Intel® instances for AI workloads on GCP

A step by step instructions user guide

Devarampati Vinod

11/29/19

V1.0

CONTENTS

DISCLAIMERS

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation.

Performance varies depending on system configuration.

No product or component can be absolutely secure. Check with your system manufacturer or retailer or learn more at intel.com

INTRODUCTION

Machine Learning (ML)/ Deep learning (DL) are next generation AI technologies bringing new revolutions in the automation. With the availability of the data and compute power, DL applications accuracy is raising above the human level accuracy in certain complex technical challenges which are difficult to handle with previous technologies.

This user guide is to recommend the right GCP cloud instances along with relevant services and tools to launch and utilize the cloud instances effectively for machine learning and deep learning workloads on intel® architectures. Here are some of the advantages of using the intel® Xeon® scalable family of servers for ML/DL workloads -

- Great for handling high memory intensive workloads and 3D-CNN topologies used in Medical imaging, GANs, seismic analysis, genomic sequencing etc.
- Great for real-time inferencing even at lower batch sizes due to flexible core control with simple numactl commands.
- Great ecosystem supported hardware for distributed training over large clusters (i.e. compute at source of data), thereby avoiding the need of adding bulk storage and expensive cache mechanisms for training over scale-up architectures (one server with many accelerator cards)
- Great TCO due to support of heterogeneous workloads (HPC/BigData/AI) on same cluster.
- SIMD capabilities to address the computational requirements of many practical deep learning applications.
- Single production level scalable infrastructure for both training (also transfer learning) and inference.

Typical Deep learning application development and deployment involves the following stages –

These stages require multiple resources along with their orchestration mechanisms given below -

- Computational power
- Memory
- Storage for Datasets
- Communication link between compute nodes
- Optimized software

Selection of the right resources needs comprehensive study and benchmarking for great TCO (Total Cost of Ownership). Here are the technical capabilities and the ecosystem that comes along with Xeon® Scalable family of servers due to Intel® legacy in computation -

COMPUTE POWER AND MEMORY

Usually the requirement for compute power and its resources varies depending on the type of algorithm (ML) or neural network topology (DL) used for your application. Although, compute power is measured with raw flops, arithmetic intensity of an algorithm decides the utilization of raw flops. Xeon® Scalable family of servers are designed to handle both serial and parallel workloads with flexible core control to get full utilization of memory and cores for most of the workloads. Here are some capabilities which gives a glimpse of their strength to use for deep learning -

- 512-bit SIMD capability (AVX-512 engines) with which each core can compute up to 64 flops per cycle which is enough for most of the practical deep learning applications [1].
- DL Boost with VNNI instruction fused into AVX-512 pipeline which can compute up to 256 tops per core per cycle accelerating the INT8 inference throughput and reducing the latency (available in second generation of Xeon® scalable family i.e. cascade lake and future generation of servers).
- Increase in memory capacity per socket (up to 4.5TB) with DCPMM in the Cascade lake servers to support 3D CNN and high dimensional tensor computation.

STORAGE

Often datasets size required for deep learning training will be hundreds of GB which requires fast storage access mechanisms. Due to the non-availability of efficient scalability methods for distributed training, many deployments in the early stages of DL revolution encouraged towards the scaleup architecture adding to the below overheads –

- limiting the scalability to single node with multiple accelerator cards.
- Latency involved in distributed datasets to move to single node (server) before training
- Adding bulk storage at single node or expensive cache mechanisms for fast training

Recent development of MPI based distributed training with Horovod made the scaling easy across nodes. Horovod with Xeon® clusters can be utilized to scale over thousands of nodes, thereby reducing the time to train and overheads mentioned above. Xeon® ecosystem has the support for multiple storage mechanisms (block/file/object storage) to go for production level scalability.

