ISSN 2186-7437

NII Shonan Meeting Report

No. 2018-7

Meta-Programming for Statistical Machine Learning

Oleg Kiselyov
Tiark Rompf
Jennifer Neville
Yukiyoshi Kameyama

May 22–25, 2018

National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-Ku, Tokyo, Japan

Meta-Programming for Statistical Machine Learning

Organizers:
Oleg Kiselyov (Tohoku University, Japan)
Tiark Rompf (Purdue University, USA)
Jennifer Neville (Purdue University, USA)
Yukiyoshi Kameyama (University of Tsukuba, Japan)

May 22–25, 2018

Statistical machine learning (ML) is a broad branch of machine learning aiming at drawing conclusions and learning from inherently uncertain data, using "ideas from probability theory and statistics to address uncertainty while incorporating tools from logic, databases, and programming languages to represent structure." (Getoor, Taskar, 2007).

Just as importance of ML increases, the scalability problem of developing ML applications becomes more and more pressing. Currently, applying a non-trivial machine learning task requires expertise both in the modeled domain as well as in probabilistic inference methods and their efficient implementations on modern hardware. The tight coupling between the model and the efficient inference procedure hinders making changes and precludes reuse. When the model changes significantly, the inference procedure often has to be re-written from scratch.

Probabilistic programming – which decouples modeling and inference and lets them be separately written, composed and reused – has the potential to make it remarkably easier to develop new ML tasks and keep adjusting them, while increasing confidence in the accuracy of the results. That promise has been recognized by the U.S. Defense Advanced Research Projects Agency (DARPA), which has initiated the broad program[1] "Probabilistic Programming for Advancing Machine Learning (PPAML)", started in March 2013 and running through 2017. The range of targeted applications can be seen from PPAML Challenge problems[2]. The first organizer is a part-time participant in one of the PPAML teams.

Developing the potential of probabilistic programming requires applying the recent insights from programming language (PL) research such as supercompilation from metaprogramming (with very promising results shown in Lingfeng Yang et al., AISTATS 2014). A surprising challenge is correctness: it turns out that a number of well-known and widely-used libraries and systems such as STAN may produce patently wrong results on some problems (as well-demonstrated in Hur et al., FSTTCS 2015).

[1] `http://www.darpa.mil/program/probabilistic-programming-for-advancing-machine-Learning`
[2] `http://ppaml.galois.com/`

Hand-in-hand with the interest of ML researchers in programming language topics (evidenced from PPAML PI meetings at which one of the organizers participated) is the growing interest of programming language researchers in probabilistic programming – if the record attendance of the first two POPL-affiliated workshops[3] "Probabilistic Programming Semantics" are of any indication. (The first organizer was a presenter at both workshops.)

We proposed a discussion-heavy workshop to promote the evident and growing interest of the developers of ML/probabilistic domain-specific languages in program generation and transformation, and programming language researchers in ML applications. As expected, many participants come from PPAML teams and participants of the PPS workshops. The Shonan meeting coincides with the conclusion of PPAML program. We hoped it to be a venue to discuss the not-yet-answered challenges and the issues raised at PPS workshops, but in more depth and detail.

We anticipated the workshop participants to consist of three groups of people: Statistical Machine Learning, researchers and practitioners building, using, and adjusting probabilistic learning systems, and PL researchers with some connections to ML.

- Probabilistic programming is coming of age and could really help real ML people in some cases. Selling points: correctness by construction (ML codes are very hard to debug and test) and some consistency in performance (saves time in many optimizations and writing custom code).
- Many implementors of probabilistic languages and libraries come to realize the importance of meta-programming and PL research in general (determining the validity of optimizations/transformations, knowledge of transformation techniques and good ways/algorithms of applying them, knowing tools like Lightweight Modular Staging (LMS), partial evaluators, staged languages).
- Treating programs as subjects of probabilistic computation, in the sense of learning facts about programs from data, i.e. learning from "big code".

Our goal was to bring three groups together and see what probabilistic programming can do more, and mainly how we can apply advances in PL and meta-programming more consciously and profitably (and if we cannot, what the PL community should be investigating then).

