habana GOYA™

Goya™ Inference Platform White Paper

Aug 2019

Habana Goya™ Inference Platform

Table of Contents

1. Introduction 3
2. Deep Learning Workflows – Training and Inference 4
   3.1. SynapseAI® - Optimizer and Runtime 6
   3.1.1. Example SynapseAI® Workloads 6
4. GOYA™ Processor High-level Architecture 7
   4.1. Software Development Tools 8
   5. GOYA™ Inference Performance 9
   5.1 Googlenet 10
   5.2 BERT 11
      5.2.1. BERT Inference Using GOYA™ 12
6. Summary 14

List of Figures and Tables

Figure 1 - Deep Learning Workflows – Training and Inference 4
Figure 2 - GOYA™ Inference Platform – Software Stack 5
Figure 3 - GOYA™ High-level Architecture 7
Figure 4 - GOYA™ Platform Software Developments Tools 8
Figure 5 - BERT Two Phases: Pre-Trained Model and Specific Task Fine-Tuning 11
Figure 6 - BERT- Base SQuAD Inference Benchmark 13
Table 1 - GOYA™ Inference Performance Benchmarks 10

1. Introduction

Machine Learning (ML), a subfield of Artificial Intelligence (AI), is no longer science-fiction. One prominent field within ML is Deep Learning, in which the models are Deep Neural Networks (DNNs). Many problems in multiple domains (e.g., object detection and classification in images/videos, speech recognition and more), which several years ago were considered difficult for machines to solve, are now solved as accurately as by human beings and more, using deep learning models. As such, deep learning is a transformational technology.

A typical deep learning algorithm comprises of multiple operators, such as matrix multiplication, convolutions and other tensor operators, which add up to billions of compute-intensive operations. The execution of this massive amount of operations can be accelerated by using the inherent parallel processing that advanced GPUs offer. However, GPUs, which are primarily designed to render graphics in a super-fast way, are not optimized for deep learning workloads. The existing solution's inefficiency for deep learning workloads has a severe impact on the operational costs of cloud providers and data centers. To address this issue, a new class of software programmable AI processors are emerging, designed from the bottom-up for DNN workloads – Pure AI™ Processors.

Habana's Goya is a product line of AI processors dedicated to inference workloads. The HL-1000 processor is the first commercially available, deep learning inference processor, designed specifically to deliver superior performance, power efficiency and cost savings for cloud, data centers and other emerging applications.

For Information about the HL-100/102 line-cards, incorporating the HL-1000 processor, please see www.habana.ai

2. Deep Learning Workflows – Training and Inference

A deep learning workflow consists of two conceptual steps:
• Training - adjusts neural network model parameters to perform well on given data
• Inference - executes a trained neural network model on new data to obtain the output

For a model to address a specific use case, one first needs to train the model. Once the model is trained, it can be used (for inference). Both training and inference have similar characteristics, but different hardware resource requirements.

During training, a large dataset is processed to train a neural network model so that the model will distinguish between different statistical properties of the samples within the dataset. An example: recognizing images of apples from an arbitrary set of input images. After the model is ready for use, i.e., the model meets the accuracy goals set for successfully recognizing which images contain an apple and which do not, the model is ready for deployment. In a production environment, the model is used to efficiently recognize a new set of images to which it was not exposed during training. This operation is called inference and the goal of this phase is to infer attributes in the new data using the trained model (in our case, whether an apple appears in the image).

Training workloads require high-bandwidth memories with large capacity, in addition to the memory requirements for chip-to-chip communication. These requirements greatly increase solution BOM and its power consumption.

Inference is critical to support real-time applications such as neural machine translation, virtual assistant and many others. Therefore, inference is required to complete with low latency. In addition, providing high throughput with low batch sizes is also critical for inference of many applications. To provide comprehensive inference capabilities, an inference solution should provide high throughput, low latency, low power and be cost effective. Enter Goya!

