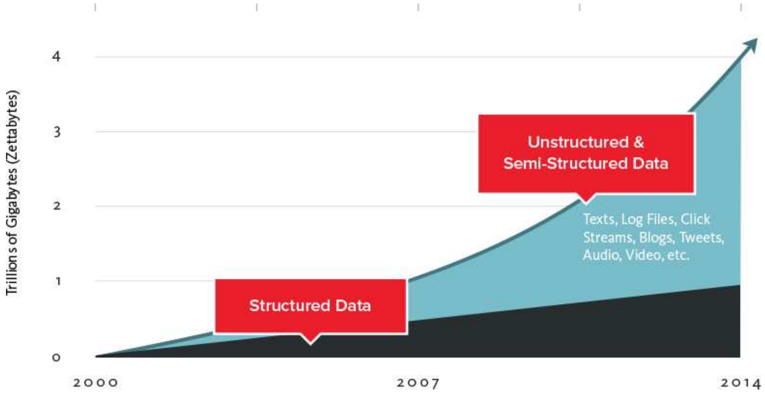
Building distributed machine learning pipeline

Vojtěch Juránek

JBoss - a division by Red Hat

27. 1. 2017, DEVCONF.CZ, Brno

Data today



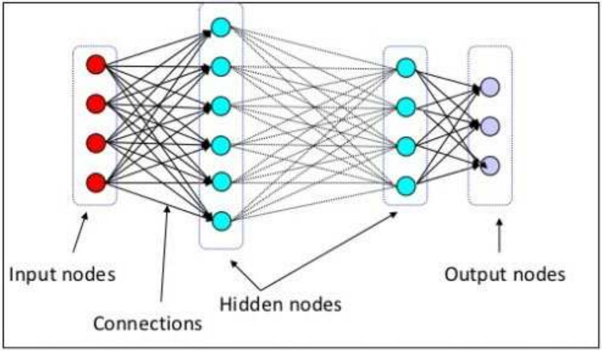
Source: `http://www.couchbase.com/nosql-resources/what-is-no-sql`

Dealing with unstructured data

- There are classes of problems where simple (e.g. linear) models work fine, but there are also classes there liner models fail.

Deep learning

Deep learning



Source: `http://www.kdnuggets.com/2016/10/deep-learning-key-terms-explained.html`

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

- Process data once it arrives (data/event streaming).
- Do the data analysis with frameworks which keep the data in memory during processing.
- If possible, keep data in memory during whole application stack.
- More in my DevConf 2016 talk (slides).

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

- Process data once it arrives (data/event streaming).
- Do the data analysis with frameworks which keep the data in memory during processing.
- If possible, keep data in memory during whole application stack.
- More in my DevConf 2016 talk (slides).

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

- Process data once it arrives (data/event streaming).
- Do the data analysis with frameworks which keep the data in memory during processing.
- If possible, keep data in memory during whole application stack.
- More in my DevConf 2016 talk (slides).

Big Data → Fast Data

- Pressure to react/process data immediately as it arrives, provide user immediate feedback.
- Big Data → Fast Data

- Process data once it arrives (data/event streaming).
- Do the data analysis with frameworks which keep the data in memory during processing.
- If possible, keep data in memory during whole application stack.
- More in my DevConf 2016 talk (slides).

What we have so far?

What we have so far?

Problem

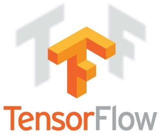
What we have so far?

Problem

- Data which requires (complicated) non-liner model to sort out.
- We'd like to process and pass incoming data through whole application stack immediately.

