Machine Learning Systems Design

Lecture 8: Deployment

CS 329 | Chip Huyen

Logistics

- No assignment 3 so you have more time for final project
- If need help with final project, private Piazza
- Mentorship with Pete Warden?

Project’s general feedback

- Healthcare
  - Be very, very careful of what predictions to show to users
  - Data privacy!
- Evaluation
  - Might need to try different metrics to see what works best for your use case
  - Critical slices
- Mobile app
  - If not familiar with Android/iPhone app development, try simple web interface

Project’s general feedback

- MVP not MVP enough
  - Online learning
  - Federated learning
  - Knowledge distillation
  - Databases
  - CI/CD

Project’s general feedback

- MVP not MVP enough
  - ~~Online learning~~
  - ~~Federated learning~~
  - ~~Knowledge distillation~~
  - ~~Databases~~
  - ~~CI/CD~~

Suggested P0s

1. Get data
2. Have a baseline with a metric that you think is the best right now
    - [P0] Better than random (for milestone 2)
    - [P1] Good enough (for demo)
    - [P2] Great (can possibly get a paper/startup out of this)
3. Understand data
4. Extract features
5. Train a simple model that is better than baseline P0 to validate data
    - Don't worry about the latency, compression just yet
6. Hook it into a simple pipeline
    - Simple user interface (even just a website for users to enter input and get input back is fine)
    - Do inference on each incoming request (no need to wait and do them in batch)
7. Print things on screen or in a log file

Compute tips (on top of class credits)

1. Colab has access to free GPU/TPU
2. GCP credits for new user $300
3. GCP free-tier always-free f1-micro
    - Should be enough to run simple apps

Machine name	Description	vCPUs	Fractional vCPUs[1]	Memory (GB)	Max number of persistent disks (PDs)[2]	Max total PD size (TB)	Local SSD	Maximum egress bandwidth (Gbps)[3]
`f1-micro`	Micro machine type with 0.2 vCPU and 0.6 GB of memory, backed by a shared physical core.	1	0.2[1]	0.60	16	3	No	1

Agenda

1. Model compression & optimization
2. Simple deployment
3. Containerized deployment
4. Test in production
5. OpenAI API workshop

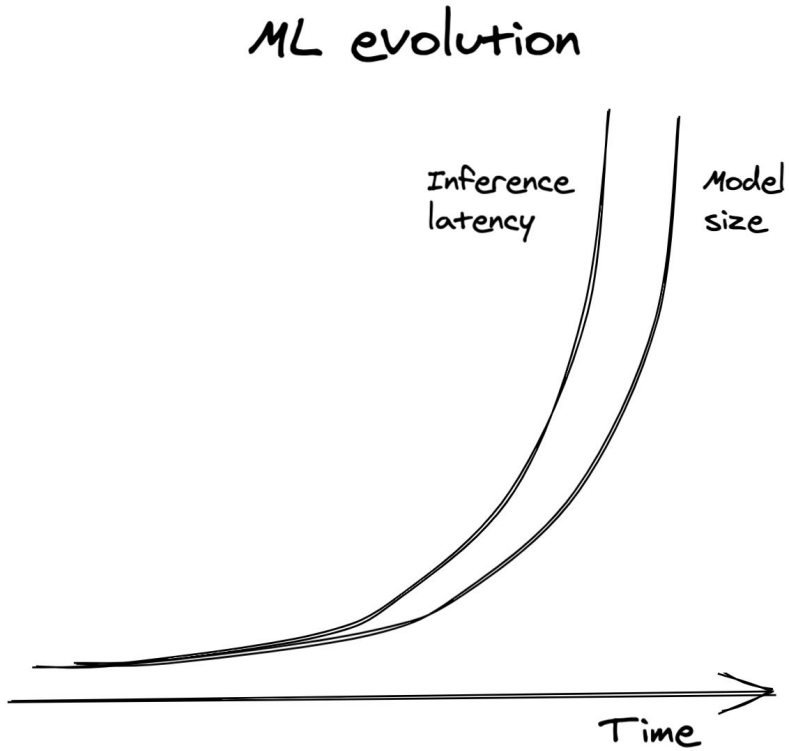
Model compression & optimization

Latency matters

- Latency 100 -> 400 ms reduces searches 0.2% - 0.6% (2009)

- 30% increase in latency costs 0.5% conversion rate (2019)

No matter how great your ML models are,
if they take just milliseconds too long,
users will click on something else.