COMMUNICATION LINK

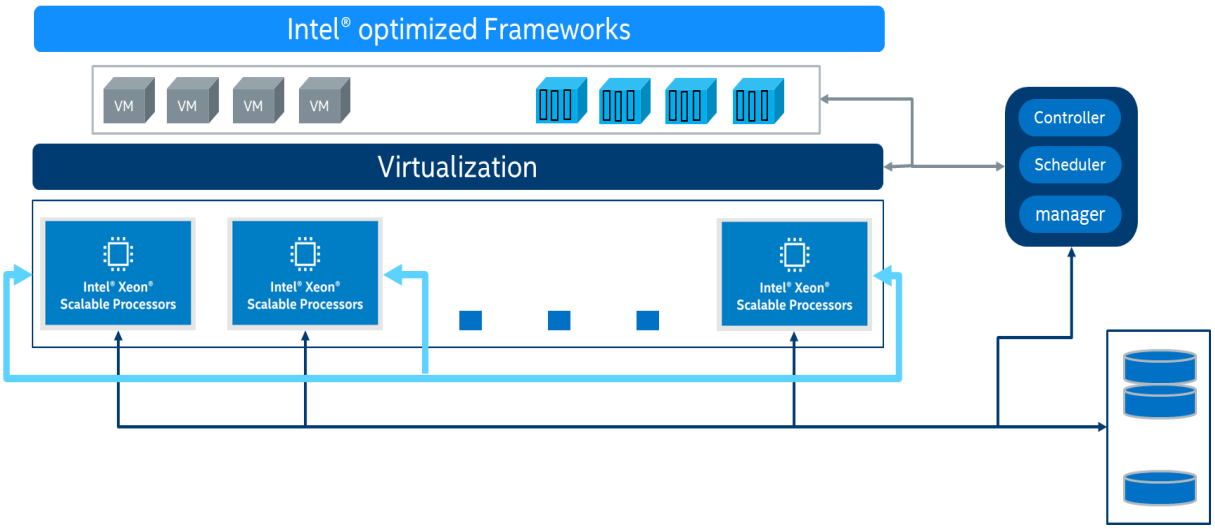
For DL inference and Single node training, there won't be any need of high network Band width communication between compute nodes. So 10Gig link will be good enough for most of the development and deployment. But for distributed computing, communication latency decides the scalability performance. Practical experimentation showed that 25 Gig ethernet will be enough to get above 90% scaling up to 8 nodes (varies based on computation latency of a topology). But beyond 8 nodes, it is recommended to use Ethernet with RDMA, IB or OPA as the communication link. Intel® Xeon® Servers supports all these mechanisms seamlessly.

OPTIMIZED SOFTWARE

Intel® optimized many publicly available popular frameworks by integrating open source MKL-DNN library which extracts the full potential of Xeon® servers for deep learning algorithms [2]. Along with optimized frameworks, intel® has heterogeneous inferencing toolkit like openVINO™ to provide the seamless deployment across multiple hardware [3]. Bigdata analytics users can move from ML analytics to sophisticated intelligent analytics with BigDL library fused seamlessly into spark framework along with Analytics Zoo [4].

Below is the block diagram which stitches all these resources together for scalable deep learning training and inference. Here are the some of the prominent mechanisms used for scaling DL applications –

- **Compute nodes** – Gold(6 series) and platinum servers of Intel® Xeon® Scalable family
- **Management and storage nodes** – Bronze or Silver servers of Xeon® scalable family or Xeon® Broadwell
- **Storage mechanisms** – CEPH, S3, NAS, Luster, GlusterFS etc.
- **Virtualization Mechanisms** - KVM, VMware, XenServer, Hyper-V etc.
- **Container orchestration** - Docker Swarm, Kubernetes*, Mesos, OpenShift etc.
- **Schedulers and Management** – PBS, SLURM, VMSphere, XenCentre, OpenStack* etc.



All the above technology ecosystem is adopted with intel® Xeon® servers to implement scalable deep learning pipeline (collect, store, preprocess, analyze and scale) efficiently.

Here is the list of GCP tools and services to accomplish the same -

- Compute nodes – Google Compute Engine (GCE)
- Storage – Google cloud storage
- Management – Google Kubernetes* Engine (GKE), AI platform, AI Hub
- Virtualization – KVM
- Container orchestration – Kubernetes*

GCP CLOUD INSTANCES FOR AI APPLICATIONS

There are the many cloud instances available in GCP to develop and productize multiple applications. But below sections give information on the instances suitable for ML/DL applications which are equipped with intel® Xeon® family of servers having the DL computation capabilities explained in previous sections. Here are the various GCP cloud instances suitable for ML/DL applications. All these instances may not be available in all regions and for detailed info please refer to the link https://cloud.google.com/compute/docs/machine-types

COMPUTE-OPTIMIZED

- **C2** (Xeon® SP, up to 60 vCPUs, 3.1 GHz Base and 3.8 GHz all-core turbo frequency – **Cascade lake**)

