Bring Your Own Datatypes: Enabling Number Format Exploration with TVM

Gus Smith
University of Washington

sampl

PAUL G. ALLEN SCHOOL
OF COMPUTER SCIENCE & ENGINEERING

tvm.ai

In this session

In this session

- Why you should care about new datatypes

In this session

- Why you should care about new datatypes
- How TVM supports datatype exploration

In this session

- Why you should care about new datatypes
- How TVM supports datatype exploration

    Currently: **software** support only

In this session

- Why you should care about new datatypes
- How TVM supports datatype exploration
  - Currently: **software** support only
  - Future directions: **hardware** support

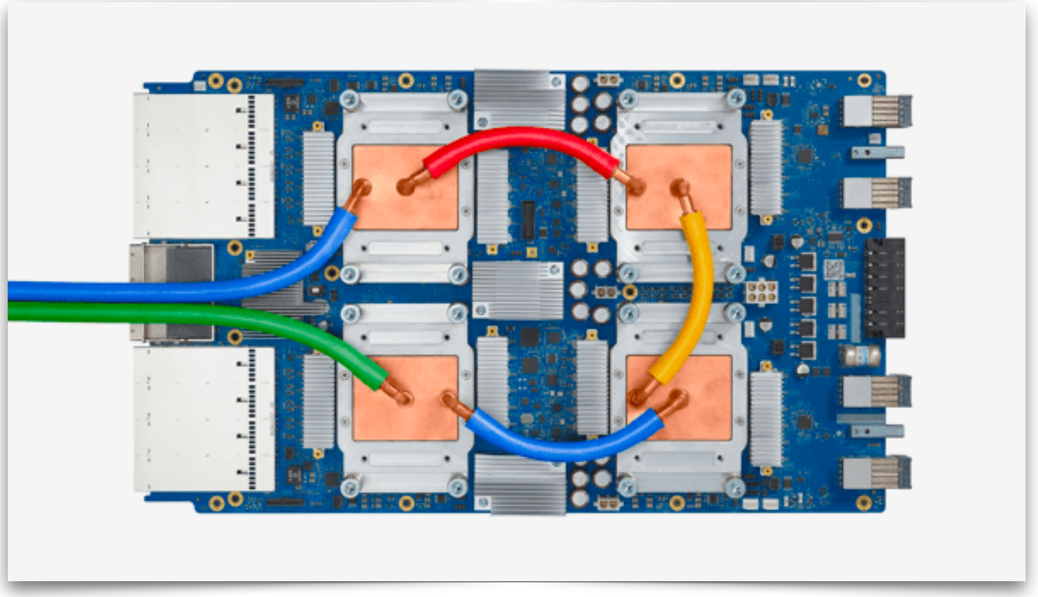
In this session

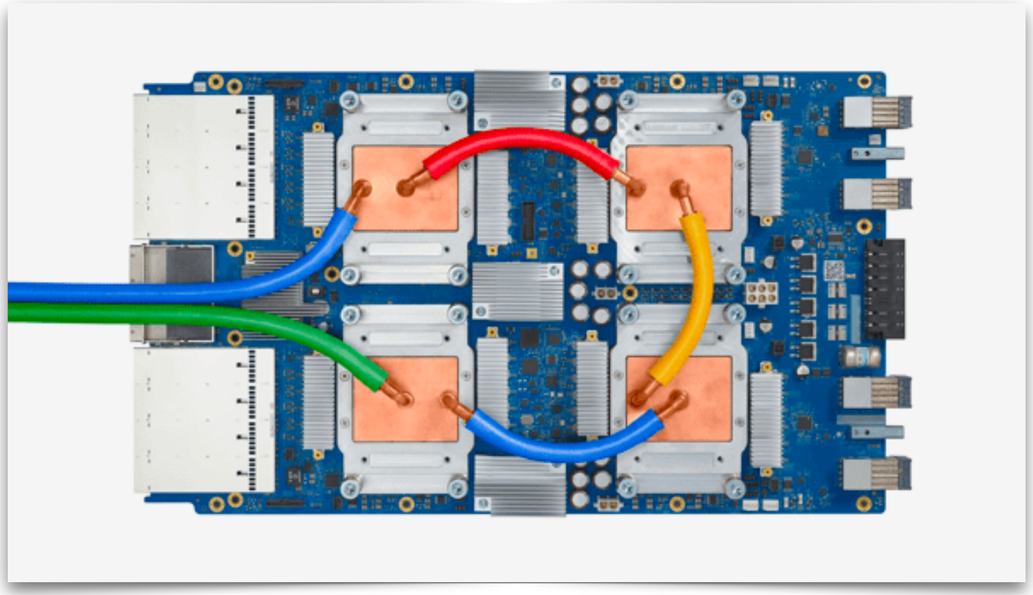
- Why you should care about new datatypes
- How TVM supports datatype exploration

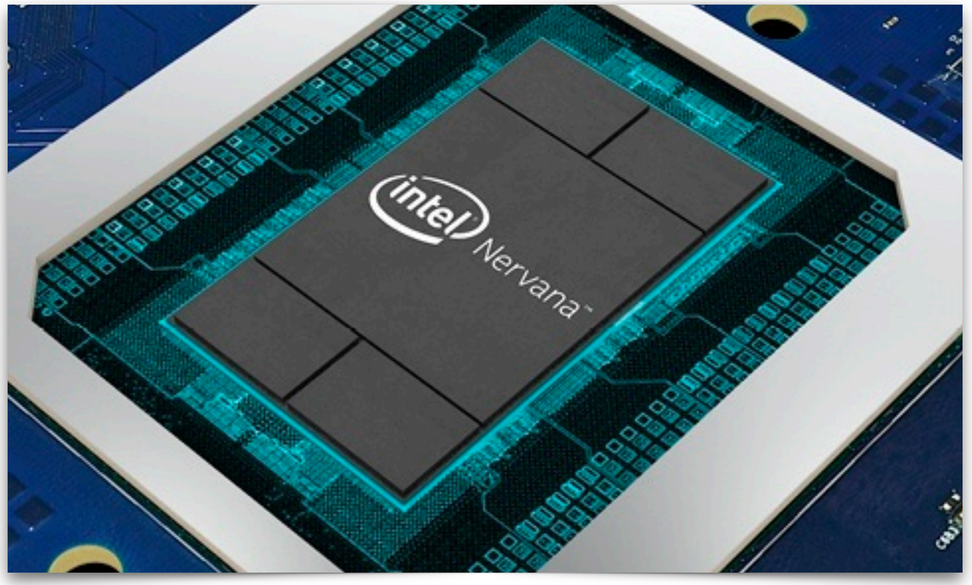
  Currently: **software** support only

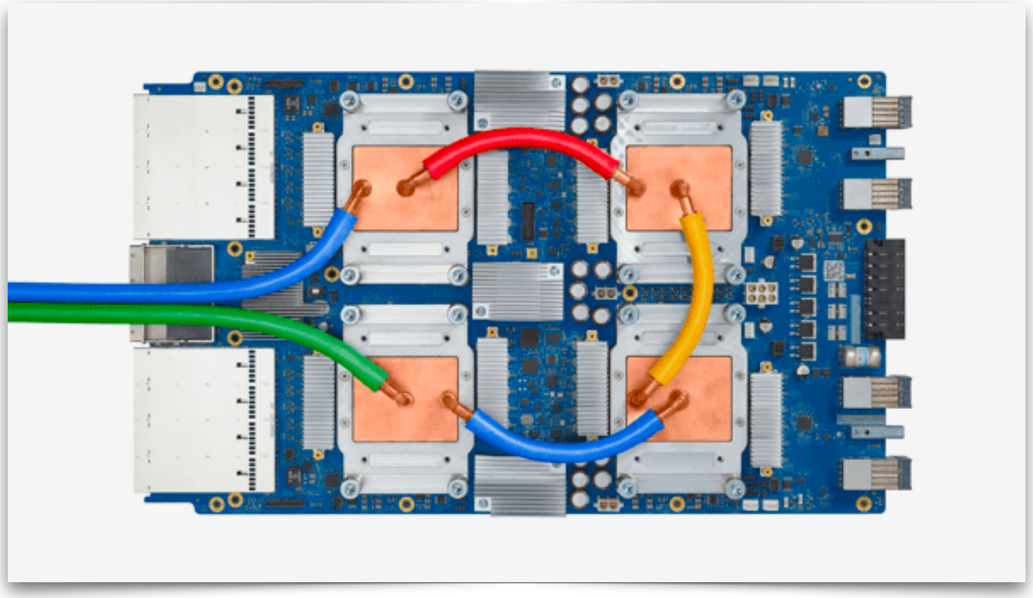
  Future directions: **hardware** support

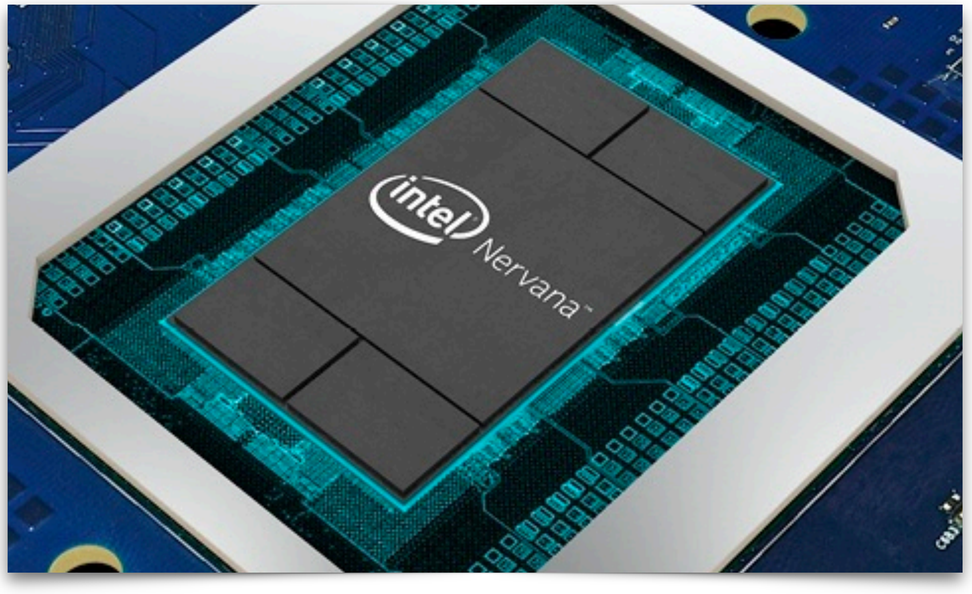
- …and of course, a coding session!