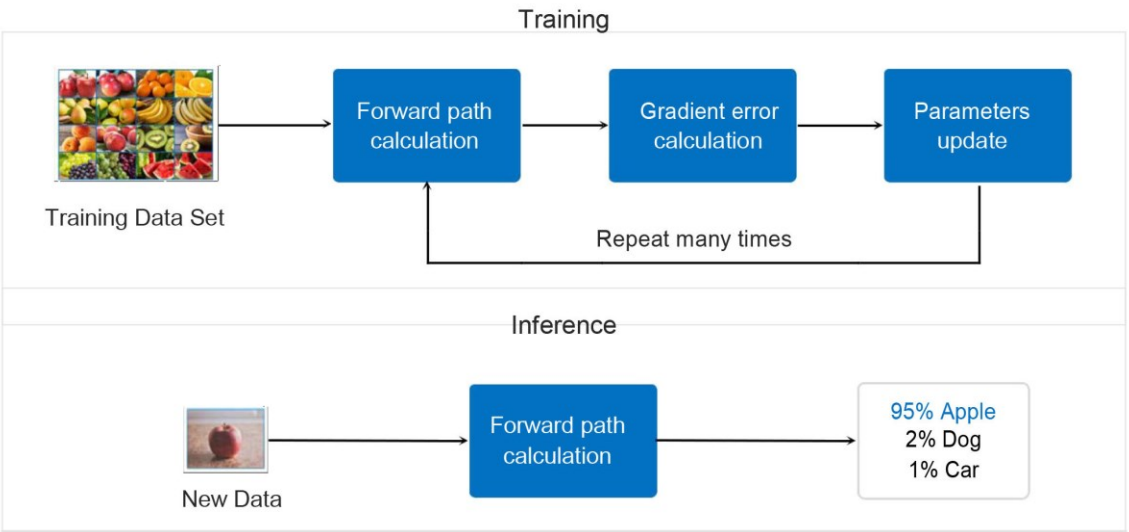

Figure 1 - Deep Learning Workflows – Training and Inference

3. GOYA™ Deep Learning Inference Platform

The Goya platform architecture has been designed from the ground up for deep learning inference workloads. It comprises a fully programmable Tensor Processing Core (TPC™) along with its associated development tools, libraries and compiler. The TPC™ with its accompanying software collectively deliver a comprehensive, flexible platform that greatly simplifies the development and deployment of deep learning systems for mass markets and cloud computing. The platform is capable of massive data crunching with low latency and high accuracy, as required by the workloads.

All major deep learning frameworks are supported, including TensorFlow, MXNet, Caffe2, Microsoft Cognitive Toolkit, PyTorch and Open Neural Network Exchange Format (ONNX). Habana also supports the Glow Machine Learning Compiler (HL-100 was the first AI Processor to be integrated as backend for the Glow ML compiler) and the Habana-Glow integration was open-sourced in Q1 2019. Starting with Linux 5.1 (released in May 2019) Habana HL-100 drivers are included in the official Linux distribution. The kernel mode drivers were upstreamed on Q4 2018.

Habana Lab's software stack seamlessly interfaces with all deep learning frameworks. A trained DNN model is first converted into an internal representation. Following this step, ahead-of-time (AOT) compilation is used to optimize the model and create a working plan for the network model execution on the Goya hardware.

