INTERSPEECH 2017: Show & Tell Contribution
August 20–24, 2017, Stockholm, Sweden

Speech Recognition and Understanding on Hardware-Accelerated DSP

Georg Stemmer, Munir Georges, Joachim Hofer, Piotr Rozen, Josef Bauer, Jakub Nowicki, Tobias Bocklet, Hannah R Colett, Ohad Falik, Michael Deisher, Sylvia J Downing

Intel Corporation

firstname.lastname@intel.com

Abstract

A smart home controller that responds to natural language input is demonstrated on an Intel® embedded processor. This device contains two DSP cores and a neural network co-processor which share 4MB SRAM. An embedded configuration of the Intel® RealSpeech™ speech recognizer and intent extraction engine runs on the DSP cores with neural network operations offloaded to the co-processor. The prototype demonstrates that continuous speech recognition and understanding is possible on hardware with very low power consumption. As an example application, control of lights in a home via natural language is shown. An Intel® development kit is demonstrated together with a set of tools. Conference attendees are encouraged to interact with the demo and development system.

Index Terms: speech recognition, natural language understanding, neural network hardware

1. Introduction

Speech-enabled user interfaces have become increasingly popular in recent years. Most successful are intelligent personal digital assistants like Apple Siri, Microsoft Cortana, Amazon Alexa, and others. These assistants run on remote servers and are often designed in a way to be perceived as a personality which fulfills tasks on the user's order, simulating a butler. Correspondingly, these digital assistants cover a wide range of services, such as weather information, calendar management, media playback and messaging. Some services are customizable, e.g. to connect devices, enabling usages such as speech controlled lights.

The demonstration system described here follows a more lightweight usage model. No personal assistant is simulated, but a combined hardware/software solution enables a device to understand spoken requests. For instance, the user could ask a speech-enabled coffee maker for "an extra strong espresso", or use speech to control the lighting: "switch to reading mode". All processing is done locally on the device and requires no Internet connection. Depending on how the application is implemented, local processing has various potential advantages in privacy, security, latency, power consumption, and cost compared to a server-based approach. Hybrid combinations of local and server based speech services are also possible.

The system described in this paper showcases our approach to speech-enable devices with a dedicated chip. We demonstrate natural speech interaction for a sample target domain. It differs from old-fashioned command & control style speech interaction which has less acceptance in consumer applications as the user must memorize commands. For demonstration purposes at the conference we will use our system to provide a speech user interface for a Philips Hue smart light. Users can speak naturally to the system. For example, "it is too dark in here" triggers an action such as increasing the lamp's brightness.

Our demonstration system combines a hardware platform for low-power speech processing with continuous speech recognition and understanding software that minimizes memory and compute footprint. Although a lot of related work has been published in both the hardware and software domains, a comprehensive review is beyond the scope of this article. Nevertheless, we cite the work of Price et al. [2] who developed an embedded speech recognition chip that includes a neural network scoring accelerator and Viterbi decoder. In contrast, we present a design that is flexible enough to run speech recognition together with natural language understanding and acoustic preprocessing. Our speech recognizer uses techniques similar to [3], [4] in order to reduce the memory consumption and compute load, for example, dynamic graph composition, language model compression and fixed-point representations of neural networks. But unlike these publications our target platform is not a smartphone but a digital signal processor (DSP) which has much less memory and compute available. Correspondingly the demonstration system uses a much smaller recognition vocabulary.

The following sections present an overview of the hardware platform, followed by a description of the software and system, and finally experimental results and conclusions.

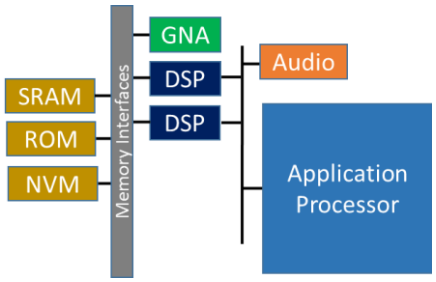

Figure 1 : *Development platform block diagram*

2. Overview of Development Platform

The platform on which the smart home user interface runs is a pre-release Intel® development board. At the time of submission we were not allowed to fully disclose the details and specifications. However, it will be publicly announced before Interspeech 2017 and a complete description will be available at the demonstration. As shown in Figure 1, the system contains two DSP cores and a neural network accelerator called GNA (Gaussian mixture model and neural network accelerator) [1] in in addition to an application processor. While the system is listening and waiting for user input, the application processor is held in low power sleep state. Only the audio, DSP, GNA, and memory subsystems are powered.

The neural network accelerator block (GNA) is a key part of the system-on-chip (SoC) design leading to extended battery life and very low thermal design power. GNA is a flexible, low-power, high performance streaming co-processor that offloads neural network inference operations from the DSP and application processor. GNA can operate while other parts of the SoC are in low power deep sleep mode. Figure 2 shows how GNA is used. Memory is pre-loaded by the DSP or application processor with the descriptors that define the required network topology. Parameter arrays describe functional properties of each layer (e.g., weights, biases, activation function) and data buffers hold inputs and outputs of each layer. Inference is initiated by the DSP or application processor during which the graph nodes (e.g., layers) are processed one after the other, until all are completed. During processing, data and parameters are streamed using multiple DMA engines that enable minimal buffering. After the inference operation is complete the DSP or application processor is notified to use the output data stored in memory.
The Memory Mapping Unit (MMU) enables mapping of memory in a paged system, and enables multiple applications to time share the use of the GNA block.

