PROCESS

Requirements Workshop

PROviding Computing
solutions for ExaScale
challengeS

Use Case #1:

EXASCALE LEARNING ON MEDICAL IMAGE DATA

Use Case Motivation:

- Everyday 3 women die in Israel for breast cancer
- 10 million women in Kenya go to mammography, 2 physicians are available on average to inspect them
- The disagreement between pathologist is generally really high, with 85 % false positive rate

Use Case Goals:

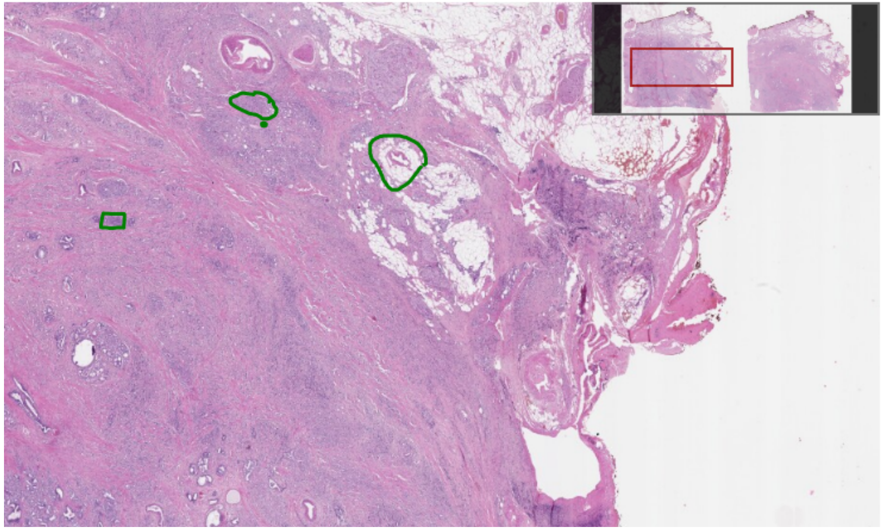
- Develop high performance image analysis algorithms
- Increasing dataset sizes, models complexity
- Improve current performances and decrease the turnaround time of experiments

What is it like to work on medical images?

- Image types can be different

What is it like to work on medical images?

- Image types can be different



Whole Slide Images :
Different Resolution Levels
~ 100K x100K (gigapixel)

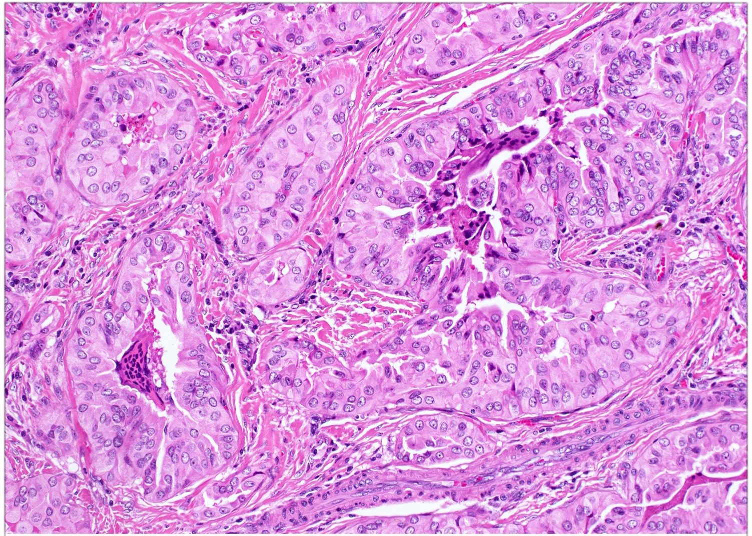
Annotations:
XMLs, CSVs, TXTs..

One WSI may occupy up to 5 GB
Need for specific tools to process data, e.g. OpenSlide, ASAP

Storage requirements:
Camelyon17: 1000 WSIs, > 3TB database

What is it like to work on medical images?