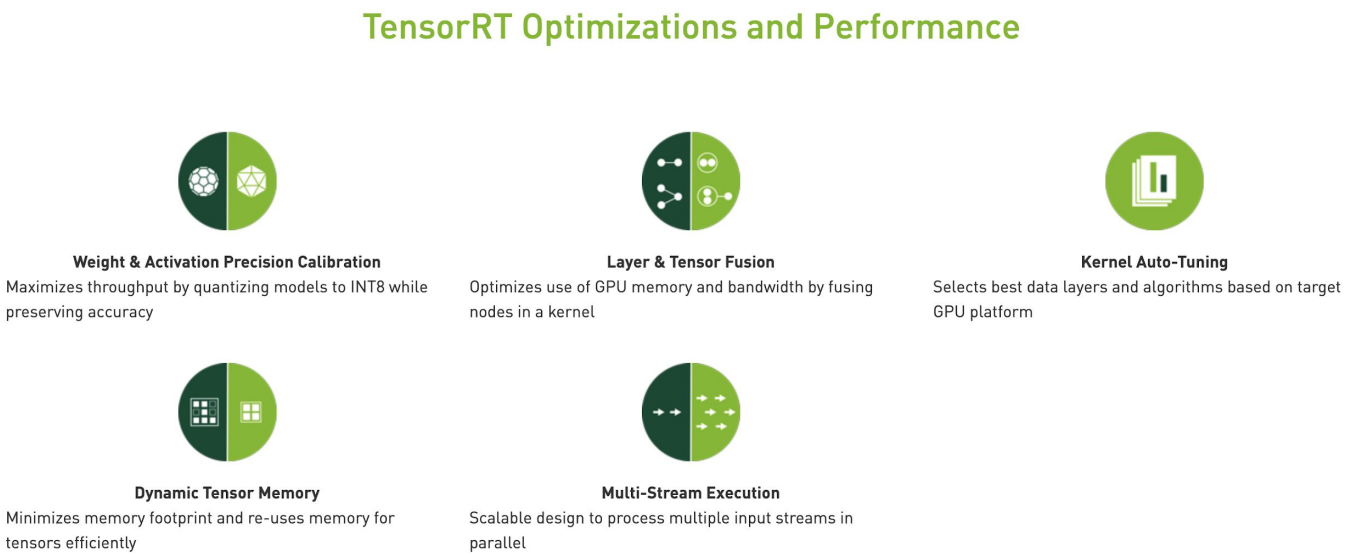
Bigger, better, slower

Fast inference

1. Make models faster
2. Make models smaller
3. Make hardware more powerful

Make models faster

- Inference optimization



Make models smaller

- Model Compression

Quantization	Knowledge distillation
Pruning	Low-ranked factorization

The Top 42 Model Compression Open Source Projects

Categories > Machine Learning > **Model Compression**

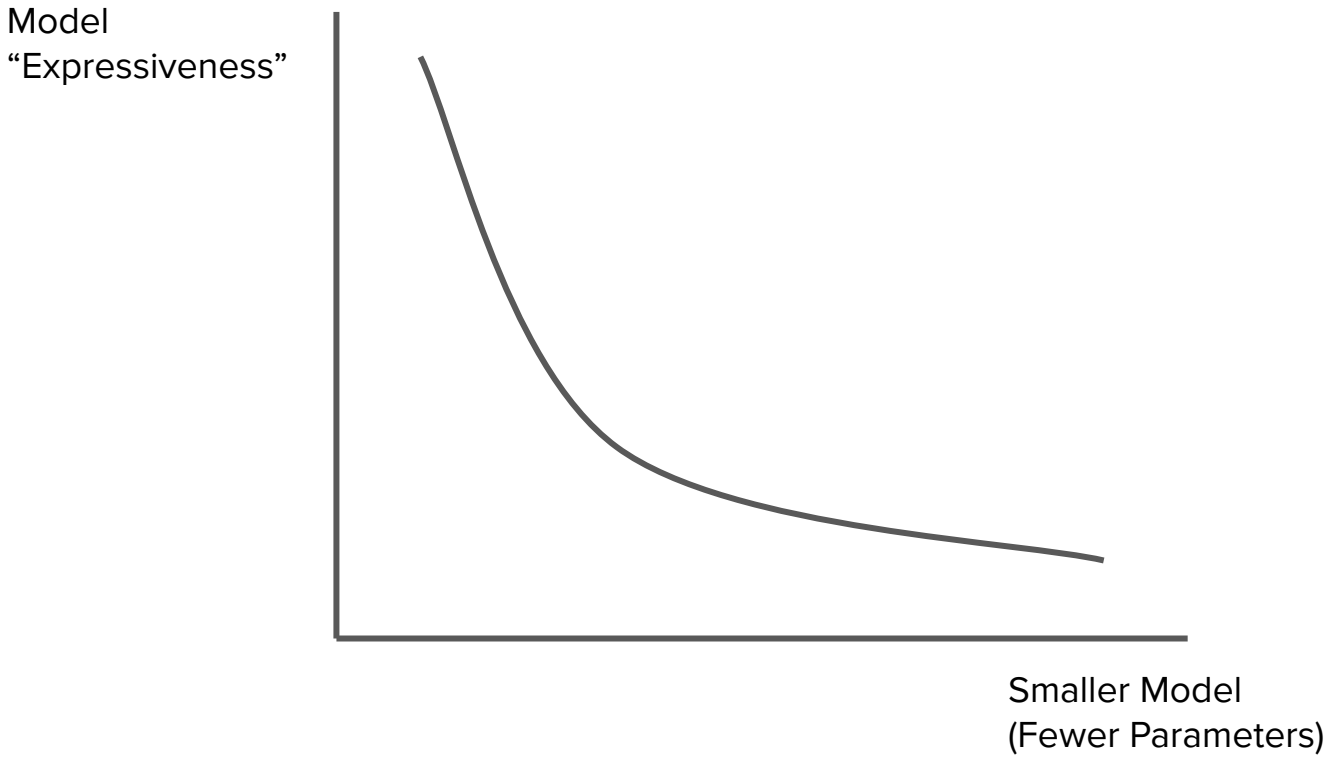
Nni ☆ 8,622
An open source AutoML toolkit for automate machine learning lifecycle, including feature engineering, neural architecture search, model compression and hyper-parameter tuning.

Pocketflow ☆ 2,583
An Automatic Model Compression (AutoMC) framework for developing smaller and faster AI applications.

Pretrained Language Model ☆ 1,449
Pretrained language model and its related optimization techniques developed by Huawei Noah's Ark Lab.

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

There is No Free Lunch!



Model compression: quantization

- Reduces the size of a model by using fewer bits to represent parameter values.
  - E.g. half-precision floating point (16-bit), or integer (8-bit) representations

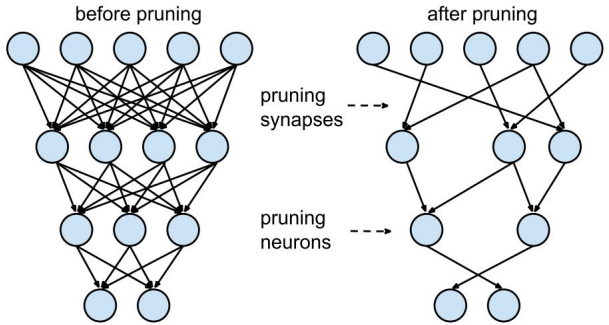
Model compression: knowledge distillation

- Train a small model (“student”) to mimic the results of a larger model (“teacher”).
  - Teacher & student can be trained at the same time.
  - E.g. DistillBERT, reduces size of BERT by 40%, and increases inference speed by 60%, while retaining 97% language understanding.

Source: Han et al., 2015 (arXiv:1506.02626)

Model compression: pruning

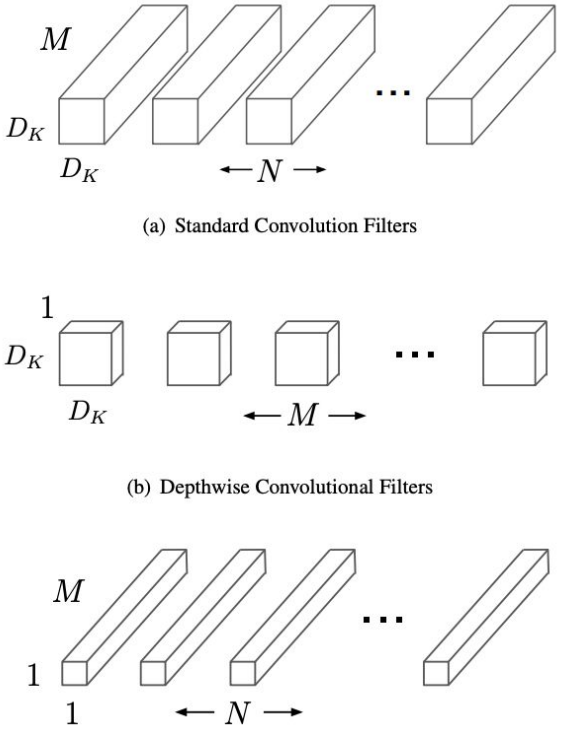
- Doesn’t reduce number of parameters; reduces the number of *nonzero* parameters.
  - Efficient storage schemes for sparse data.
- Can prune either weights or neurons:
  - Weights: set weights which are almost zero to zero.
  - Neurons: neurons with invariant activations can be removed.



Learning both Weights and Connections for Efficient Neural Networks, Han et al., 2015.

Model compression: factorization

- Insight: 3 x 3 matrix can be written as a product of 3 x 1 and 1 x 3
  - 6 params instead of 9
- Replace convolution filters (many parameters) with compact blocks (fewer parameters)
  - E.g. MobileNets: The standard convolutional filters in (a) are replaced by depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter.



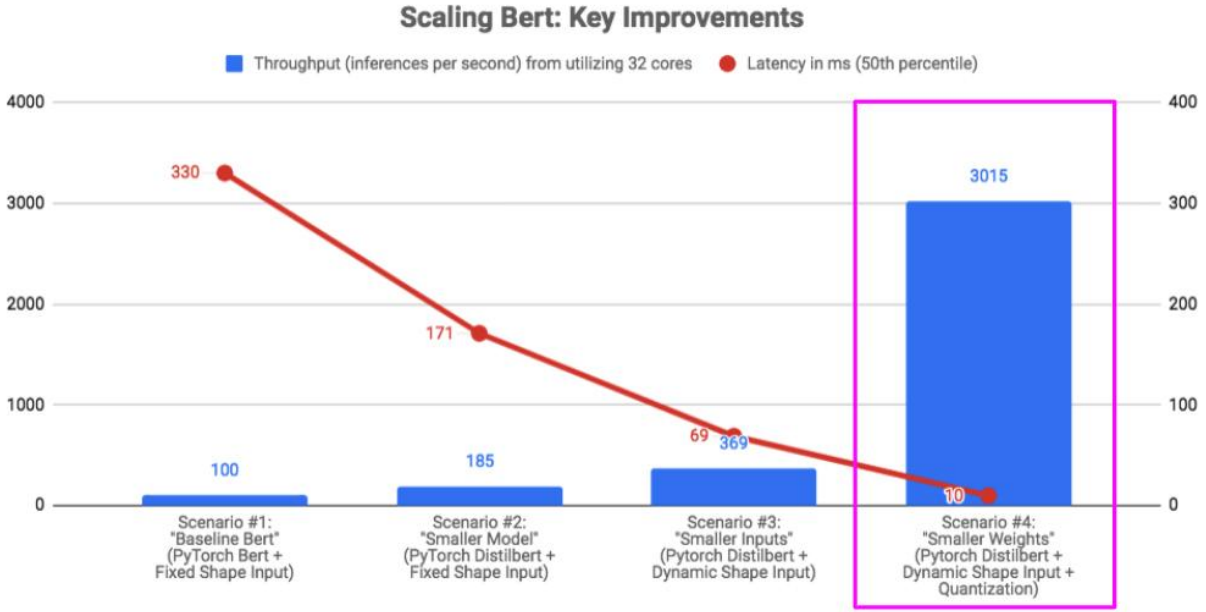
(a) Standard Convolution Filters

(b) Depthwise Convolutional Filters

(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, Howard et al., 2017.

