UChile Peppers 2017 Team Description Paper

Matías Pavez, Rodrigo Muñoz, Cristopher Gomez, Diego Baño, Lukas Pavez, José Astorga, Patricio Loncomilla, and Javier Ruiz-del-Solar

Department of Electrical Engineering - Advanced Mining Technology Center
Universidad de Chile
http://robotica-uchile.amtc.cl/pepper-index.html

Abstract. The UChile Peppers team is an effort of the Department of Electrical Engineering and the Advanced Mining Technology Center of the Universidad de Chile. The team is composed by former members of UChile Homebreakers team who worked with the Bender service robot. Furthermore, the UChileRT of the SPL soccer league brings support to our team in topics like the Naoqi framework (used by both the NAO and Pepper robot). Our main technological and scientific contribution will be the application of the deep learning paradigm in the Pepper robot. We will make possible the development of deep learning based vision applications for Pepper, that can work without any external hardware by using novel compression and quantization implementations of existing deep neural networks (DNNs). Moreover, at least two vision applications highly relevant to the RoboCup@Home community will be developed using DNNs, and made available as open source: semantic segmentation of indoor environments and object recognition. Additionally we want to contribute improvements to the ROS implementation of the Pepper robot.

1 Introduction

Even though the UChile Peppers is a newly founded team, there are a lot of ongoing investigations to make use of Pepper's full potential as a service robot, as can be seen in the following TDP. The main focus of the team is to develop different technologies so the robot can interact with the world around it in the best way possible and thus allowing better interaction between humans and robots, and improved functionalities in object recognition and navigation. Participating in the RoboCup also allows the team to gather and share information with all the other attending teams about different investigations regarding the development of a service robot using Pepper as a standard platform.

2 Background

UChile's team members have served the RoboCup organization in many ways: The UChile Robotics Team (UChileRT) has been involved in RoboCup competitions since 2003 in different leagues: Four-legged 2003-2007, @Home in 2007-

2015, Humanoid in 2007-2009, and Standard Platform League (SPL) in 2008-2016, Dr. Javier Ruiz-del-Solar was the organizing chair of the Four-Legged competition in 2007, TC member of the Four-Legged league in 2007, TC member of the @Home league in 2009, Exec Member of the @Home league between 2009 and 2015, and co-chair for the RoboCup 2010 Symposium. Among the main scientific achievements of the group to be highlighted, are four important RoboCup awards: RoboCup 2004 Best Paper Award, RoboCup @Home Innovation Award in 2007 and in 2008, and RoboCup 2015 Best Paper Award. In addition, UChile's team members have published a total of 36 papers in RoboCup Symposium between 2003 and 2016, 17 of them corresponding to oral presentations. In the SPL league where Nao robots are used, UChileRT reached the fourth place in the last three RoboCup World Competitions (2014 in Brazil, 2015 in China and 2016 in Germany). In addition, given our work in domestic robotics during the past 9 years, and our participation in the RoboCup@Home with our own-designed Bender robot, in which we use standard development tools and middleware such as ROS, we are in a privilege position to bring together our experience of software development from the Nao platform, and our experience in domestics robotics using our own developments and standard tools (e.g. ROS) into Pepper. Moreover, we have already developed ROS packages for the Nao [1], which we have made available as open source [2]. In summary, we feel that we are very prepared to start developing software for the Pepper robot, because we have a vast experience working with Naoqi at different levels, and we have already developed software for a service robot in the @Home league using ROS, which is the framework that we want to use in Pepper.

3 Current research

3.1 Application of the Deep Learning Paradigm in the Pepper Robot (Main research)

Deep Learning in Computer Vision and Robotics Deep learning has allowed a paradigm shift in pattern recognition, from using hand-crafted features together with statistical classifiers, to using general-purpose learning procedures to learn data-driven representations, features, and classifiers together. The use of the deep learning paradigm has facilitated addressing several computer vision problems in a more successful way than with traditional approaches. In fact, in several computer vision benchmarks, such as the ones addressing image classification, object detection and recognition, semantic segmentation, and action recognition, just to name a few, most of the competitive methods are now based on the use of deep learning techniques. In addition, most of the recent presentations at the flagship conferences in this area (e.g. CVPR, ECCV, ICCV) use deep learning methods or hybrid approaches that incorporate deep learning. Deep learning has already attracted the attention of the robot vision community [3][4]. However, given that new methods and algorithms are usually developed within the computer vision community and then transferred to the

robot vision community, the question is whether or not new deep learning solutions to computer vision and recognition problems can be directly transferred to robot vision applications. We believe that this transfer is not straightforward considering the multiple requirements of current deep learning solutions in terms of memory and computational resources, which in many cases include the use of GPUs. We want to address this important challenge in our work with the Pepper robot, by developing vision applications for the Pepper that can work without any external hardware. In order to achieve this, we are developing novel compression and quantization implementations of existing deep networks, which will be incorporated in one of the standard development frameworks used for implementing Deep Neural Networks (DNNs). Moreover, at least two vision applications highly relevant for the RoboCup@Home community are being develop using DNNs: semantic segmentation of indoor environments and object recognition.

