WE WILL START AT 8:02 PM

ML SYSTEMS DESIGN MEETUP GROUP

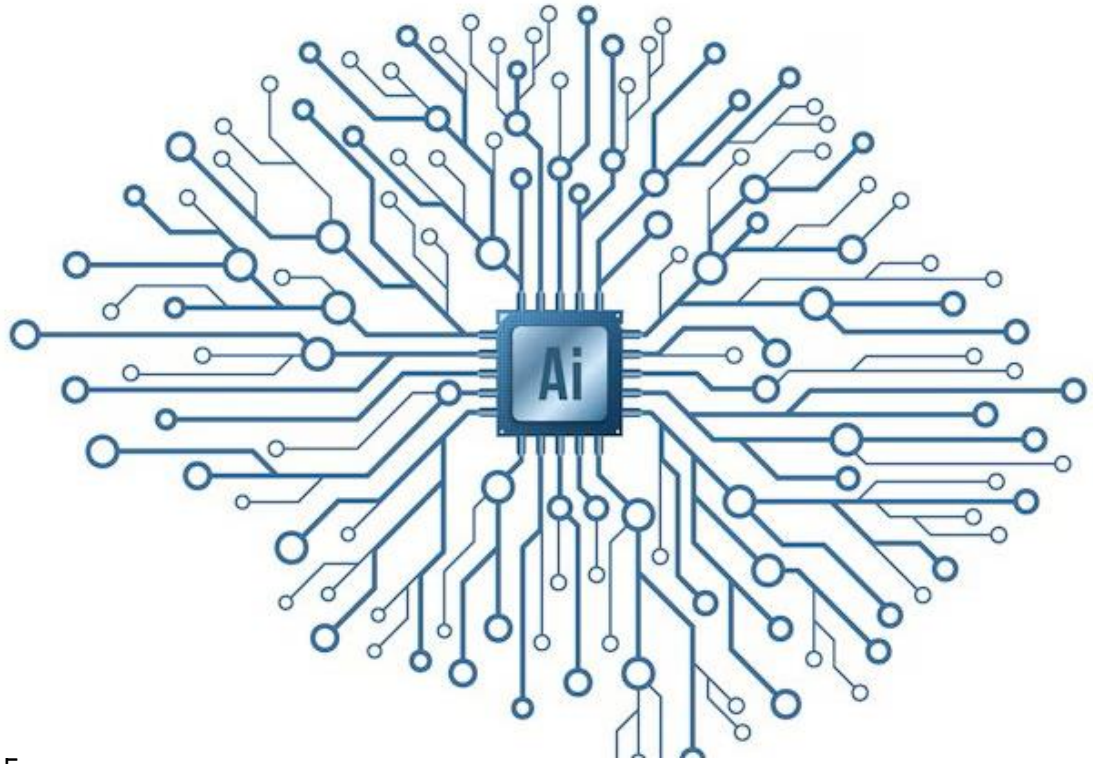
HETAV PANDYA

THIS WORKSHOP WILL BE RECORDED

HETAV PANDYA

AGENDA

- ✓ INTRODUCTION
- ✓ SERIALIZATION
- ✓ ML DEPLOYMENT MYTHS
- ✓ BATCH PREDICTION
- ✓ ONLINE PREDICTION
- ✓ UNIFYING BATCH PIPELINE AND STREAMING PIPELINE
- ✓ MODEL COMPRESSION
- ✓ COMPILING ON EDGE DEVICES
- ✓ ML IN BROWSERS
- ✓ Q&A



INTRODUCTION

- Welcome to ML Systems Design Meetup Group
- Designing Machine Learning Systems – Chip Huyen
- Two chapters every meeting
- Free Access – City Library
- Frequency – Biweekly - Monthly
- Questions: pandyahetav1@gmail.com

FREE ACCESS

Via Burnaby Public Library

SERIALIZATION

The process of converting an ML model into a format that can be used by another application