Make models smaller: case study



How We Scaled Bert To Serve 1+ Billion Daily Requests on CPUs (Roblox, 2020)

Make hardware more powerful

- Training & inference
- Server & edge

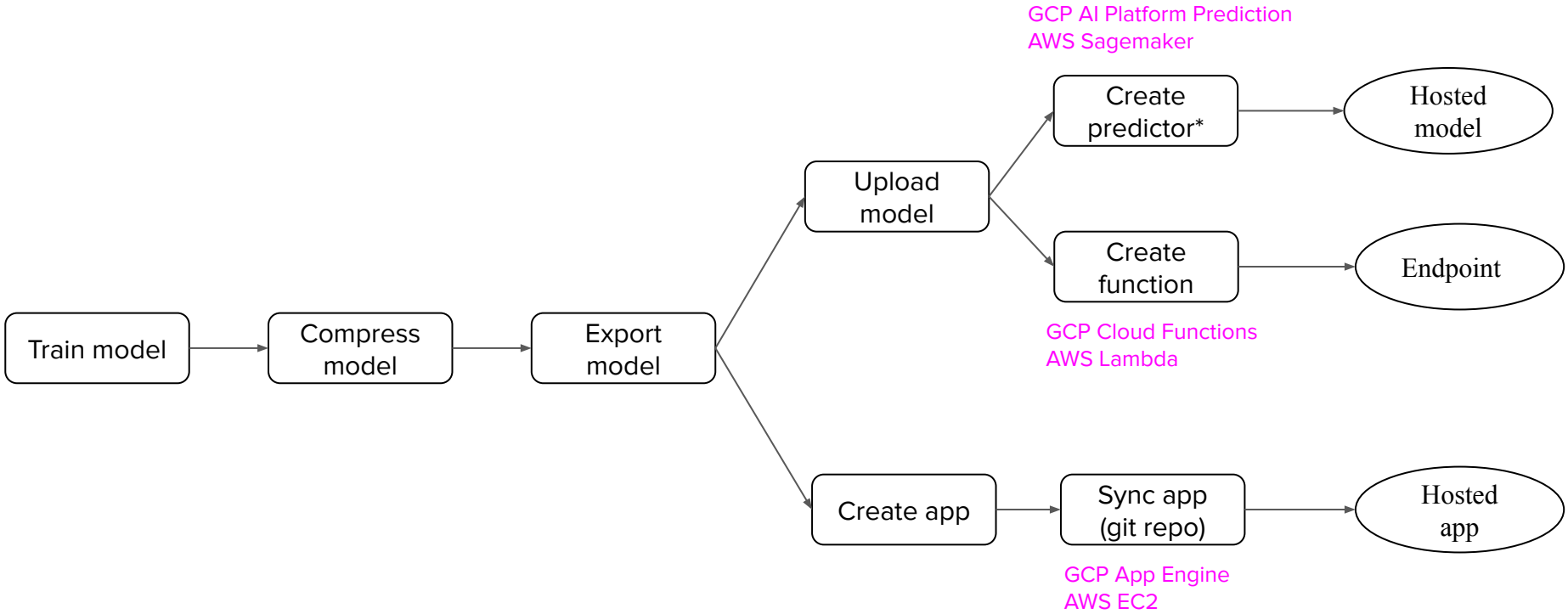
Will cover this in TinyML lecture!

Hardware startup	Raised ($M)	Year founded	Location
SambaNova	1100	2017	Bay Area
Graphcore	682	2016	UK
Groq	362	2016	Bay Area
Nuvia	293	2019	Bay Area
Wave Computing	203	2008	Bay Area
Cambricon	200	2016	China
Cerebras	112	2016	Bay Area
Hailo	88	2017	Israel
Habana Labs	75	2016	Israel
Kneron	73	2015	San Diego
Prophesee	65	2014	France
Syntiant	65	2017	LA
Groq	62	2016	Bay Area
EdgeQ	53	2018	Bay Area
LeapMind	50	2012	Japan

Simple deployment

We’ve trained a model. Now what?

Deploy a model on cloud

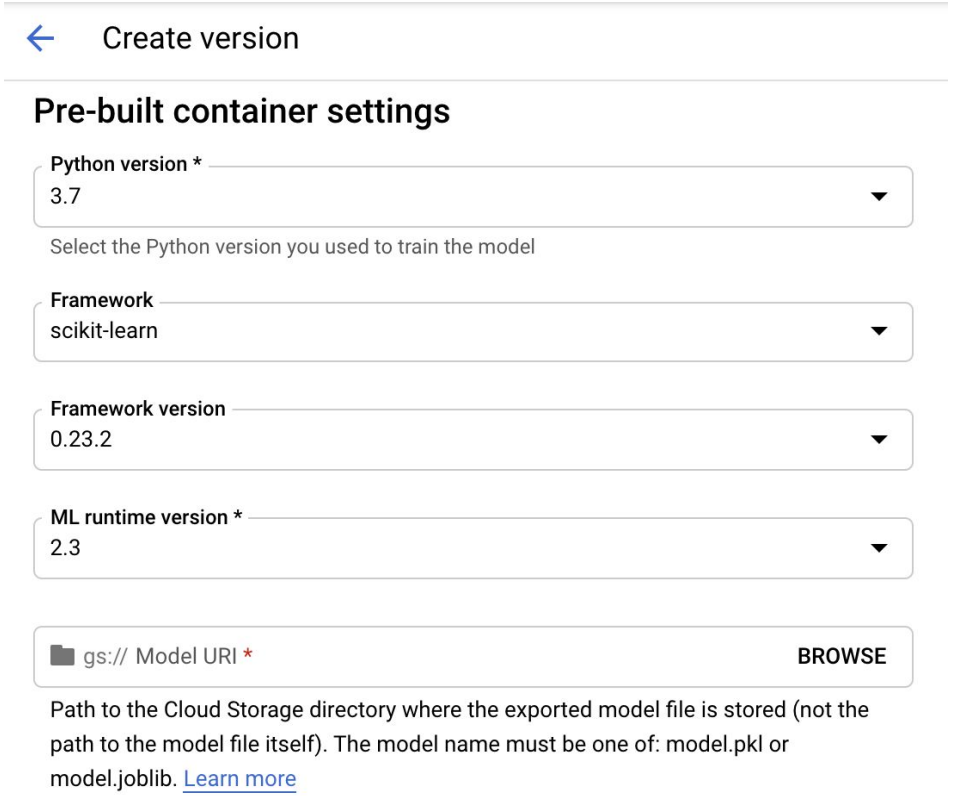


Export a model

- Turn models into binary formats
  - scikit-learn, XGBoost -> joblib, pickle
  - TensorFlow -> .save()
  - PyTorch -> .save()