Building neural network pipeline in era of fast data

Building neural network pipeline in era of fast data



Building neural network pipeline in era of fast data

Building neural network pipeline in era of fast data

TensorFlow
+
Infinispan
+
ceph

Deep NN/learning frameworks

- There are many . . .
- . . . Caffe, CNTK, Deeplearning4j, TensorFlow, . . .
- Brief comparison on Wikipedia:
  https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software

Deep NN/learning frameworks

- There are many . . .
- . . . Caffe, CNTK, Deeplearning4j, TensorFlow, . . .
- Brief comparison on Wikipedia: `https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software`

Deep NN/learning frameworks

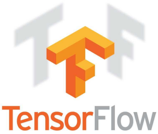
- There are many . . .
- . . . Caffe, CNTK, Deeplearning4j, TensorFlow, . . .
- Brief comparison on Wikipedia:
  https://en.wikipedia.org/wiki/Comparison_of_deep_learning_software

TensorFlow

- Open source software library for machine learning developed by Google Brain team, white paper: arXiv:1603.04467
- Google's second generation machine learning system (first one was DistBelief), open sourced more then year ago (Nov. 2015).
- Used by Google speech recognition, Google photos and other Google products.
- Used also by many other projects, e.g. Mozilla DeepSpeech project (there's a talk by Tilman Kamp about this project here at DevConf on Sunday)

TensorFlow

- Open source software library for machine learning developed by Google Brain team, white paper: arXiv:1603.04467
- Google's second generation machine learning system (first one was DistBelief), open sourced more then year ago (Nov. 2015).
- Used by Google speech recognition, Google photos and other Google products.
- Used also by many other projects, e.g. Mozilla DeepSpeech project (there's a talk by Tilman Kamp about this project here at DevConf on Sunday)



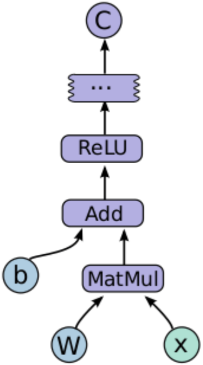
- Open source software library for machine learning developed by Google Brain team, white paper: arXiv:1603.04467
- Google's second generation machine learning system (first one was DistBelief), open sourced more then year ago (Nov. 2015).
- Used by Google speech recognition, Google photos and other Google products.
- Used also by many other projects, e.g. Mozilla DeepSpeech project (there's a talk by Tilman Kamp about this project here at DevConf on Sunday)

TensorFlow

- Open source software library for machine learning developed by Google Brain team, white paper: arXiv:1603.04467
- Google's second generation machine learning system (first one was DistBelief), open sourced more then year ago (Nov. 2015).
- Used by Google speech recognition, Google photos and other Google products.
- Used also by many other projects, e.g. Mozilla DeepSpeech project (there's a talk by Tilman Kamp about this project here at DevConf on Sunday)

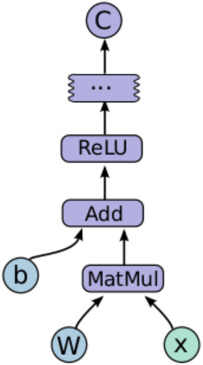
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



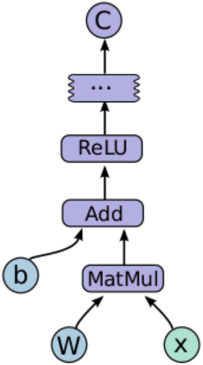
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



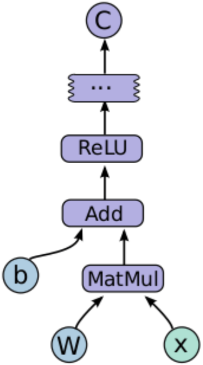
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



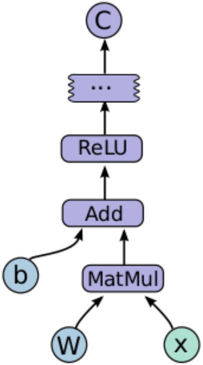
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



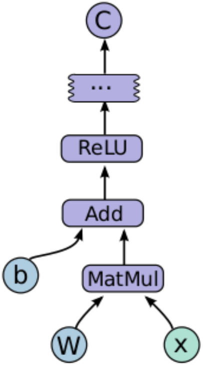
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



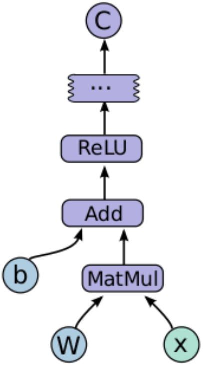
TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.