Model Definition

Model Parameters

SERIALIZATION - PYTORCH

PyTorch does not provide a built-in way to save just the model architecture (i.e., the structure without the weights). You usually save the entire model (weights + architecture)

```
import torch

# Define and train your model
model = MyModel()
# Train your model
# ...

# Save only the model's weights
torch.save(model.state_dict(), 'model_weights.pth')
```

SERIALIZATION - TENSORFLOW

TensorFlow allows you to store weights separately! However, like PyTorch you can save the entire model together too.

```python
import tensorflow as tf

# Define and compile your model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(10, activation='relu'),
    tf.keras.layers.Dense(1)
])
model.compile(optimizer='adam', loss='mse')

# Train your model
# ...

# Save only the model's weights
model.save_weights('model_weights.h5')
```

```python
import tensorflow as tf

# Define your model (assuming you have a `tf.keras.Model` instance)
model = tf.keras.Sequential([
    tf.keras.layers.Dense(10, activation='relu'),
    tf.keras.layers.Dense(1)
])

# Compile and train the model
# ...

# Save the model
model.save('model.h5')
```

ML DEPLOYMENT MYTHS

You only deploy one or two ML models at a time...

ML DEPLOYMENT MYTHS

If we don't do anything, model performance remains the same...

ML DEPLOYMENT MYTHS

You won't need to update your models as much...

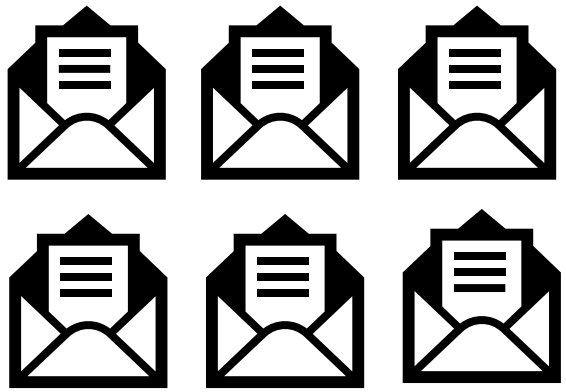
ML DEPLOYMENT MYTHS

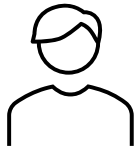
Most ML engineers don't need to worry about scale...

BATCH PREDICTION

Batch prediction is when predictions are generated periodically or whenever triggered. The predictions are stored somewhere, such as in SQL tables or an in-memory database, and retrieved as needed.

Batch prediction is also known as *asynchronous prediction*





ONLINE PREDICTION

Online prediction is when predictions are generated and returned as soon as requests for these predictions arrive.

For example, you enter an English sentence into Google Translate and get back its French translation immediately.

BATCH VS ONLINE

LET'S DIFFERENTIATE...

Where can BATCH prediction go wrong?

Where can online prediction go wrong?