GCP AI Platform: hosted model

1. Upload model to cloud storage
2. Create a version for your model



GCP AI Platform: hosted model

1. Upload model to cloud storage
2. Create a version for your model

Output looks like this

```
{
  "name": "projects/[YOUR-PROJECT-ID]/operations/create_[YOUR-MODEL-NAME]_[YOUR-VERSION-NAME]-[TIMESTAMP]",
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.ml.v1.OperationMetadata",
    "createTime": "2018-07-07T02:51:50Z",
    "operationType": "CREATE_VERSION",
    "modelName": "projects/[YOUR-PROJECT-ID]/models/[YOUR-MODEL-NAME]",
    "version": {
      "name": "projects/[YOUR-PROJECT-ID]/models/[YOUR-MODEL-NAME]/versions/[YOUR-VERSION-NAME]",
      "deploymentUri": "gs://your_bucket_name",
      "createTime": "2018-07-07T02:51:49Z",
      "runtimeVersion": "2.3",
      "framework": "[YOUR_FRAMEWORK_NAME]",
      "machineType": "[YOUR_MACHINE_TYPE]",
      "pythonVersion": "3.7"
    }
  }
}
```

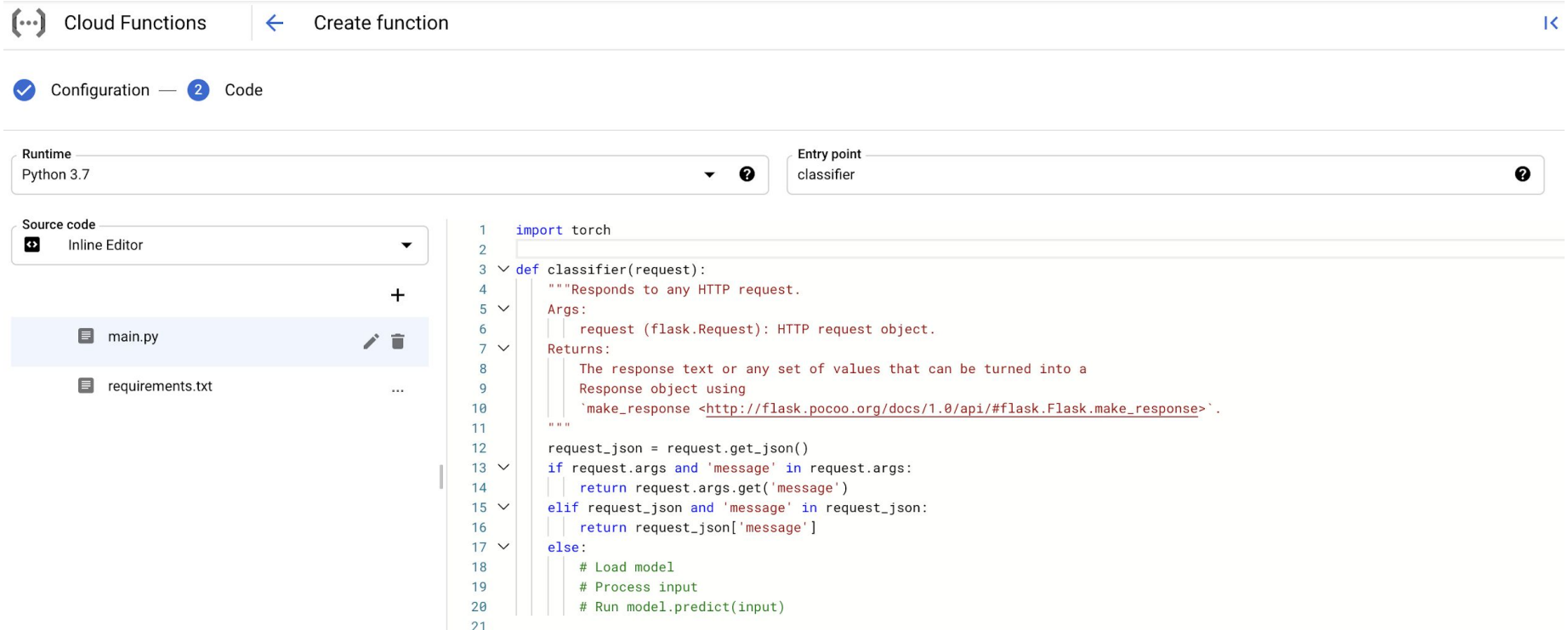
GCP AI Platform

1. Train model
2. Deploy a model & version
3. Make online/batch predictions

```python
import googleapiclient.discovery

def predict_json(project, model, instances, version=None):
    """Send json data to a deployed model for prediction.
    Args:
        project (str): project where the AI Platform Prediction Model is deployed.
        model (str): model name.
        instances ([[float]]): List of input instances, where each input
           instance is a list of floats.
        version: str, version of the model to target.
    Returns:
        Mapping[str: any]: dictionary of prediction results defined by the
            model.
    """
    # Create the AI Platform Prediction service object.
    # To authenticate set the environment variable
    # GOOGLE_APPLICATION_CREDENTIALS=<path_to_service_account_file>
    service = googleapiclient.discovery.build('ml', 'v1')
    name = 'projects/{}/models/{}'.format(project, model)

    if version is not None:
        name += '/versions/{}'.format(version)

    response = service.projects().predict(
        name=name,
        body={'instances': instances}
    ).execute()

    if 'error' in response:
        raise RuntimeError(response['error'])

    return response['predictions']
```

GCP Cloud Functions: endpoint

1. Create a function



GCP Cloud Functions: endpoint

1. Create a function -> hosted endpoint (e.g. HTTP trigger)
2. Access endpoints

```
curl -X POST "https://YOUR_REGION-YOUR_PROJECT_ID.cloudfunctions.net/FUNCTION_NAME" -H
"Content-Type:application/json" --data '{"sentence":"This tweet is supposed to be very angry."}'
```

GCP App Engine: hosted app

1. Clone your GitHub repo into your Gcloud project directory
    - main.py
    - app.yaml
    - requirements.txt

GCP App Engine: hosted app

1. Clone your GitHub repo into your Gcloud project directory
    - main.py: define endpoint for predict

```
@app.route('/predict', methods=['POST'])
def predict():
    X = request.get_json()['X']
    y = MODEL.predict(X).tolist()
    return json.dumps({'y': y}), 200
```

GCP App Engine

1. Clone your GitHub repo into your Gcloud project directory
    - main.py: define endpoint for predict
    - app.yaml: define app environment e.g. variables, where the exported model is located

```
env_variables:
    # The app will look for the model file at: gs://MODEL_BUCKET/MODEL_FILENAME
    MODEL_BUCKET: BUCKET_NAME
    MODEL_FILENAME: sentiment_classifier.pkl
```

GCP App Engine

1. Clone your GitHub repo into your Gcloud project directory
    - main.py: define endpoint for predict
    - app.yaml: define app environment e.g. variables, where the exported model is located
    - requirements.txt: dependencies

```
numpy
scikit-learn
torch
fastapi
```

GCP App Engine: hosted app

1. Clone your GitHub repo into your Gcloud project directory
2. Initialize: `gcloud init`
3. Deploy: `gcloud app deploy`
4. View your app in browser: `gcloud app browse`

`https://PROJECT_ID.REGION_ID.r.appspot.com`

Containerized deployment

GCP runtime version

GCP AI Platform Prediction revisited

```
{
  "name": "projects/[YOUR-PROJECT-ID]/operations/create_[YOUR-MODEL-NAME]_[YOUR-VERSION-NAME]-[TIMESTAMP]",
  "metadata": {
    "@type": "type.googleapis.com/google.cloud.ml.v1.OperationMetadata",
    "createTime": "2018-07-07T02:51:50Z",
    "operationType": "CREATE_VERSION",
    "modelName": "projects/[YOUR-PROJECT-ID]/models/[YOUR-MODEL-NAME]",
    "version": {
      "name": "projects/[YOUR-PROJECT-ID]/models/[YOUR-MODEL-NAME]/versions/[YOUR-VERSION-NAME]",
      "deploymentUri": "gs://your_bucket_name",
      "createTime": "2018-07-07T02:51:49Z",
      "runtimeVersion": "2.3",
      "framework": "[YOUR_FRAMEWORK_NAME]",
      "machineType": "[YOUR_MACHINE_TYPE]",
      "pythonVersion": "3.7"
    }
  }
}
```

GCP runtime version

- Runtime version is container image
- Each AI Platform Prediction model runs in a container

Supported AI Platform Prediction runtime versions

The following versions are supported in AI Platform Prediction:

Version	Package	Released On	Last Updated
2.3	TensorFlow 2.3.1 scikit-learn 0.23.2 XGBoost 1.2.1 GPUs are supported for training and online prediction in this runtime version. *Runtime version 2.3 does not support batch prediction.* Python 3.7 is the only version of Python available for training and online prediction with runtime version 2.3. You cannot use Python 2 with runtime version 2.3.	December 9, 2020	December 9, 2020

Package lists

PyPI packages	Ubuntu packages
xgboost 1.2.1	curl
numpy 1.18.5	libcurl3-dev
pandas 1.1.3	wget
scipy 1.4.1	zip
scikit-learn 0.23.2	unzip
sympy 1.6.2	git
statsmodels 0.12.0	vim

GCP runtime version

- How to use a model with PyTorch?
- How to use other packages?
- How to use different combinations of packages?
- How to specify other stuff e.g.:
  - CUDA version
  - Python version
  - Download random things from the Internet?