MEMORY-OPTIMIZED

- **M1*** (Xeon® SP, 96 vCPUs, 2 GHz Base and 2.7 GHz all-core turbo frequency – Skylake)
  - **m1-megamem** only
- **M2** (Xeon® SP, up to 416 vCPUs, 2.5 GHz base and 3.4 GHz all-core turbo frequency - **Cascade lake**)
  - **m2-ultramem** only

GENERAL PURPOSE INSTANCES

- **N1*** (Xeon® SP, up to 96 vCPUs, 2 GHz Base and 2.7 GHz all-core turbo frequency – Skylake)
  - **highcpu**
- **N2** (Xeon® SP, up to 80 vCPUs, 2.8 GHz Base and 3.4 GHz all-core turbo frequency – **Cascade lake**)
  - **highcpu**

*These instances consist of various versions of Xeon® family too. Xeon® Scalable Platform (Skylake, cascade lake) based servers are mainly recommended for DL workloads as given in the below table

CPU platform	Supported machine types
Intel Xeon Scalable Processor (Cascade Lake)	• N2 predefined machine types • N2 custom machine types
	• C2 machine types
	• Memory-optimized machine types (`m2-ultramem` only)
Intel Xeon Scalable Processor (Skylake)	• N1 predefined machine types • Memory-optimized machine types (`m1-megamem` only) • N1 custom machine types

RECOMMENDATION FOR VARIOUS ML/DL WORKLOADS

- ✓ **N1 instances** are suitable ML applications and low compute DL inference applications.
- ✓ **M2, C2, N2 instances** are suitable for all kinds of Deep learning inference workloads
- ✓ **N2, M2 (m2-Ultramem) and C2** are upgraded with 2nd Generation of Xeon® SP which will benefit with **DL Boost (VNNI)** for Inference. These are suitable for Single node training too and are suitable for Distributed Deep learning training by selecting high NW performance required for inter-node communication.
- ✓ **M2 instances** are suitable for memory intensive workloads which use 3D-CNN, BERT, genomic analysis related topologies with memory requirement more than 192GB. Also, these instances come with higher sockets to support distributed training with in a node.

For ease of understanding, below table gives recommendation based on the ML/DL topologies. This table suggests the minimum number of vCPUs with each instance types to get reasonable performance and latency requirements. Users are expected to benchmark their topology and select the instance suitable for production.

Topology	GCP Instance
Training and inference of ML and small NN topologies Ex : Regression, K-means, KNN, SVD, PCA, LDA, RCF, XGBoost, Lenet, Alexnet, Squeezenet etc..	All N1 Skylake instances and C2/M2 with **<=8** vCPUs
Inference of medium NN topologies for classification, object detection, segmentation Ex: Resnet-50/101/152, Inception-v3/v4, VGG-16/19, SSD-mobilenet, Yolo-tiny-v2/v3 etc..	N2/C2/M2 **>=8** vCPUs for FP32 and INT8
Inference of other Deep NN topologies for object detection, segmentation, NLP, Reinforcement learning, GANs Ex: Faster-RCNN, SSD-VGG, Yolov2,v3, GNMT, Transformer, U-net, V-net, NCF, W&D, A3C, BERT, GAN etc..	N2/C2/M2 **>=16** vCPUs for FP32 and INT8
Inference on memory and compute intensive NN topologies Ex: 3D-U-Net, 3D-GAN etc...	C2, M2 **>=16** vCPUs for inference
Single node Training of medium and Deep NN topologies	C2 **>=32** vCPUs
Distributed training on multiple nodes with all medium and deep NN topologies	C2, M2 **>=32** vCPUs
Distributed training with further memory intensive like 3D CNN topologies	C2, M2 **>= 32** vCPUs

Rule of thumb

- use **N2**, **C2** instances which are suitable for Deep learning inference and single node and distributed training over a large cluster
- Use **M2** instances if distributed training requires memory intensive workload with huge number of parameters to synchronize during training.

GCP TOOLS & SERVICES FOR DEEP LEARNING

GCP has multiple tools and services to ease the deep learning application development and deployment.

Deep Learning images

Deep Learning VM – an image with pre-installed frameworks. Supports multiple framework versions for TensorFlow*, MxNet*, PyTorch*, Chainer*, CNTK* etc.

Recommendation: use the VM images with **Intel® MKL-DNN on CPUs**

Containers

To ease the job of installation, GCP is providing multiple container images with MKL-DNN optimized frameworks.