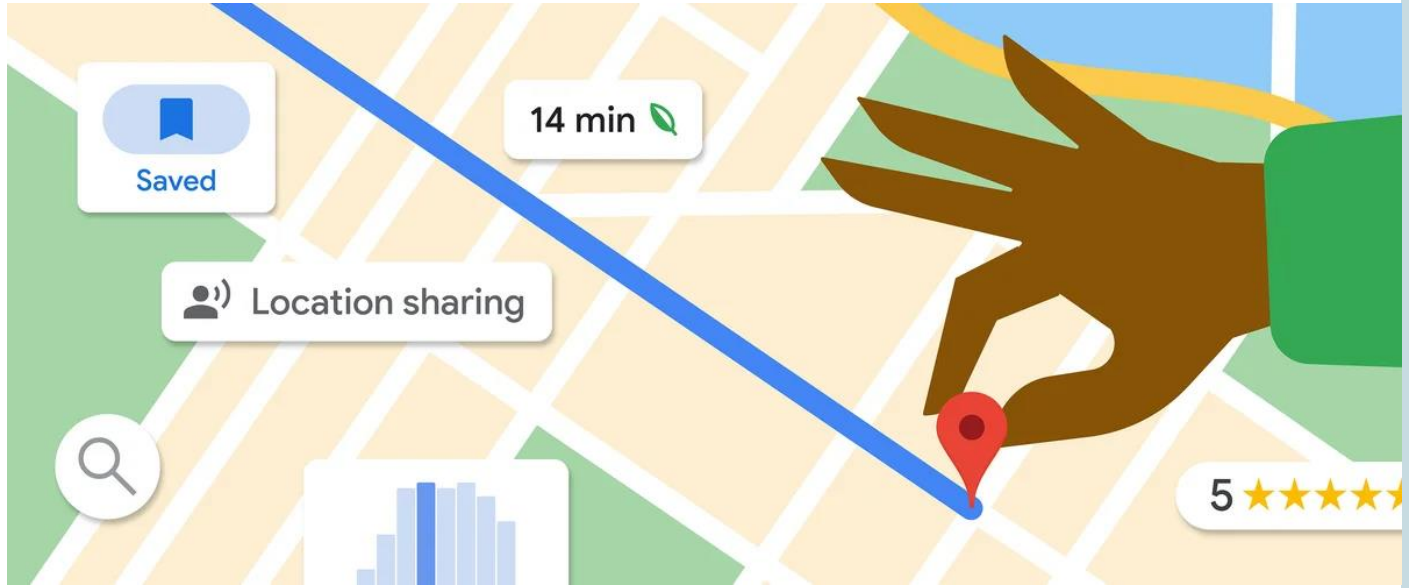
	Batch prediction (asynchronous)	**Online prediction (synchronous)**
Frequency	Periodical, such as every four hours	As soon as requests come
Useful for	Processing accumulated data when you don't need immediate results (such as recommender systems)	When predictions are needed as soon as a data sample is generated (such as fraud detection)
Optimized for	High throughput	Low latency

GOOGLE MAPS EXAMPLE
YOUR BRANCH HAS DIVERGED!

A feature you might want to use is the average speed of all the cars in your path in the last five minutes.

AND

You want to exploit the past training data collected from other users.



GOOGLE MAPS EXAMPLE
YOUR BRANCH HAS DIVERGED!