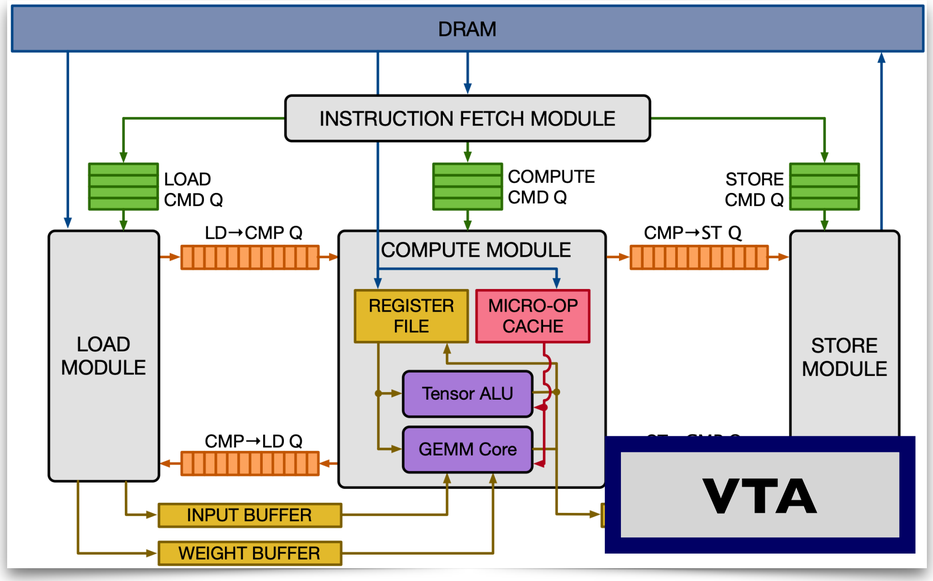


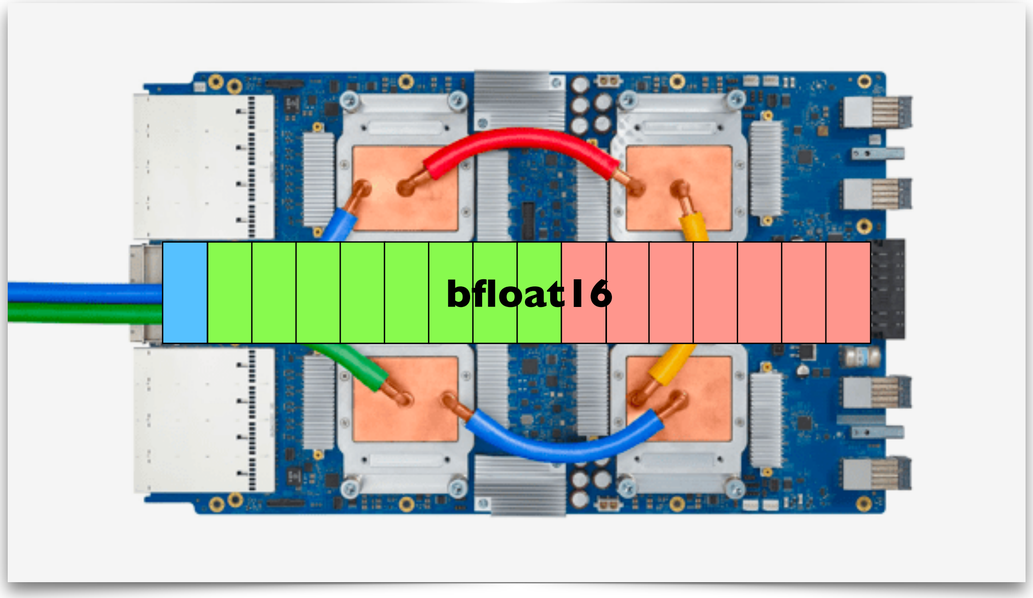


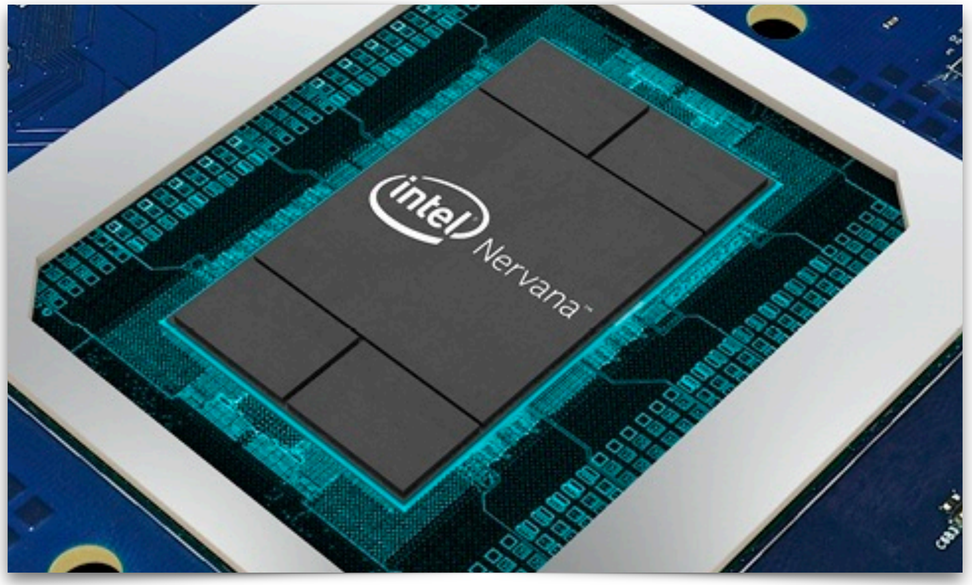

intel
Nervana

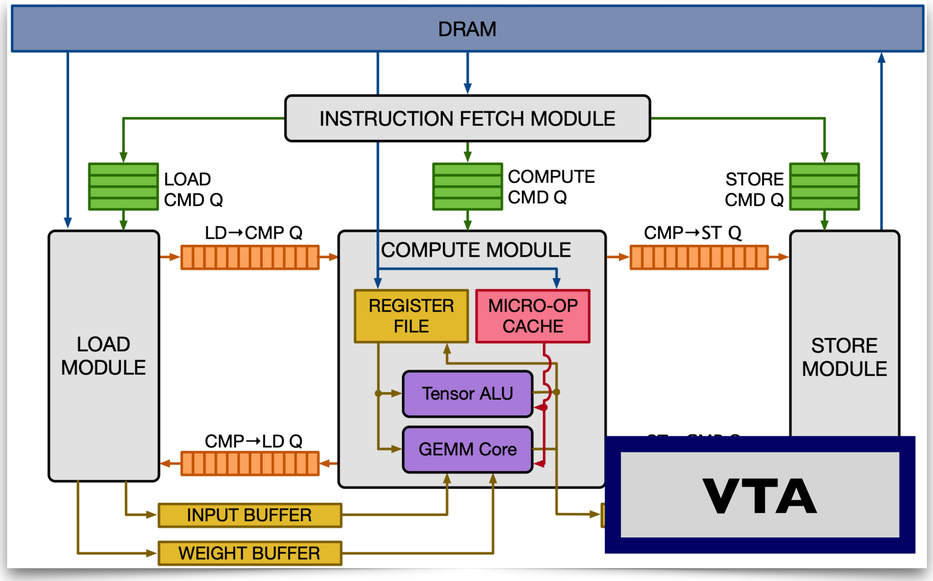


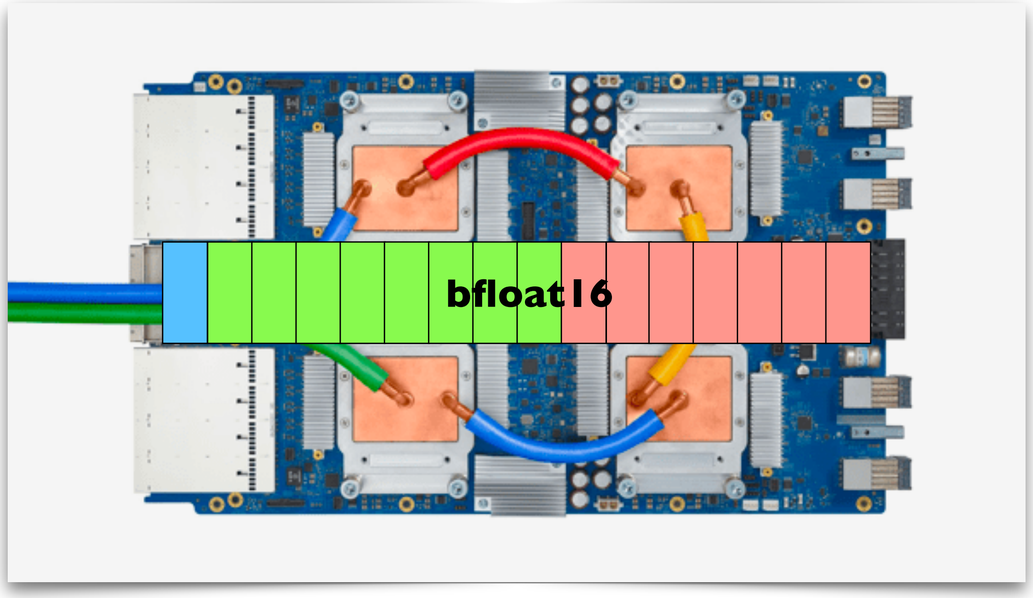

intel
Nervana

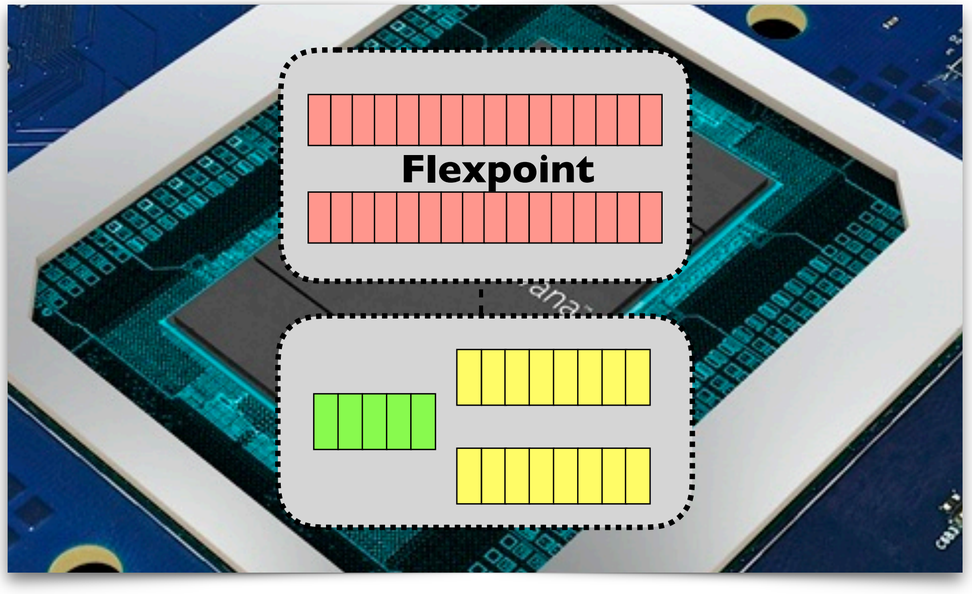

DRAM
INSTRUCTION FETCH MODULE
LOAD CMD Q
COMPUTE CMD Q
STORE CMD Q
LD→CMP Q
CMP→ST Q
COMPUTE MODULE
LOAD MODULE
REGISTER FILE
MICRO-OP CACHE
STORE MODULE
Tensor ALU
GEMM Core
CMP→LD Q
INPUT BUFFER
WEIGHT BUFFER
VTA

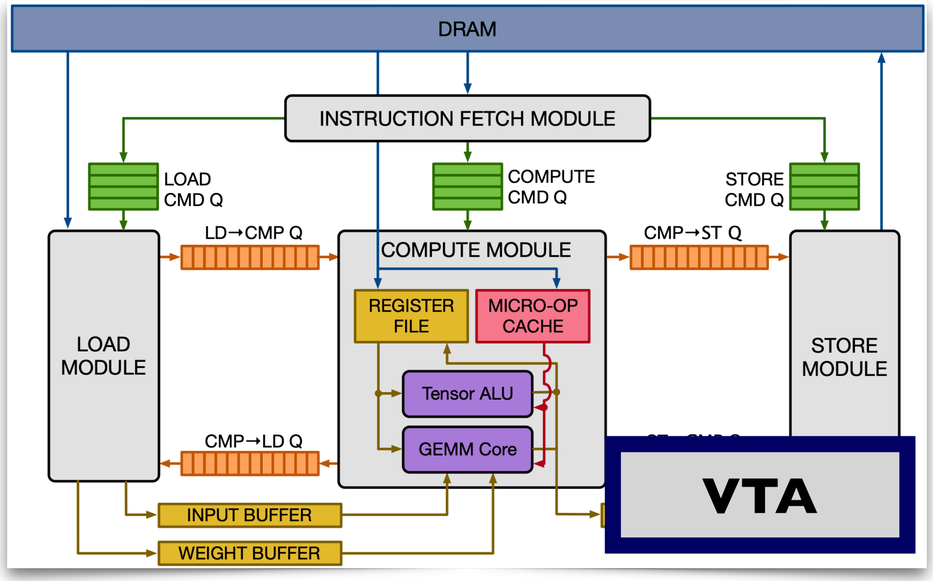

bfloat16

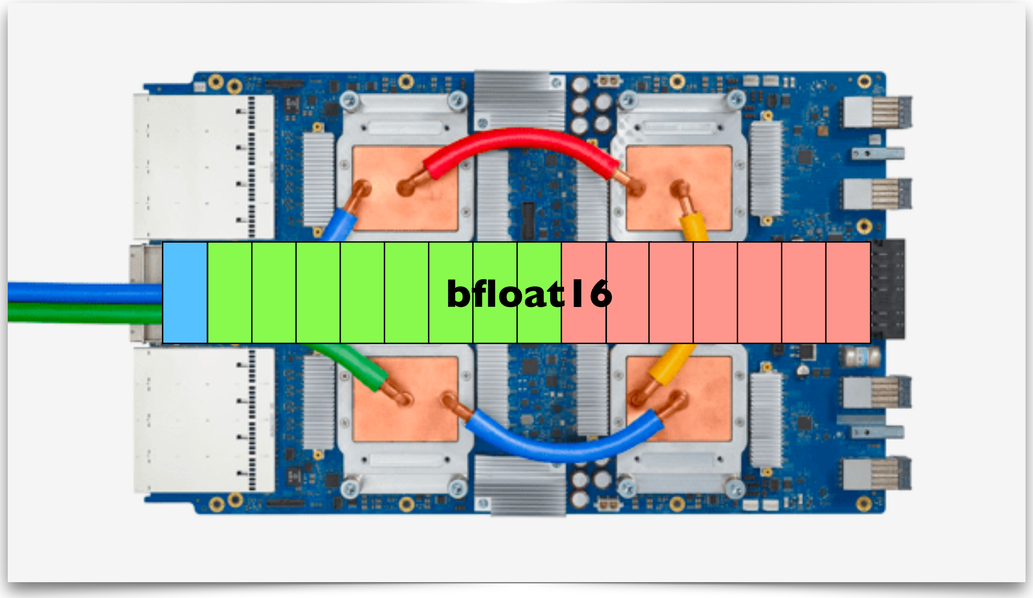

intel
Nervana

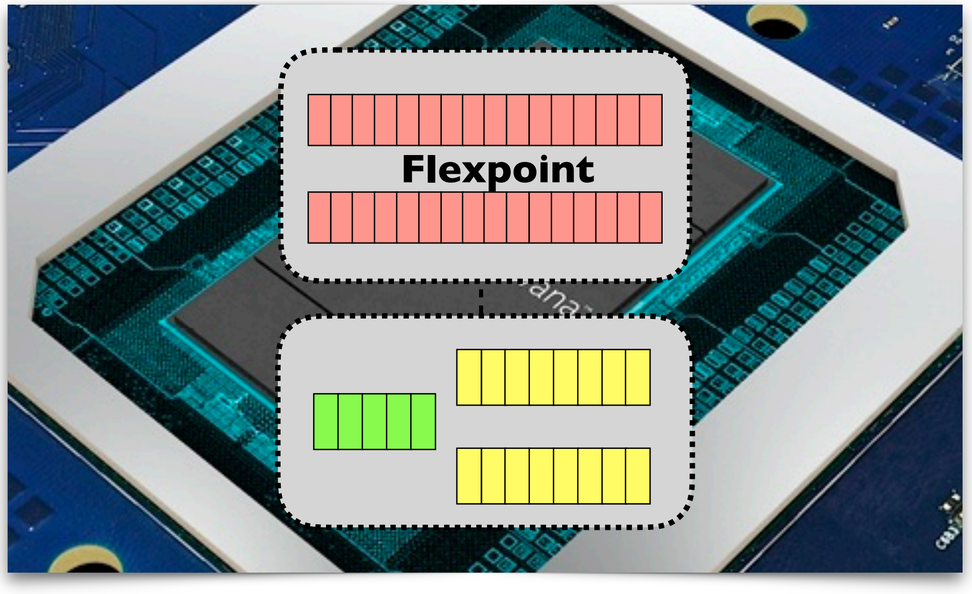

DRAM
INSTRUCTION FETCH MODULE
LOAD CMD Q
COMPUTE CMD Q
STORE CMD Q
LD→CMP Q
CMP→ST Q
COMPUTE MODULE
LOAD MODULE
REGISTER FILE
MICRO-OP CACHE
STORE MODULE
Tensor ALU
GEMM Core
CMP→LD Q
INPUT BUFFER
WEIGHT BUFFER
VTA