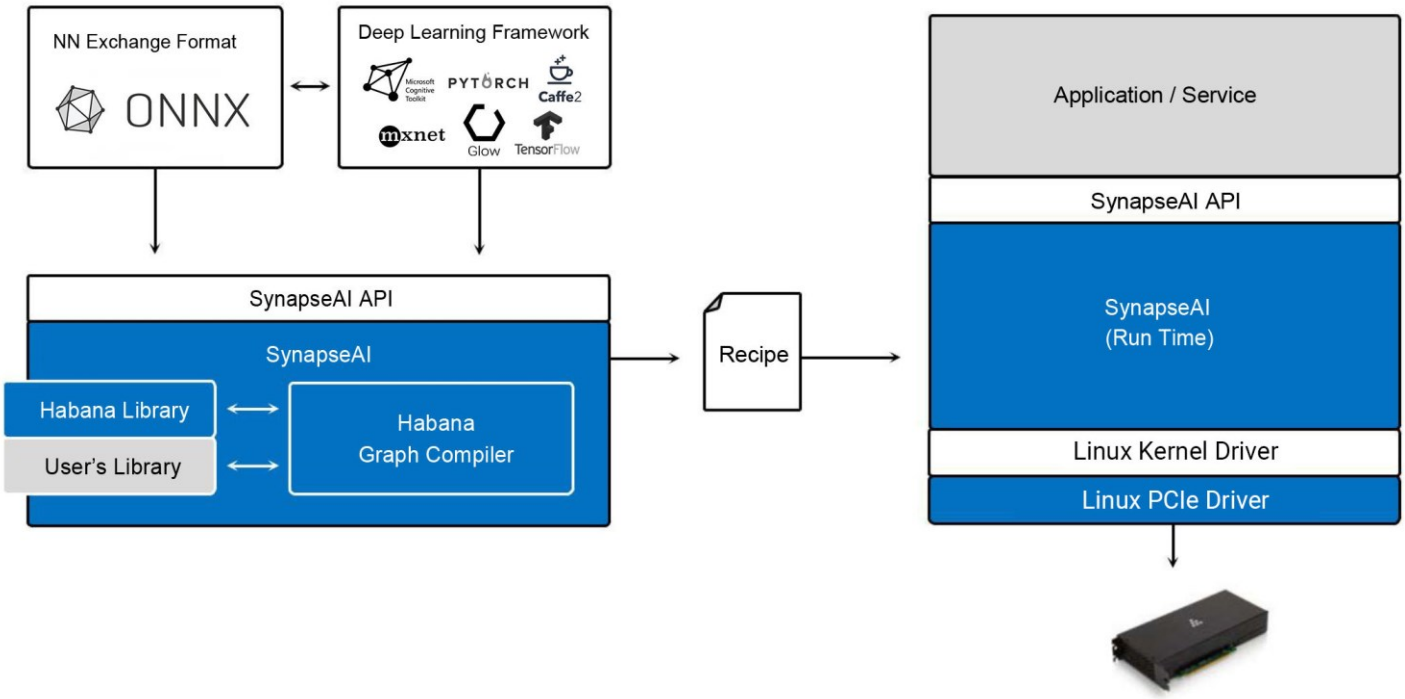

Figure 2 - GOYA™ Inference Platform – Software Stack

3.1. SynapseAI® - Optimizer and Runtime

Habana Lab's SynapseAI® is a comprehensive inference software toolkit that simplifies the development and deployment of deep learning models for mass-market use. The SynapseAI® software provides inference network model compilation (Graph Complier) and runtime.

The Goya™ platform is training platform-agnostic. A DNN can be trained on any hardware platform (GPU, TPU, CPU or any other platform) to obtain a model. SynapseAI® imports the trained model and compiles it for use on the Goya™ platform. The result is an optimized execution code in terms of accuracy, latency, throughput and efficiency.

SynapseAI® supports automatic quantization of models trained in floating-point format with near-zero accuracy loss. It receives a model description and representative inputs, and automatically quantizes the model to fixed-point data types, thus greatly reducing execution time and increasing power efficiency.

The user can specify the required level of performance gain and whether some accuracy may be sacrificed to improve performance.

SynapseAI® provides two APIs:

• C API for describing a neural network to be executed on the platform.
• Python API that can load an existing native framework (TensorFlow, MXNet, etc.) or via ONNX (that can be imported from other frameworks).

The SynapseAI Run Time is the user mode driver. It is a layer between the user's code to Goya's PCIe driver that is used when inference is executed.

3.1.1. Example SynapseAI® Workloads

Given TPC's programmability, Goya™ is a very flexible platform. It enables quick adoption of different deep learning models and is not limited to supporting specific workloads or workloads from a specific domain. Goya™ supports models from various domains, including, but not limited to, vision (for example, object detection, classification, segmentation), NLP (for example, Neural Machine Translation, text classification) and speech (recognition, synthesis) and Recommender systems.

4. GOYA™ Processor High-level Architecture

Goya™ is based on the scalable architecture of the TPC™, which uses a cluster of eight TPC™ cores. The TPC™ was designed to support deep learning workloads. It is a VLIW SIMD vector processor with ISA and hardware that was tailored to serve deep learning workloads efficiently.