Supported AI Platform Prediction runtime versions

The following versions are supported in AI Platform Prediction:

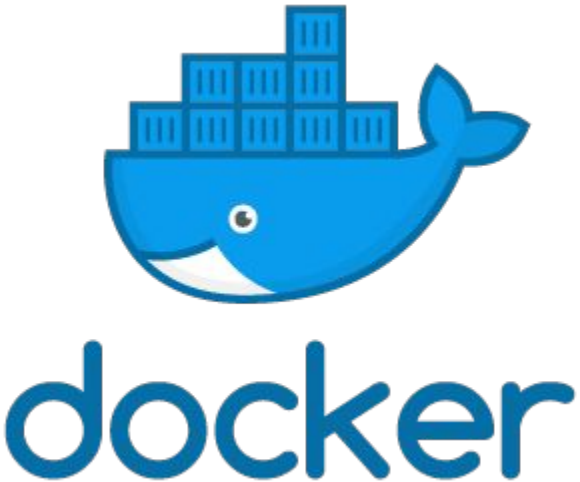
Version	Package	Released On	Last Updated
2.3	TensorFlow 2.3.1 scikit-learn 0.23.2 XGBoost 1.2.1 GPUs are supported for training and online prediction in this runtime version. *Runtime version 2.3 does not support batch prediction.* Python 3.7 is the only version of Python available for training and online prediction with runtime version 2.3. You cannot use Python 2 with runtime version 2.3.	December 9, 2020	December 9, 2020

Package lists

PyPI packages	Ubuntu packages
xgboost 1.2.1	curl
numpy 1.18.5	libcurl3-dev
pandas 1.1.3	wget
scipy 1.4.1	zip
scikit-learn 0.23.2	unzip
sympy 1.6.2	git
statsmodels 0.12.0	vim

Custom container

- An isolated, lightweight environment for running an application on the host OS



Creating a Docker image with Dockerfile

master

transformers / docker / transformers-gpu / Dockerfile

```
FROM nvidia/cuda:10.2-cudnn7-devel-ubuntu18.04
LABEL maintainer="Hugging Face"
LABEL repository="transformers"

RUN apt update && \
    apt install -y bash \
                   build-essential \
                   git \
                   curl \
                   ca-certificates \
                   python3 \
                   python3-pip && \
    rm -rf /var/lib/apt/lists

RUN python3 -m pip install --no-cache-dir --upgrade pip && \
    python3 -m pip install --no-cache-dir \
    jupyter \
    tensorflow \
    torch

RUN git clone https://github.com/NVIDIA/apex
RUN cd apex && \
    python3 setup.py install && \
    pip install -v --no-cache-dir --global-option="--cpp_ext" --global-option="--cuda_ext" ./
```

Docker image

```
FROM nvidia/cuda:10.2-cudnn7-devel-ubuntu18.04
LABEL maintainer="Hugging Face"
LABEL repository="transformers"

RUN apt update && \
    apt install -y bash \
                   build-essential \
                   git \
                   curl \
                   ca-certificates \
                   python3 \
                   python3-pip && \
    rm -rf /var/lib/apt/lists

RUN python3 -m pip install --no-cache-dir --upgrade pip && \
    python3 -m pip install --no-cache-dir \
    jupyter \
    tensorflow \
    torch

RUN git clone https://github.com/NVIDIA/apex
RUN cd apex && \
    python3 setup.py install && \
    pip install -v --no-cache-dir --global-option="--cpp_ext" --global-option="--cuda_ext" ./
```

Base image

Layers

Docker: layers (immediate images)

- A layer is files generated from running a command in Docker

```
FROM node:argon

# Create app directory
RUN mkdir -p /usr/src/app
WORKDIR /usr/src/app

# Install app dependencies
COPY package.json /usr/src/app/
RUN npm install

# Bundle app source
COPY . /usr/src/app

EXPOSE 8080
CMD [ "npm", "start" ]
```

```
$ docker build -t expressweb .
Step 1 : FROM node:argon
argon: Pulling from library/node...
...
Status: Downloaded newer image for node:argon
 ---> 530c750a346e
Step 2 : RUN mkdir -p /usr/src/app
 ---> Running in 5090fde23e44
 ---> 7184cc184ef8
Removing intermediate container 5090fde23e44
Step 3 : WORKDIR /usr/src/app
 ---> Running in 2987746b5fba
 ---> 86c81d89b023
Removing intermediate container 2987746b5fba
Step 4 : COPY package.json /usr/src/app/
 ---> 334d93a151ee
Removing intermediate container a678c817e467
Step 5 : RUN npm install
 ---> Running in 31ee9721cccb
 ---> ecf7275feff3
Removing intermediate container 31ee9721cccb
Step 6 : COPY . /usr/src/app
 ---> 995a21532fce
Removing intermediate container a3b7591bf46d
Step 7 : EXPOSE 8080
 ---> Running in fddb8afb98d7
 ---> e9539311a23e
Removing intermediate container fddb8afb98d7
Step 8 : CMD npm start
 ---> Running in a262fd016da6
 ---> fdd93d9c2c60
Removing intermediate container a262fd016da6
Successfully built fdd93d9c2c60
```

Digging into Docker layers (Jessica G, 2016)

Docker: layers (immediate images)