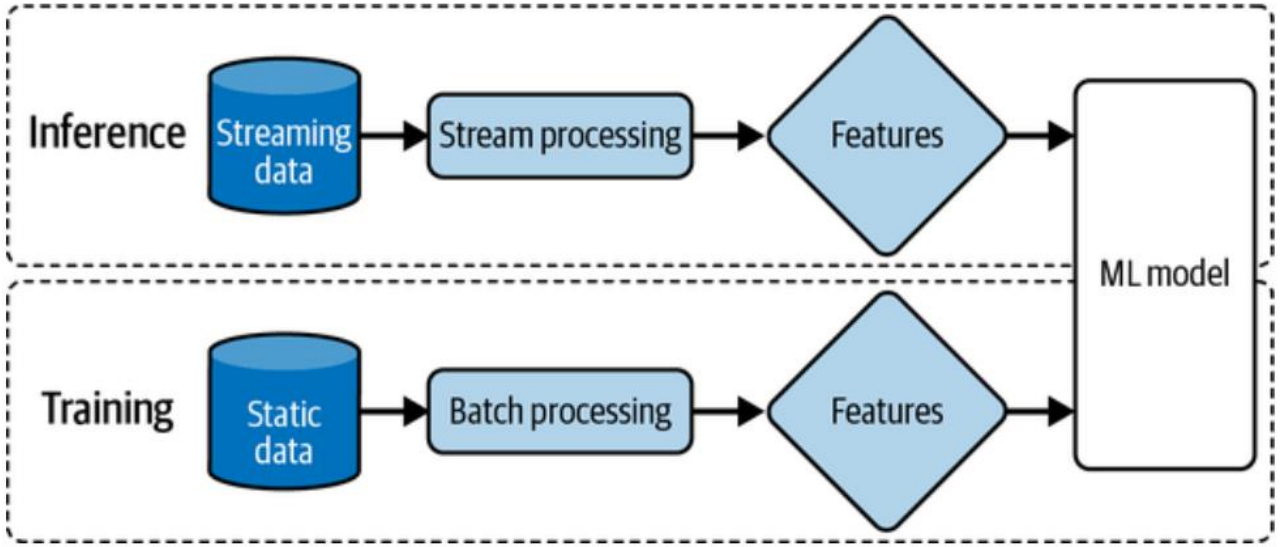


Figure 7-7. Having two different pipelines for training and inference is a common source for bugs for ML in production

