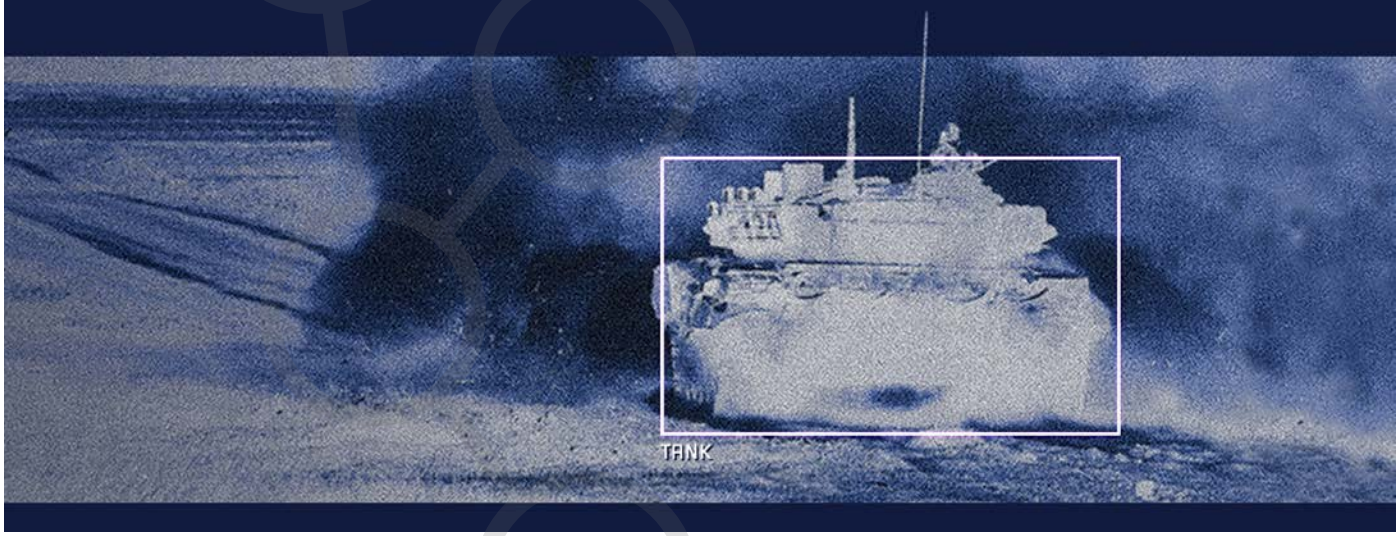
AI Workflows for Defense and Aerospace Embedded Systems



A WOLF ADVANCED TECHNOLOGY and DOWSON ROBOTICS WHITEPAPER

Setting the Stage for a Successful AI Project

AI and machine vision projects are no different from other embedded processor projects that need to meet budget, performance, and the operational goals of a program. But the path to that success incorporates decidedly different development workflows. As an example, unlike typical embedded system deployments, which may require occasional hardware or software updates, the updates required for AI inference engines must be regularly scheduled and built into the operational plan to incorporate deep learning results from new (or changing) data. Another common occurrence in AI projects is the underestimation of the amount of data required for successful AI training.

To help set the stage for success, Dowson Robotics (Dowson) and Wolf Advanced Technology (WOLF) have teamed up to share our understanding of the common development workflows used to mitigate risk and to help accommodate resource requirements.

As a matter of background, Dowson has provided technical consulting and management services for a number of successfully deployed applications based on artificial intelligence and computer vision, and WOLF creates GPU and FPGA-based modules that are currently being used in a number of AI and computer vision programs for defense and aerospace.

Introduction

Artificial Intelligence is a term used to describe the capability of computer systems to perform tasks that normally require human intelligence such as perception, conversation and decision-making [01]. At this time, the universe of artificial intelligence solutions is continuing to expand and have included such capabilities as [02]:

- **computer vision** to perceive objects;
- **natural language processing** to understand and communicate in human languages;
- **knowledge representation** to store what it knows and perceives;
- **automated reasoning** to use stored information to answer questions and to draw new conclusions;
- **machine learning** to adapt to new circumstances and to detect and extrapolate patterns; and
- **robotics** to manipulate objects and move about.