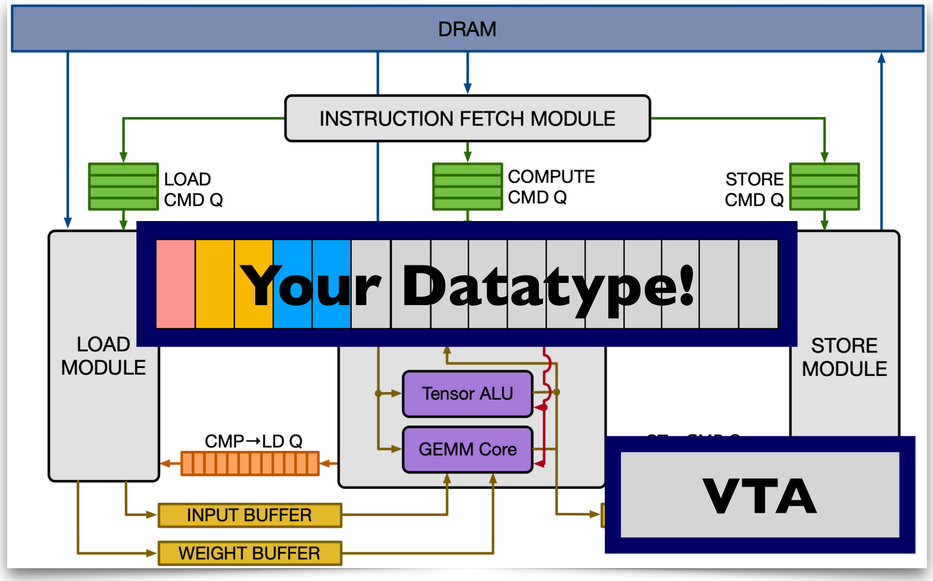

bfloat16


Flexpoint

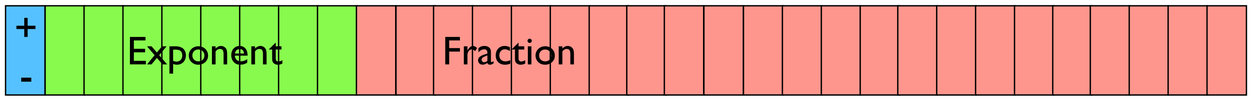

DRAM
INSTRUCTION FETCH MODULE
LOAD CMD Q
COMPUTE CMD Q
STORE CMD Q
LD→CMP Q
CMP→ST Q
COMPUTE MODULE
REGISTER FILE
MICRO-OP CACHE
LOAD MODULE
STORE MODULE
Tensor ALU
GEMM Core
CMP→LD Q
INPUT BUFFER
WEIGHT BUFFER
VTA


bfloat16

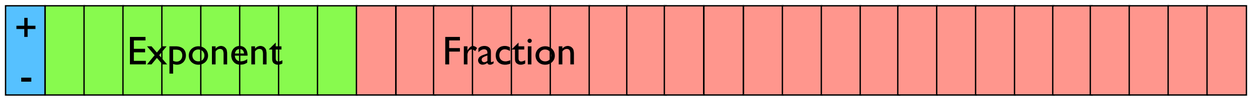

Flexpoint


DRAM
INSTRUCTION FETCH MODULE
LOAD CMD Q
COMPUTE CMD Q
STORE CMD Q
Your Datatype!
LOAD MODULE
STORE MODULE
Tensor ALU
GEMM Core
CMP→LD Q
INPUT BUFFER
WEIGHT BUFFER
VTA

What's wrong with IEEE floats?

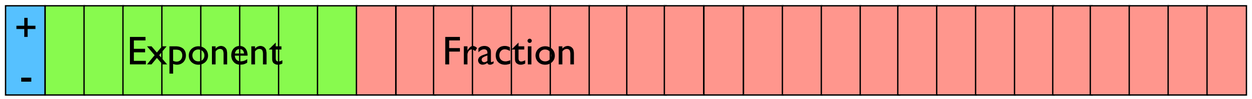


What's wrong with IEEE floats?



IEEE floats…

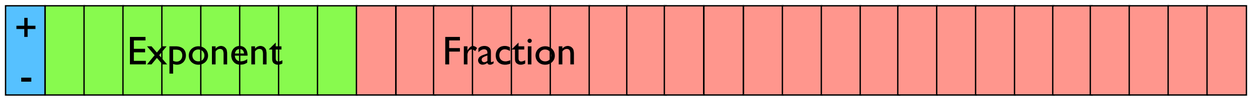
What's wrong with IEEE floats?



IEEE floats…

- take up too much space,

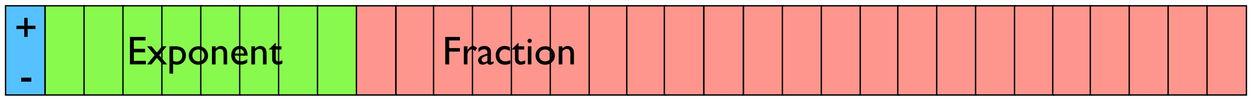
What's wrong with IEEE floats?



IEEE floats…

- take up too much space,
- have poor *dynamic range* for their size, and

What's wrong with IEEE floats?

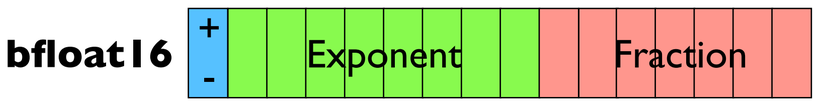


IEEE floats…

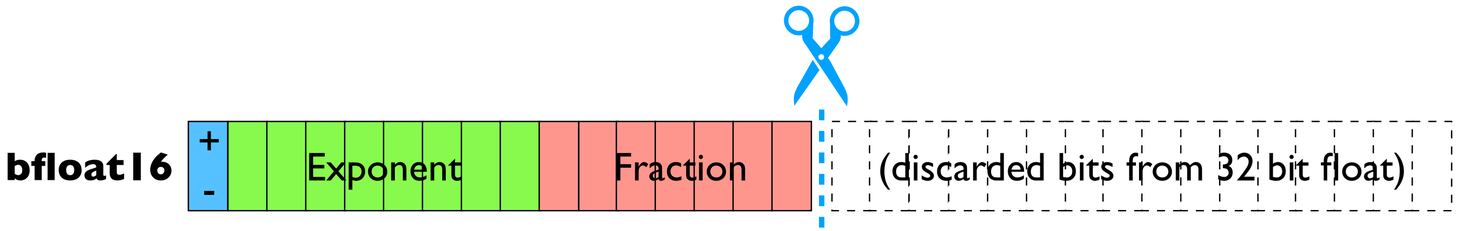
- take up too much space,
- have poor *dynamic range* for their size, and
- require overly-complex hardware