TensorFlow computation graphs

- **Graph** represents TF computation.
- **Graph nodes** act as mathematical operations.
- **Graph edges** represent matrices (tensors).
- **Session** represents a client accessing particular TF runtime.
- **Variable** is a variable with pre-defined value (pre-defined parameter).
- **Placeholder** is a variable placeholder and its value will be injected into the session.
- **Checkpoint** use for storing variable/trained model/etc.

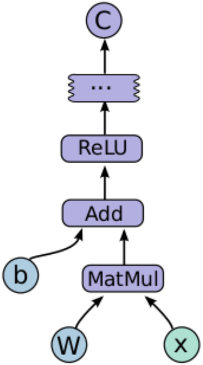


TensorFlow computation graphs

```
import tensorflow as tf

b = tf.Variable(tf.zeros([100]))
W = tf.Variable(tf.random_uniform \
  ([784,100],-1,1))
x = tf.placeholder(name="x")
relu = tf.nn.relu(tf.matmul(W, x) + b)
C = [...]

s = tf.Session()
for step in xrange(0, 10):
  x_in = [...] #construct 100-D input array
  result = s.run(C, feed_dict={x: x_in})
  print step, result
```



Brief list of some others TF features

- Storing graphs and checkpoints into language neutral files (`protobuf` format).
- Supports computation on CPU as well as on GPU, optionally supports CUDA.

```
with tf.Session() as sess:
  with tf.device("/gpu:1"):
```

- Can be run across the cluster, uses `grpc` for communication

```
with tf.device("/job:ps/task:0"):
  weights_1 = tf.Variable(...)
  biases_1 = tf.Variable(...)
[...]
with tf.device("/job:worker/task:7"):
  layer_1 = tf.nn.relu(tf.matmul(input, weights_1) +
      biases_1)
  logits = tf.nn.relu(tf.matmul(layer_1, weights_2) +
      biases_2)
[...]
with tf.Session("grpc://worker7.example.com:2222") as sess:
  sess.run(train_op)
```

Brief list of some others TF features

- Storing graphs and checkpoints into language neutral files (`protobuf` format).
- Supports computation on CPU as well as on GPU, optionally supports CUDA.

```
with tf.Session() as sess:
  with tf.device("/gpu:1"):
```

- Can be run across the cluster, uses `grpc` for communication

```
with tf.device("/job:ps/task:0"):
  weights_1 = tf.Variable(...)
  biases_1 = tf.Variable(...)
[...]
with tf.device("/job:worker/task:7"):
  layer_1 = tf.nn.relu(tf.matmul(input, weights_1) + biases_1)
  logits = tf.nn.relu(tf.matmul(layer_1, weights_2) + biases_2)
[...]
with tf.Session("grpc://worker7.example.com:2222") as sess:
  sess.run(train_op)
```

Brief list of some others TF features

- Storing graphs and checkpoints into language neutral files (`protobuf` format).
- Supports computation on CPU as well as on GPU, optionally supports CUDA.

```
with tf.Session() as sess:
  with tf.device("/gpu:1"):
```

- Can be run across the cluster, uses `grpc` for communication

```
with tf.device("/job:ps/task:0"):
  weights_1 = tf.Variable(...)
  biases_1 = tf.Variable(...)
[...]
with tf.device("/job:worker/task:7"):
  layer_1 = tf.nn.relu(tf.matmul(input, weights_1) +
      biases_1)
  logits = tf.nn.relu(tf.matmul(layer_1, weights_2) +
      biases_2)
[...]
with tf.Session("grpc://worker7.example.com:2222") as sess:
  sess.run(train_op)
```

TensorFlow integration with Infinispan/Java

- Typically you create the model in Python and run it in C++ or Java app.
- TensorFlow Java API support still TBD, see TF issue #5and issue #3
- C++ API via JNI can be used as a workaround for Java.
- You can use some existing library, e.g. javacpp-presets.

```
GraphDef graph = new GraphDef();
ReadBinaryProto(Env.Default(), modelPath, graph);
SessionOptions options = new SessionOptions();
this.session = new Session(options);
Status status = session.Create(graph);
[...]
Tensor img = new Tensor(DT_FLOAT, new TensorShape(1, image.
    length));
FloatBuffer imgBuff = img.createBuffer();
imgBuff.put(image);
[...]
TensorVector outputs = new TensorVector();
Status status = session.Run(new StringTensorPairVector(new
    String[] { "images" }, new Tensor[] { img }),
new StringVector("softmax_linear/logits"), new StringVector("
    softmax_linear/logits"), outputs);
```