Based on customer knowledge and feedback, we are focusing this paper on the development workflow examples using WOLF GPU-based systems for **computer vision, machine learning** and **robotics** applications.

In addition to the costs of traditional systems development, the majority of the investment required for artificial intelligence projects is in the resources required for training, testing and validation. An artificial intelligence or computer vision application is also highly dependent on the amount (and quality) of the data used for training. The ImageNet Large Scale Visual Recognition Challenge, as an example, contains 14,197,122 images with labeled objects, bounding boxes, descriptive words, and features. [03]

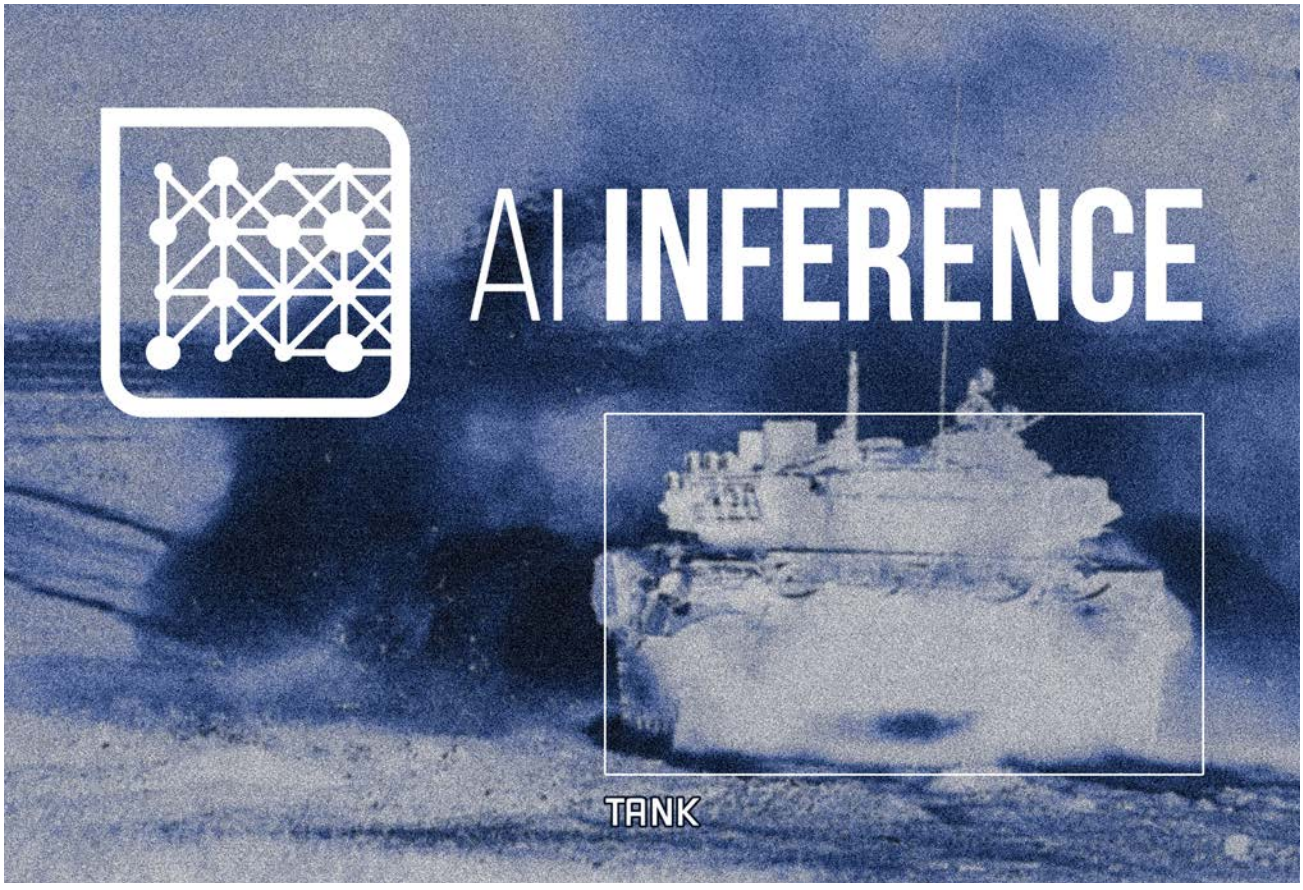


Figure 1: Object Detection using AI Inference

Computer vision

Computer vision is an important enabling technology for aircraft and autonomous systems for many aerospace and defense applications.