Modern Datatype Examples

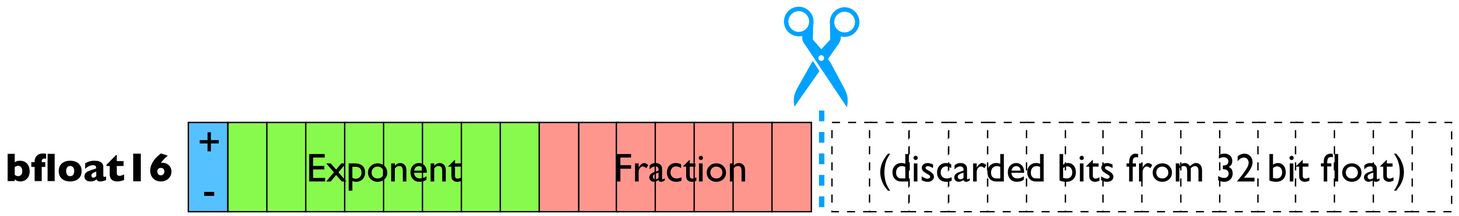
Modern Datatype Examples

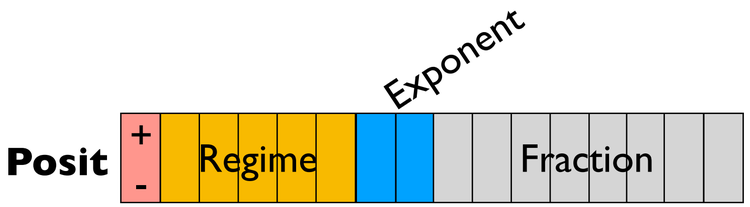


Modern Datatype Examples

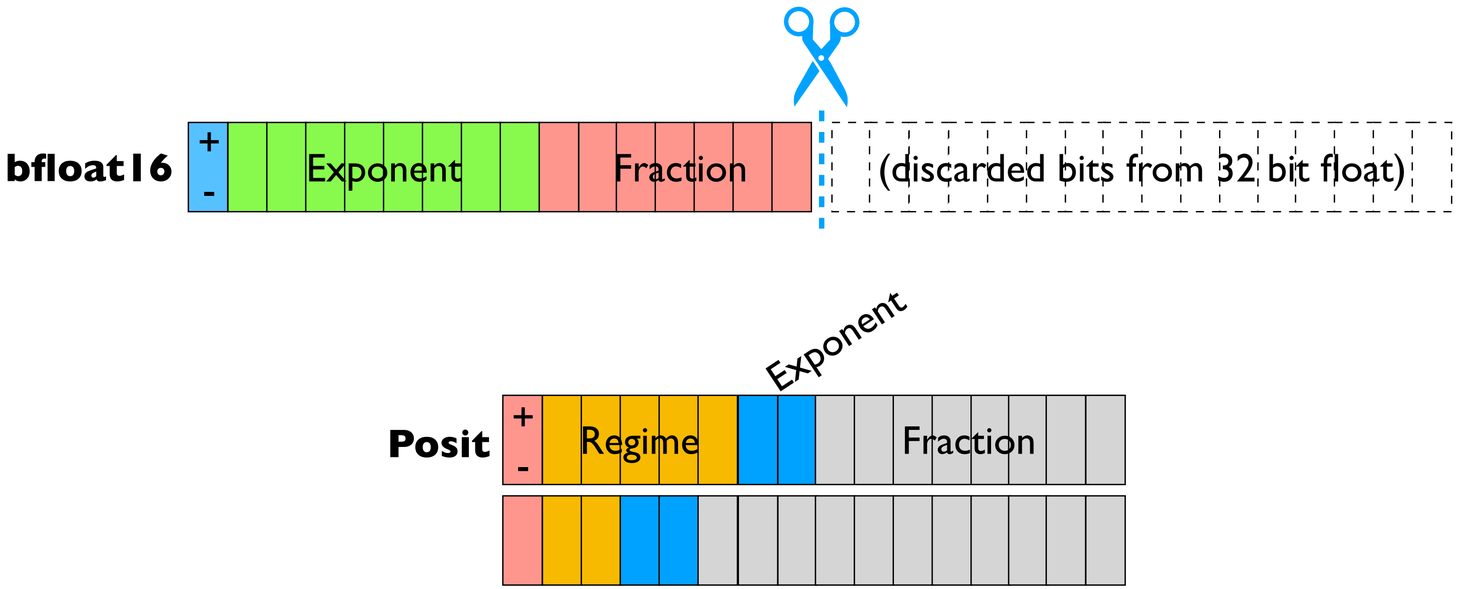


Modern Datatype Examples

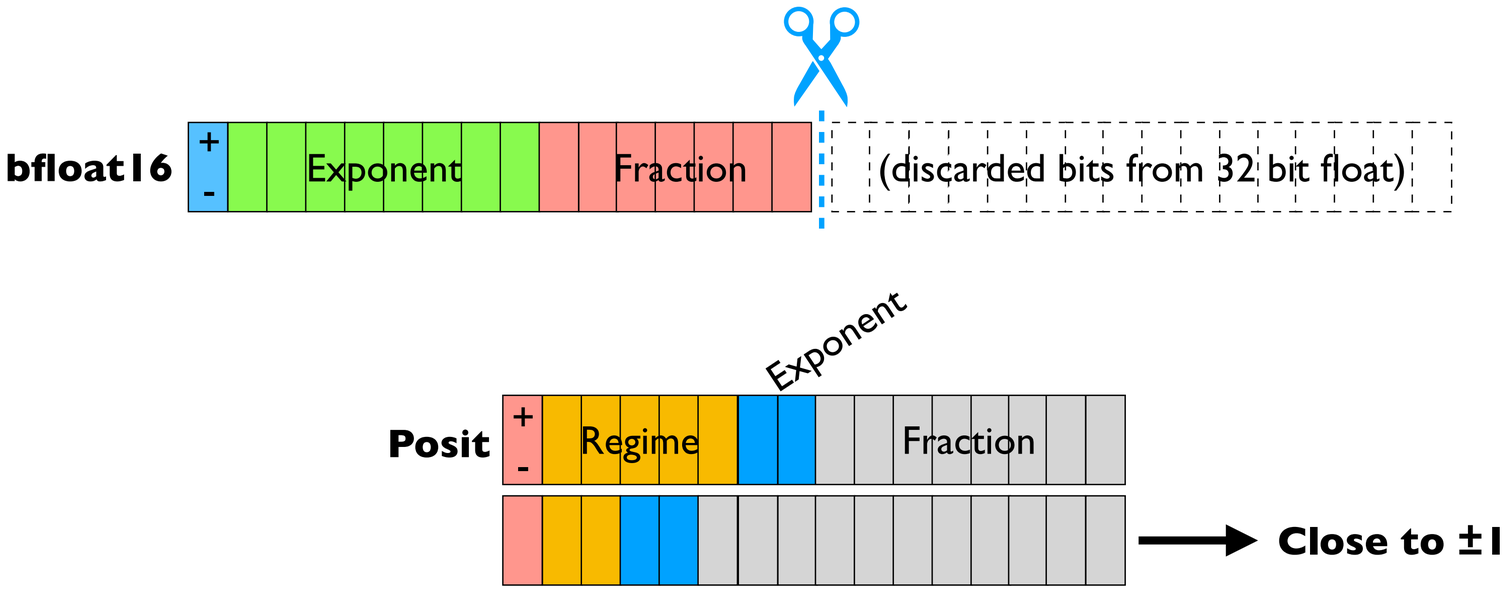




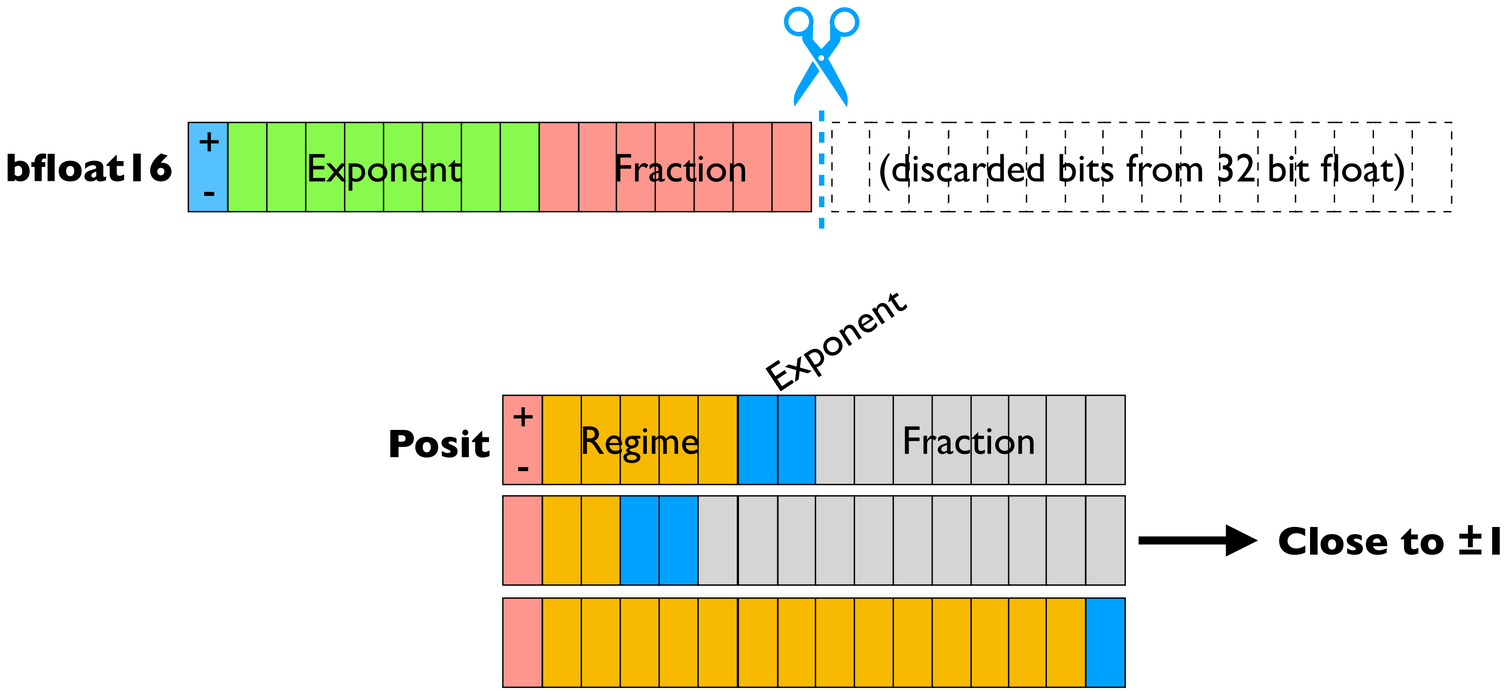
Modern Datatype Examples



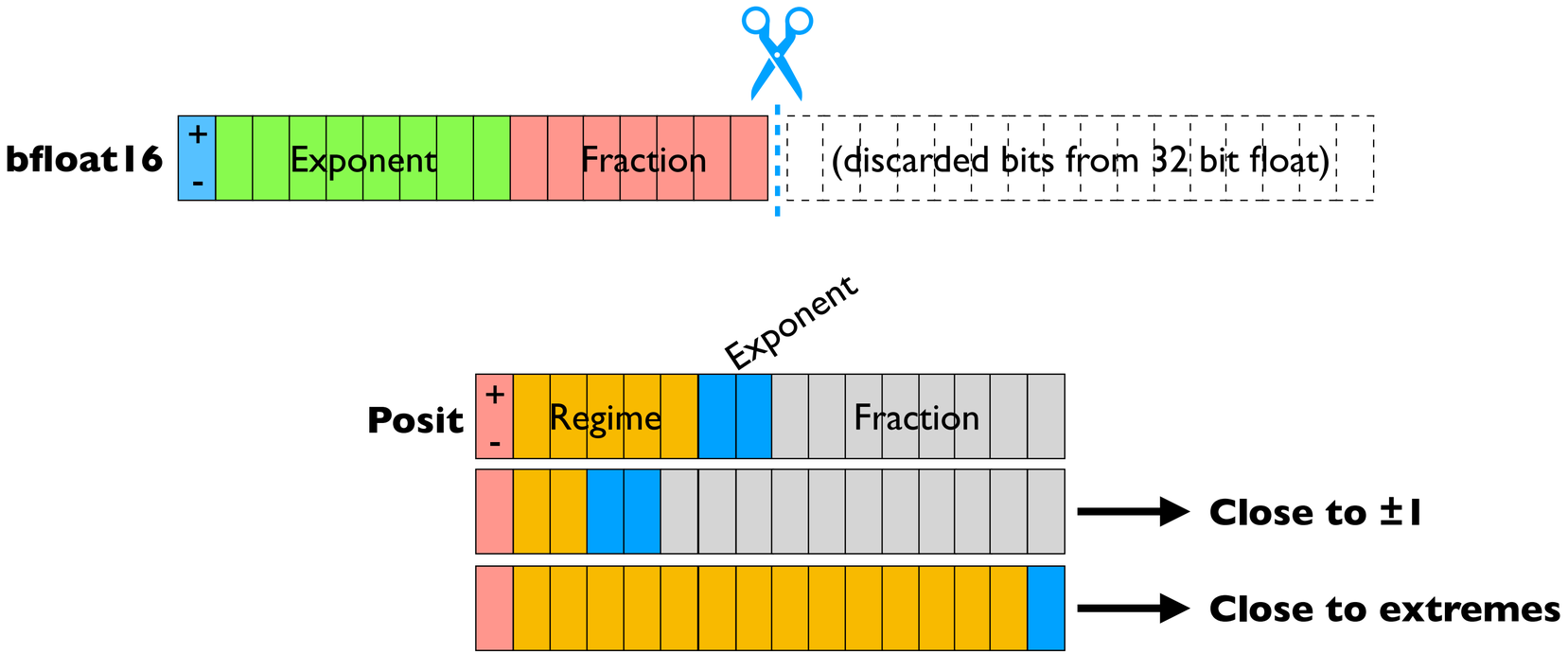
Modern Datatype Examples



Modern Datatype Examples



Modern Datatype Examples



Deep learning needs hardware specialization

Deep learning needs hardware specialization
…and hardware specialization needs new datatypes!

Our goal is to **make datatype exploration easy** with TVM.

Our goal is to **make datatype exploration easy** with TVM.

This helps

Our goal is to **make datatype exploration easy** with TVM.

This helps

- **datatype researchers,** who need easy ways to test their datatypes,

Our goal is to **make datatype exploration easy** with TVM.

This helps

- **datatype researchers,** who need easy ways to test their datatypes,
- but also, **all TVM users,** who can easily use new datatypes in their models.

Bring Your Own Datatypes

How to Use New Datatypes (without TVM!)

How to Use New Datatypes (without TVM!)

How to Use New Datatypes (without TVM!)

To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

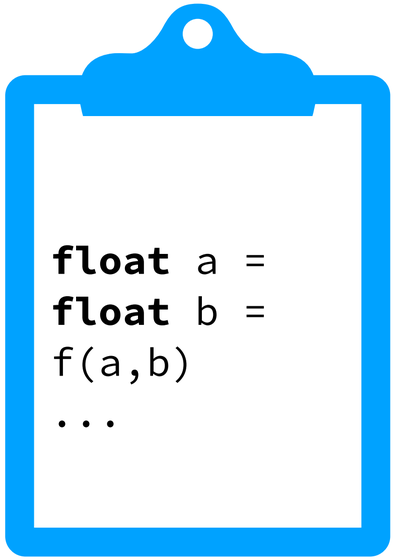
How to Use New Datatypes (without TVM!)

To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

How to Use New Datatypes (without TVM!)

To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

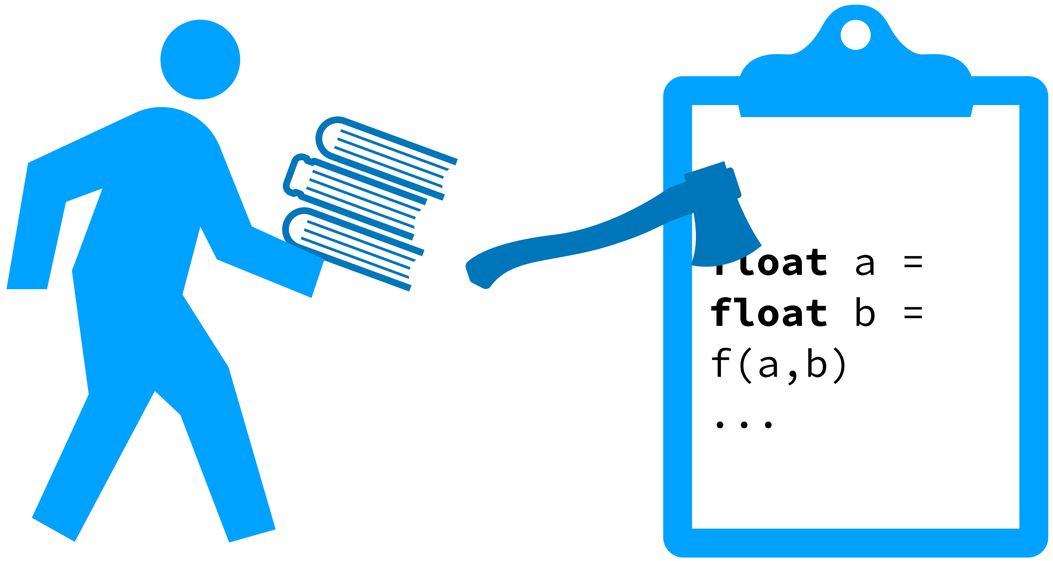
To use the datatype in real workloads, such as deep learning models, users will have to hack the library into the workload.



How to Use New Datatypes (without TVM!)

To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

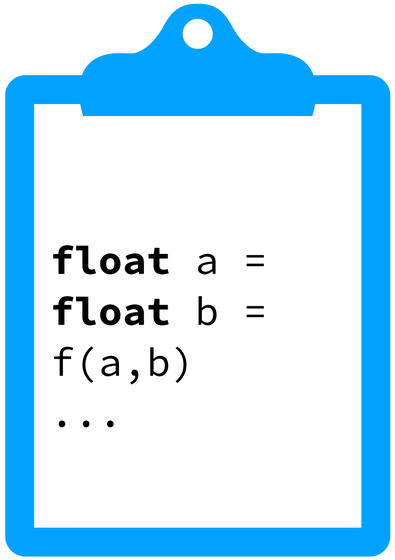
To use the datatype in real workloads, such as deep learning models, users will have to hack the library into the workload.



How to Use New Datatypes (without TVM!)

To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

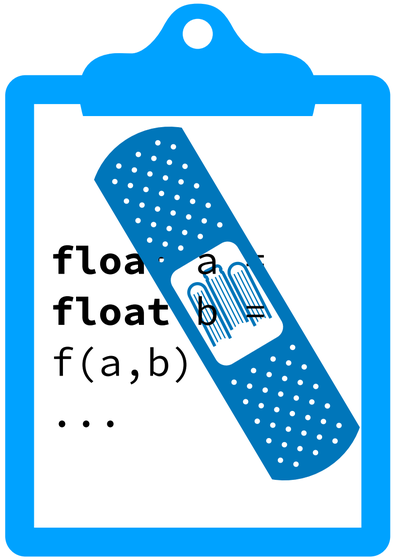
To use the datatype in real workloads, such as deep learning models, users will have to hack the library into the workload.



