PART I – AI TECHNICAL CONSIDERATIONS
CH.6 – DATA STORAGE, CLOUD USAGE AND AI PIPELINE

P.M.A van Ooijen[1,2] **Erfan Darzidehkalani**[1,2] **Andre Dekker**[3]

[1]Machine Learning Lab, Data Science Center in Health (DASH),University Medical Center Groningen, University of Groningen, Hanzeplein 1, the Netherlands

[2]Department of Radiation Oncology, University Medical Center Groningen, University of Groningen, Hanzeplein 1, Groningen, the Netherlands

[3]Department of Radiation Oncology (MAASTRO), GROW School for Oncology and Developmental Biology, Maastricht University Medical Centre+, Maastricht, The Netherlands

{e.darzidehkalani,p.m.a.van.ooijen}@umcg.nl

November 12, 2021

ABSTRACT

Artificial Intelligence, and especially Deep Learning, requires vast amounts of data for training, testing and validation. Collecting these data and the corresponding annotations requires the implementation of imaging biobanks that provide access to these data in a standardized way. This in turn requires careful design and implementation based on the current standards and guidelines and complying to the current legal restrictions. However, just the realization of proper imaging data collections is not sufficient to train, validate and deploy AI as resource demands are high and requires a careful hybrid implementation of AI pipelines both on-premise and in the cloud. This chapter aims to help the reader when technical considerations have to be made about the AI environment by providing a technical background of different concepts and implementation aspects that are involved in data storage, cloud usage and AI pipelines.

Keywords Cloud Computing, Artificial Intelligence, Federated Learning, Distributed Learning, Data Storage, Biobank, Imaging Biobank, AI Pipeline.

1 Introduction

It is a well-known fact that Artificial Intelligence (AI) performance depends heavily on the availability of data and the corresponding annotations or labels that define the ground-truth about the (clinical) question at hand. To obtain this data they need to be made available by multiple institutions geographically distributed all over the world. One option is to centralize data which not only requires de-identification of this data, but also an informed consent that includes the possibility of wide distribution and sharing of the acquired data [1]. A second option is to apply federated learning principles in which data does not have to move but does require wide distribution of the AI application and computing resources [2]. In both options, the gathering of large collections of imaging data of the same pathology or disease and of a pre-defined population is a challenging task and many projects suffer from the limitations of a small dataset. Not only because of the limited availability of the data but also because the willingness of healthcare institutes to share medical data is low. To tackle this problem and to enable the collection of large imaging databases, careful technical considerations are required concerning the IT infrastructure. Three important parts of this IT infrastructure that can help driving AI forward are data storage, cloud usage and AI pipeline implementation. This chapter will dive into these three

larger topics by addressing more specific topics such as imaging biobanking, cloud storage and compute, federated learning, and different implementation versions of the AI pipeline in medical imaging.

2 Data storage

In 2010 the Quantitative Imaging Network (QIN) performed a questionnaire among their members to explore the field of informatics methods in imaging research [3]. One of their goals was to share data among institutes in order to accelerate quantitative imaging research. Important findings were a large variation in tools used for the de-identification, local image file storage was varying from XNAT and (commercial) Picture Archiving and Communication System (PACS) solutions to open consumer platforms such as Dropbox and local image meta-data databases varied from dedicated tools like RedCAP to simple spreadsheets on a local hard drive or USB stick.

2.1 Imaging Biobanking

According to the ESR position paper on imaging biobanks, a biobank is an "Organised database of medical images and associated imaging biomarkers (radiology and beyond) shared among multiple researchers and linked to other biorepositories" [4]. The key points resulting from this work were:

- Imaging biobanks are "shared databases of imaging biomarkers, linked to biorepositories"
- Exploitation of traditional and imaging biobanks is meaningful for "personalised medicine"
- A European imaging biobank network would significantly boost research in the imaging domain

The immediate purpose of an imaging biobank is to allow the generation of imaging biomarkers for use in research studies (either using 'conventional' techniques or AI/Deep Learning) and to support biological validation of existing and novel imaging biomarkers. The long-term scope of imaging biobanks is the creation of a network/federation of such repositories. An imaging biobank can exist in different scenarios. First, an imaging biobank could contain clinical research data gathered in clinical research/trials. Second, it could contain disease-specific data with data from clinical practise or screening programmes (e.g. breast, lung and colon cancer) based on disease characteristics. These data collections are not necessarily directly connected to a clinical research question. Third and final, an imaging biobank can contain general population data. In this case data is collected from the general population (so not only patients or population at risk), with no specific goal or disease-oriented approach. This usually involves the collection of long-term longitudinal data.