Deep Compression and Quantization State-of-the-art vision systems based on DNNs require large memory and computational resources, such as those provided by high-end GPUs. For this reason CNN-based methods are unable to run on devices with low resources, such as smartphones or mobile robots, limiting their use in real-world applications. Thus, the development of mechanisms that allow DNNs to work using less memory and fewer computational resources, such as compression and quantization of DNNs, is an important area to be addressed. Mathieu et al. [5] propose computing convolutions by using FFTs. This technique enables reusing transformed filters with multiple images. FFT is efficient when computing responses for shared convolutional layers. Also, inverse FFT is required only once for each output channel. However, the use of FFTs requires additional memory for storing the layers in the frequency domain. Libraries such as CuDNN [6] are able to use FFTs when training and testing the networks. FFT is more efficient than normal convolutions when filter masks are large. However, the state-of-the-art ResNet [7] uses small 3x3 masks. The use of sparse representations is a useful approach for network compression. In [8], kernels are factorized into a set of column and row filters, by using an approach based on separable convolutions. A 2.5x speedup is obtained with no loss of accuracy, and a 4.5x speedup with a loss in accuracy of 1%. A procedure for reducing the redundancy in a Convolutional Neural Network (CNN) by using sparse decomposition is proposed in [9]. Both the images and the kernels are transformed onto a different space by using matrix multiplications, where convolutions can be computed by using sparse matrix multiplication. This procedure is able to make more than 90% of the parameters zero in an AlexNet [10], with a drop in accuracy that is less than 1%. In [11], a methodology for compressing CNNs is proposed, which is based on a three-stage pipeline: pruning, trained quantization, and Huffman coding. In the first stage, the original network is pruned by learning only the important connections, and a compressed sparse row/column format is used for storing the non-zero components. In the second stage, weights are quantized by using a 256-word dictionary for convolutional layers, and a 32-word dictionary for fully connected layers, enabling weight sharing. In the

third stage, the binary representation of the network is compressed by using Huffman coding. By using the proposed approach, AlexNet can be compressed from 240MB to 6.9MB, achieving a 35x compression rate, and VGG can be compressed from 552MB to 11.3MB, achieving a 49x compression rate, without loss of accuracy. In [12], an EIE (Efficient Inference Engine) is proposed for processing images while using the compressed network representation. EIE is 189x faster, and 13x faster when compared to CPU and GPU implementations of AlexNet without compression, without loss of accuracy. Two binary-based network architectures are proposed in [13]: Binary-Weight-Networks and XNOR-Networks. In Binary-Weight-Networks, the filters are approximated with binary values in closed form, resulting in a 32x memory saving. In XNOR-Networks, both the filters and the input to convolutional layers are binary, but non-binary non-linearities like ReLU can still be used. This results in 58x faster convolutional operations on a CPU, by using XNOR-Bitcounting operations. The classification accuracy with a Binary-Weight-Network version of AlexNet is only 2.9% less than the full-precision AlexNet (in top-1 measure); while XNOR-Networks have a larger, 12.4%, drop in accuracy. The compression and quantization of DNNs is an area under development, and of high relevance for allowing the use of DNNs in mobile robots. We have already started the testing of XNOR-Networks in the Nao robots. After this, the obtained results will be transferred for the case of Pepper.

3.2 Development Frameworks

Given the complexity in the design and training of DNNs, special learning frameworks/tools are used for these tasks. There are a wide variety of frameworks available for training and using deep neural networks (see a list in [14]), as well as several public databases suited to different applications and used for training (see a list in [15]). Four frameworks are under analysis for developing vision applications for the Pepper: Caffe, Torch, Theano and TensorFlow. One of them will be selected for our participation in the Robocup 2017. Our evaluation considers the availability of pre-trained models, and the obtained training/testing speed.

3.3 Vision Applications under development

A vision library based on DNNs will be developed. The first two applications to be developed for RoboCup 2017 are object recognition and semantic segmentation. Object recognition is a key ability in domestic robotic applications. DNNs obtain state of the art results in this task. We will use the Faster R-CNN network, which implements a Region Proposal Network (RPN) that is computed by applying a per-pixel fully connected network over the last shared convolutional layer [16]. The RPN network is in charge of generating region proposals. In the proposed research an already trained Faster R-CNN network will be used. It will be fine-tuned to recognize objects of domestic environments, and compressed/quantized to operate in real-time.

