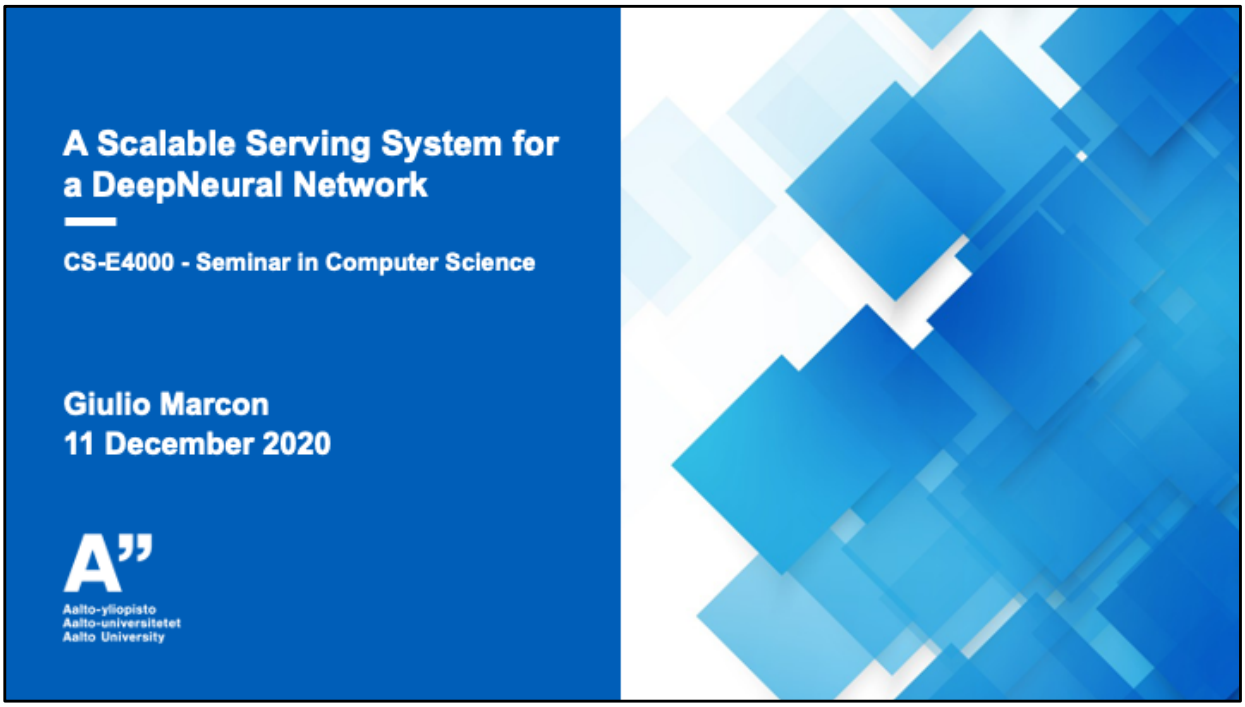


Good morning everyone! A fundamental problem in Machine Learning is to distribute the large incoming work-load onto the machines whilst improving the workflow of the data scientist that has to develop models that will be deployed into production.

My name is Giulio Marcon, 2nd year student of the master's degree in Autonomous Systems, and the topic of this project was proposed by my tutor Phuong Pham, a phd candidate of the Department of Computer Science here at Aalto.

Overview on today's presentation:

- Serving
- Related work solutions
- Comparing Tensorflow Serving and BentoML
- Possible Improvements
- Conclusions