How to Use New Datatypes (without TVM!)

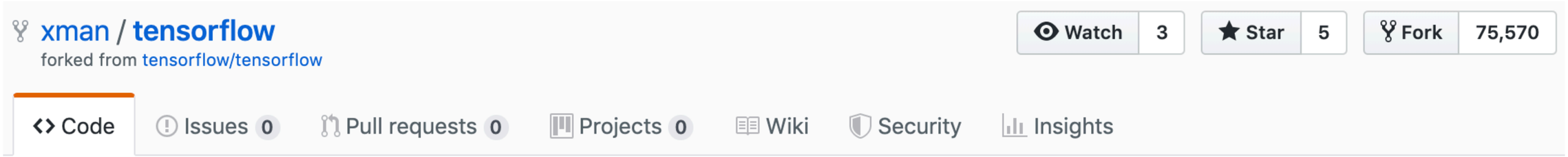
To prototype a hardware datatype, researchers will emulate the datatype in software by building a software library.

To use the datatype in real workloads, such as deep learning models, users will have to hack the library into the workload.



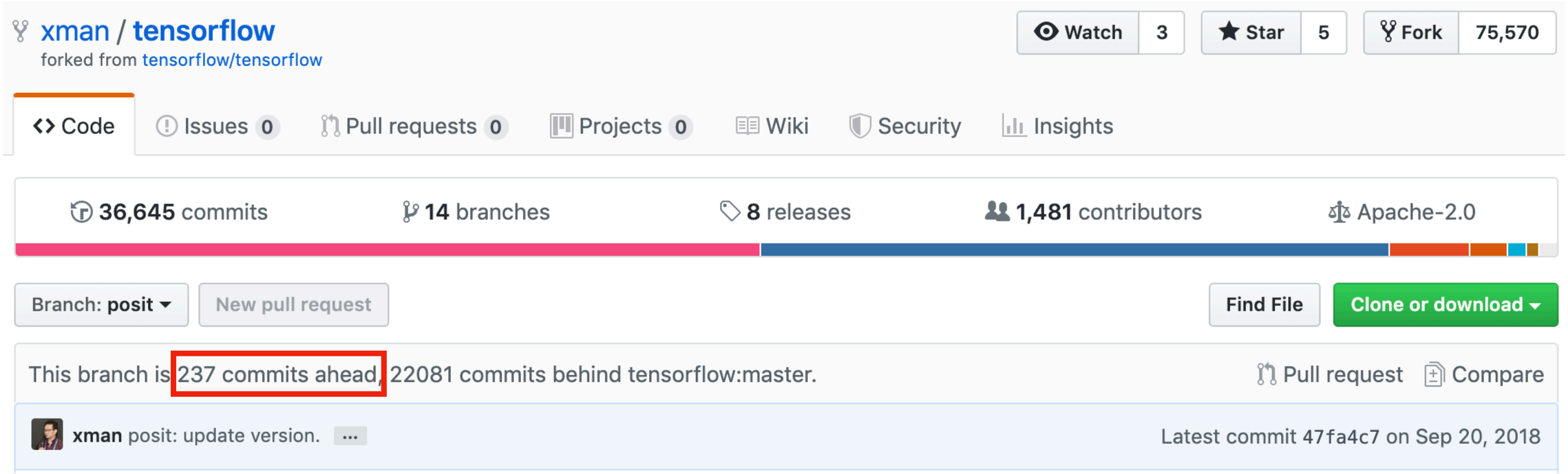
An Example in the Wild!

An Example in the Wild!



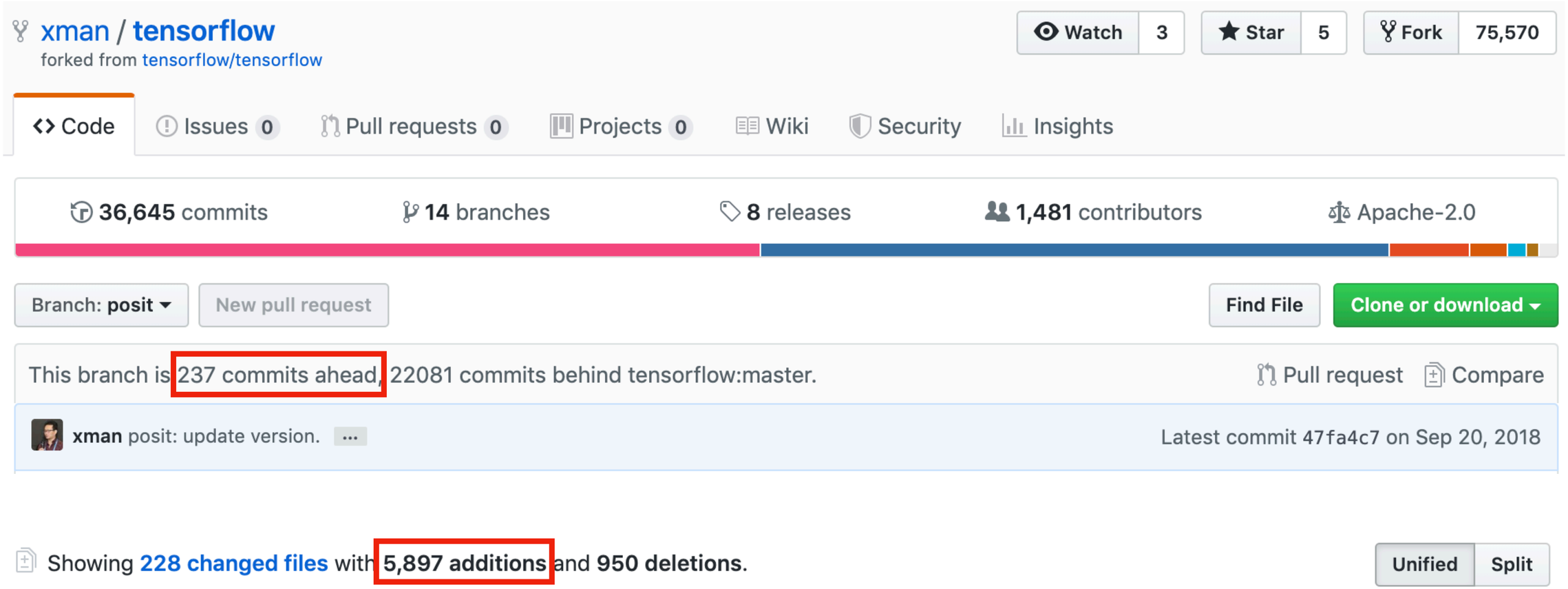
https://github.com/xman/tensorflow/tree/posit

An Example in the Wild!



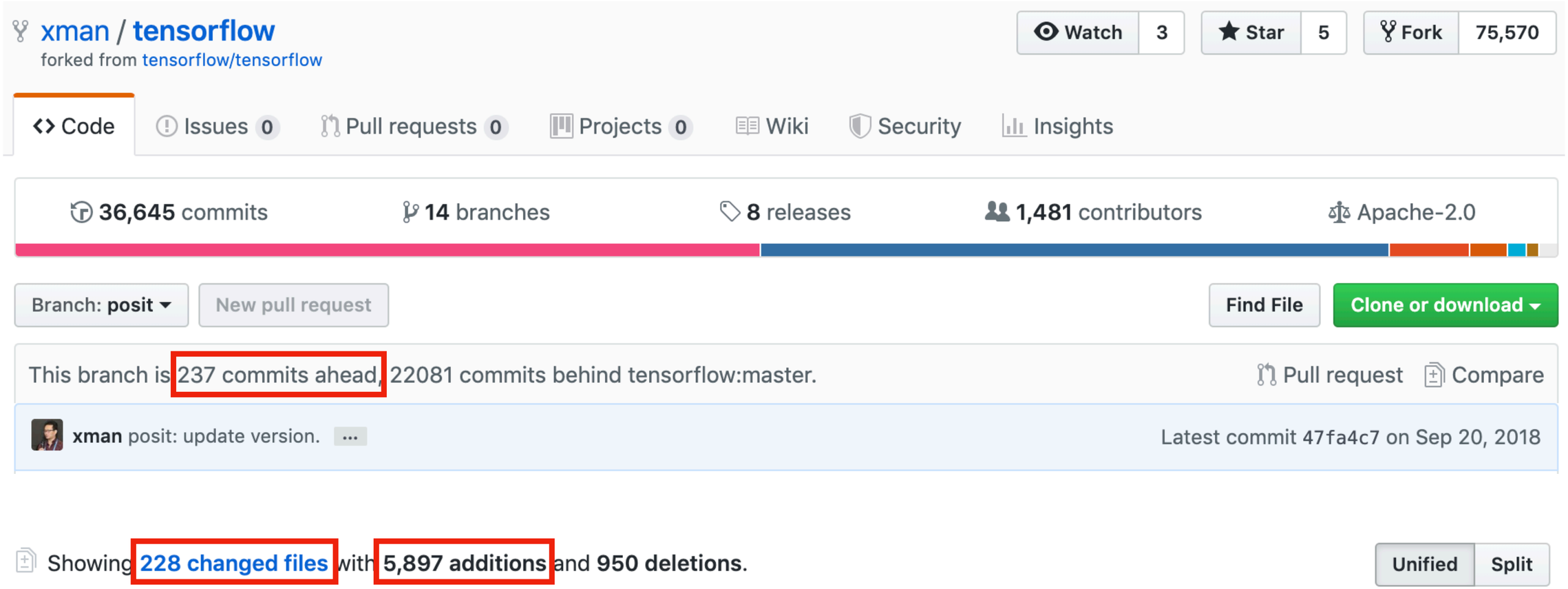
https://github.com/xman/tensorflow/tree/posit

An Example in the Wild!



https://github.com/xman/tensorflow/tree/posit

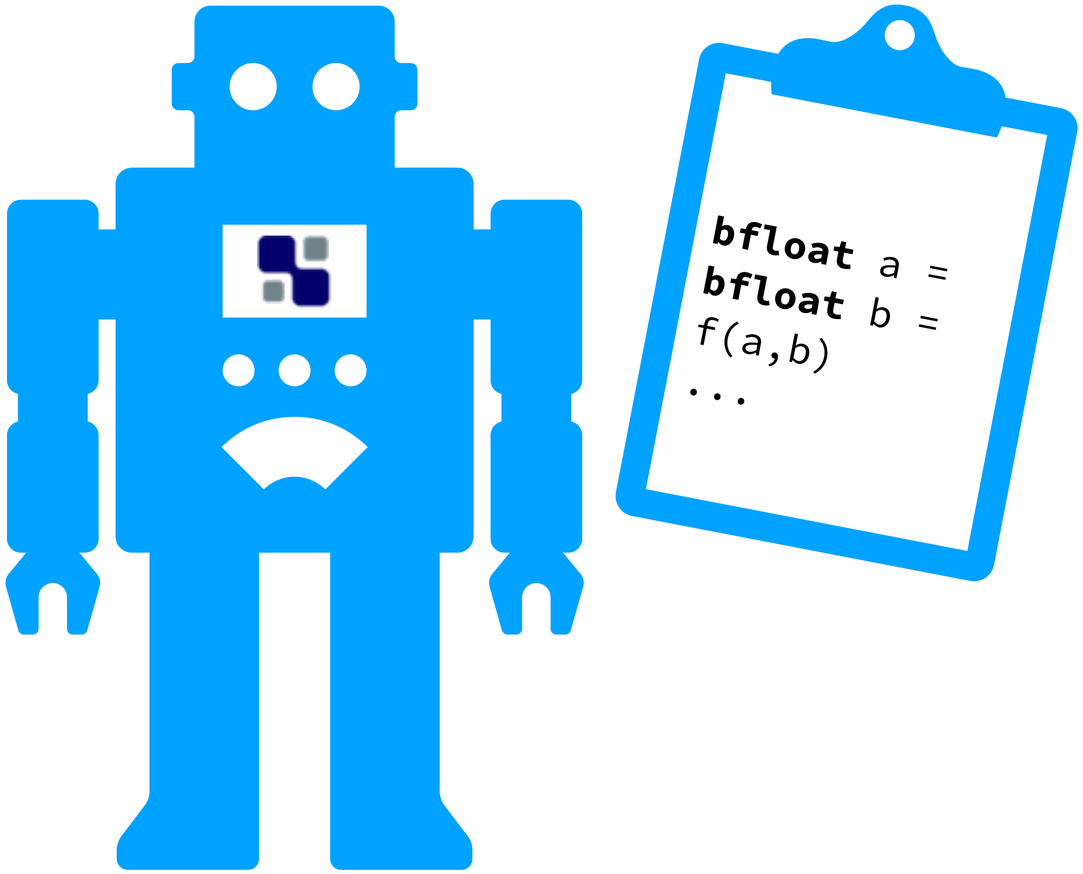
An Example in the Wild!



