Towards Neural Network Inference on Programmable Switches

Jonatan Langlet
Karlstad University
Karlstad, Sweden
jonatan.langlet@gmail.com

Andreas Kassler
Karlstad University
Karlstad, Sweden
andreas.kassler@kau.se

Deval Bhamare
Karlstad University
Karlstad, Sweden
deval.bhamare@kau.se

ABSTRACT

Recent advances in machine learning (ML) have gained a lot of interest in the domain of computer networks and network security. Machine learning techniques can be used to understand normal behavior, detect anomalies, infer root causes of such anomalies and take proper actions. Learning-based approaches are very effective, since the underlying models could be re-trained to counter a large amount of evolving and complex data using comprehensive datasets. Training a machine learning model and applying the trained model inside the network on real traffic are two completely different tasks which require different resources. Training of machine learning models is very complex, requires a large amount of resources and is typically done offline on specialized equipment (e.g. TensorFlow, etc.). On contrary, applying a ML model in a real networking environment requires substantially different packet processing capabilities than the ones available from traditional IP-oriented networking equipment. Therefore, ML based inference together with the collection of proper features, which are required for the inference, is typically done in software on dedicated equipment (e.g. inside a separate Intrusion Detection System or Flow Classification box), which comes with several drawbacks. First, adding additional equipment to traditional networking gear leads to complex deployment. Second, the additional hop introduces additional latency, which impacts the end-to-end performance, and leads to higher detection latency for e.g. critical events. To overcome the aforementioned problems, in this paper we propose to perform feature collection and artificial neural network (ANN) inference directly in the data plane of programmable packet processors.

The development of an ANN inference capable data plane is faced with non-trivial challenges due to the complex processing logic involved. First, for supporting different use cases and neural network models, the data plane needs to be able to dynamically collect a variety of different features, which requires a flexible parsing of packet headers, including IP, TCP options and application specific protocol fields. Second, the complex mathematical operations involved in ANN inference are challenging to run effectively in the data plane, due to its processing and memory limitations. The requirement to process packets at line rate further exacerbates the challenge. However, the recent emergence of programmable packet-processing pipelines [2], together with high-level language P4 [1] and compiler support, creates new opportunities for innovation in networking. Once basic support for feature collection and inference is implemented in P4, the operator can modify the program and propagate the tables and registers so that it fits her ANN model and scenario and compile it to different hardware (e.g. NIC, switch).

We have designed and implemented traffic flow feature collection together with feedforward neural network inference on a Netronome Agilio CX 2x40G SmartNIC. The ANN model is trained offline and then applied as a Micro-C external function to perform inference on the collected flow features which can perform, for example, flow classification, intrusion detection or QoE estimation. Only the first few packets in each traffic flow are recorded to gather enough metadata for performing ANN inference, where the number of packets to inspect for feature extraction is specified in a P4 register. Our implementation can flexibly be instructed which features to record for each flow. When feature extraction is complete, ANN inference is triggered and its output is stored per-flow using register arrays. This value can then be used by other P4 actions for determining forwarding rules for subsequent flow packets, and for example redirect time-sensitive flows through a faster route or immediately block unwanted traffic. ANN parameters and activation levels are typically stored as floating point values, which are not supported in P4. Instead, we use fixed-point approximation using 64-bit integers and bit-shifting. Because P4 lacks support for exponential functions, we use ReLU as activation function. We are currently working on implementing a sigmoid approximation as an alternative to ReLU. In our implementation, the ANN is stored directly in the memory of the SmartNIC, and its structure (i.e. number of hidden layers and neurons) is defined at compile-time due to the lack of dynamic memory support. The ANN parameter values are also currently set during compile-time, however it should be possible to update these values at run-time as the SmartNIC host has built in functionality to modify these memory regions. This means that we can update the model dynamically without reloading the firmware, as long as the structure of the ANN is unchanged.

Using packet traces, we find that extracting a total of 43 flow features results in additional packet latency of $16\mu s$ for the first few packets in each flow, compared to the base latency of $6\mu s$ that is required by the simple P4 forwarding switch. ANN inference results in additional latency depending on the model complexity. For example, for an ANN with two hidden layers with 10 neurons each, the inference latency is $65\mu s$ while it increases to $915\mu s$ for larger ANNs having three hidden layers with 30 neurons each. Because the inference is done on metadata for a single packet per flow once features are collected, our implementation has a negligible impact on the latency of subsequent packets of the given flow.

ACKNOWLEDGMENTS

Parts of this work has been funded by the Knowledge Foundation of Sweden through the Profile HITS.

REFERENCES

[1] Pat Bosshart, Dan Daly, Glen Gibb, Martin Izzard, Nick McKeown, Jennifer Rexford, Cole Schlesinger, Dan Talayco, Amin Vahdat, George Varghese, et al. 2014. P4: Programming Protocol-Independent Packet Processors. *ACM SIGCOMM Computer Communication Review* 44, 3 (2014), 87–95.

[2] Pat Bosshart, Glen Gibb, Hun seok Kim, George Varghese, Nick Mckeown, Martin Izzard, O Mujica, and Mark Horowitz. 2013. Forwarding Metamorphosis: Fast Programmable Match-Action Processing in Hardware for SDN.