Modern aircraft are equipped with enhanced synthetic vision systems which synthesize flight information from multiple sensors and on-board databases and are designed to replicate a heads-up cockpit view as closely as possible on a heads-down display. It provides pilots with a realistic view of the surrounding environment, irrespective of the time of day and weather conditions and helps improve situational awareness and safety during all phases of flight.

Autonomous vehicles rely heavily on computer vision and processed sensor information to perceive their environment and detect and avoid obstacles.

Computer vision applications may include:

- Image processing
- Visual image classification: classifying images based on its content
- Visual Object Detection: detecting and locating objects in an image
- Image Segmentation: partitioning an image into different segments on a per-pixel basis and assigning a label for each pixel that shares certain characteristics
- Image stitching: turning overlapping pictures into a mosaic
- 3D point cloud generation: A point cloud is a set of data points in space, generated using cameras or laser scanners. 3D point clouds are used to capture information about a scene at a very high resolution and used as a starting point for generating more sophisticated representations such as 3D meshes.
- 3D scene reconstruction: Several techniques exist for reconstructing a scene in 3D. These include using photogrammetry techniques and multiple images of a scene to generate a 3D point cloud and subsequently converting the 3D point cloud representation to a textured 3D surface.

The Graphics Processing Unit (GPU) has became essential for accelerating computer vision and machine learning algorithms. With its high performance and flexibility, GPU computing has seen its application in computer vision evolve from accelerating 3D graphics to realizing advanced vision, perception and AI-based systems.