The TPC™ is C-programmable, providing the user with maximum flexibility to innovate, coupled with many workload-oriented features such as:
- GEMM operation acceleration
- Special functions dedicated hardware
- Tensor addressing
- Latency hiding capabilities

The TPC™ natively supports the following mixed-precision data types:
- FP32
- INT32
- INT16
- INT8
- UINT32
- UINT16
- UINT8

To achieve maximum hardware efficiency, the SynapseAI™ quantizer selects the appropriate data type by balancing throughput and performance versus accuracy.

For predictability and low latency, Goya™ is based on software-managed, on-die memory along with programmable centralized DMAs. For robustness, all memories are ECC-protected.

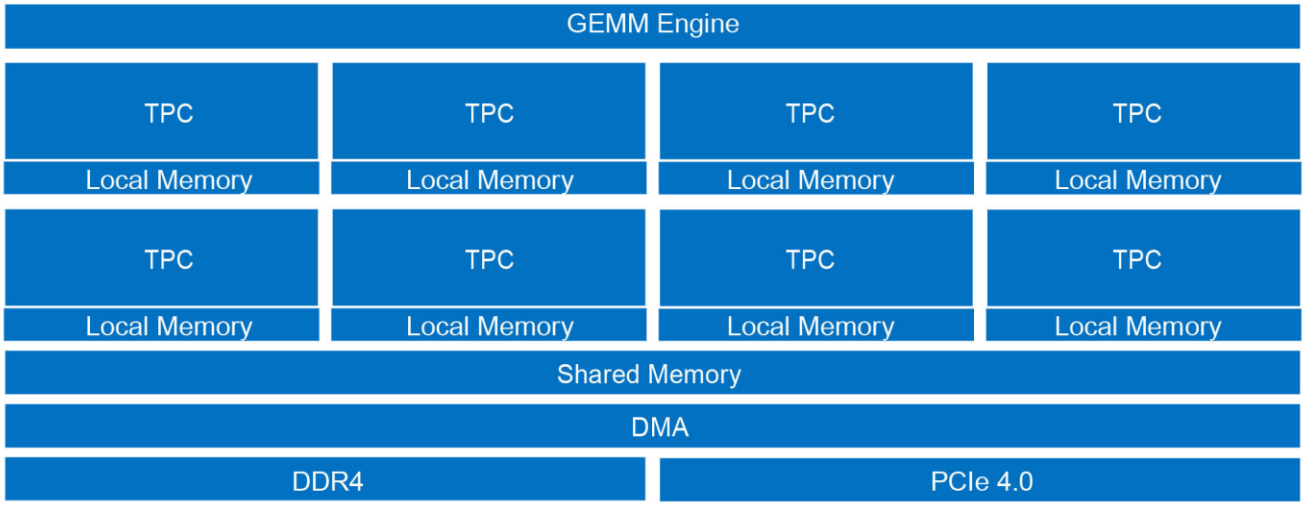

Figure 3 - GOYA™ High-level Architecture

4.1. Software Development Tools

SynapseAI™ enables users to unleash the power of deep learning by executing the algorithms efficiently using its high-level software abstraction. However, advanced users can perform further optimizations and add their own proprietary code using the provided software development tools. The Goya™ platform comes with state-of-the-art development tools, including visual real-time performance profiling and TPC™ development tools for advanced users (including an LLVM-C compiler) to combine third-party TPC™ kernels. Thus, users can quickly and easily deploy a variety of network models and algorithms.