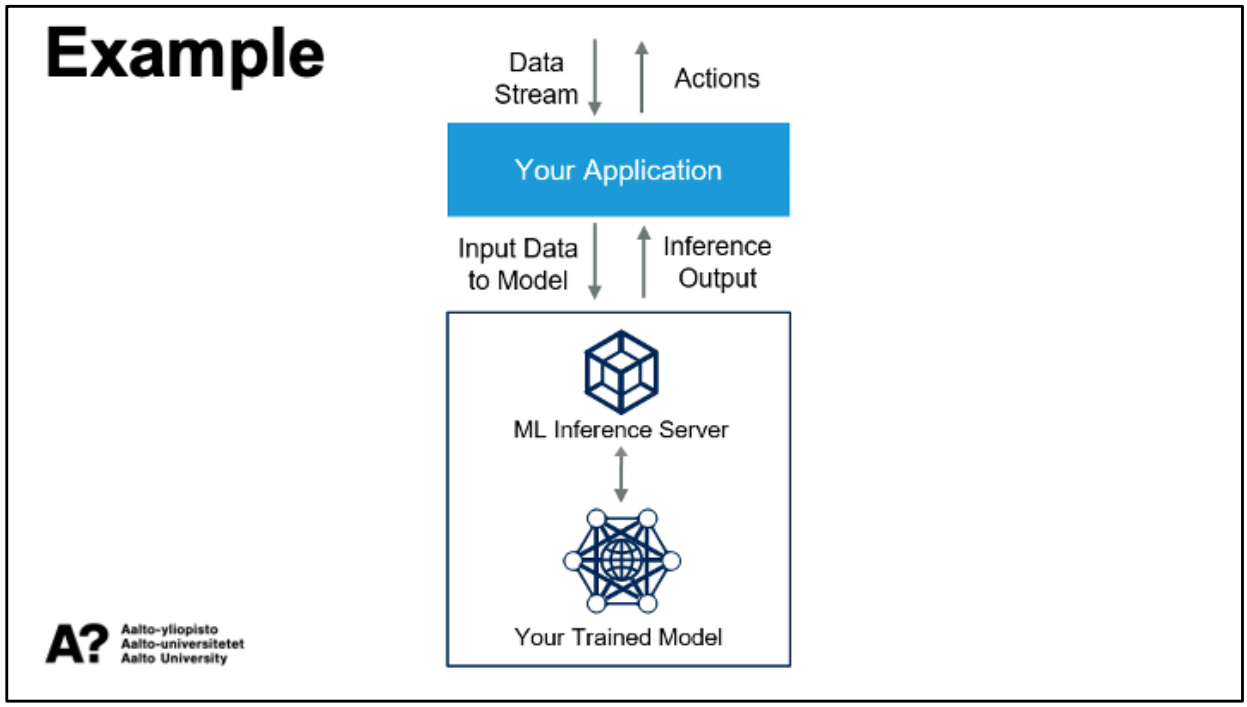
Aalto-yliopisto
Aalto-universitetet
Aalto University

So, I have divided this talk into five parts.
First, we will introduce the topic by talking about what is serving and why is it important.
Second, we will discuss the existing solutions to tackle this problem, with a focus on two of the most popular ones.
Third, we will talk about the difference between these two frameworks.
Fourth, we will examine the possible improvements that could impact this workflow.
Finally, I'll present my conclusions.

What is Serving?

Aalto-yliopisto
Aalto-universitetet
Aalto University

We'll start by explaining what is serving



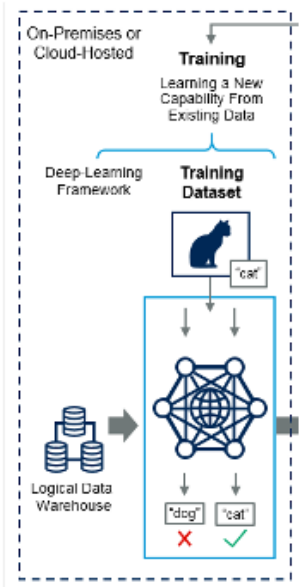
Serving is the process of applying machine learning models after they have been trained.

Consider the example of a machine learning inference server that executes the model algorithm given data gathered from an IoT sensor, for instance a camera, and returns the inference output.

The image illustrates the infrastructure, and it shows the two stages of this process: training and inference.

Training

- **Computationally Expensive**
- **Requires Multiple Passes on the Dataset**



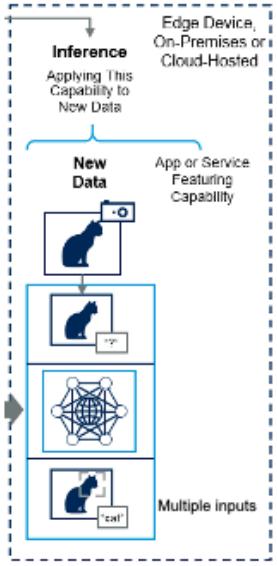
Aalto-yliopisto
Aalto-universitetet
Aalto University

Training refers to the process of creating a machine learning algorithm. In our example, it involves the use of a deep-learning framework (for exampleTensorFlow) and training dataset. In this case, IoT data provides a source of training data that data scientists and engineers can use to train the models for a variety of use cases.