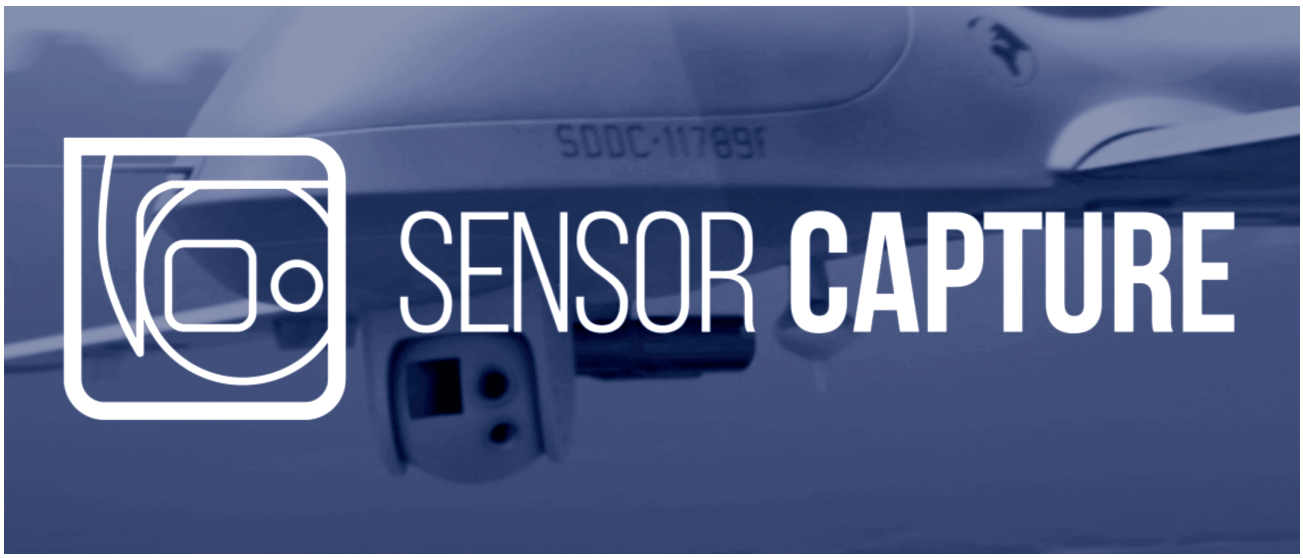


Figure 2: Sensor image video capture on a drone

Workflow

Dataset Preparation

Successful artificial intelligence training relies on the quality and quantity of data input into the model. In supervised learning, the data must be labelled in a form that includes the input and the output expected from the network. An example of this is in an object detection dataset. Not only do you need images, but you also need labels to indicate where in the image an object of interest can be found, typically indicated by the coordinates of a 2D bounding box. You will typically require a dataset that is split into three (3) using a ratio that is dependent on the size of the dataset (e.g. 60/20/20, 70/15/15 or 80/10/10 for a large dataset):

- Training Dataset
  a. Used to train a neural network (nn) model.
  b. The results of the training process are a set of trained nn model weights and biases.
- Validation Dataset
  a. Used during the training process to evaluate the performance of a neural network model and tune model hyper-parameters.
- Test Dataset
  a. Is a small dataset that is typically held back and used by an end-user to evaluate the final performance of a model.

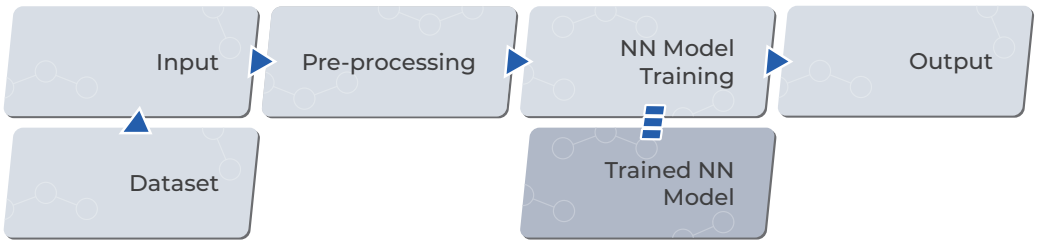


Figure 3: Model training

Model Training

We use the training and validation datasets to train the neural network model. The input data goes through a pre-processing step for image adjustment and data-set augmentation.

During the forward pass of the training, the neural network makes a prediction for the output and a loss function is used to predict the errors made by the neural network. The weights and biases of the neural network are then updated using backpropagation. This process is repeated until model convergence and you get an acceptable level of loss and accuracy for the predictions made by the neural network model.

When you train a neural network model, you save or export the model configuration and weights in a particular format (e.g. *.hdf5, *.pth, *.onnx, etc).

Model Inference

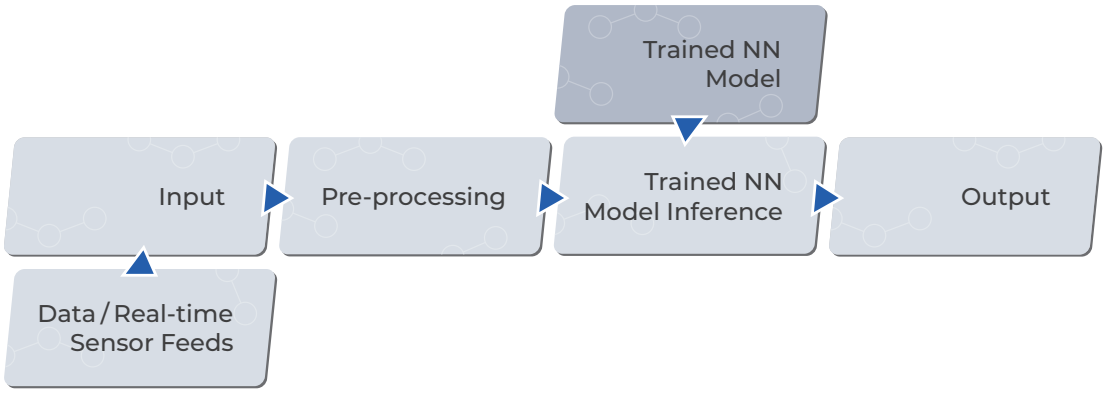


Figure 4: Model inference

The trained neural network can now be tested with the test dataset to check its performance. If the performance of the model is satisfactory in terms of accuracy of its predictions, you can proceed with further testing with real-world data and perform further optimization of the model for run-time deployment in a production environment.