Recommendation: use the below container images while using the CPU instances which are integrated with MKL-DNN or generate MKL-DNN optimized framework containers

- **TensorFlow*** - gcr.io/deeplearning-platform-release/tf-cpu-<version>
- **PyTorch*** - gcr.io/deeplearning-platform-release/pytorch-cpu-<version>

Tools and Resources

GCP provides many services to ease the job of deep learning development and deployment. Here are some of the tools and resources that can help novice to expert data scientists to play with Deep learning.

GKE - Google Kubernetes* Engine – This service helps in deploying and management of a cluster for distributed training using Kubernetes*. GKE uses kubeflow for distributed TensorFlow* training. Users with little knowledge on container management can use this service to deploy and scale.

GCP market place – Here intel® optimized containers and images are available to use on the CPU instances.

Recommendation: Please use CPU instances with Skylake/Cascade lake instances with MKL-DNN optimized containers and images and for both training and inference.

AI services

AI Hub - AI Hub is a hosted repository of plug-and-play AI components, including end-to-end AI pipelines and out-of-the-box algorithms.

AI Hub offers a collection of components for developers and data scientists building artificial intelligence (AI) systems as give below -

- Find, deploy, and use Kubeflow pipelines and components.

- Explore code and learn in interactive Jupyter notebooks.
- Explore and reuse TensorFlow* modules.
- Explore, deploy, and use trained models.
- Use prepackaged virtual machine (VM) images to quickly set up AI environment.
- Share AI components with colleagues.

AI building blocks - AI building blocks make it easy to add sight, language, conversation, and structured data into your applications.

These services can help you to quickly deploy for multiple AI use cases -

- Cloud speech to text
- Cloud vision
- Cloud NLP
- Cloud translation
- Cloud video intelligence
- Cloud AutoML

AI platform - AI Platform makes it easy for machine learning developers, data scientists, and data engineers to take their ML projects from ideation to production and deployment, quickly and cost-effectively.

Recommendation: use the Intel® Xeon® based instances to avail these services to get the advantage mentioned previously.

INSTRUCTIONS TO DEVELOP AND DEPLOY ML APPLICATIONS ON GCP

Development and deployment of machine learning applications is straight forward by choosing relevant compute instances from N2/C2/M2 mentioned above. To get the optimum performance for NumPy, SciPy and Scikit-learn based applications, install Intel® distribution of Python (IDP). IDP contains Intel® Data analytics acceleration library (DAAL) to get optimized machine learning performance on GCP Intel® instances.

- Launch compute instances mentioned in the previous section depending on the use case and computational requirement. https://console.cloud.google.com/compute/instances?_ga=2.159358521.-2092433894.1565692760
- Download and install IDP which consists of DAAL too https://software.intel.com/en-us/distribution-for-python
- To install only DAAL, please follow the link https://software.intel.com/en-us/daal.
- Here is the developer guide for DAAL https://software.intel.com/en-us/daal-programming-guide
- Check references [5], [6] for IDP and DAAL benchmarks.

INSTRUCTIONS TO DEVELOP AND DEPLOY DL APPLICATIONS ON GCP

After choosing the right compute instance, communication network, storage and optimized software, GCP services are required to automate and scale on a production cluster.

In this section we discuss on the procedure to develop and productize the deep learning training and inference applications using GCP CPU instances.

Rule of thumb

Install frameworks on basic VMs or containers with below commands and set **KMP_AFFINITY, KMP_BLOCKTIME** and **OMP_NUM_THREADS.**

- **TensorFlow***: pip(3) install intel-tensorflow
- **MxNet*** : pip(3) install mxnet-mkl
- **PyTorch*** : pip(3) install torch
- **bigDL** : pip(3) install BigDl

Use Horovod with Intel® MPI for distributed training over a single node or a cluster of nodes.
As VM instances are based on hyper threads, use **inter_op_parallelism_threads=1** with ConfigProto()

Whatever may be the service from GCP, use the Intel® GCP instances mentioned in above sections and follow these guidelines -

- After launching an instance, run **lscpu | grep avx** and make sure AVX-512 is available in flags list
- After launching a script, set **export MKL_VERBOSE=1** and make sure MKL logs are seen (unset after testing as logging will add latency)

STEP BY STEP GUIDE TO SETUP A CLUSTER FOR DL TRAINING OVER GCP

Most of the GCP services like AI hub and AI Building blocks do not give the flexibility to choose the underlying Hardware. So, customers need to query for intel® Xeon® instances for great TCO. But AI platform service comes with options to use Deep learning VMs and containers over Xeon® instances of compute engines. Following sections give instructions to use AI platform service to train and deploy DL applications.