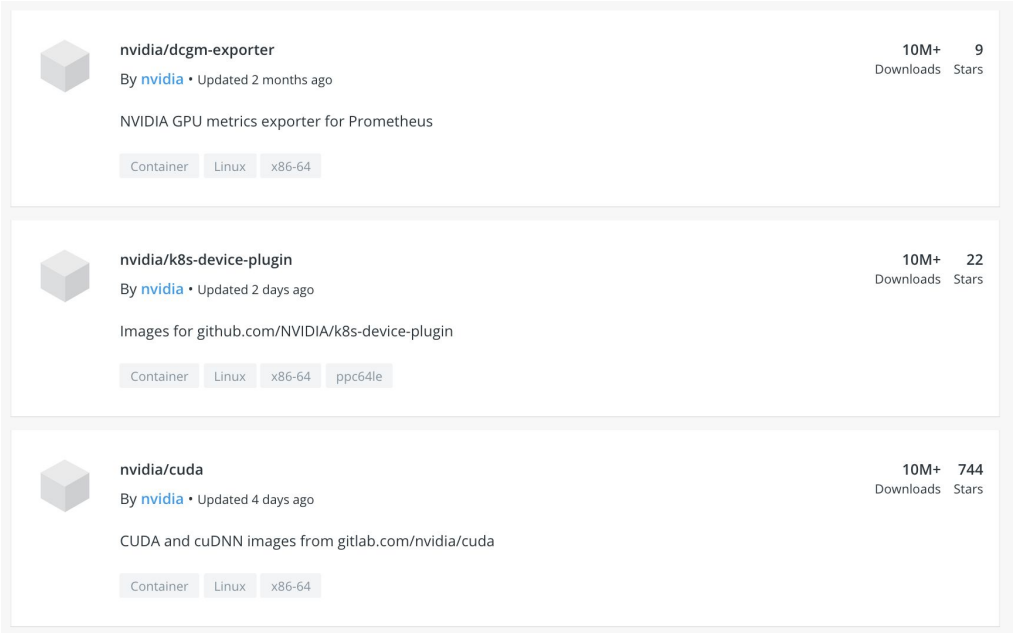
- Faster rebuild
- Layers can be reused by different containers
- Easier to revert back to the last layer

```
$ docker build -t expressweb .
Step 1 : FROM node:argon
argon: Pulling from library/node...
...
Status: Downloaded newer image for node:argon
 ---> 530c750a346e
Step 2 : RUN mkdir -p /usr/src/app
 ---> Running in 5090fde23e44
 ---> 7184cc184ef8
Removing intermediate container 5090fde23e44
Step 3 : WORKDIR /usr/src/app
 ---> Running in 2987746b5fba
 ---> 86c81d89b023
Removing intermediate container 2987746b5fba
Step 4 : COPY package.json /usr/src/app/
 ---> 334d93a151ee
Removing intermediate container a678c817e467
Step 5 : RUN npm install
 ---> Running in 31ee9721cccb
 ---> ecf7275feff3
Removing intermediate container 31ee9721cccb
Step 6 : COPY . /usr/src/app
 ---> 995a21532fce
Removing intermediate container a3b7591bf46d
Step 7 : EXPOSE 8080
 ---> Running in fddb8afb98d7
 ---> e9539311a23e
Removing intermediate container fddb8afb98d7
Step 8 : CMD npm start
 ---> Running in a262fd016da6
 ---> fdd93d9c2c60
Removing intermediate container a262fd016da6
Successfully built fdd93d9c2c60
```

Digging into Docker layers (Jessica G, 2016)

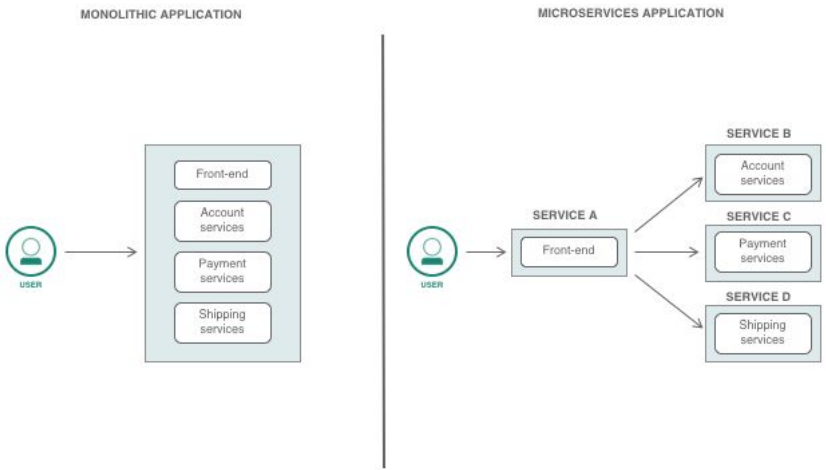
Container registry

- Single place to manage docker images
  - Docker Hub
  - Private solution



Container: microservices

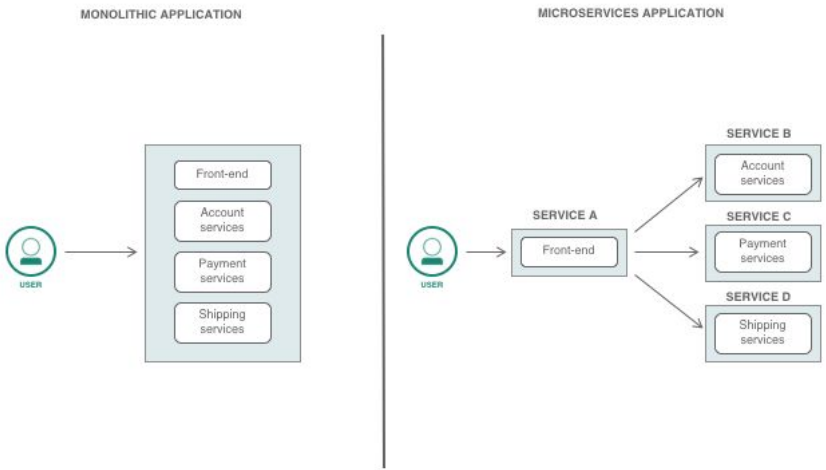
- Monolith: all application logics in one instance
- Microservices:
  - break application logics into smaller services
  - each service in its own container



[Why should you use microservices and containers?](#) (IBM)

Container: microservices

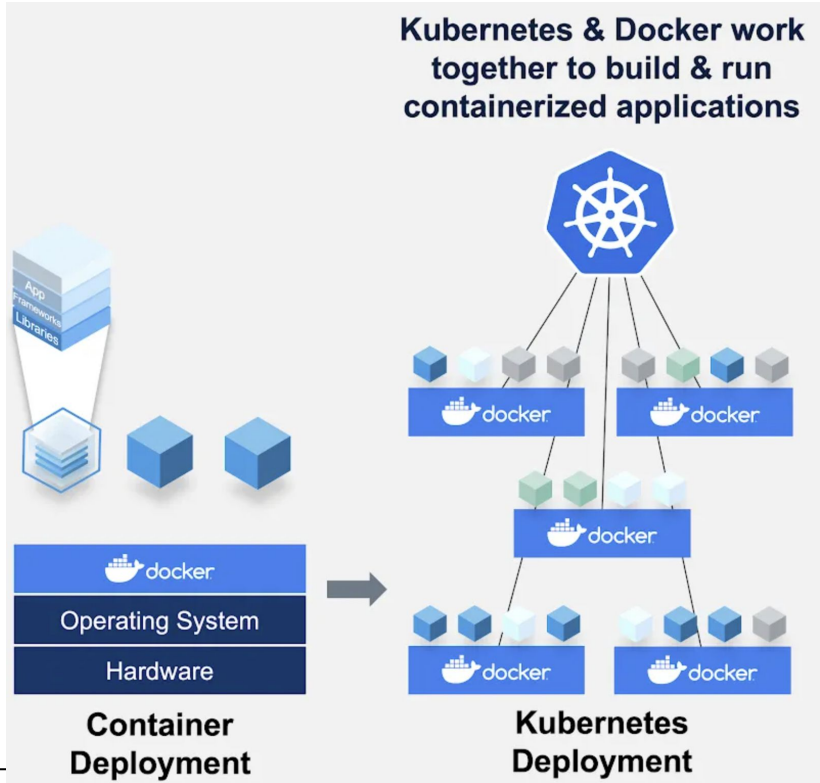
- **Reduced complexity**: each developer works on a smaller codebase
- **Faster development cycle**: easier review process
- **Flexible stack**: different microservices can use different technology stacks



[Why should you use microservices and containers?](#) (IBM)