This process is often computationally expensive and requires multiple passes over potentially large datasets.

Inference

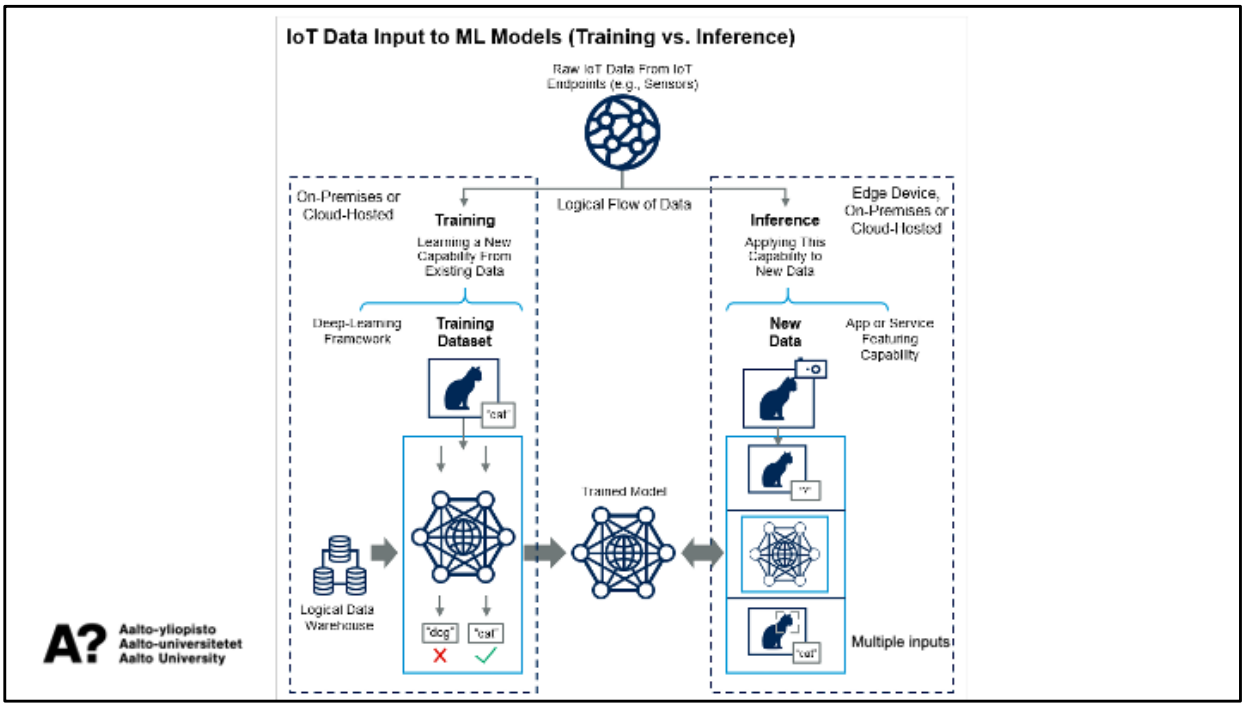
- **User-facing Applications**
- **Runs in real-time**



Aalto-yliopisto
Aalto-universitetet
Aalto University

Inference refers to the process of using a trained machine learning algorithm to make a prediction given an input. In our example, IoT data can be used as the input to a trained machine learning model, enabling predictions that can guide decision logic on the device, at the edge gateway or elsewhere in this IoT system.
It is important to notice that the input can is usually very diverse and significant in its size, which is one of the main problems that needs to be tackled when designing this type of infrastructure.

So Inference is typically part of user-facing applications, and must run in real-time.



Here we can see the whole infrastructure of the example just explained.

If we analyze what a typical workflow of a data scientist looks like, we will notice that the process of deploying models requires additional effort and attention, which results in them being distracted from their problem at hand. Apart from that, having many data scientists build and maintain their own serving solutions means that there may be a lot of duplicated effort.

Related work solutions

Aalto-yliopisto
Aalto-universitetet
Aalto University

Most of the solutions that try to tackle this problem are done using custom code developed by individual teams for specific use cases, leading to duplicated effort and fragile systems with high technical debt.