Xeon® SP CPU instances are enough to run the scalable distributed deep learning training for most of the practical neural networks. No need of accelerator cards unless a special requirement is there. So, while launching the training instances as given below, skip the steps to add accelerator cards and use only high vCPU-based Xeon® SP instances.

- Here are the instructions to create and start a VM instance and install the libraries of your choice https://console.cloud.google.com/compute/instances?_ga=2.67056173.-2092433894.1565692760 . Select the OS and setup the python environment as mentioned here https://cloud.google.com/python/setup and use the pip install instructions mentioned above to get intel® optimized frameworks. There will be option to select the containers too while creating the instance in which DL CPU containers can be selected.

- Here are the instructions to create instance with deep learning VMs which will come with optimized libraries to install TensorFlow* and PyTorch* to run Deep learning - https://console.cloud.google.com/marketplace/details/click-to-deploy-images/deeplearning?_ga=2.121411083.-2092433894.1565692760 . Chose the option without GPUs. While choosing the framework, get the Intel® MKL supported only and choose number of GPUs is None. Install the desired framework in those instances using the python commands given above.

    - TensorFlow* VM : https://cloud.google.com/ai-platform/deep-learning-vm/docs/tensorflow_start_instance
    - PyTorch* VM : https://cloud.google.com/ai-platform/deep-learning-vm/docs/pytorch_start_instance.

- Here are the instructions to create and use the Deep learning containers https://cloud.google.com/ai-platform/deep-learning-containers/docs/getting-started-local. You can choose the desired version of TensorFlow* and PyTorch* containers from https://console.cloud.google.com/gcr/images/deeplearning-platform-release?project=deeplearning-platform-release&_ga=2.64330540.-2092433894.1565692760 . Suggested to use CPU versions of the containers as given below -

- gcr.io/deeplearning-platform-release/pytorch-cpu
- gcr.io/deeplearning-platform-release/pytorch-cpu.1-0
- gcr.io/deeplearning-platform-release/pytorch-cpu.1-1
- gcr.io/deeplearning-platform-release/pytorch-cpu.1-2
- gcr.io/deeplearning-platform-release/tf-cpu
- gcr.io/deeplearning-platform-release/tf-cpu.1-13
- gcr.io/deeplearning-platform-release/tf-cpu.1-14
- gcr.io/deeplearning-platform-release/tf-cpu.1-15

STEP BY STEP GUIDE TO RUN DL TRAINING OVER GCP CPU INSTANCES

Single node training:

After launching the VM or container instance mentioned above, initiate training through command line or cloud shell or Rest API or Jupyter lab.

Distributed training over VMs:

Distributed training requires multiple VM instances to be launched and kept ready with all framework dependent libraries. But multiple frameworks use different mechanism for distributed training. To make it simple we suggest use the MPI based library Horovod provided by uber and run the training on multiple VM instances as given in the instructions document - https://www.intel.ai/using-intel-xeon-for-multi-node-scaling-of-tensorflow-with-horovod/#gs.cf2huc.

Alternate is to launch basic VMs and use the Deep learning containers on VMs for training over a cluster as explained below.

Distributed training over Containers:

Here is the training overview on GCP - https://cloud.google.com/ml-engine/docs/training-overview

Create a Kubernates cluster by choosing the CPU intensive VMs https://cloud.google.com/kubernetes-engine/docs/how-to/creating-a-cluster

Here are the step by step instructions to run the training over container cluster with GKE https://cloud.google.com/ai-platform/deep-learning-containers/docs/kubernetes-container.

STEP BY STEP GUIDE TO RUN DEEP LEARNING INFERENCE

Instructions to setup the inference is almost like the training on VMs. Use the GCP CPU instances along with Deep learning VMs or containers for ease of installation of frameworks. Depending on the computational requirement of the topology the number vCPUs required will vary. Users are recommended to do experiment with multiple vCPU sizes and fix to an instance which gives the optimum performance for production environment. Use this guide to split the cores for a bigger VM and try for optimized inference with multiple streams even at lower batch sizes - https://www.intel.ai/accelerating-deep-learning-training-inference-system-level-optimizations/#gs.3d03w7

INFERENCE WITH DL FRAMEWORKS

Here are the instructions to run inference - https://cloud.google.com/ml-engine/docs/prediction-overview. There are two modes of inference online prediction https://cloud.google.com/ml-engine/docs/online-predict and batch prediction https://cloud.google.com/ml-engine/docs/tensorflow/batch-predict. Most of the practical applications needed real time response and Xeon® CPUs can be tuned flexibly for online or real-time prediction.