Model Optimization

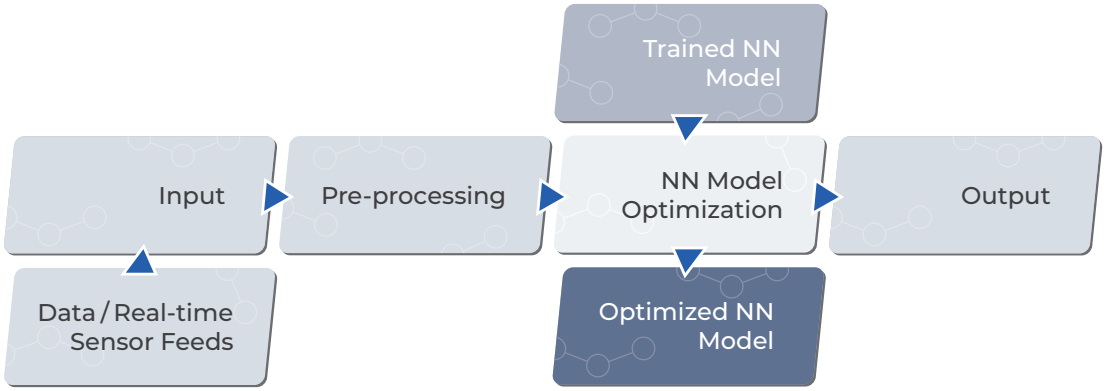


Figure 5: Model optimization

The model can then be further optimized in terms of:

- Energy consumption
  a. Power consumption, which is an important consideration for data-center applications and deployment onto edge devices
- Resource optimization
  a. Size of the model and amount of memory consumed. This is important in order to run the model on resource constrained edge devices
- Performance
  a. Throughput
  b. Latency, in terms of the time to make a prediction

Optimizing a model for inference will require retraining the model once, with the original dataset, to recover some of the accuracy lost as a result of the optimization before deployment.[04]

Model Deployment

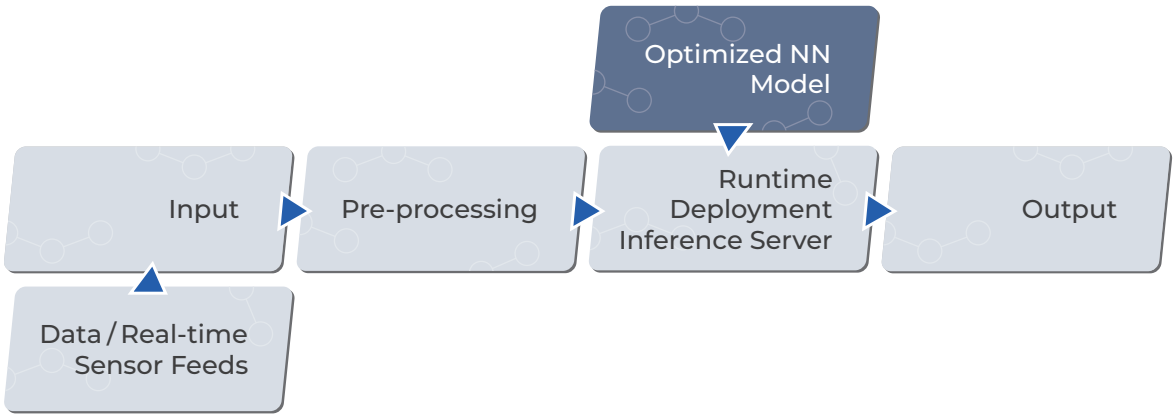


Figure 6: Model deployment

A neural model can then be deployed into a production environment. In order to simplify setting up and configuring development, staging, and runtime production environments, we can utilize embedded docker runtime containers for supported edge platform devices. This docker container will encapsulate the neural network model along with all dependencies required to run the model in production such as an inference engine, required software libraries, and operating system packages.