https://github.com/xman/tensorflow/tree/posit

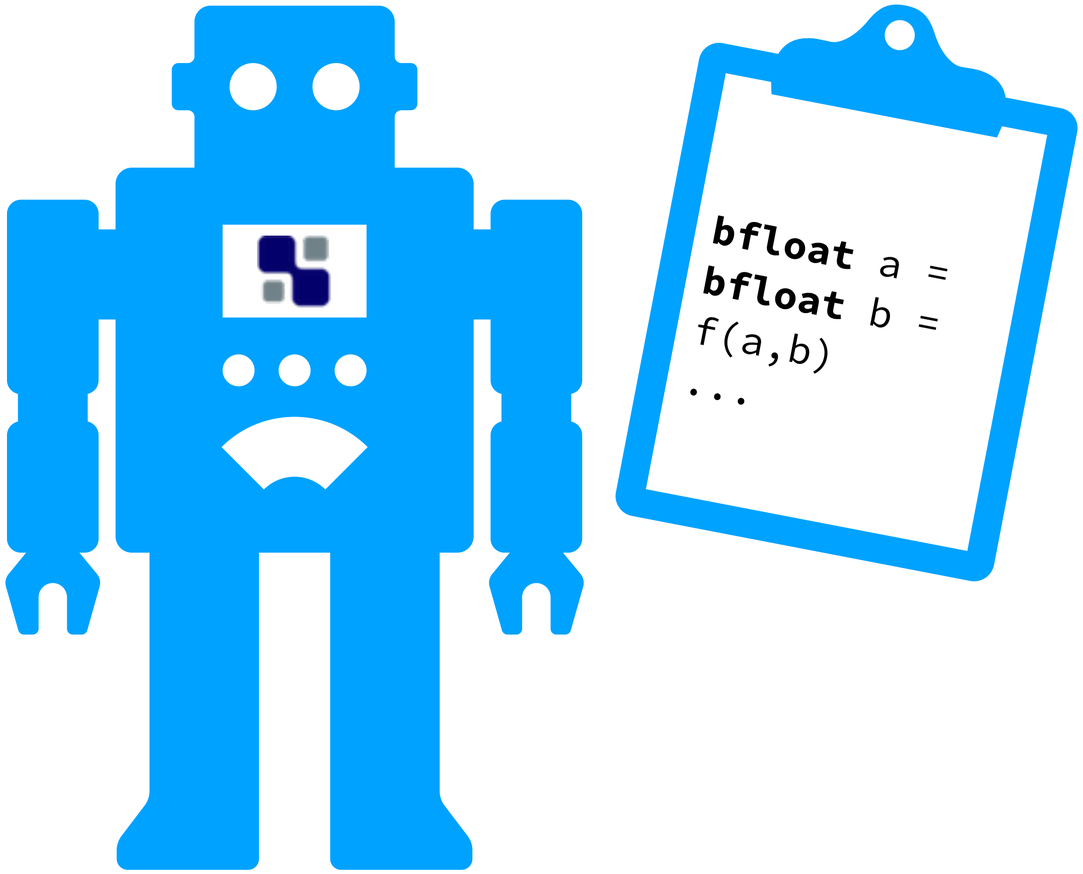
Can we do better? Can we let users bring their own datatypes to TVM, and have TVM do the rest of the work?

How to Use New Datatypes (with TVM!)



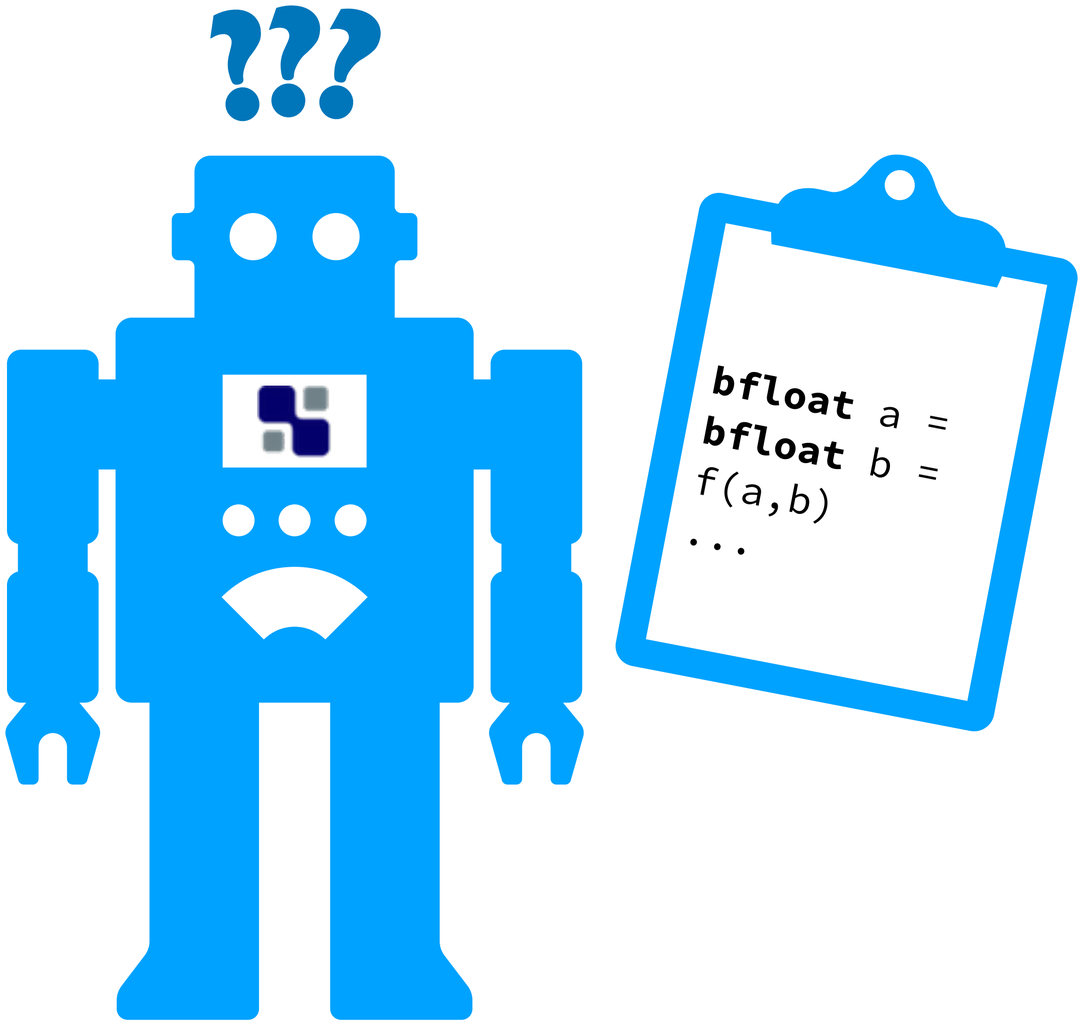
How to Use New Datatypes (with TVM!)

Out of the box, TVM does not know how to interpret programs with custom datatypes.



How to Use New Datatypes (with TVM!)

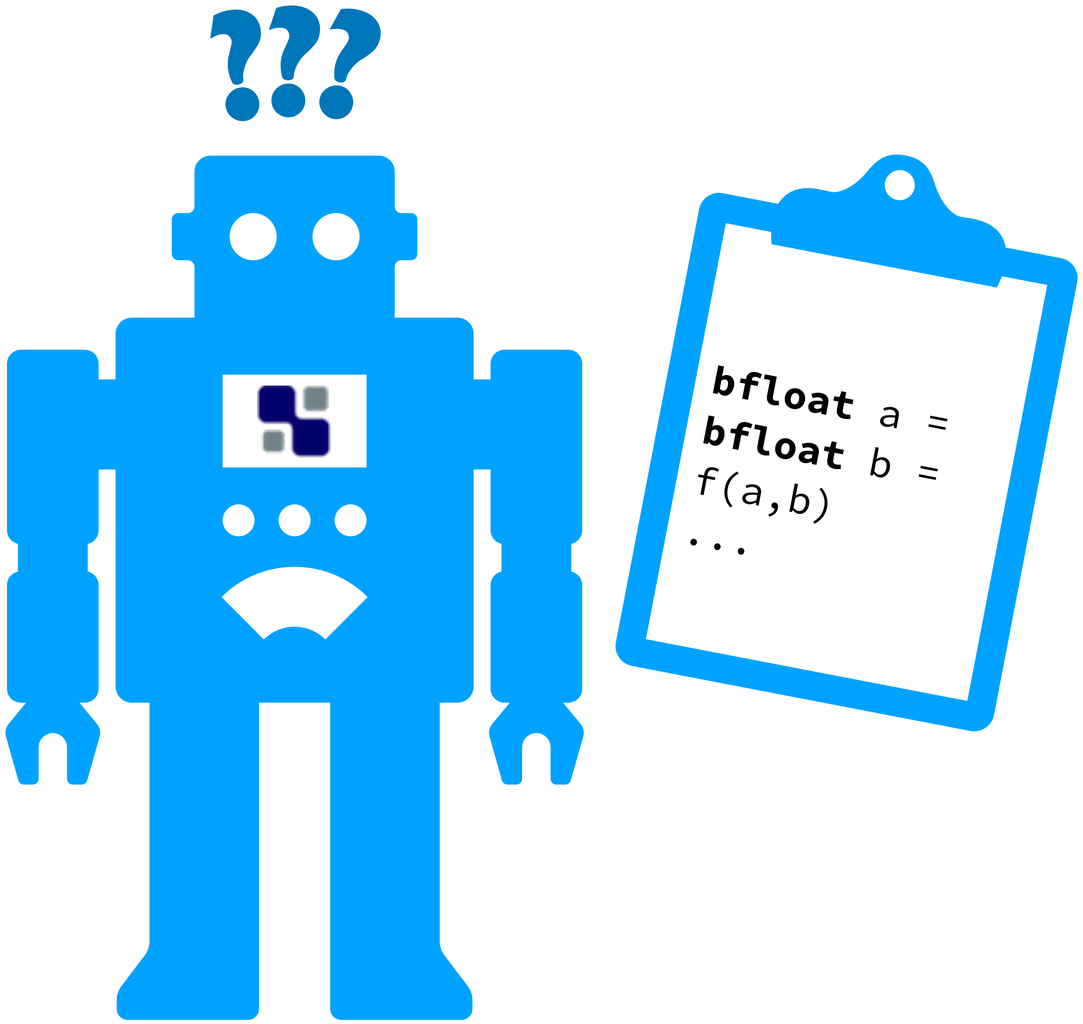
Out of the box, TVM does not know how to interpret programs with custom datatypes.



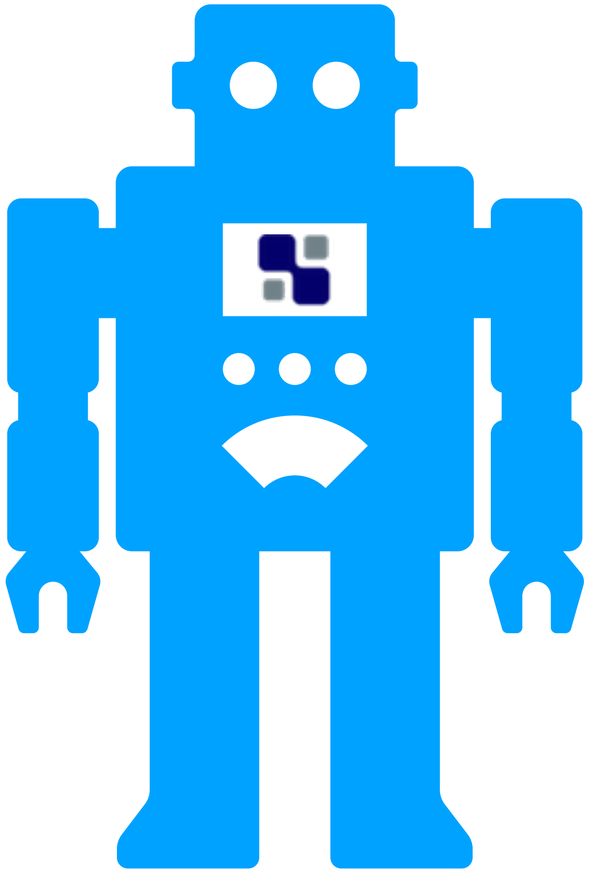
How to Use New Datatypes (with TVM!)

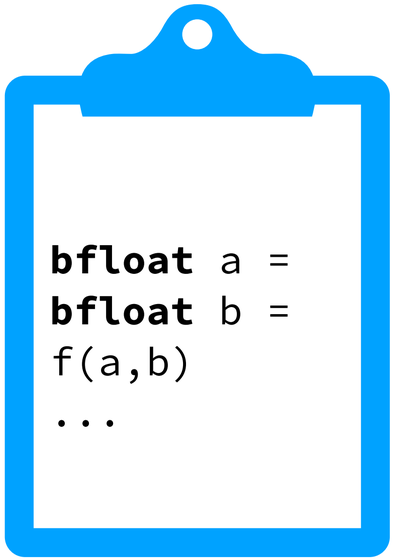
Out of the box, TVM does not know how to interpret programs with custom datatypes.

But, with a bit of information, TVM can compile and run these programs easily!