Semantic segmentation of indoor environments is a key ability to recognize different kinds of rooms and environment-objects such as the furniture. For this task we will use the SegNet network [17], which is based on an encoder and a decoder. The encoder corresponds to a set of convolutional layers with decreasing spatial size. The decoder takes the output data of the encoder as input, and applies a set of convolutional layers that scale up the data to become the same size as the original input image. In the proposed research an already trained SegNet network will be used. It will be fine-tuned for domestic environments, and compressed/quantized to operate in real-time.

4 Approach to be implemented on the Pepper

In order to develop software for the Pepper robot we have analyzed the following two options:

1. To use the ROS framework. ROS can be compiled in a virtual machine with Pepper's Operating System, and then transferred to the real robot [18]. There is a package named `naoqi_drivers` in ROS for publishing all the sensors and actuators [19], or the `pepper_dcm_bringup` [20] can be used to communicate directly with the NAOqi Device Communication Manager (DCM).
2. Regarding the Naoqi framework, we use it in our UChileLib software library, which controls our Nao robots for playing soccer in the SPL soccer league.

We have experience with the use of ROS in service robots and most functionalities that we have developed are ROS compatible, so at first we would prefer the ROS option, because it will allow using all software available in the ROS ecosystem (SLAM, navigation, arm movement, etc.) and the packages we have developed for our Bender robot. However, the limited computational power of Pepper can be a serious problem at the moment of running the required algorithms. In behalf of this, a process of customizing the correspondent ROS packages will be required. A final decision about which choice to use will require experimenting with a real Pepper (at the moment, we still have not receive our Pepper robot!). In the development of a vision library we will use OpenCV, and one of the mentioned DNNs development frameworks (Caffe, Torch, Theano or TensorFlow).

5 Re-usability of the System for other Research Groups

Our goal is that all our developments will be open source. That means that the following components will be open source and will be made available:

- Vision library based on DNNs for Pepper.
- Facial features recognition system based on DNNs library.
- Image segmentation system based on DNNs library.

6 Applications in the real world

Pepper is a social robot designed to charismatically relate to people, as it has many skills with the potential for great social impact. The UChile Peppers team will disseminate the associated technologies, contributing to the development of robotics itself and motivating the interest of children and young people, through talks at different schools and universities, continuing and supplementing the efforts started with the Bender robot of the UChile Homebreakers team[21].

Taking advantage of its social skills and their impact on children, Pepper can help teach different things in schools, not just robotics. In addition, his great charisma and the fact of being striking allows him to participate in social causes and entertainment such as being involved in a play, visiting children in a hospital or giving support to an aid campaign.

Moreover, Pepper will be very relevant in the development of students at university level as a research subject in different fields, especially in those dedicated to human-robot interaction, due to its vanguard technology.

7 Conclusions and future work

In this TDP, the plans and goals of the new UChile Peppers robotic team have been described. As a new team, they count on the support provided by the Universidad de Chile's other robotics teams, namely UChile Homebreakers and UChileRT. The team has great ambitions for the 2017 RoboCup Competition, and they are motivated to excel at the Standard Social Platform League with the Pepper robot.

To this end, various lines of research are being pursued, with the common goal of implementing powerful solutions in computationally limited platforms. This is important research because by their very nature, mobile robots such as Pepper are computationally limited.

The research efforts of the UChile Peppers team are focused around the application of the Deep Learning Paradigm through the use of Deep Neural Networks (DNNs), where the team is investigating novel implementations of compression and quantizations of DNNs. Alongside this research, the team is developing algorithms for Semantic Segmentation of indoor environments and for Object Recognition, using DNNs.

References

1. J. Ruiz-del-Solar L. Leottau-Forero, J.M. Yañez. Integration of the ros framework in soccer robotics: The nao case. *RoboCup Symposium 2013*, july 2013.
2. UChile Robotics Team. Uchile robotics team: B-human motion ros package. `https://github.com/uchile/bh-motion_ros-pkg`.
3. RSS 2016. Workshop deep learning for autonomous robots. `http://www.umiacs.umd.edu/~yzyang/deeprobotics.html`.
4. RSS 2016. Workshop are the skeptics right? limits and potentials of deep learning in robotics. `http://juxi.net/workshop/deep-learning-rss-2016/`.

