

# Complex Features Extraction in Real Time

## *A Part III project proposal*

T. T. Bui (*tbb29*), Downing College

Project Supervisor: Dr Noa Zilberman

Director of Studies: Dr Robert K. Harle

### Abstract

*In-network computing is an emerging research area in systems and networking, where applications traditionally running on the host are offloaded to the network hardware (e.g., switch, NIC). In the past, many applications have been offloaded [1, 2, 3], even machine learning (ML) on a commodity programmable switch [4]. However, the in-network classification discussed in [4] only used features extracted from packet headers. Many ML models require more complex features beyond those obtained from packet headers and metadata. There has been no attempts to look into extracting more complex features. This project aims to explore the possibility and the extent to which we can extract these features.*


## 1 Introduction, approach and outcomes (500 words)

In-network computing is an emerging research area in systems and networking, where applications traditionally running on the host are offloaded to the network hardware (e.g., switch, NIC). Examples of applications offloaded in the past include network functions (DNS server [1]) and distributed systems functions such as consensus (P4xos [2, 3]). In-network computing offers benefits in the form of better performance, both in terms of throughput and latency, and power efficiency [5]. Typical numbers range from  $\times 10$ –100 latency,  $\times 10,000$  throughput and  $\times 1,000$  power efficiency [5].

Recently, a paper published by Z. Xiong and N. Zilberman [4] discussed the possibility of in-network classification, i.e., performing Machine Learning (ML) on a commodity programmable switch. Doing ML within the network offers a potential of 10's of ns to  $\mu s$ , 10M's of images/sec and 10K–100K of images/sec/watt, which are more superior than what most GPUs can offer (100's of  $\mu s$  to ms, 10K's of images/sec, 100 images/sec/watt [4]). However, one limitation of the prototype was that inference was performed using information from packet headers and metadata, e.g., source and destination IP address, source and destination port number, etc. This limits it to just packet classification. Many ML models require features that are more complex than just information from packet headers and metadata.

At the moment, there is no previous work that looks into extracting these complex features.





The main goal of my project is to explore the feasibility and the extent to which we can extract more complex features beyond headers and metadata. To approach the problem, I first have to identify the different features and their use cases with regard to ML, e.g.,


 duration of a flow or the flow size. This is done based on previous works in the working context and in-network computing. Then, I will need to implement the extraction functionality for each of the features. Some of these complex features may require the use of counters or externs, whose functionality might be target-specific, to store some certain states across packets for computation. It might also be possible that some complex features cannot be extracted at all on a switch. In this case, I will evaluate the difficulties and limitations of the platform.



The outcome of the project will focus on demonstrating the possible functionality and exploring the limit to which feature extraction is capable of, rather than achieving a great performance. The project will be done in P4 programming language on one of the two available platforms: bmv2, a P4 software switch, or NetFPGA, which I am familiar with from my Part II project.

## 2 Workplan (500 words)

1. **Michaelmas vacation weeks 1–2 [5/12–18/12]:** Set up the working environment to work with the bmv2 and NetFPGA platform. 
2. **Michaelmas vacation weeks 3–4 [19/12–1/1]:** Literature review to identify the possible features and the use cases for their extraction. Determine, for each feature, the potential difficulty in extracting it (“Medium” or “Hard”). 
3. **Michaelmas vacation weeks 5–6 [2/1–15/1]:** Start implementing the algorithm to extract the ‘Medium’ features.
4. **Lent weeks 1–2 [16/1–29/1]:** Buffer weeks for leisure time spent during Michaelmas vacation. 
5. **Lent weeks 3–4 [30/1–12/2]:** Continue working on implementing the extraction of the “Medium” features.  
**Milestone:** A working code to extract the “Medium” features. 
6. **Lent weeks 5–6 [13/2–26/2]:** Start exploring the possibility of extracting the “Difficult” features.
7. **Lent weeks 7–8 [27/2–11/3]:** Evaluate the difficulties/limitations in the extraction of the “Difficult” features.

**Milestone:** Being able to either extract the “Difficult” features or to evaluate the difficult  in extracting them.

8. **Easter vacation weeks 1–2 [12/3–25/3]:** ~~Work on possible extensions (if any).~~



~~Otherwise, continue focusing on evaluation of the feasibility of extracting the ‘Difficult’ features.~~

9. **Easter vacation weeks 3–4 [26/3–8/4]:** Possible overflow from the previous weeks. Clean up codes and repository. Start writing dissertation main chapters.



10. **Easter vacation weeks 5–6 [9/4–22/4]:** Possible overflow from the previous weeks. Clean up codes and repository. Continue writing dissertation.

**Milestone:** Complete working prototype with power analysis performed and results evaluated. First draft of dissertation.

11. **Easter weeks 1–2 [23/4–6/5]:** Continue writing dissertation. Review cycles and corrections to dissertation.



12. **Easter weeks 3–4 [7/5–20/5]:** Continue writing dissertation. Review cycles and corrections to dissertation.

**Milestone:** Complete dissertation. Proof reading and submission.

13. **Easter weeks 5 [21/5–27/5]:** Buffer week.

## References

- [1] Sapio, Amedeo, I. Abdelaziz, A. Aldilaijan, M. Canini, and P. Kalnis, “In-Network Computation is a Dumb Idea Whose Time Has Come,” in *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, pp. 150–156, ACM, 2017.
- [2] H. T. Dang, P. Bressana, H. Wang, K. S. Lee, H. Weatherspoon, M. Canini, F. Pedone, N. Zilberman, and R. Soulé, “P4xos: Consensus as a Network Service,” Tech Report, May 2018.
- [3] H. T. Dang, M. Canini, F. Pedone, and R. Soulé, “Paxos made switch-y,” vol. 46, pp. 18–24, May 2016.
- [4] Z. Xiong and N. Zilberman, “Do switches dream of machine learning?: Toward in-network classification,” in *Proceedings of the 18th ACM Workshop on Hot Topics in Networks*, HotNets ’19, (New York, NY, USA), pp. 25–33, ACM, 2019.
- [5] Noa Zilberman, “In-Network Computing.” <https://www.sigarch.org/in-network-computing-draft/>.