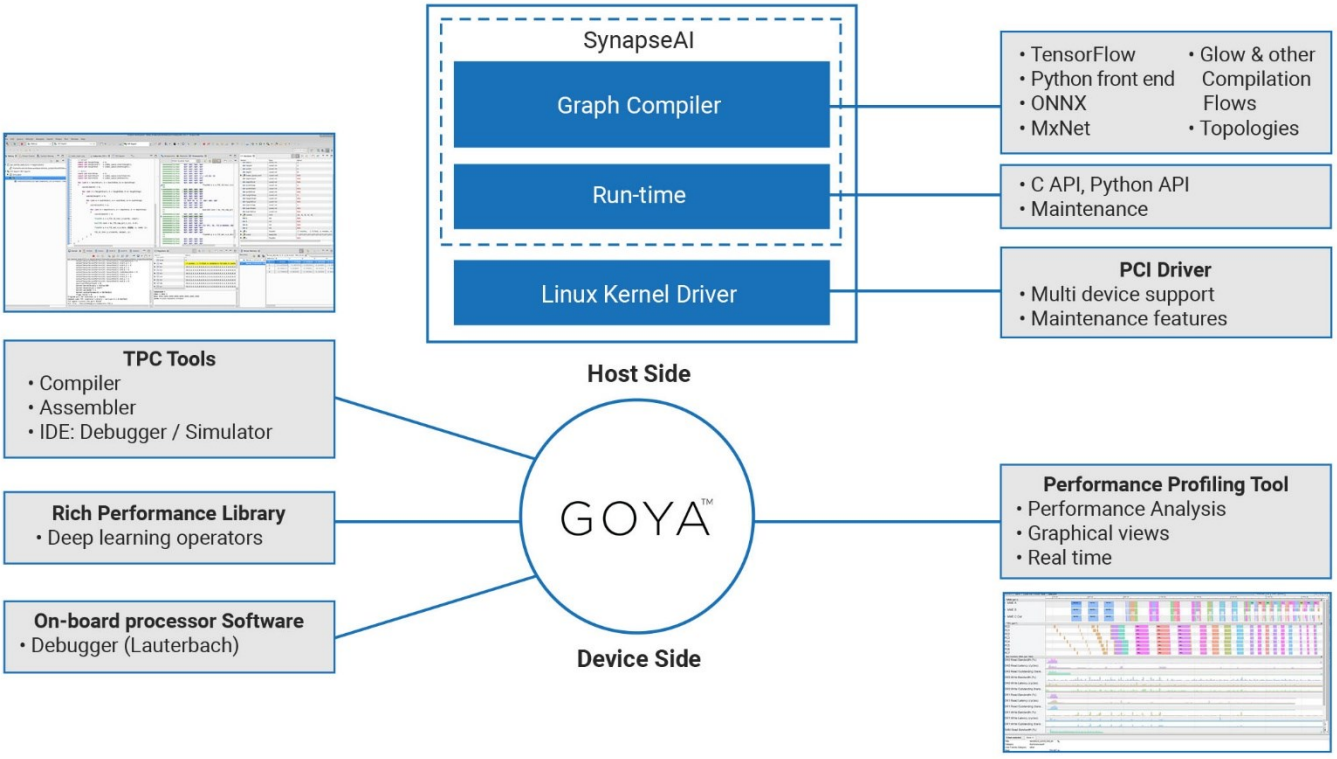

Figure 4 - GOYA™ Platform Software Developments Tools

5. GOYA™ Inference Performance

The key factors used in assessing the performance of the Goya™ inference platform compared to other solutions are throughput (speed), power efficiency, latency and the ability to support small batch sizes. The Goya HL-100 PCIE card provides throughput of 15,453 images/second for a ResNet-50 workload at a latency of ~1 msec, well below the industry requirement of 7 msec, Moreover, Goya's performance is sustainable at a small batch size, making it effective in various scenarios.
Below are performance results for various topologies from Tensorflow, ONNX public repositories and in-house topologies based on public sources.

Software: Ubuntu v-16.04.4 , SynapseAI v-0.2.0-1173
Hardware: Goya HL-100 PCIE card, Host: Xeon Gold 6152@2.10Ghz