In-memory data grid: Infinispan

http://infinispan.org/

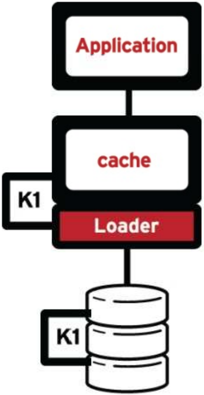
- Data grid platform, written in Java
- In-memory No-SQL key-value data store, (optionally) schema-less
- Distributed cache - offers massive memory
- Elastic and scalable - can run on hundreds of nodes
- Highly available - no SPOF, resilient to node failures
- Transactional
- Supports indexing and searching
- Many other features

Infinispan eviction and cache stores

Eviction: removing entries from the cache.

```
ConfigurationBuilder().eviction().size(5).
    strategy(EvictionStrategy.LRU)
```

Cache store: a way how to store cache content in some external (persistent) storage.
There are many, JDBC, JPA, clouds, LevelDB, Cassandra . . . and also **Ceph** cache store.



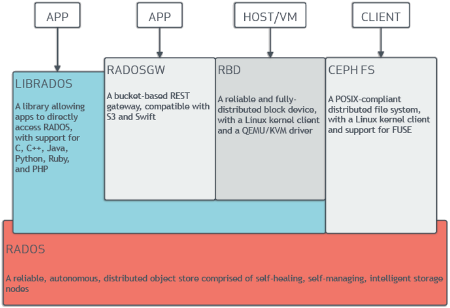
Ceph

http://ceph.com/

- Distributed object storage.
- Provides interfaces to object, block and file system storage in unified data cluster.
- Scalable to the exabyte level.
- Highly available - no SPOF, resilient to node failures.
- Open source.

Ceph architecture and Infinispan cache store

- `rados` is distributed object store
- Infinispan cache store accesses `rados` directly via `librados` (using Java JNI client).



APP

APP

HOST/VM

CLIENT

LIBRADOS

A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP

RADOSGW

A bucket-based REST gateway, compatible with S3 and Swift

RBD

A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver

CEPH FS

A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE

RADOS

A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes

Ceph cache store configuration:
(details on https://github.com/vjuranek/infinispan-cachestore-ceph).

```
<local-cache name="cachestore" start="EAGER">
    <store class="org.infinispan.persistence.ceph.CephStore">
        <property name="monitorHost">192.168.122.145:6789</property>
        <property name="userName">admin</property>
        <property name="key">mykey</property>
    </store>
</local-cache>
```

”Hello world” Demo

Sources available on `https://github.com/vjuranek/tf-ispn-demo`

NN ”Hello world” - MNIST data sample

- Recognition of hand written digits.
- Besides simplicity, TF has very detailed tutorial for MNIST (for beginners as well as for experts)

”Hello world” Demo

Sources available on **https://github.com/vjuranek/tf-ispn-demo**

NN ”Hello world” - MNIST data sample

- Recognition of hand written digits.
- Besides simplicity, TF has very detailed tutorial for MNIST (for beginners as well as for experts)