Just as the two Shonan meetings (No.2012-4 "Bridging the Theory of Staged Programming Languages and the Practice of High-Performance Computing" and No. 2014-7 "Staging and High-Performance Computing: Theory and Practice") aimed to solicit and discuss real-world applications of assured code generation in HPC (High-Performance Computing) that would drive PL research in meta-programming, we proposed a similar direction for ML and meta-programming.

To promote mutual understanding, we planned for the workshop to have lots of time for discussion. We emphasized tutorial, brainstorming and working-group sessions rather than mere conference-like presentations.

[3] `http://pps2016.soic.indiana.edu/` `http://pps2017.soic.indiana.edu/`

List of Participants

The following people have participated in the seminar, beside the organizers.

1. Chung-chieh Shan, Indiana University (USA)
2. Nada Amin, University of Cambridge (UK)
3. Ohad Kammar, University of Oxford (UK)
4. Praveen Narayanan, Indiana University (USA)
5. Kohei Suenaga, Kyoto University (JP)
6. Shin-ya Katsumata, National Institute of Informatics (JP)
7. Hiroshi Unno, University of Tsukuba (JP)
8. Rob Zinkov, Indiana University (USA)
9. Adam Ścibior, University of Cambridge (UK)
10. Luke Ong, University of Oxford (UK)

Main Questions

The following questions were raised repeatedly during the seminar:

What is correctness and how to measure it? Different communities have different takes. In programming languages we consider correctness as program (e.g., an implementation of inference algorithm) satisfying its specification. On the other hand, modeling and machine learning community looks at correctness as model adequately representing data and generating useful predictions. Modeling community is hence rather accepting towards 'faulty' implementations (which fail in some cases) so long as they are useful in evaluating the model. We have to be very clear what notion of correctness is at hand at each point in a discussion.

How to represent complex data? ML learning community has need to represent complex data (for example, complicated graphs); in contrast, many probabilistic programming languages support only very simple data types (in the extreme, only collections of floating-point numbers). Related is the question of a suitable notation for these complex data types.

How to compose inference algorithms and control them? On one hand we would like to compose inference procedures as programs from library modules, where modules are treated as black box and freely substitutable (provided they support the same interface). On the other hand, there are compelling examples for knowing the details of the inference algorithm so to 'guide' it (e.g., to 'weight' the distributions used therein).

Tangible Outcomes

The repository of challenge problems The goals and the content of the repository has been discussed at length. The repository should contain the variety of models (from simple ones to research topics) plus the sample input data for them. It is intended that participants and the community at large will submit various inference procedures (or the same procedure but implemented in different languages or approaches).

Adam Ścibior has volunteered to start up the repository and add one sample model.

The repository of typical bugs Rob Zinkov volunteered to start the repository with examples for typical bugs encountered while implementing probabilistic programming languages and various inference procedures. Hopefully, the repository would turn into a paper in "Software: Practice and Experience" and help with the very tough problem of determining what went wrong.

What the existing tools are lacking in, exactly In other words, what makes ML practitioners curse at their computer. Jennifer Neville will ask her graduate students.

Meeting Schedule

May 22 (Tuesday)

Theme: Introductions, background tutorials

- Self-introductions
- Machine Learning 101 (Jennifer Neville)
- Probabilistic Programming still matters (Rob Zinkov)

May 23 (Wednesday)

Theme: Last Tutorials. What are the pressing problems

- Probabilistic Metaprogramming Tutorial (Tiark Rompf)
- Discussion: Crystallizing Future Directions (Ken Shan)
- Discussion: What Repository do we want (Ken Shan)
- Verifiable and Reusable Metaprograms for Bayesian Inference (Praveen Narayanan)
- Inference Programming (Adam Ścibior)

May 24 (Thursday)

Theme: Correctness, Excursion

- Discussion: What do we mean by 'correctness' and what do we want it to mean (Ken Shan, Ohad Kammar, Rob Zinkov)
- Excursion and Main Banquet

May 25 (Friday)

Theme: Formality, Conclusions

- Disintegration with more general measures (Luke Ong, Ken Shan)
- Group discussion: How to continue, which papers to write, which languages to try