Topology	Frame Work	Perf Metric	Batch Size	Throughput	Latency (msec)	Model Source
BERT SQuAD Base, Intermediate Size=3,072; Max Seq Len = 128	MXnet	Sentences/Sec	8	1108	7.2	https://gluon-nlp.mxnet.io/model_zoo/bert/index.html
			12	1273	9.4	
			24	1527	15.7	
Resnet18 v1	ONNX	Images/Sec	1	13156	0.1	https://s3.amazonaws.com/onnx-model-zoo/resnet/resnet18v1/resnet18v1.onnx
			4	26384	0.3	
			8	29961	0.5	
			10	30726	0.6	
Resnet34 v1	ONNX	Images/Sec	1	8345	0.2	https://s3.amazonaws.com/onnx-model-zoo/resnet/resnet34v1/resnet34v1.onnx
			4	15050	0.5	
			8	17254	0.8	
			10	18016	0.9	
Resnet50 v1	ONNX	Images/Sec	1	7466	0.2	https://s3.amazonaws.com/onnx-model-zoo/resnet/resnet50v1/resnet50v1.onnx
			4	13221	0.5	
			8	14546	0.9	
			10	15453	1.0	
Resnet101 v1	ONNX	Images/Sec	1	4533	0.4	https://s3.amazonaws.com/onnx-model-zoo/resnet/resnet101v1/resnet101v1.onnx
			4	8062	0.9	
			8	9086	1.3	
			10	9705	1.5	
Resnet152 v1	ONNX	Images/Sec	1	2912	0.6	https://s3.amazonaws.com/onnx-model-zoo/resnet/resnet152v1/resnet152v1.onnx
			4	5138	1.2	
			8	5737	1.9	
			10	6075	2.2	

Topology	Frame Work	Perf Metric	Batch Size	Throughput	Latency (msec)	Model Source
Inception v1	TF	Images/Sec	1	11296	0.2	http://download.tensorflow.org/models/inception_v1_2016_08_28.tar.gz
			4	15116	0.5	
			8	15881	0.9	
			10	16011	1.0	
Inception v3	TF	Images/Sec	1	2587	0.6	https://storage.googleapis.com/download.tensorflow.org/models/inception_v3_2016_08_28_frozen.pb.tar.gz
			4	4217	1.3	
			8	4389	2.3	
			10	4412	2.8	
SSD300 vgg16	Mxnet	Images/Sec	1	1447	1.1	VGG16 VGG backbone: http://github.com/zhreshold/mxnet-ssd/blob/master/symbol/legacy_vgg16_ssd_300.py
Googlenet BN no lrn	ONNX	Images/Sec	1	12573	0.1	Developed inhouse based on Googlenet with batch norm and w/o LRN (pls refer to section 5.1 below)
			4	16962	0.5	
			8	18243	0.8	
Yolo v2 1088x1920	pytorch	Images/Sec	1	166	6.6	https://github.com/marvis/pytorch-yolo2
Yolo v2 960x960	pytorch	Images/Sec	1	290	4.0	https://github.com/marvis/pytorch-yolo2
Tiny Yolo v2 960x960	pytorch	Images/Sec	1	1772	1.1	https://github.com/marvis/pytorch-yolo2
Yolo v3 960x960	pytorch	Images/Sec	4	361	12.0	https://github.com/eriklindernoren/PyTorch-YOLOv3

Table 1: GOYA™ Inference Performance Benchmarks

5.1 Googlenet

The Googlenet topology was developed in 2014, opting to use Local Response Normalization (LRN) over Batch Normalization. LRN is more computationally expensive. We have improved the throughput when replacing LRN with BN, which also results in improved accuracy, 72% top1 versus 68% originally. As this replacement reduces the amount of computation while improving accuracy, we expect it to perform better on any inference processor. Therefore, we are offering this improved topology (Googlenet_bn_no_lrn) on demand. Other than replacing LRN with BN, the topology has exactly the same structure.

5.2 BERT

One of challenges implementing natural language processing (NLP) applications is the cost of training resources. Many NLP (natural language processing) use cases require large amounts of data in ordered to be trained. In order to reuse and save training compute, time and costs, researchers have developed a pre-training method, training general purpose language representation models, based on enormous amount of unannotated text. This pre-trained model can then be fine-tuned using small amounts of data and with an additional, lean, output layer create state-of-the-art models for a wide range of tasks like question answering, text summation, text classification (e.g. spam or not) and sentiment analysis, resulting also in substantial accuracy improvement, compared to training on these datasets from scratch.