Related work solutions

- **Tensorflow Serving**
- **BentoML**
- **Clipper**
- **Nexus**
- **InferLine**
- **TorchServe**
- **…and many more**

Aalto-yliopisto
Aalto-universitetet
Aalto University

For example we have Tensorflow Serving, BentoML (which will both be the focus on this presentation) but also services such as Clipper, Nexus, Inferline, TorchServe, among many others.
All these different solutions try to tackle different parts of this problem and present their own advantages and disadvantages.

Tensorflow Serving

- High performance
- High availability
- Actively maintained

Aalto-yliopisto
Aalto-universitetet
Aalto University

Tensorflow Serving is an open-source ML model serving project by Google. It aims to be a lexible, high-performance serving system, designed for production environments, that facilitates the deployment of new algorithms and experiments, while keeping the same server architecture and APIs. Tensorflow Serving provides out-of-the-box integration with Tensorflow models, but can be easily extended to serve other types of models and data.
Some of the advantages of using this technology include:
High performance
High availability
Actively maintained

BentoML

- **Ability to create basic API**
- **High performant**
- **Highly flexible**

Aalto-yliopisto
Aalto-universitetet
Aalto University

BentoML is an open-source platform for high-performance ML framework for serving, managing, and deploying machine learning models

The main advantages of the project are:
Ability to create basic API endpoints
High-Performant online API for serving with adaptive micro-batching support.
Flexibility, by providing support for all major machine learning training frameworks.

What are the differences between the two platforms?

Aalto-yliopisto
Aalto-universitetet
Aalto University

The two frameworks have been compared using the Fashion MNIST dataset , which often used to compare machine learning models

Comparing the two platforms

BentoML	TF Serving
• Multi-framework support • Support a great variety of model types • Open Source	• Currently supports only Tensorflow models • Versioning • Open Source

Aalto-yliopisto
Aalto-universitetet
Aalto University

The most significant differences between the two platforms are:

BentoML has multi-framework support, and is fully compatible with Tensorflow, PyTorch, Scikit-Learn, and many others. On the other hand TS Serving, only supports Tensorflow framework as of now. There are some work arounds to avoid this but

Tensorflow loads the model from a specific format, which means that all the graphs and computations must be compiled into the Saved Model. BentoML keeps the Python run time in serving time, making it possible to do pre-processing and post-processing in serving endpoints.

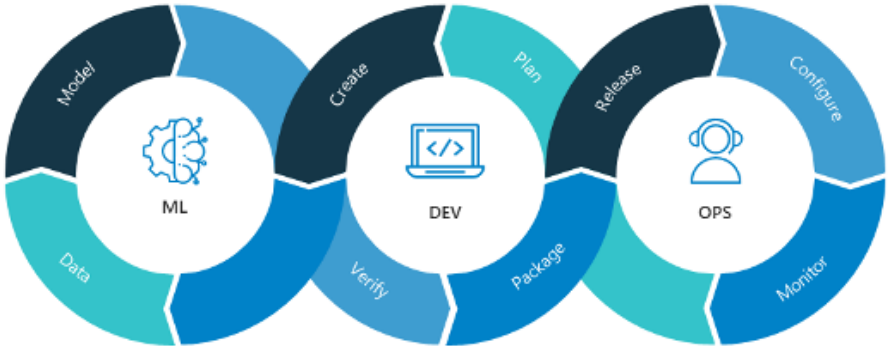
finally, one of the most useful features for production purposes is Tensorflow-serving's ability to serve different versions of the same model, which can be useful when comparing changes during testing.