- Image types can be different



Papillary thyroid carcinoma:
giant cells are typically found within a lumen of papillary structure (H&E, ×20)

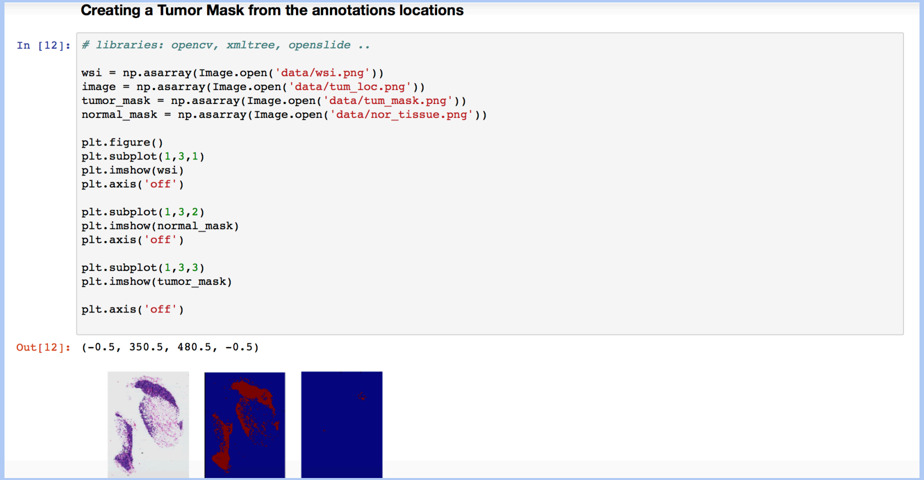
PubMed Central Images :
Low Resolution
Multiple formats

Annotations:
Natural Language Processing of image captions with Deep Learning

Approx. 5 million images

What is it like to work on medical images?

DEMO



Creating a Tumor Mask from the annotations locations

In [12]:
```
# libraries: opencv, xmltree, openslide ..

wsi = np.asarray(Image.open('data/wsi.png'))
image = np.asarray(Image.open('data/tum_loc.png'))
tumor_mask = np.asarray(Image.open('data/tum_mask.png'))
normal_mask = np.asarray(Image.open('data/nor_tissue.png'))

plt.figure()
plt.subplot(1,3,1)
plt.imshow(wsi)
plt.axis('off')

plt.subplot(1,3,2)
plt.imshow(normal_mask)
plt.axis('off')

plt.subplot(1,3,3)
plt.imshow(tumor_mask)

plt.axis('off')
```

Out[12]:
```
(-0.5, 350.5, 480.5, -0.5)
```

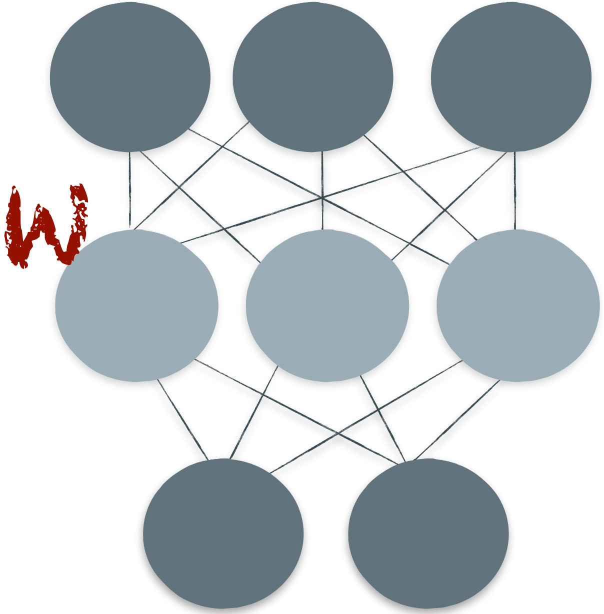
What is it like to work on medical images?