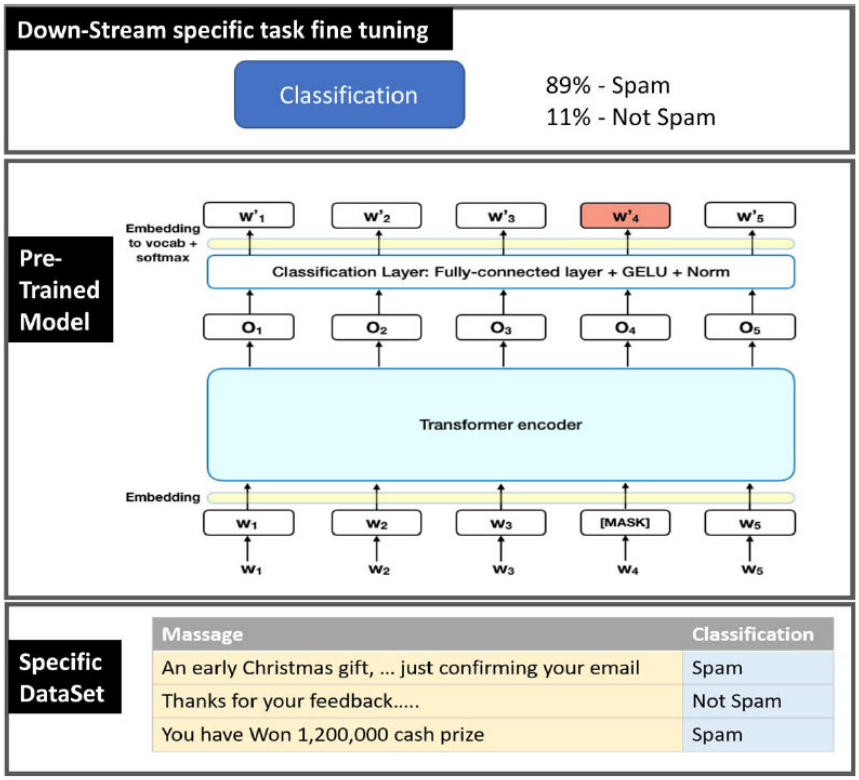

Figure 5 - BERT Two Phases: Pre-Trained Model and Specific Task Fine-Tuning

The approach pioneered by BERT constitutes a paradigm shift in language modelling using deep learning models, by offering a strong representation that can be used in transfer-learning new tasks. New models and techniques based on the BERT architecture have quickly taken their place as the de-facto standard in text understanding. BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. It consists of multiple, multi-head self-attention layers, that leverage a bidirectional context conditioning on the input text, allowing the model to learn the context of a word based on all its surroundings (beginning and end of each sentence).

5.2.1. BERT Inference Using GOYA™

The GOYA inference architecture delivers exceptional inference performance for the BERT workloads. All of BERT operators are natively supported by the Goya HL-1000 Inference processor and can be executed on Goya without any host intervention. A mixed precision implementation is deployed, using Habana's quantization tools which set the required precision per operator to maximize performance while maintaining accuracy.
BERT is amenable to quantization, with either zero or negligible accuracy loss, while GEMM operations are execute at INT16, Some other operations, like Layer Normalization, are done in FP32.

Goya's heterogenous architecture is an ideal match to the BERT workload, as both the GEMM engine and the Tensor Processing Cores (TPCs) are fully utilized, supporting low batch sizes at high throughput. Goya's TPC provides significant speedup when calculating BERT's nonlinear functions resulting in leading benchmark results. In addition, the Goya's software-managed SRAM allows increased efficiency between different memory hierarchies, while executing.
Below are performance results measured on Goya, for a question answering task, identifying the answer to the input question within the paragraph, based on Stanford Question Answering Database (SQuAD). The tested topology is BERTBase encoder, Layers=12 Hidden Size=768 Heads=12 Intermediate Size=3,072 Max Seq Len = 128.

To quantify and benchmark this task, we used Nvidia's demo release and model:
https://github.com/NVIDIA/TensorRT/commits/release/5.1 commit:
7a0772f06997e3bdd1e68e55a52f9851e17aadc8

Model used: https://api.ngc.nvidia.com/v2/models/nvidia/bert_tf_v1_1_base_fp32_128/versions/1/zip

Below are the platform configurations and results.