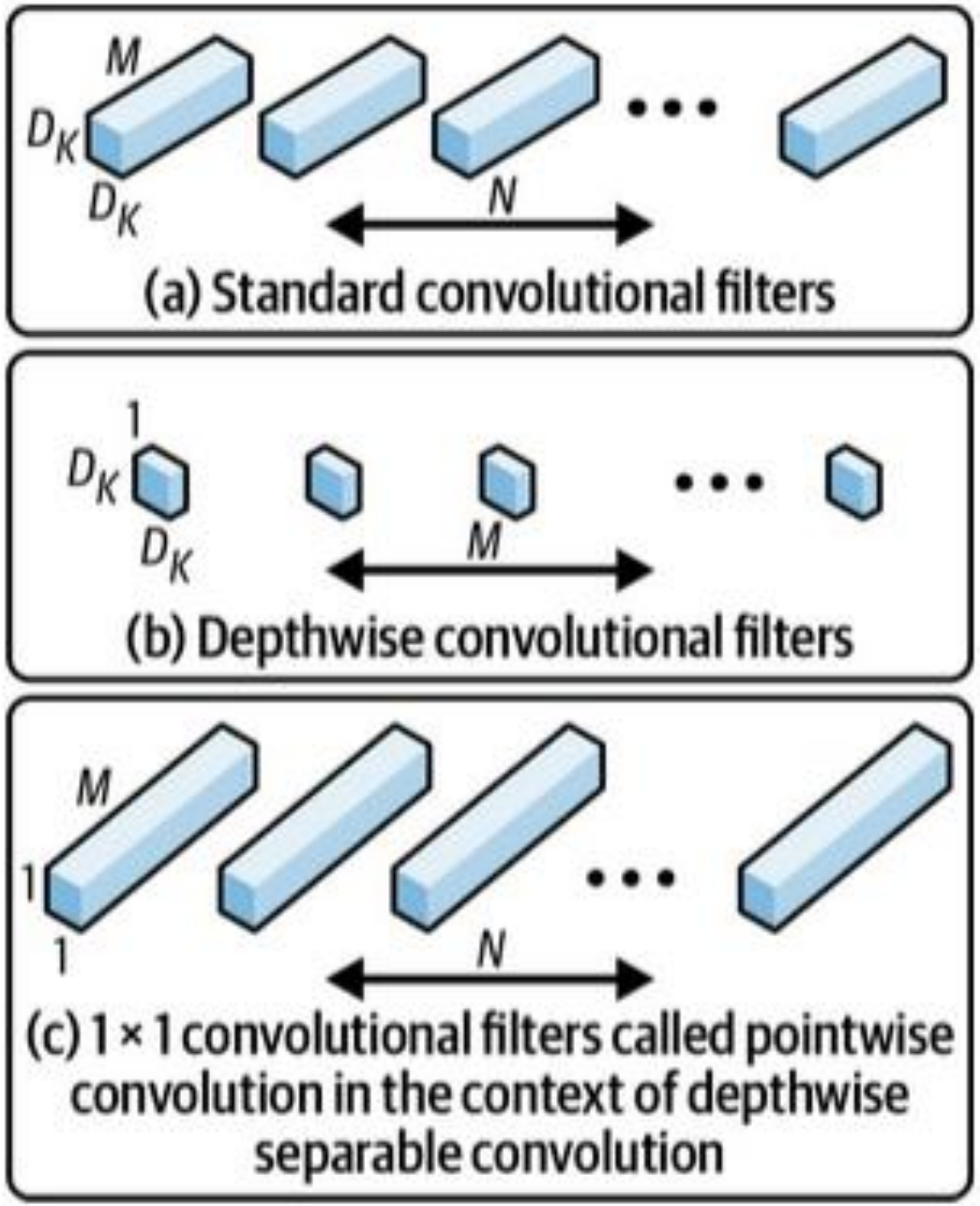
MODEL COMPRESSION

The process of making a model smaller is called model compression, and the process to make it do inference faster is called inference optimization.

https://awesomeopensource.com/projects/model-compression

LOW-RANK FACTORIZATION

The key idea behind low-rank factorization is to replace high-dimensional tensors with lower-dimensional tensors



DEMO: LOW-RANK FACTORIZATION

KNOWLEDGE DISTILLATION

Knowledge distillation is a method in which a small model (student) is trained to mimic a larger model or ensemble of models (teacher). The smaller model is what you'll deploy.

The advantage of this approach is that it can work regardless of the architectural differences between the teacher and the student networks. For example, you can get a random forest as the student and a transformer as the teacher.

Who is the teacher? Large scale open-source models fine-tuned on **your** data!

PRUNING

Pruning was a method originally used for decision trees where you remove sections of a tree that are uncritical and redundant for classification.

Pruning, in the context of neural networks, has two meanings:

- to remove entire nodes of a neural network: Rare
- to find parameters least useful to predictions and set them to 0: Common

QUANTIZATION

Quantization reduces a model's size by using fewer bits to represent its parameters.