INFERENCE WITH DL BOOST

Intel® DL boost added in 2nd generation of Xeon® scalable platform has high throughput and low latency INT8 inference capability with VNNI instructions [7]. Here is the complete guide to convert the trained model from FP32 to INT8 and use for high performance inference on the GCP cluster with Second generation of Xeon® servers - https://www.intel.ai/tensorflow-containers-optimized-intel/#gs.2ass5x

INFERENCE WITH OPENVINO

OpenVINO is an optimization tool which converts the trained model from multiple frameworks and runs efficiently on Intel® Xeon® CPUs.

Here are the instructions to install the openVINO on Ubuntu/CentOS/Windows based AMI and use it for optimized inference - https://docs.openvinotoolkit.org/latest/index.html

Here are the instructions to build docker images to run on containers using GKE services – https://docs.openvinotoolkit.org/2018_R5/_docs_install_guides_installing_openvino_docker.html

Here is calibration tool to run the inference with DL Boost - https://docs.openvinotoolkit.org/latest/_inference_engine_tools_calibration_tool_README.html

Here is the step-by-step instructions for openVINO model server to go for production with openVINO - https://www.intel.ai/openvino-model-server-boosts-ai-inference-operations/#gs.3cwfl4

OpenVINO can also be accessed through kubeflow pipelines too as given in the link here which provides the flexibility to use GCS and openVINO model server to deploy the scalable applications on GCP CPU instances- https://github.com/kubeflow/pipelines/tree/master/contrib/components/openvino

INSTRUCTIONS TO DEVELOP AND DEPLOY BIGDL APPLICATIONS ON GCP

BigDL is a distributed deep learning library developed and open-sourced by Intel® Corporation to bring native deep learning support to Apache Spark. By leveraging the distributed execution capabilities in Apache Spark, BigDL can help you take advantage of large-scale distributed training in deep learning.

GCP supports BigDL through Cloud Dataproc. Here are the instructions to train and deploy AI applications on Spark cluster through BigDL https://gweb--cloudblog--publish-appspot-com.cdn.ampproject.org/v/s/gweb-cloudblog-publish.appspot.com/products/gcp/using-bigdl-for-deep-learning-with-apache-spark-and-google-cloud-dataproc/amp/?usqp=mq331AQCCAE%3D&amp_js_v=0.1#referrer=https%3A%2F%2Fwww.google.com&amp_tf=From%20%251%24s

STEP BY STEP GUIDE TO DEVELOP AND DEPLOY AI APPLOCATIONS USING GCP SERVICES

AI Hub:

AI Hub offers a collection of assets for developers and data scientists building artificial intelligence (AI) systems. Train your machine learning (ML) model in a notebook or deploy it to a managed service. AI hub provides the production ready and cutting-edge research models for AI services on GCP cluster with simple steps. Here are the instructions to get use of resources in AI Hub https://cloud.google.com/ai-hub/docs/introduction

AI Building Blocks:

AI building blocks make it easy to add sight, language, conversation, and structured data into AI applications with already trained models. Using simple API mechanism intelligence can be added into your applications. Here are the instructions to use these services on GCP https://cloud.google.com/products/ai/building-blocks/

REFERENCES

[1] Avx-512 info: https://colfaxresearch.com/skl-avx512/

[2] intel® optimized frameworks: https://software.intel.com/en-us/frameworks

[3] Intel® distribution of OpenVINO™ toolkit: https://docs.openvinotoolkit.org/

[4] Intel® Analytics zoo: https://github.com/intel-analytics/analytics-zoo

[5] Hands-on IDP and DAAL : https://software.intel.com/en-us/videos/get-your-hands-dirty-with-intel-distribution-for-python

[6] IDP benchmarks: https://software.intel.com/en-us/distribution-for-python/benchmarks

[7] Intel® DL Boost : https://www.intel.ai/increasing-ai-performance-intel-dlboost/#gs.3cxhiw