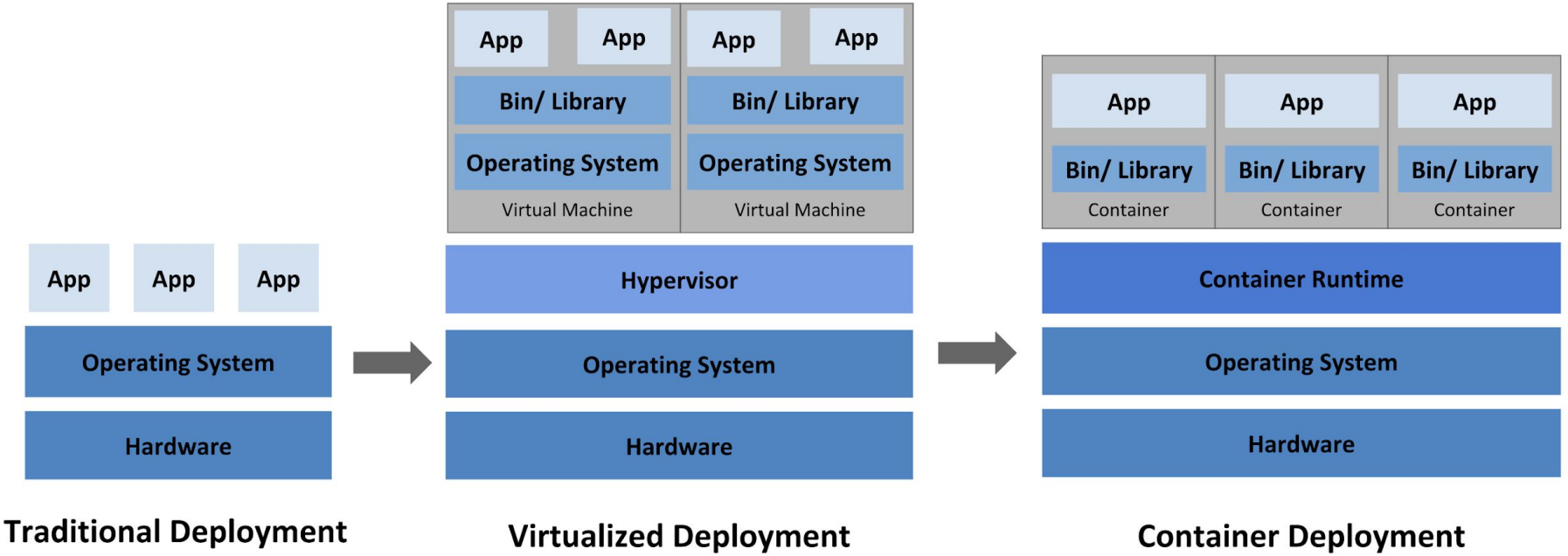
Kubernetes: container orchestration

- How to manage multiple containers?
- If a container goes down, how to restart/replace it?
- How to allocate resources between containers?
- Security access
- ...



Top Questions Answered: Docker and Kubernetes? I Thought You Were Competitors! (Docker blog, 2019)

Container isn't the only virtualization technology



https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/

VMs vs. containers

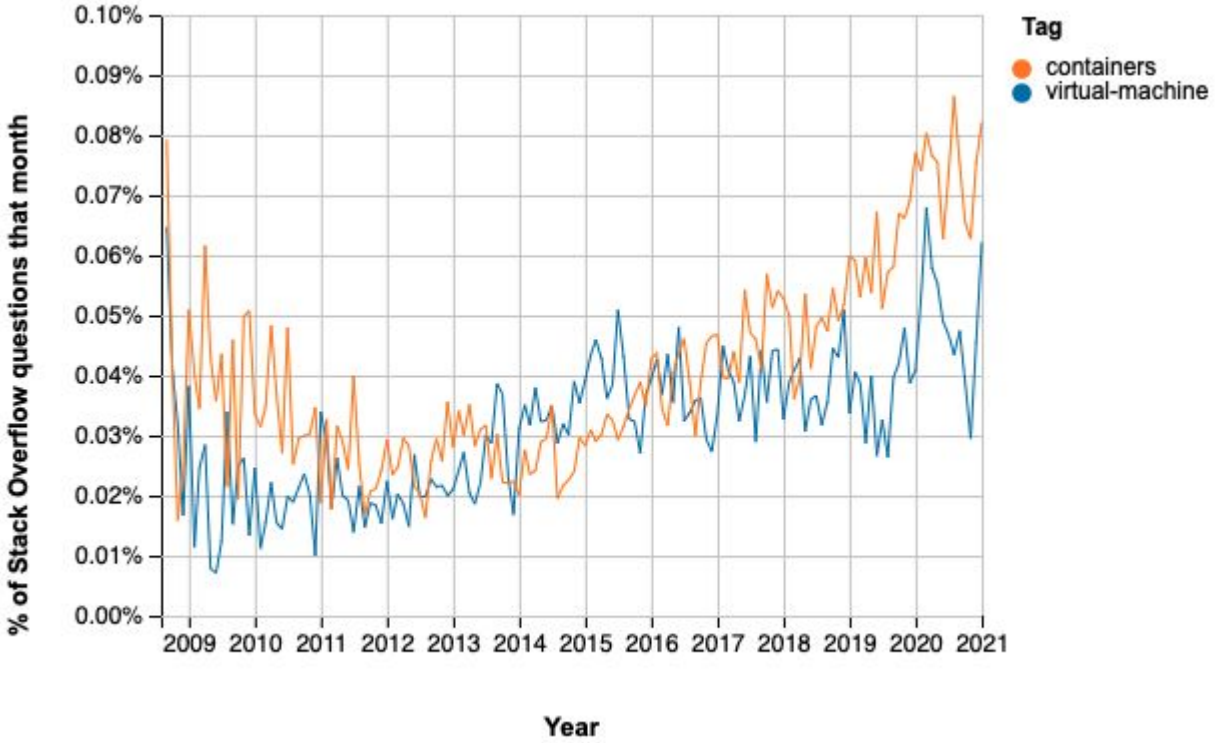
Virtual machine	**Container**
Full virtualization	OS-level virtualization
Simulates unmodified “guest” OS	Creates isolated user space on same OS
Can run any OS (e.g. Windows on Linux)	Can only accommodate different distributions of same OS
Heavy	Lightweight
Startup time: minutes	Startup time: seconds

VMs vs. containers

Fascinating history on how legacy systems motivated virtualization technology

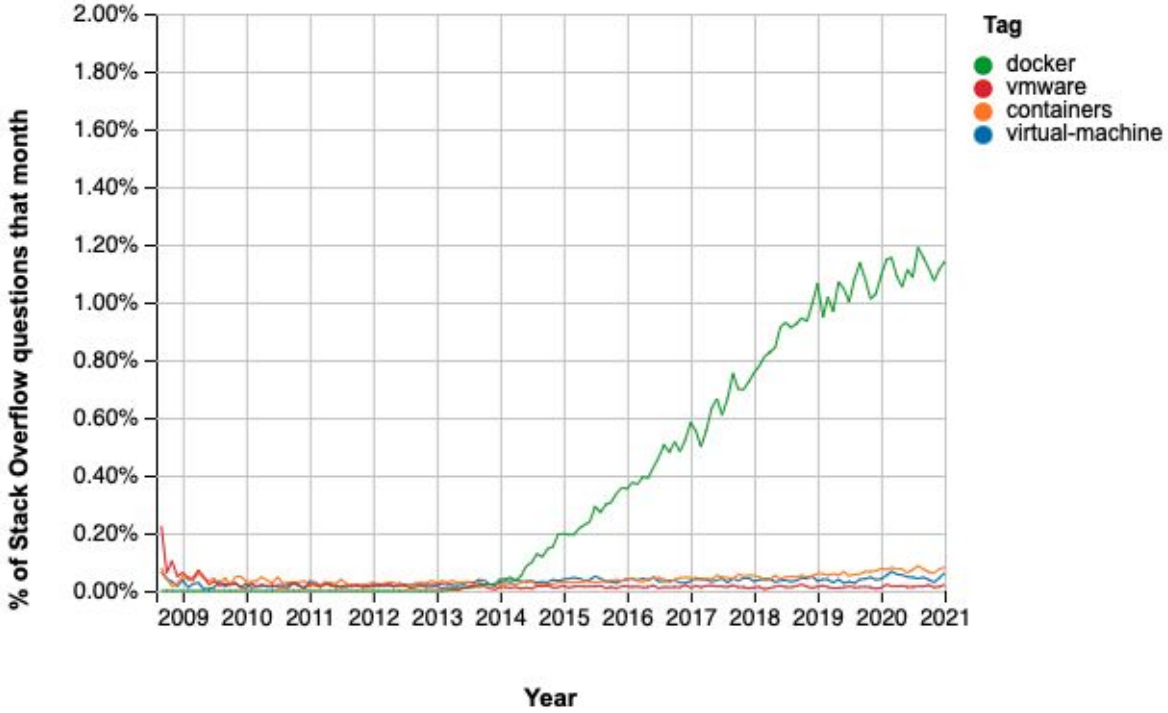
Virtual machine	**Container**
Full virtualization	OS-level virtualization
Simulates unmodified “guest” OS	Creates isolated user space on same OS
Can run any OS (e.g. Windows on Linux)	Can only accommodate different distributions of same OS
Heavy	Lightweight
Startup time: minutes	Startup time: seconds

