014.

[13] Itamar Arel, Derek C. Rose, and Thomas P. Karnowski., "Deep Machine Learning—A New Frontier in Artificial Intelligence Research", IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE, Nov. 2010.