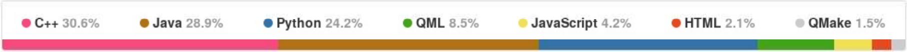
”Hello world” Demo

Sources available on **`https://github.com/vjuranek/tf-ispn-demo`**

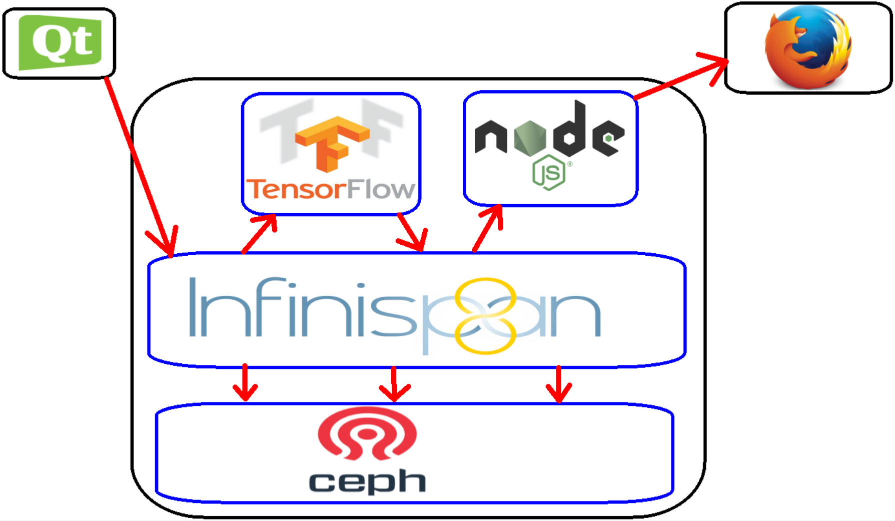
NN ”Hello world” - MNIST data sample

- Recognition of hand written digits.
- Besides simplicity, TF has very detailed tutorial for MNIST (for beginners as well as for experts)

”Hello world” Demo



● C++ 30.6% ● Java 28.9% ● Python 24.2% ● QML 8.5% ● JavaScript 4.2% ● HTML 2.1% ● QMake 1.5%



Summary

- Building pipeline for complex data processing in real time can be quite easy if you use the right tools.
- TensorFlow is very powerful machine learning framework.
- Infinispan is real middleware which can glue together various pieces of your application stack and server as its backbone.

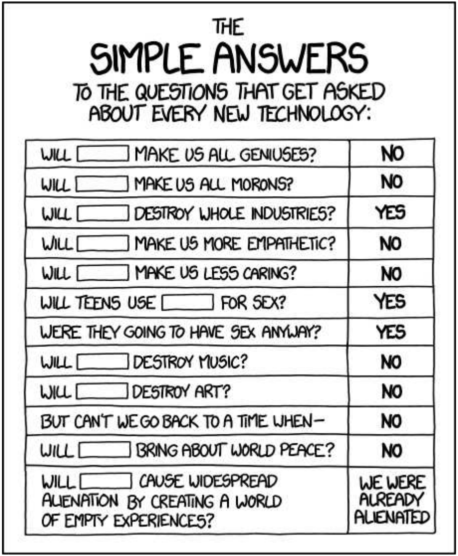
Summary

- Building pipeline for complex data processing in real time can be quite easy if you use the right tools.
- TensorFlow is very powerful machine learning framework.
- Infinispan is real middleware which can glue together various pieces of your application stack and server as its backbone.

Summary

- Building pipeline for complex data processing in real time can be quite easy if you use the right tools.
- TensorFlow is very powerful machine learning framework.
- Infinispan is real middleware which can glue together various pieces of your application stack and server as its backbone.

Question?



THE
SIMPLE ANSWERS
TO THE QUESTIONS THAT GET ASKED ABOUT EVERY NEW TECHNOLOGY:

WILL □ MAKE US ALL GENIUSES?	NO
WILL □ MAKE US ALL MORONS?	NO
WILL □ DESTROY WHOLE INDUSTRIES?	YES
WILL □ MAKE US MORE EMPATHETIC?	NO
WILL □ MAKE US LESS CARING?	NO
WILL TEENS USE □ FOR SEX?	YES
WERE THEY GOING TO HAVE SEX ANYWAY?	YES
WILL □ DESTROY MUSIC?	NO
WILL □ DESTROY ART?	NO
BUT CAN'T WE GO BACK TO A TIME WHEN—	NO
WILL □ BRING ABOUT WORLD PEACE?	NO
WILL □ CAUSE WIDESPREAD ALIENATION BY CREATING A WORLD OF EMPTY EXPERIENCES?	WE WERE ALREADY ALIENATED

Source: `https://xkcd.com/1289/`

Infinispan

http://infinispan.org/

Thank you for your attention!