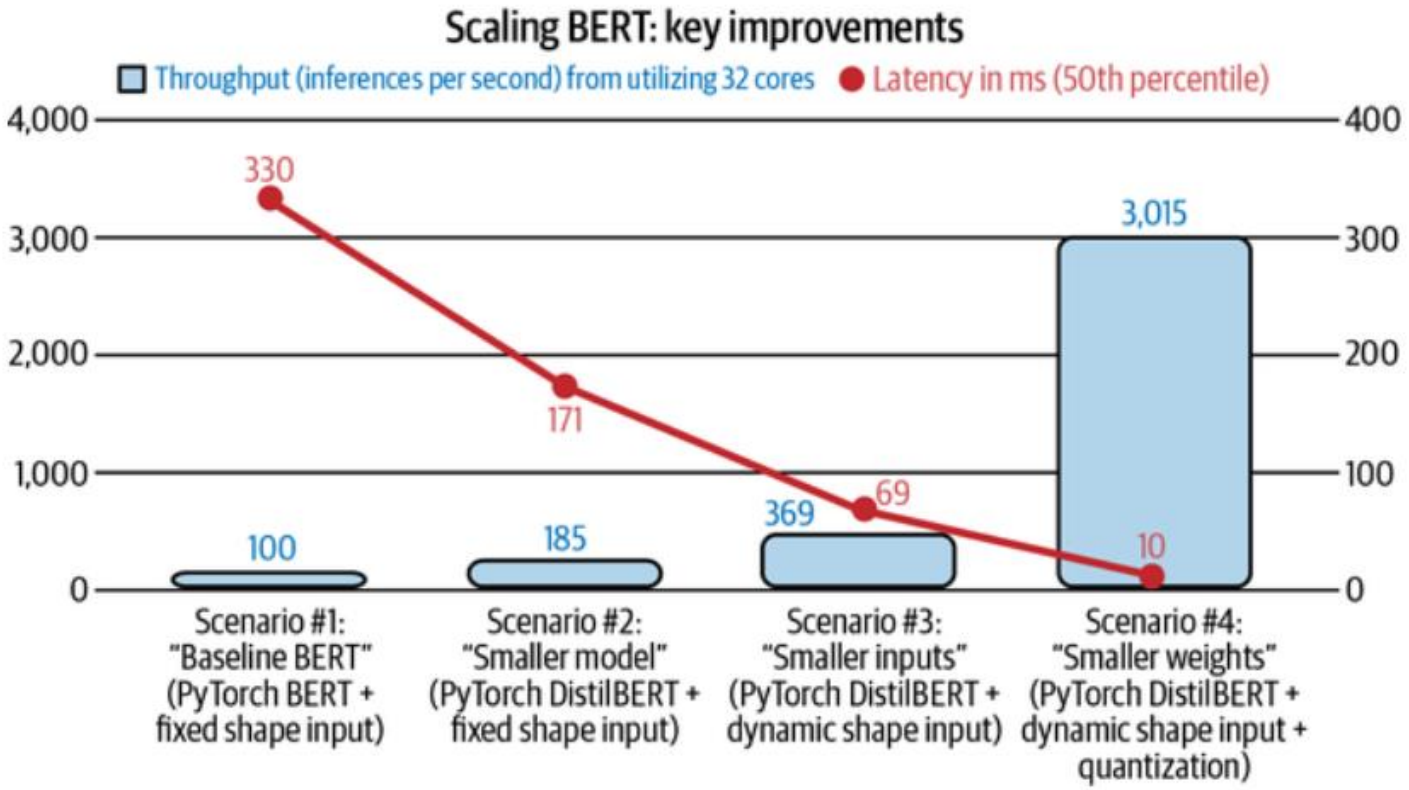
By default, most software packages use 32 bits to represent a float number.

If a model has 100M parameters and each requires 32 bits to store, it'll take up 400 MB. If we use 16 bits to represent a number, we'll reduce the memory footprint by half.

Pros	Cons
Less memory footprint	Less precision
Faster training	Less learning capacity
Faster Inference	Overflow/underflow risks

QUANTIZATION – TFLITE DEMO

PRACTICAL IMPACTS OF COMPRESSION



CLOUD VS EDGE

Aspect	Cloud Deployment	Edge Deployment
Deployment Ease	Easy with managed cloud services like AWS or GCP.	More complex; requires handling diverse hardware and software environments.
Cost	Can be expensive due to compute-intensive operations; large annual bills possible.	Potentially lower cost by reducing cloud usage; however, requires investment in edge hardware.
Internet Dependency	Requires stable internet connections for data transfer.	Operates without internet or with unreliable connections.
Network Latency	Potentially high due to data transfer delays between cloud and users.	Low latency as computation is done locally on the device.
Data Privacy	Higher risk of data breaches; sensitive data transferred over networks.	Better privacy; data remains on local devices, reducing risk of interception.
Device Requirements	No specific hardware requirements; cloud infrastructure handles computation.	Edge devices need to be powerful enough for computations, with sufficient memory and energy supply.
Battery Consumption	Not applicable; computation is server-side.	Critical factor; computation can drain device batteries quickly.
Scalability	Highly scalable with cloud resources; can handle large-scale computations.	Scalability depends on deploying and managing a large number of edge devices.
Use Cases	Suitable for applications requiring significant computation power and data aggregation.	Ideal for applications needing real-time processing, low latency, and offline capabilities.

OPTIMIZING ON EDGE DEVICES

Vectorization

Given a loop or a nested loop, instead of executing it one item at a time, execute multiple elements contiguous in memory at the same time to reduce latency caused by data I/O.

OPTIMIZING ON EDGE DEVICES

Parallelization

Given an input array (or *n*-dimensional array), divide it into different, independent work chunks, and do the operation on each chunk individually.