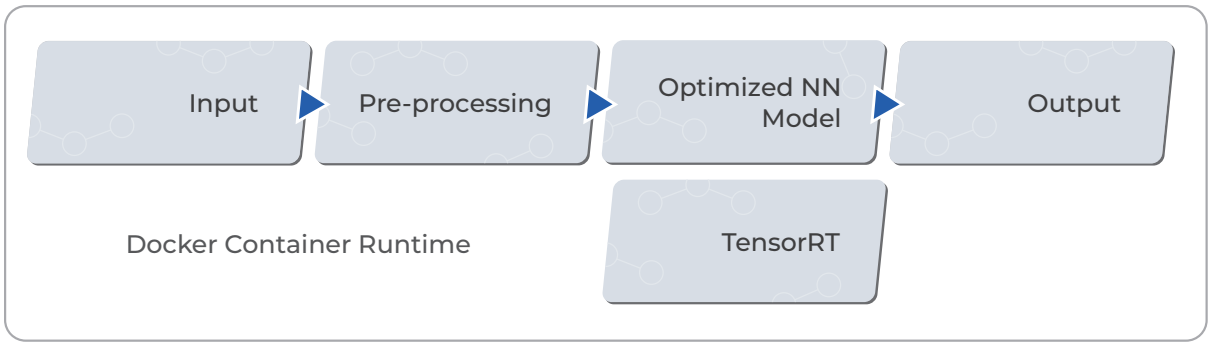


Figure 7: Trained model in Docker container

Four Deployment Options:

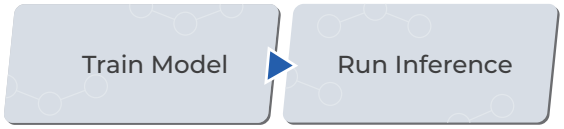


Figure 8: Unoptimized inference model

01. Use an un-optimized model as is: Simply use the saved weights file from the training process on the edge device and use it for inference.

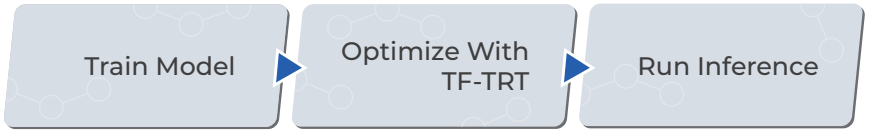
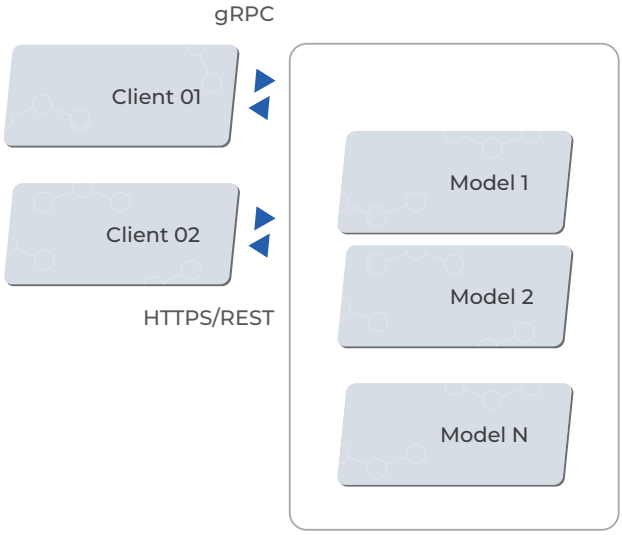


Figure 9: Optimized inference model with TensorRT

02. Convert the trained model to TensorRT and then run that on the edge device.

03. Construct a neural network model that supports optimization, train the model, convert to TensorRT and do one final retraining pass, and then deploy to the edge device.

04. Deploy optimized models to an embedded inference server to support time-shared inferencing.



The edge device effectively becomes an end-point, to which multiple, say camera sources, send individual frames to the inference server for batch inference.

Data Distributed Service (DDS), gRPC, and HTTPS/REST end-points can be used for command, control, configuration and status reporting for a service on an edge device. DDS and gRPC can also be used for streaming sensor data and sensor streams to an inference server.

Figure 10: Embedded inference server using time-sharing

Conclusion

The opportunity to advance embedded systems applications using artificial intelligence and computer vision is upon us. And although both WOLF and Dowson have fulfilled demand for these types of applications, and this demand is growing, we wanted to highlight the unique development workflows and resources AI projects need in order to become a success, especially in respect to the volume of data required to begin a project and the ongoing retraining (and refreshed data) that must be incorporated into any project plan.