Possible Improvements

Aalto-yliopisto
Aalto-universitetet
Aalto University

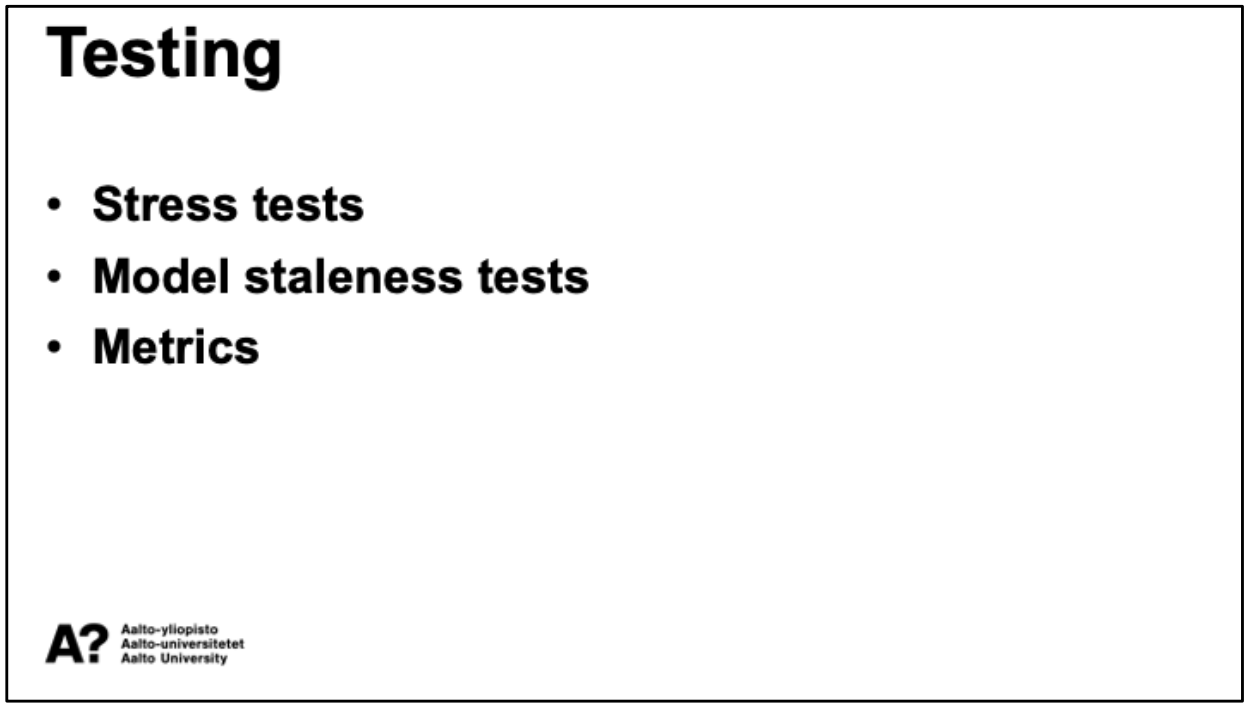
So what are possible improvements that could be made in this workflow?

MLOps



Aalto-yliopisto
Aalto-universitetet
Aalto University

I'll start first by introducing the concept of MLOps, an emerging field that aims to solve the technical debt that currently exists in machine learning systems
MLOps consist of a combination of philosophies and practices designed to enable data science and IT teams to rapidly develop, deploy, maintain, and scale out Machine Learning models. By following these guidelines, new versions of the models can be deployed into production constantly and reliably



One part where the current process could be improved is testing.
The validation of the trained model is still a tedious task that takes a lot of effort and is prone to errors.

A specific testing support and methodology for detecting ML-specific errors needs to be established and followed in order to successfully tackle this problem, and the serving solutions currently available to us should provide a framework to help developers implementing this solution.

Some of the important type of testing includes:

Stress tests to ensure that the infrastructure can handle high volumes of data.

Model staleness test, to check whether the trained model includes up-to-date data.

Checking that the calculated metrics are satisfactory and improve the previous version of the same model, by measuring for example loss metrics and bias, but also the economic cost of training and serving.

Future Work

Aalto-yliopisto
Aalto-universitetet
Aalto University

Some future work could include a comparison between more framework that are currently used for serving. Moreover, a solution to successfully test ML and Deep learning models could be implemented.

So, in conclusion, today we've learned about the importance of Serving in Machine Learning, and the significant technical dept that needs to be solved. We discussed existing implementations and the main difference between them. Finally, we took a look at what could be improved in the process.

Concepts such as continuous delivery and integration, immutable infras-tructure, and serverless computing, have been the focus of DevOps engi-neers in the last few years, and they can be adapted for the development,test, and serving of DNNs and ML models.The so-called MLOps practises are still being designed and refined to thisday, due to how innovative and current they are, but improvements can already be noticed, even if the practices are still in the early stages of development.

Thank you for your attention! So, if you have any questions, I would now be happy to answer them.