Average Statistics:

- *“Deep Multimodal Case-Based Retrieval for Large Histopathology Datasets”*:
  - 2000 patches per WSI x 267 WSIs = 530 K patches
- In principle, one could cover the whole WSI are with around 66K patches..
  - 66 K patches per WSI x 500 WSI* = 33 M patches
- When applying data augmentation more patches could be generated to scale the dataset of 10 times or even more…

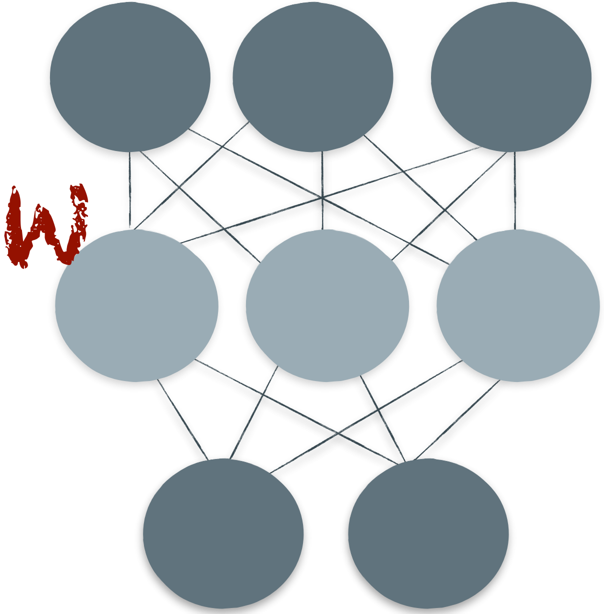
* Camelyon17 training WSIs

Deep Learning requires access to GPUs



$$W = \begin{pmatrix} w_{1,1} & . & . & . \\ . & w_{i,j} & & . \\ . & & & . \\ . & . & . & w_{m,n} \end{pmatrix}$$

Deep Learning requires access to GPUs

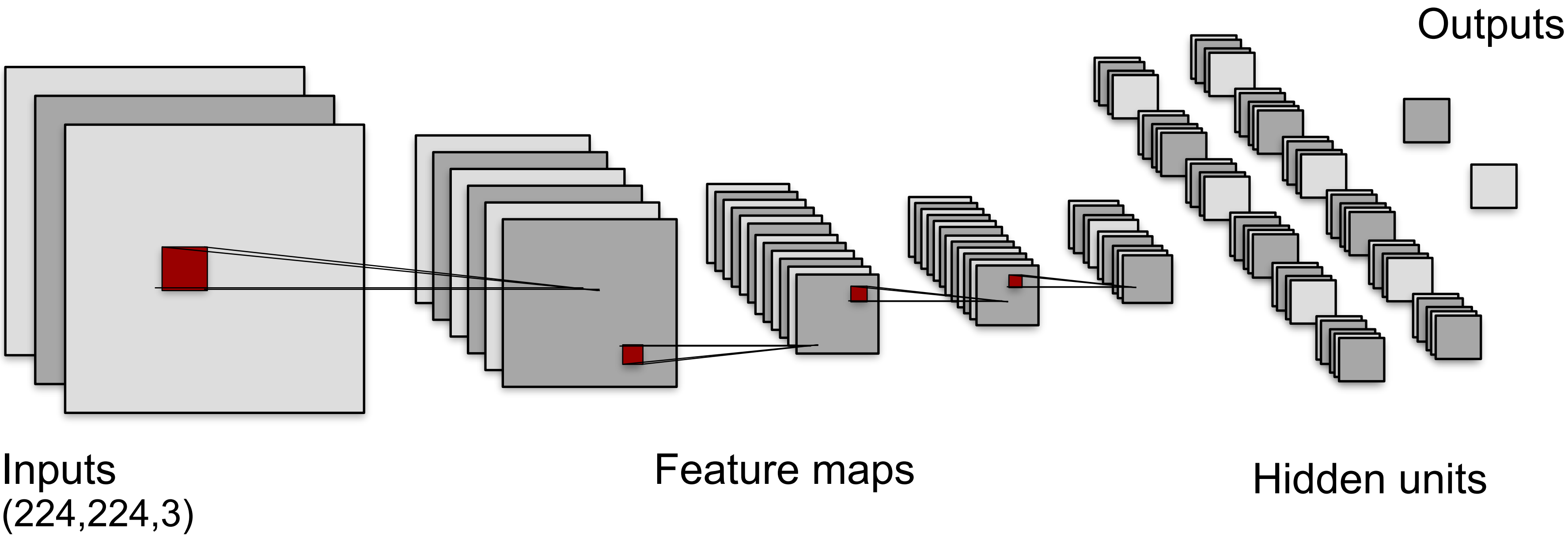


$$W = \begin{pmatrix} w_{1,1} & \cdot & \cdot & \cdot \\ \cdot & w_{i,j} & & \cdot \\ \cdot & & & \cdot \\ \cdot & \cdot & \cdot & w_{m,n} \end{pmatrix}$$