OPTIMIZING ON EDGE DEVICES

Loop tiling

Change the data accessing order in a loop to leverage hardware's memory layout and cache. This kind of optimization is hardware dependent. A good access pattern on CPUs is not a good access pattern on GPUs.

OPTIMIZING ON EDGE DEVICES

Operator fusion

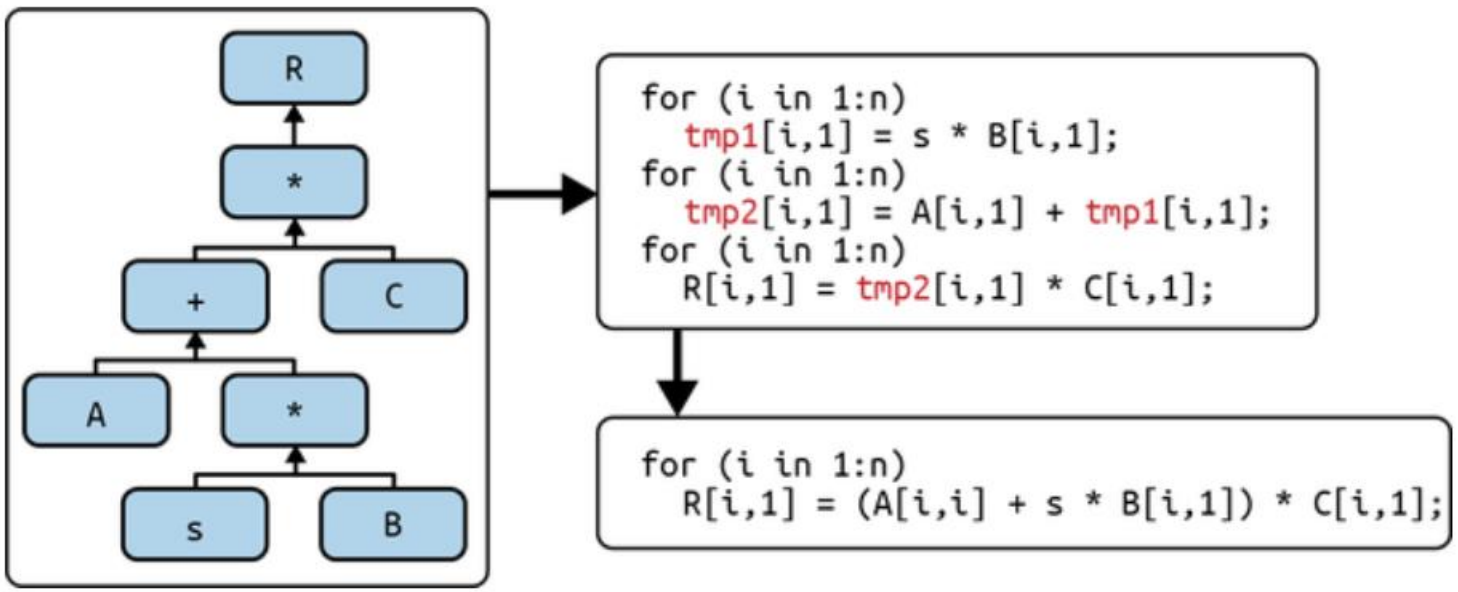


Figure 7-13. An example of an operator fusion. Source: Adapted from an image by Matthias Boehm[47]