VM vs. container



https://insights.stackoverflow.com/trends

VM vs. container

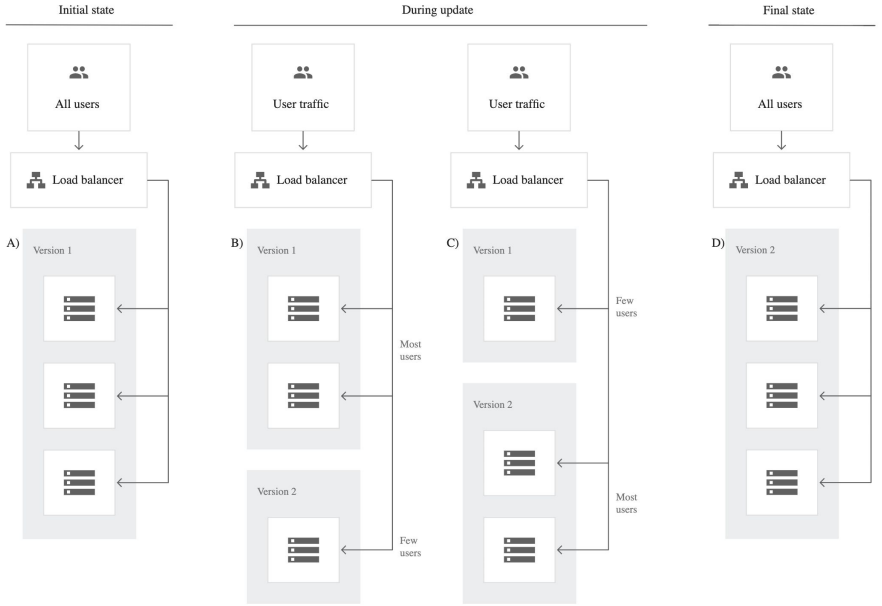


https://insights.stackoverflow.com/trends

Test in production

Canary testing

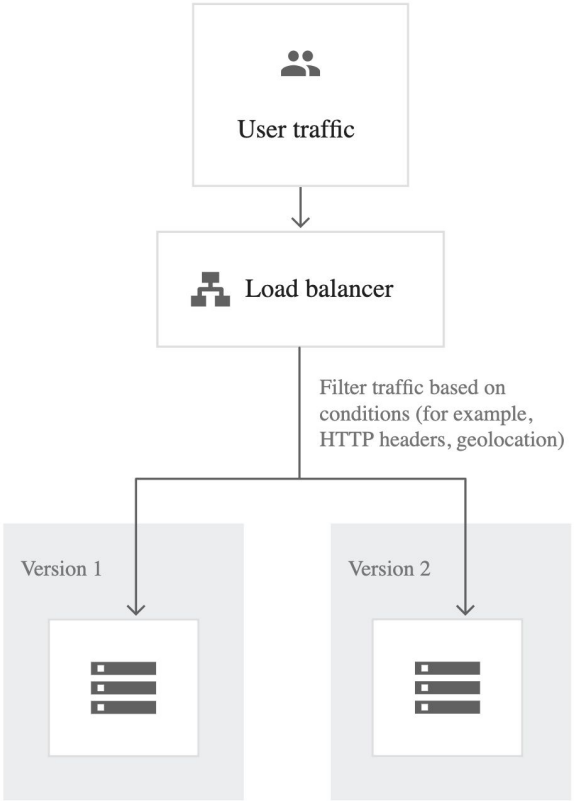
- New model alongside existing system
- Some traffic is routed to new model
- Slowly increase the traffic to new model
  - E.g. roll out to Vietnam first, then Asia, then rest of the world



Application deployment and testing strategies (Google)

A/B testing

- New model alongside existing system
- A percentage of traffic is routed to new model based on routing rules
- Control target audience & monitor any statistically significant differences in user behavior
- Can have more than 2 versions



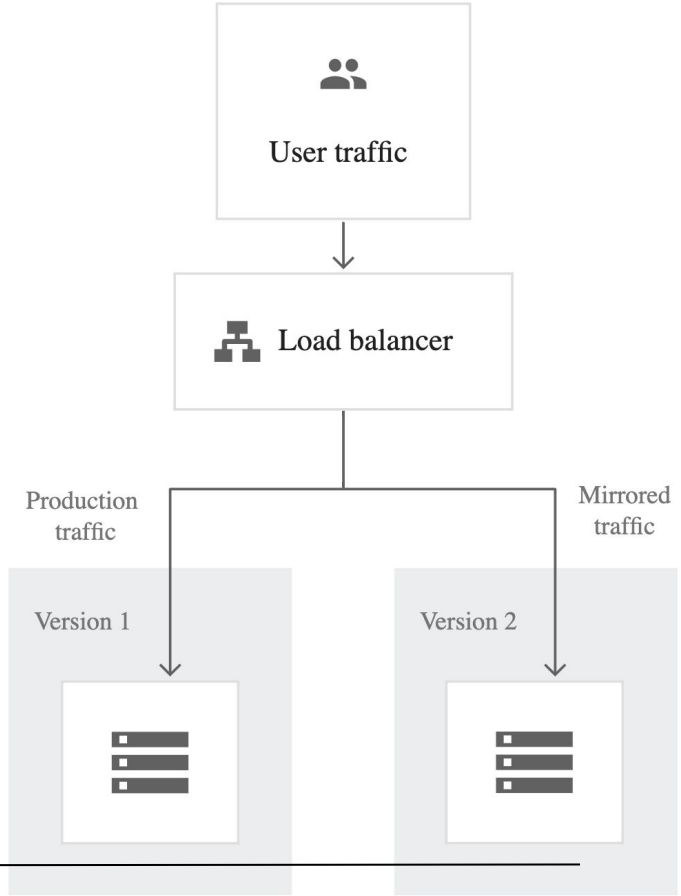
Application deployment and testing strategies (Google)

Interleaved experiments

- Especially useful for ranking/recsys
- Take recommendations from both model A & B
- Mix them together and show them to users
- See which recommendations are clicked on

Shadow testing

- New model in parallel with existing system
- New model's predictions are logged, but not show to users
- Switch to new model when results are satisfactory



Application deployment and testing strategies (Google)

Test internally first

- Use features even as they’re in development
- Share internally before externally

⚠️ You and your coworkers are not typical users ⚠️

OpenAI API workshop

Andrew Mayne

Andrew Mayne

Creative Applications @ OpenAI

GPT-3 is a prediction machine: It’s trying to predict what should come next based upon what it has learned about the world through text.

A prompt is text we send to GPT-3 for it to make a prediction about what should come next.

GPT-3 isn’t programmed to do any specific task. It can perform as a chatbot, a classifier, a summarizer and other tasks because it understands what those tasks look like on a textual level

How to create a prompt

1. Define the problem you're trying to solve (classification, Q&A, creative, etc.).

2. Ask if there is a way to get a solution with zero examples.

3. If you think you need examples, go to step 2 and think really, really hard.

4. Think of how someone might encounter this request in the world of text and write a prompt based on that.

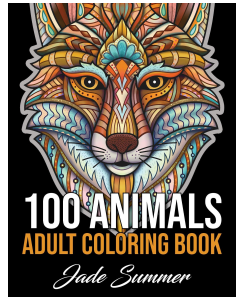
5. If you need examples, use as few as possible and add counter examples.

1

2

9

4



5

3

6

7

8

10

Machine Learning Systems Design

Next class: Distributed training & framework tutorials