2.7 Million of parameters

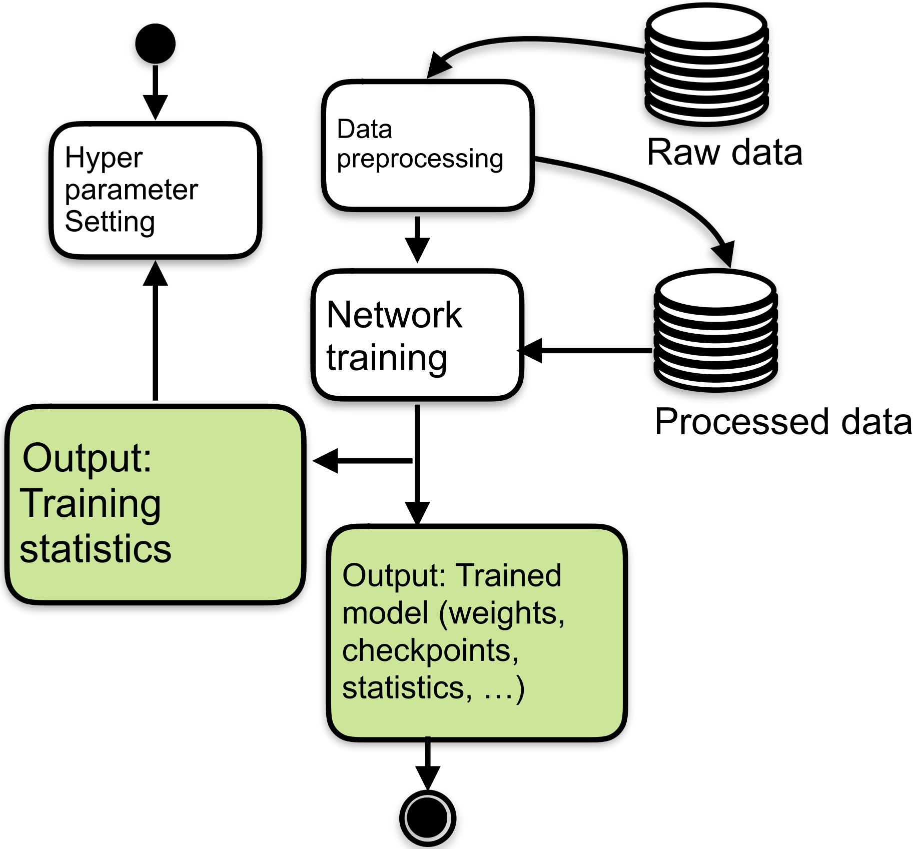
300 seconds per epoch

CNN training is highly parallelisable in GPUs



Training requires mainly one CPU thread, high RAM occupancy, high communication bandwidth between CPU and GPU memory

Network training workflow


Hyper parameter Setting
Data preprocessing
Raw data
Network training
Processed data
Output: Training statistics
Output: Trained model (weights, checkpoints, statistics, …)

System Requirements:

- Openness to programming languages, tools, frameworks

→ Virtual Machines, Docker Containers

- Flexibility in the building, deployment and management of running applications

→ Need for a **PROCESS Environments Manager**

Software Requirements:

- Deep Learning software and GPU drives

CUDA, NVIDIA, CuDNN
Tensorflow, Keras, Theano, PyTorch, …

- Support of Medical Imaging tools

OpenSlide, ASAP, DICOM, ..