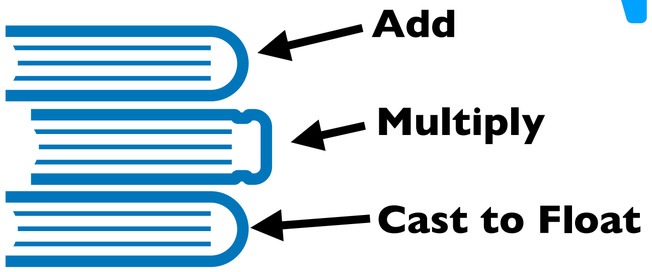


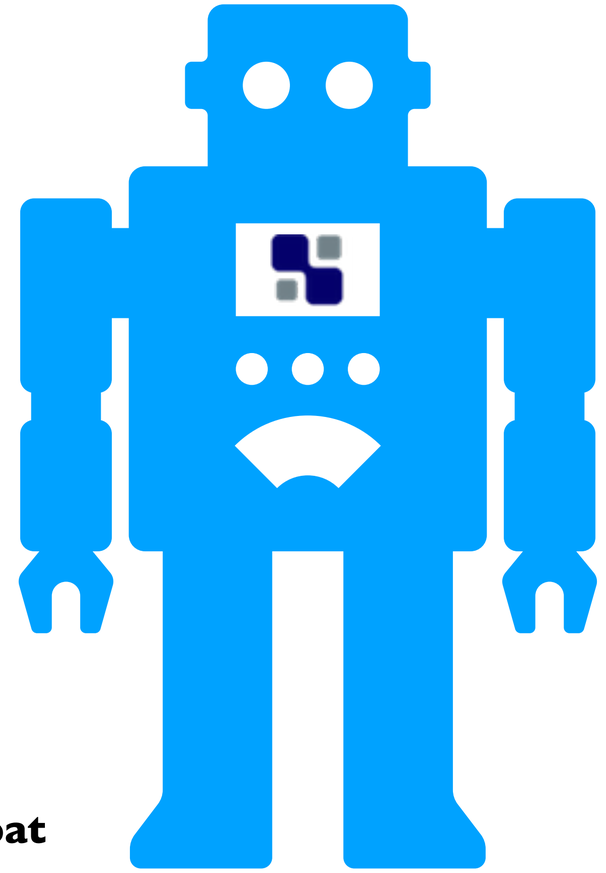
How to Use New Datatypes (with TVM!)

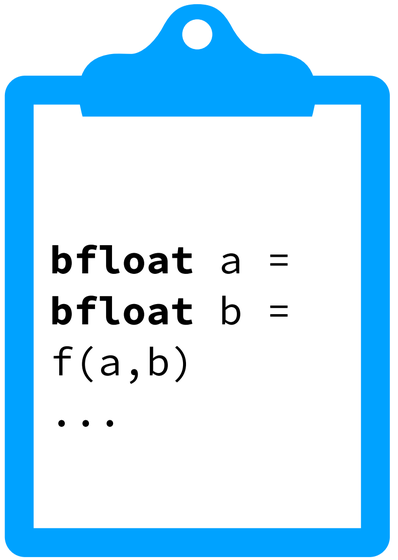




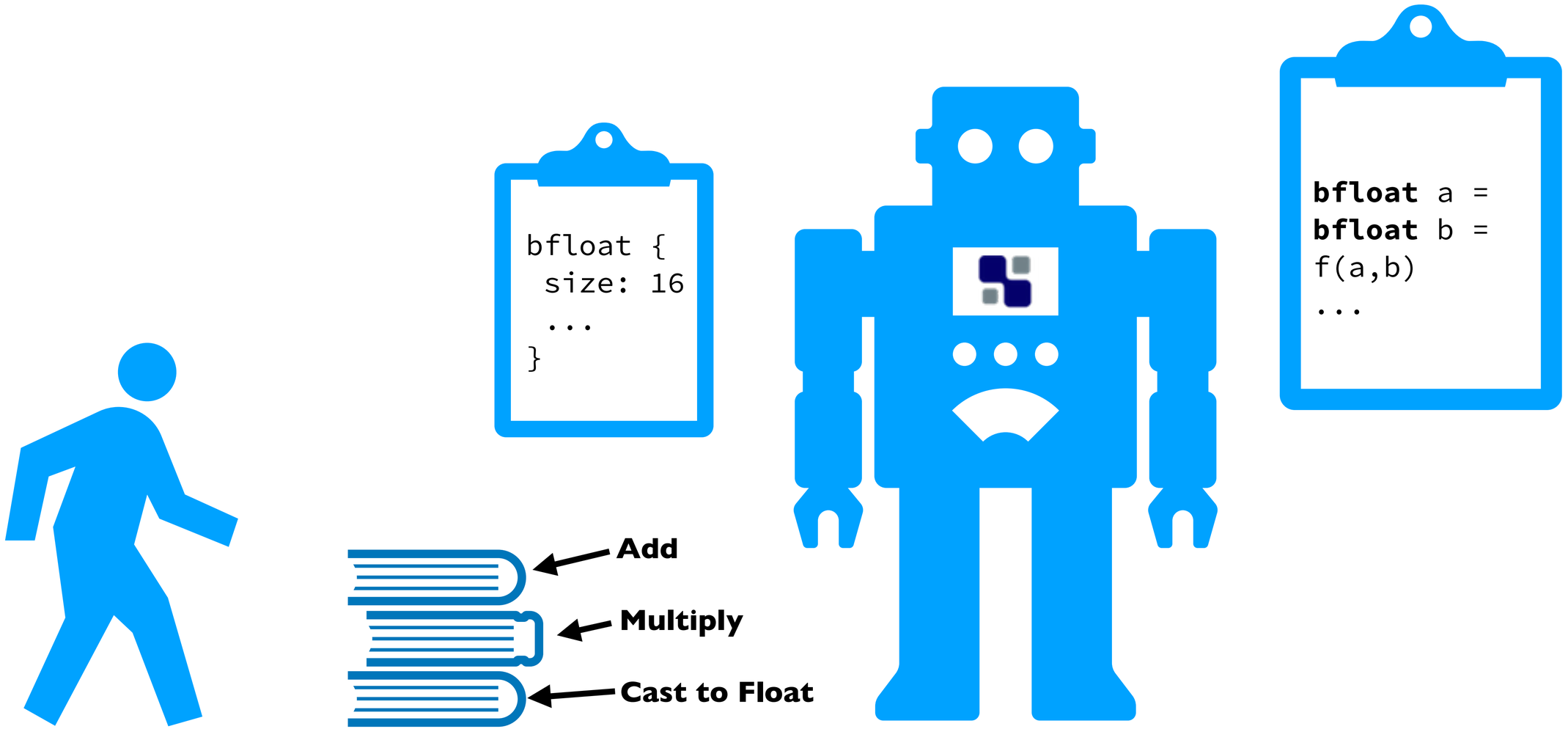
How to Use New Datatypes (with TVM!)



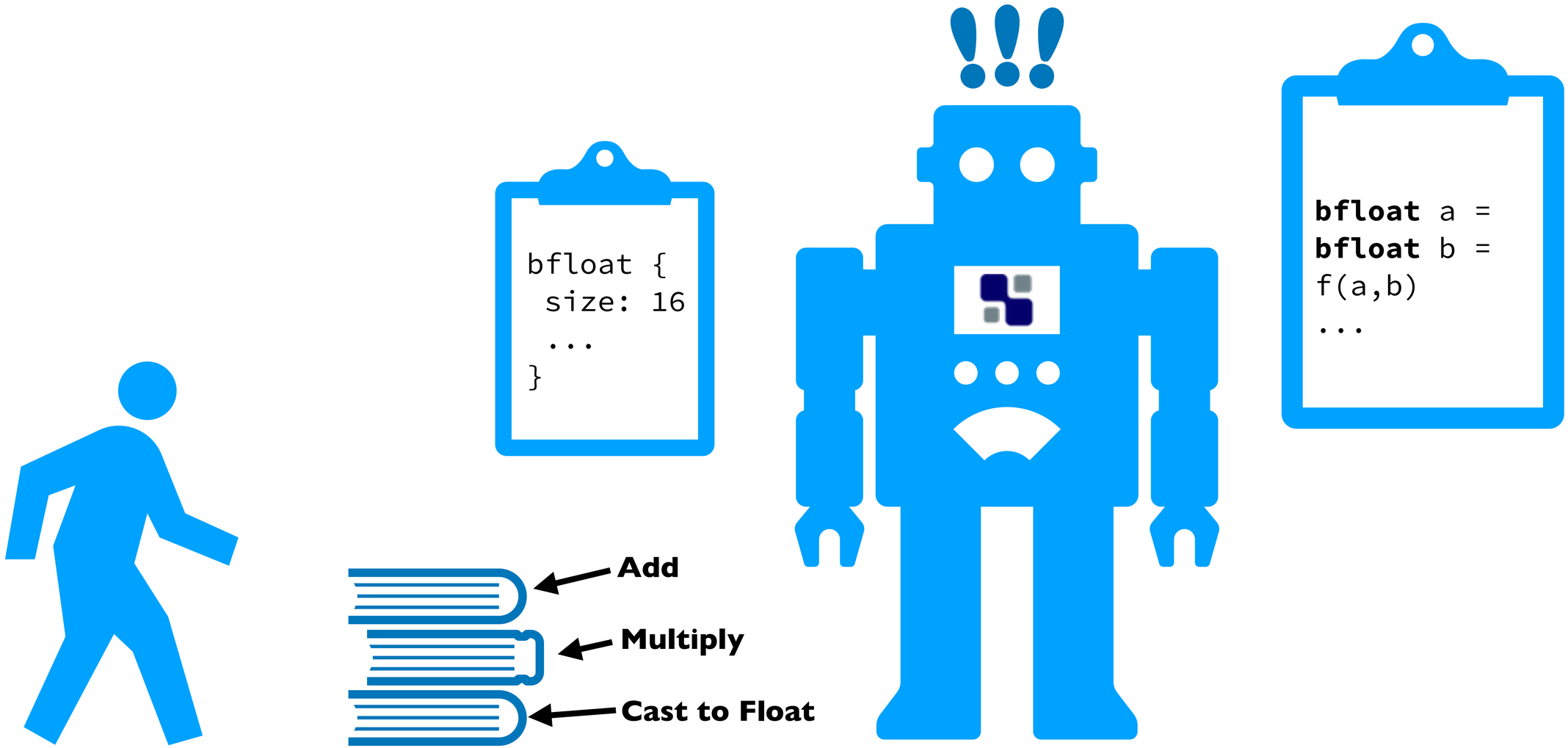




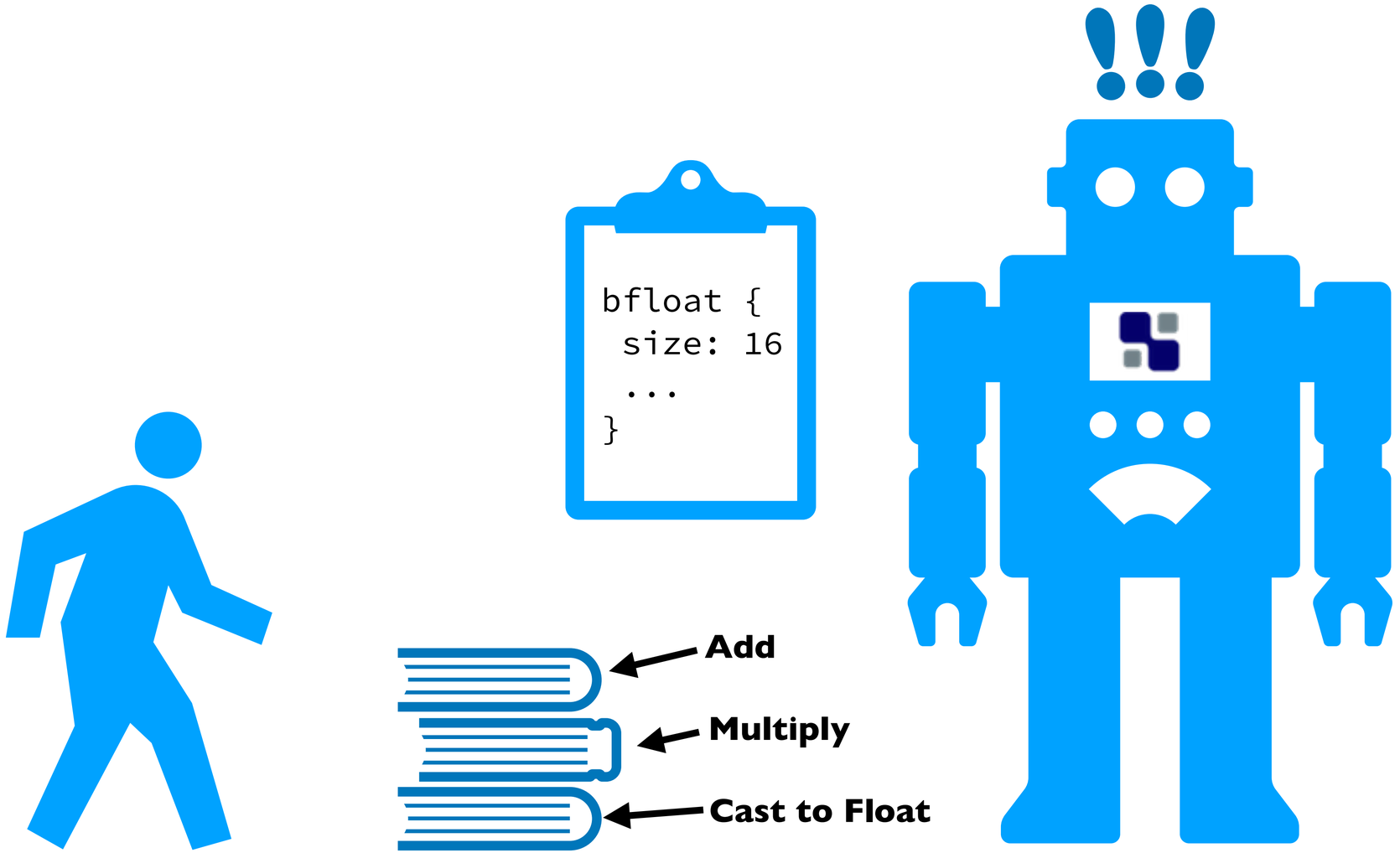
How to Use New Datatypes (with TVM!)