Please note that both Goya and T4 implementation using mixed precision of 16 bit and FP32 data types. Goya's mixed precision quantization resulted in a comparable accuracy to the original model trained in FP32, such that the error is at most 0.11% (Verified on SQuAD 1.1 and MRPC tasks).

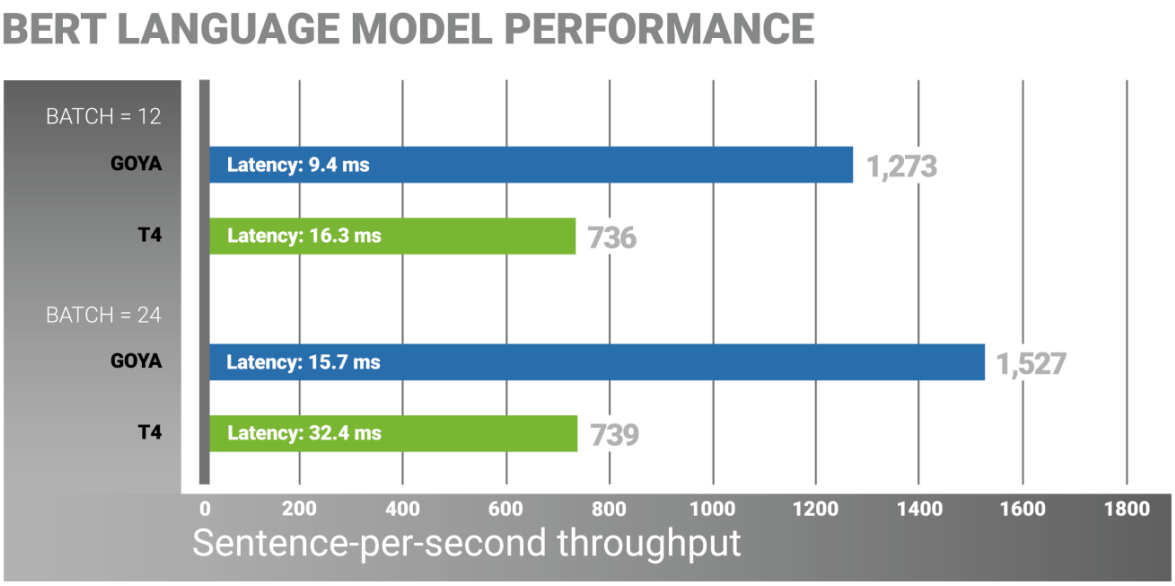

Figure 6 - BERT- Base SQuAD Inference Benchmark

Goya Configuration:

Hardware: Goya HL-100; CPU Xeon Gold 6152@2.10Ghz
Software: Ubuntu v-16.04.4; SynapseAI v-0.2.0-1173

GPU Configuration:

Hardware: T4; CPU Xeon Gold 6154 CPU @ 3Ghz/16GB/4 VMs
Software: Ubuntu-18.04.2.x86_64-gnu; CUDA Ver 10.1, cudnn7.5; TensorRT-5.1.5.0

The Goya processor delivers 1.67x to 2.06x (batch 12/24 respectively) higher throughput than the T4 on the SQuAD task, all at significant lower latency. As the BERT base model is foundational to many NLP applications, we expect similar inference speedups for other NLP applications based on it with their own fined tune layers. Moreover, the Goya architecture is scalable and keep its high efficiency while increasing the batch size.

6. Summary

Deep learning revolutionizes computing, impacting enterprises and the services and experiences they can deliver across multiple industrial and consumer sectors. Deep neural networks are becoming exponentially larger and more complex, driving massive computing demands and costs. Modern neural networks are too compute-intensive for traditional CPUs, and even GPUs. The future belongs to dedicated high throughput, low latency, low power AI processors.

Inference performance is measured by throughput, power efficiency, latency and accuracy, all of which are critical to delivering both data center efficiency and great user experiences. The Goya HL-1000 is a world class, high performance, AI processor for inference workloads. It is supported by state-of-art software development tools and a full software stack.

An effective deep learning platform must have the following characteristics:

• A processor that is custom-built for deep learning (e.g., Goya HL-1000).
• That's software-programmable (e.g., SynapseAI™)

Habana Labs' Goya™ AI Inference Platform meets all these requirements.