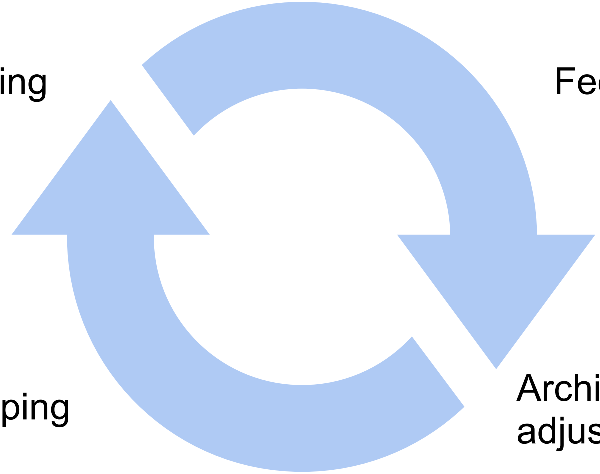
- Environment for Python development

Hardware Requirements:

- Access to GPUs for network training
- Access to CPU clusters for data preprocessing, data postprocessing, network testing
- Large RAM consumption
- High Caching to reduce the number of I/O operations
- Need for a **PROCESS Data manager**, imitating an extension of the local datacenter, although distributing the data sources.

! Data should be accessible from both CPUs and GPUs

Development should be ***iterative***



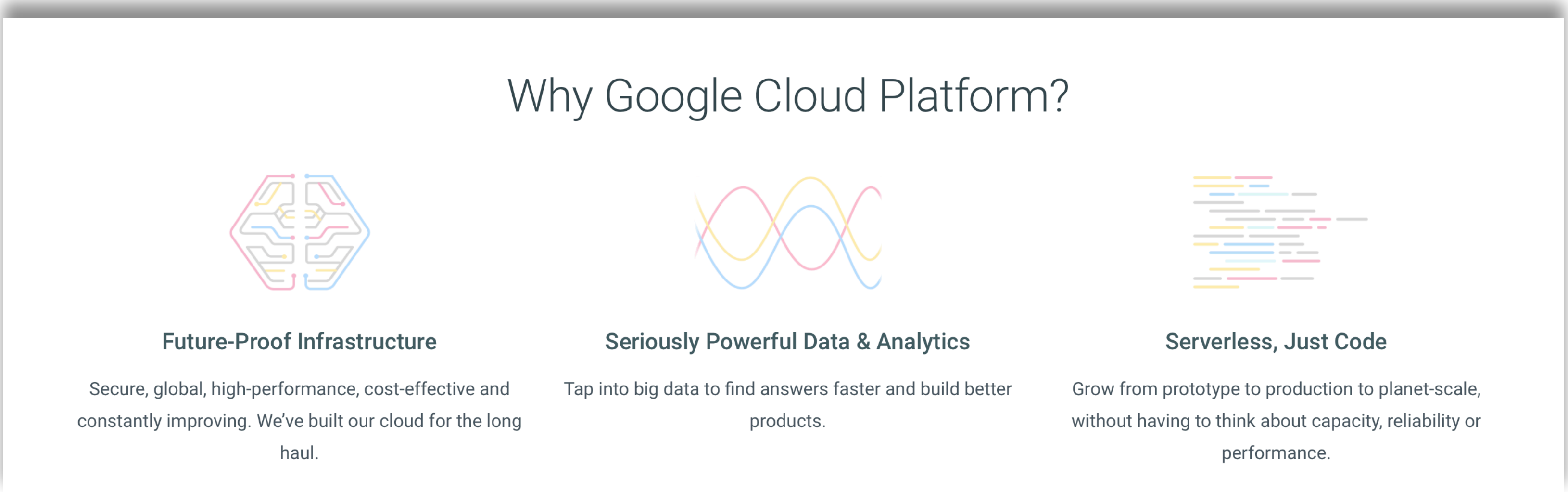
Testing

Feedback

Architecture adjustments

Prototyping

Flexibility is key in Amazon Cloud, Google Cloud, Microsoft Azure



Why Google Cloud Platform?

Future-Proof Infrastructure

Secure, global, high-performance, cost-effective and constantly improving. We've built our cloud for the long haul.

Seriously Powerful Data & Analytics

Tap into big data to find answers faster and build better products.

Serverless, Just Code

Grow from prototype to production to planet-scale, without having to think about capacity, reliability or performance.

Questions?

Thanks!