OPTIMIZING ML USING ML

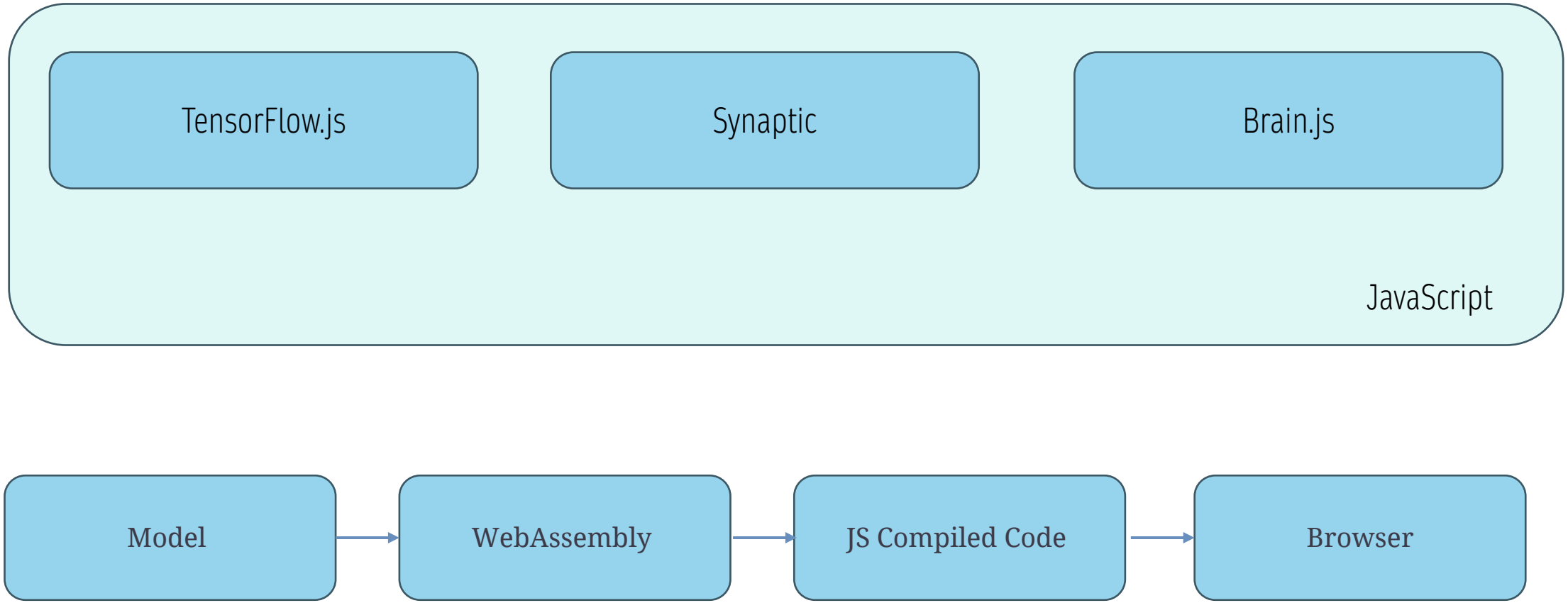
Operation optimization:

```
torch.backends.cudnn.benchmark=True
```

Subgraph optimization:

```
AutoTVM:
https://github.com/apache/tvm/
```

ML IN BROWSERS

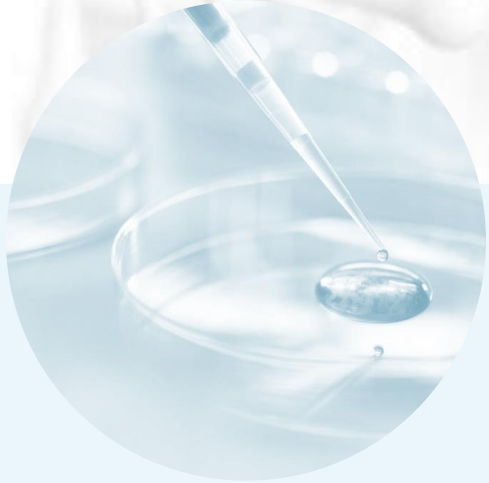


YOUR VOICE MATTERS!

Please take some time to fill up our very very short feedback form ☺

If you would like to connect with me, feel free to scan this!

THANK YOU!



Q&A TIME