If you have any questions regarding this paper, please reach out to Elvis Dowson or Keith Menezes using the contact information provided on the last page of this paper.

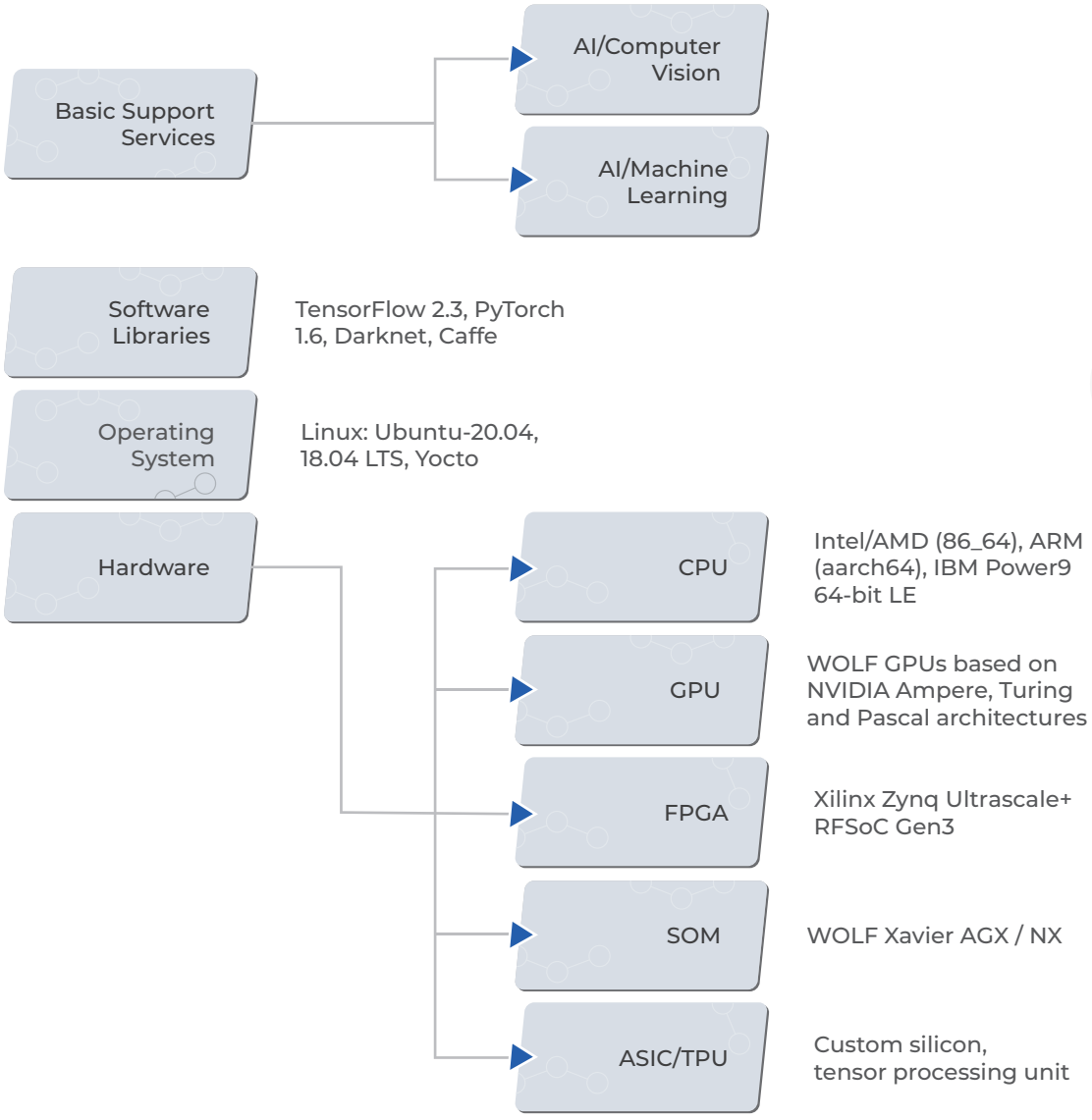
Reference

1. The US Army Robotic and Autonomous Systems Strategy - March 2017 https://www.tradoc.army.mil/Portals/14/Documents/RAS_Strategy.pdf
2. Artificial Intelligence: A Modern Approach (3rd Edition) - Stuart Russell, Peter Norvig - Prentice Hall - 2010.
3. https://www.image-net.org/
4. Compressing and Regularizing Deep Neural Networks, Improving Prediction Accuracy Using Deep Compression and DSD Training - Song Han - O'Reilly - 20161110 https://www.oreilly.com/content/compressing-and-regularizing-deep-neural-networks/

Services available from Dowson Robotics

Dowson Robotics quickstart package offers support and prototype development services for computer vision and machine learning applications for WOLF hardware products.

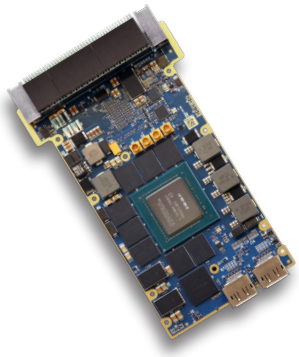
Basic support services



WOLF: Edge-AI Related Product Examples

VPX3U-XAVIER-SBC

The VPX3U-XAVIER-SBC features an NVIDIA Volta™ GPU with 512 CUDA® cores and 64 Tensor cores, 1.4 TFLOPS of peak performance, an NVIDIA Carmel ARM64 CPU with 8 cores, 32GB of LPDDR4 256-bit memory with up to 137 GB/s of bandwidth. The WOLF FGX adds additional video outputs such as SDI and CVBS.



VPX3U-RTX5000-CV

The VPX3U-RTX5000-CV features an NVIDIA Turing™ TU104 GPU with 10.9 TFLOPS peak performance, 3072 CUDA® Cores, 384 Tensor Cores, 48 RT Cores, and has 16GB of GDDR6 256-bit memory with up to 448 GB/s memory bandwidth. The WOLF FGX provides support for additional video formats such as SDI and CVBS.

VPX3U-P5000-8IO