5. Michael Mathieu, Mikael Henaff, and Yann LeCun. Fast Training of Convolutional Networks through FFTs. http://openreview.net/document/aa6ab717-ca19-47e1-a958-823b9a106ca9\#aa6ab717-ca19-47e1-a958-823b9a106ca9.
6. NVIDIA CUDNN GPU. Accelerated deep learning. www.businessinsider.com/.
7. Shaoqing Ren Jian Sun. Kaiming He, Xiangyu Zhang. Deep Residual Learning for Image Recognition. Available at: ArXiv:1512.03385.
8. A. Zisserman. M. Jaderberg, A. Vedaldi. Speeding up Convolutional Neural Networks with Low Rank Expansions. Available at: arXiv:1405.3866.
9. H. Foroosh M. Tappen B. Liu, M. Wang and M. Penksy. Sparse Convolutional Neural Networks. pages 806–814, 2015.
10. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. page 1097—1105, 2012.
11. W.J. Dally S. Han, H. Mao. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. 2016.
12. H. Mao J. Pu-A. Pedram M.A. Horowitz. S. Han, X. Liu and W.J. Dally. Eie: Efficient inference engine on compressed deep neural network. 2016.
13. J. Redmon M. Rastegari, V. Ordonez and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. 2016. Available at: arXiv:1603.05279.
14. Software links for deep learning. http://deeplearning.net/software_links/.
15. Datasets for deep learning. http://deeplearning.net/datasets/.
16. Ross Girshick Jian Sun Shaoqing Ren, Kaiming He. Faster r-cnn: Towards real-time object detection with region proposal networks. 2015. Available at: arXiv:1506.01497v3.
17. Roberto Cipolla Vijay Badrinarayanan, Alex Kendall. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. 2015. Available at: arXiv:1511.00561.
18. Pepper ros wiki. http://wiki.ros.org/pepper.
19. Naoqi_driver ros wiki. http://wiki.ros.org/naoqi_driver.
20. Pepper_dcm_bringup. http://wiki.ros.org/pepper_dcm_bringup.
21. Gonzalo Olave Mauricio Correa-Loreto Sánchez Patricio Loncomilla Luz Martínez, Matias Pavez and Javier Ruiz del Solar. Uchile homebreakers 2015 team description paper. *RoboCup Symposium 2015*, july 2015.
22. Official pepper documentation. http://doc.aldebaran.com.
23. Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.
24. Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016.
25. Gil Levi and Tal Hassner. Age and gender classification using convolutional neural networks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) workshops*, June 2015. http://www.openu.ac.il/home/hassner/projects/cnn_agegender.
26. Luca Mella and Daniele Bellavista. Recognizing emotional states in faces. https://github.com/luca-m/emotime.

Robot's Description

Pepper's hardware description is standard and can be found on the official documentation[22].

Regarding software, the Aldebaran supported ROS packages provides the core features to work with the robot (drivers and URDF model of Pepper). Futhermore, we have developed an improved version of the simulated robot for Gazebo, on top of the baseline provided by aldebaran. A major problem with the ROS approach is the limited computational resources that Pepper has. Normally the full suite of software required to accomplish a typical Robocup@Home test runs in at least two high-end computers. Considering this, an important amount of effort will be dedicated to optimize the originals ROS packages.

- Navigation: ROS Navigation stack will be used for mapping, localization and planning. However, due to the limited field of view of Pepper short-range lasers, the robot 3D camera will be used. This strategy is similar to the one implemented on the Bender robot[21].
- Face detection/recognition: For detection we use the `dlib` library[23]. For recognition we use the Openface research[24] based on the framework Torch.
- Facial features recognition: The following facial features have been developed: age and gender recognition [25] using CNN with the framework Caffe, and emotion recognition with the emotime library[26].
- Speech recognition and generation: We intend to use the already incorporated speech recognition and generation system Nuance.
- Arms control and two-hand coordination: The ROS MoveIt! package is used and has been successfully tested in the simulated version of Pepper.

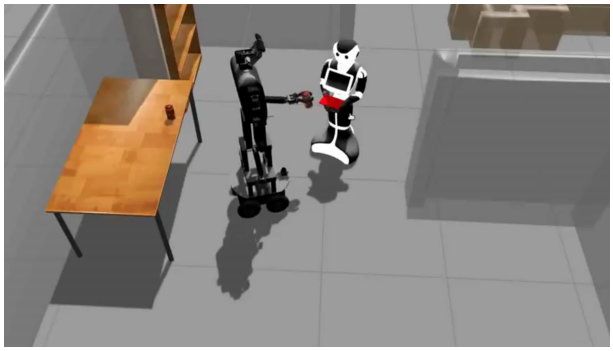

Fig. 1. Simulation of bender and pepper robots.

The implementation of high level behaviors is achieved with hierarchical state machines, programed with the `smach` library of Python. An emphasis is applied to the modularity and reutilization of code. Additionally, for the purpose of making the programming of high level behaviors more straightforward, a middle layer between the states machines and the ROS control interface exist based on the `robot_skill` interface designed by Tech United Eindhoven @HOME team.