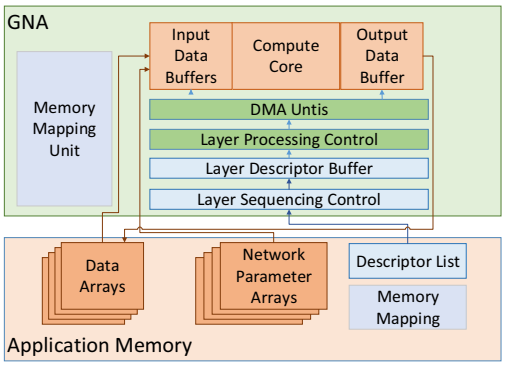

Figure 2 : *GNA block diagram and memory organization*

GNA supports many common neural network types (CNN, DNN, RNN, LSTM, GRU, etc.) via a set of fundamental layer types that may be combined to create a rich set of topologies and sizes. Invocation of neural network scoring operations is accomplished via a very simple application programming interface. The interface consists of a small number of functions to manage memory, acquire/release a handle to the device, initiate scoring, and sleep until completion or timeout. The API is accessible from C/C++ and Python as well as the Intel® Deep Learning SDK.

3. System Description

The automatic speech recognition (ASR) engine used in this paper is a continuous large-vocabulary speech recognizer which has been developed internally. It supports both HMMs with DNN and Chain models [6] based on long-short term memory networks (LSTM). The acoustic models are trained using the Kaldi open source toolkit [5] on several thousands of hours of manually and semi-automatically transcribed speech data. The acoustic training data has been expanded by adding noises and reverberation for better accuracy in noisy environments. The language model has been estimated with the MIT language modeling toolkit on a representative collection of text queries gathered from the application domain. The natural language understanding (NLU) module consists of an intent classifier and a parameter extraction component. Both are based on (recurrent) deep neural networks. As mentioned above, all neural networks are quantized to 8-16 bits for lower memory consumption, reduced memory bandwidth and faster scoring. The decoder combines dynamic composition of WFSTs together with language model compression for less memory usage and better memory locality.

4. Demonstration Flow and Results

The demonstration system consists of a development board, which is connected to a microphone array, the application processor running on a second board, and a PC which is connected to the application processor board. The PC is needed solely for demonstration purposes to provide more information about what is happening on the device. The development board runs the ASR engine and the NLU module on the DSP cores, leveraging GNA for neural network scoring. The application on the main processor captures the NLU results from the development board and triggers the corresponding actions on the Philips Hue light. The application furthermore connects to the graphical user interface on the companion PC in order to provide visual information during the demonstration.
The ASR has a vocabulary of 105 words. The NLU module recognizes six different intents for controlling a lamp: set brightness, set atmosphere, increase and decrease brightness, turn the lamp on and off. Additionally, a rejection class helps to suppress out-of-domain input and to properly handle false insertions of the ASR. The overall size of the speech recognition and understanding resources is less than 2 MB. A word error rate of 2.8% is achieved at an intent recognition error of 6% on an independent test set of 309 example utterances. The 309 utterances were recorded with a single microphone at a 60 to 120 cm distance from five unique subjects.

5. Conclusions

Our demonstration shows that a complete spoken language understanding system for home control can be realized on a low power SoC. The power consumption is reduced by running the entire system on a DSP together with a neural network scoring accelerator. The algorithms are optimized so that very little memory is required. As a next step, we are expanding the system to support vocabularies up to 100,000 words by jointly improving accelerators, platform, and software algorithms.

6. References

[1] M. Deisher and A. Polonski, “Implementation of Efficient, Low Power Deep Neural Networks on Next-Generation Intel Client Platforms,” IEEE SigPort, http://sigport.org/1777, March 2017.

[2] M. Price, J. Glass and A. P. Chandrakasan, “A Scalable Speech Recognizer with Deep-Neural-Network Acoustic Models and Voice-Activated Power Gating”, IEEE ISSCC, February 2017.

[3] X. Lei, A.W. Senior, A. Gruenstein, and J. Sorensen. "Accurate and Compact Large Vocabulary Speech Recognition on Mobile Devices", Interspeech 2013, vol. 1, 2013.

[4] R. Prabhavalkar, *et al*., "On the Compression of Recurrent Neural Networks with an Application to LVCSR Acoustic Modeling for Embedded Speech Recognition," IEEE ICASSP, Shanghai, 2016.

[5] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlıcek, Y. Qian, P. Schwarz, J. Silovský, G. Stemmer, and K. Veselý, “The Kaldi Speech Recognition Toolkit”, Proc. IEEE 2011 ASRU, December 2011.

[6] D. Povey, V. Peddinti, D. Galvez, P. Ghahrmani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur. "Purely Sequence-trained Neural Networks for ASR Based on Lattice-free MMI." Interspeech, 2016.