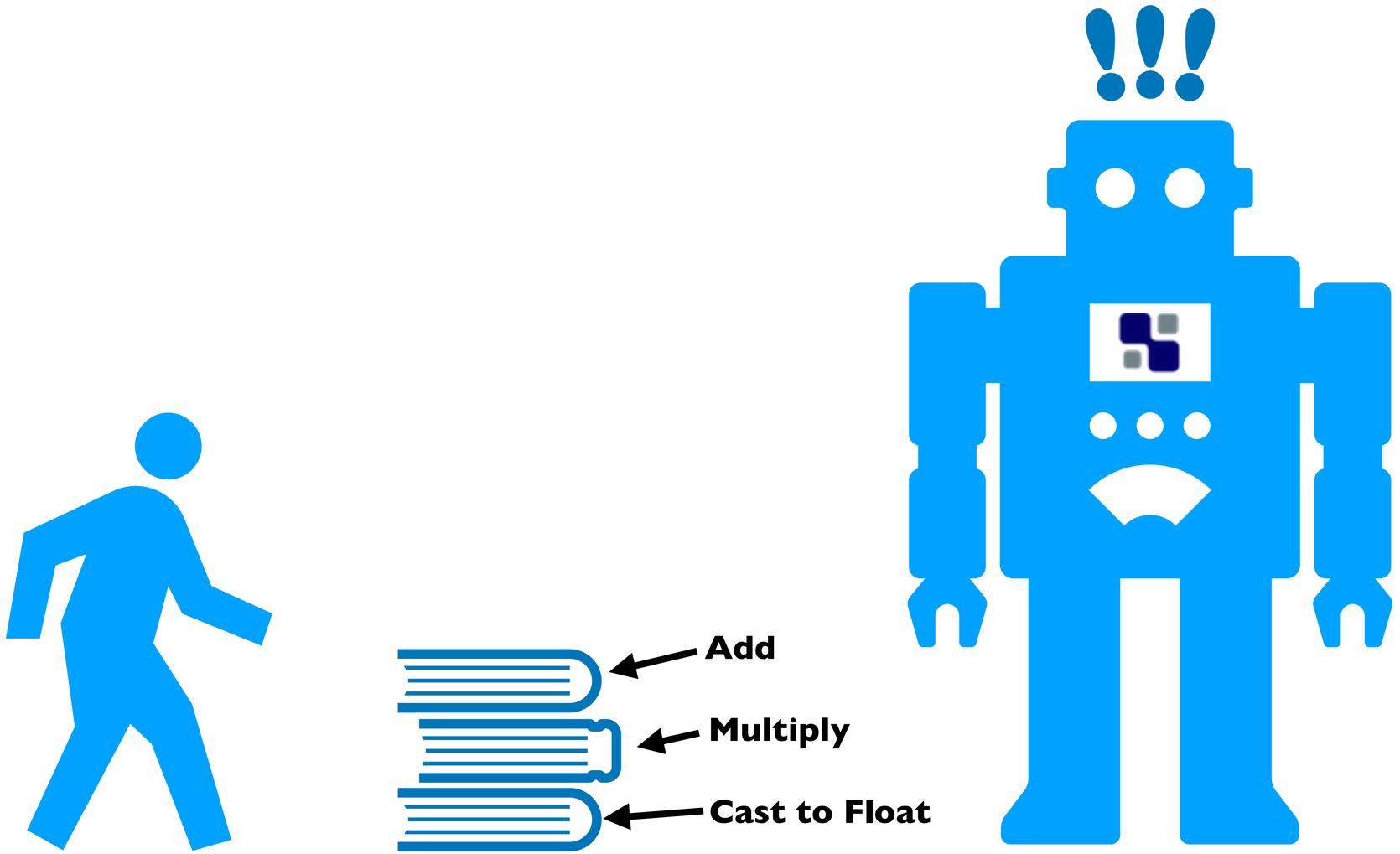
How to Use New Datatypes (with TVM!)



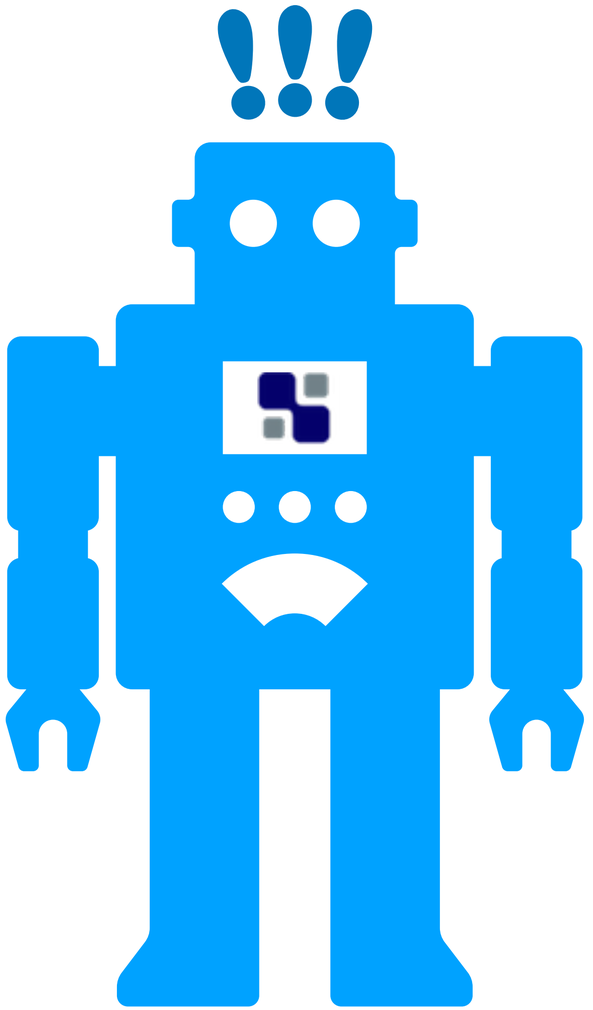
How to Use New Datatypes (with TVM!)



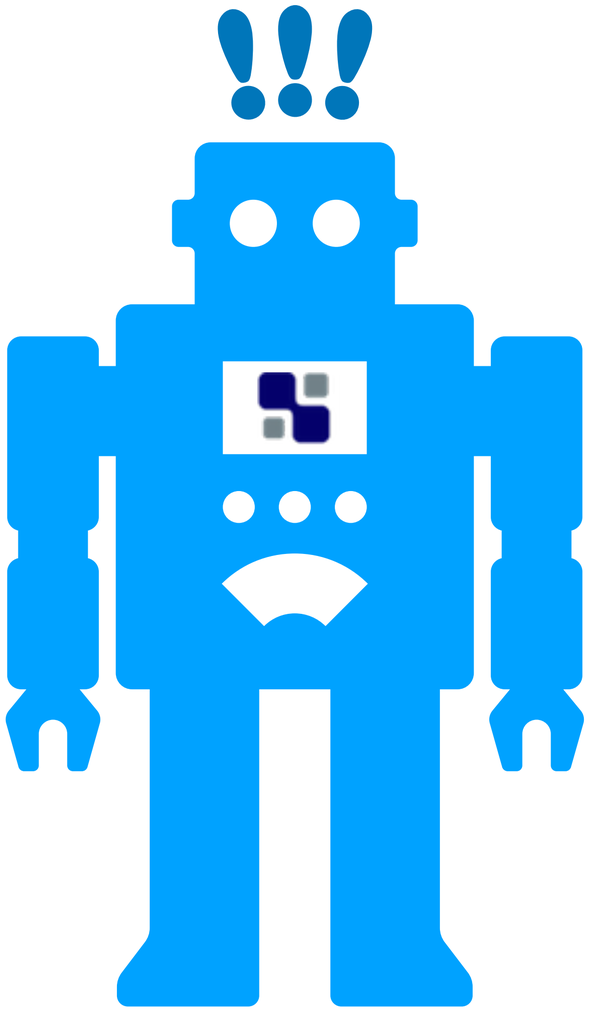
How to Use New Datatypes (with TVM!)



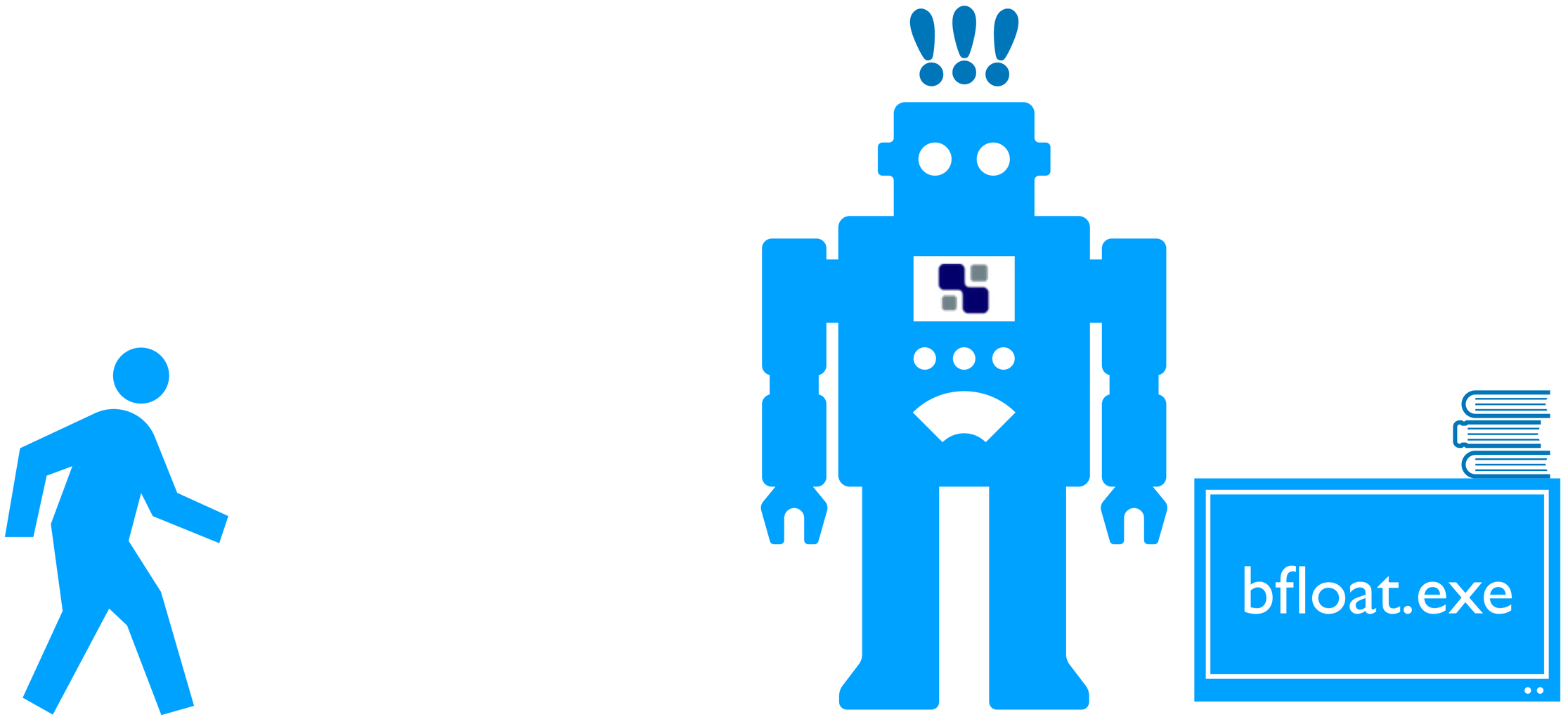
How to Use New Datatypes (with TVM!)



How to Use New Datatypes (with TVM!)



How to Use New Datatypes (with TVM!)



How to Use New Datatypes (with TVM!)

How to Use New Datatypes (with TVM!)

1. User makes or finds a library which implements their datatype in software

How to Use New Datatypes (with TVM!)

1. User makes or finds a library which implements their datatype in software
2. User writes a program which uses their custom datatype

How to Use New Datatypes (with TVM!)

1. User makes or finds a library which implements their datatype in software
2. User writes a program which uses their custom datatype
3. User points TVM to the important functions (+, $*$, cast-to-float) in the library

How to Use New Datatypes (with TVM!)

1. User makes or finds a library which implements their datatype in software
2. User writes a program which uses their custom datatype
3. User points TVM to the important functions (+, $*$, cast-to-float) in the library
4. User gives TVM other information e.g. datatype size

How to Use New Datatypes (with TVM!)

1. User makes or finds a library which implements their datatype in software
2. User writes a program which uses their custom datatype
3. User points TVM to the important functions (+, $*$, cast-to-float) in the library
4. User gives TVM other information e.g. datatype size
5. TVM compiles the program, handling the custom datatype by compiling to calls into the provided library

Limitations of this Approach

Limitations of this Approach

Currently, we're only supporting **software implementations** of datatypes.

Limitations of this Approach

Currently, we're only supporting **software implementations** of datatypes.

Compiling for **hardware implementations** is a work in progress!

Future Work

Future Work

- **Short term:** evaluating real deep learning models with software emulations of modern datatypes

Future Work

- **Short term:** evaluating real deep learning models with software emulations of modern datatypes

  Have a model or datatype you're interested in specifically? I can help!

Future Work

- **Short term:** evaluating real deep learning models with software emulations of modern datatypes

  Have a model or datatype you're interested in specifically? I can help!

- **Also short term:** improving performance in software (datatype emulation libraries are slow, and models run many operations!)

Future Work

- **Short term:** evaluating real deep learning models with software emulations of modern datatypes

  Have a model or datatype you're interested in specifically? I can help!

- **Also short term:** improving performance in software (datatype emulation libraries are slow, and models run many operations!)

- **Long term:** supporting custom datatype hardware implementations in VTA

Let's move to the notebook! (link)