The VPX3U-P5000-8IO features an NVIDIA® Quadro® P5000, 16GB of GDDR5 memory, with up to 6.2 TFLOPS of peak performance. This module can accomodate up to eight 3G-SDI inputs, eight 3G-SDI outputs, and four CVBS/STANAG inputs and/or outputs.

VPX6U-RTX5000-DUAL-CV

The VPX6U-RTX5000-DUAL-CV features two NVIDIA Turing™ TU104 GPUs with 22 TFLOPS of peak performance, 6144 CUDA® Cores, 768 Tensor Cores, 96 RT Cores, and 32 GB GDDR6 256-bit memory providing up to 448 GB/s memory bandwidth to each GPU. The WOLF FGX provides support for additional video formats such as SDI and CVBS.

About the Authors

Elvis Dowson

Founder
Dowson Robotics

Elvis Dowson, founder of Dowson Robotics, is an embedded systems and software engineer with over 21 years of work experience specializing in safety critical avionics (DO178B) software development, robotics (UAV, UGV, ROS) software development, machine learning, communication systems development using software-defined radios and radar systems integration. Elvis was awarded a Bachelor Degree (BE) in Computer Science and Engineering, from BMS College of Engineering, Bangalore, India..

elvis@dowsonrobotics.com

Keith Menezes

Technical Sales Engineer
Wolf Advanced Technology

Keith Menezes graduated from Space Engineering at the Lassonde School of Engineering, York University in Toronto, Canada, and is focused on demonstrating WOLF's capabilities in artificial intelligence on the NVIDIA AGX Xavier and Turing products. Keith has years of experience with unmanned systems research and development including an Unmanned Aerial Systems for aerial mapping and an Unmanned Marine Vehicle for bathymetry and hydrography. Keith has also worked on CubeSat design projects, vibration testing at the Canadian Space Agency's David Florida Laboratory in Ottawa, Ontario, and radiation testing TRIUMF, Canada's particle acceleration centre in Vancouver, British Columbia.

keith@wolf.ca
905-852-1163 x775