Imaging biobanks can exist as a single entity with centralized data collection or storage, but federations of biobanks are also possible in which data or research questions can be exchanged. Such a federated setup of imaging biobanks requires a centralized database or catalogue of data collections present at different local imaging biobanks [4] preferably using Findable Accessible Interoperable and Reusable (FAIR) data principles [5]. Utilization of the data in such a federated environment could involve collecting the imaging data at the site where they are needed when needed, but also to have the data remain at the site of acquisition or collection and allowing software tools to access that data to perform analysis with only returning the analysis results [6].

To allow the merging of (imaging) data collected from multiple biobanks, a common FAIR data model is required. This common data model should include common data schemes, standard nomenclatures, reference to common ontologies, etc. Furthermore, access policies should be implemented to request and grant access to data collections for specific research projects.
The ESR Position paper on Imaging Biobanks [4] lists a number of requirements for the appropriate implementation and use of imaging biobanks. First, the aim should be to achieve at an eco-system with a combination of central and federated approach where query, analysis and retrieval across multiple data repositories should be possible. Second, data access should be secure and permission based. Third, proper de-identification of both imaging and meta-data should be in place. And finally, standard information models and terminologies should be used. To achieve part of this standardization the Quantitative Imaging Network proposed a system architecture to enable data sharing, collaborative experimentation and translation of methods to clinical practice [3].

This system architecture proposes a four-layer design. It includes informatics tools to support data repositories in a data storage layer (in blue) based on standard information models and shared semantics (in red).

The data repositories in the storage layer can be accessed by different processing and analysis in the Research Methods layer. Nowadays this layer is of course extended by the training, validation and deployment of AI systems. Finally, the clinical systems layer provides the clinical users with the tools to get access to the data in the underlying layers including the results from the algorithms in the research methods layer. Although this model was proposed in 2012 it still holds and provides a clear picture of the information use in the imaging domain .

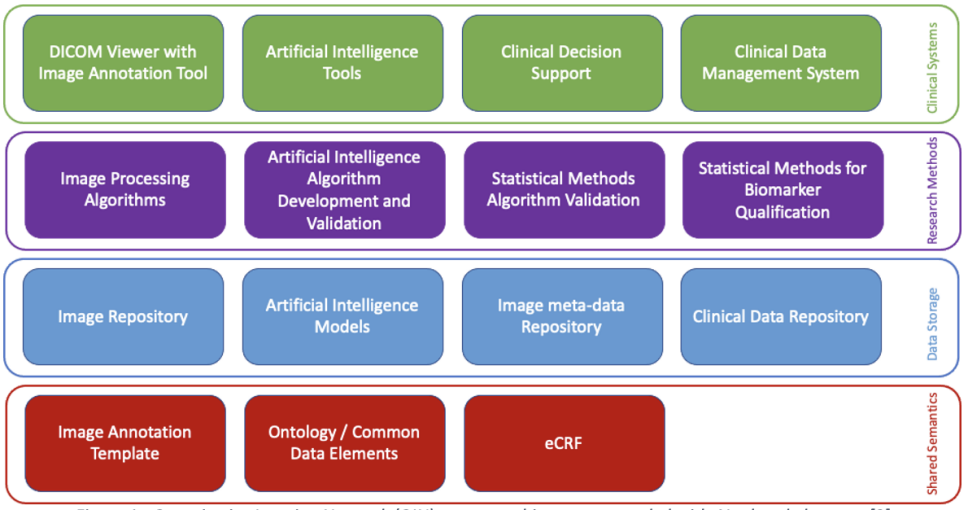

Figure 1: Quantitative Imaging Network (QIN) system architecture expanded with AI related elements

Quite a large body of work exists on the design and implementation of data environments for medical imaging, both for research and clinical practice [6] [7] [8]. Commonly used open source tools are XNAT (imaging) and RedCap (meta data), but also other environments and implementations both commercial and open source [6] [7] exist. One common denominator of the environments is that they can all operate in the 'cloud' of the world wide web. Often mentioned advantages of cloud usage are economy of scale, improved performance, data portability, increased and flexible storage capacity, data migration, and patient-centric connected systems [9].

2.2 Data storage: Challenges and Considerations

In order for AI pipelines to work properly, data must be annotated which introduces the challenge of obtaining a proper ground truth. Ground-truth images in imaging biobanks are discrepant most of the times as medical experts are not fully concordant in their annotations.
Also, the quality of the data collection is vital, and several aspects have to be considered when collecting and storing data. In medical imagining domain, numerous scanners, imaging protocols and parameter choices exist, which results in images of significantly different distributions. It is important to store the provenance of the images, i.e. by which protocol, scanners, parameters they were obtained, whether they contain missing slices or artefacts etc. It enables AI algorithms to decide what data to query and to avoid improper samples. As an example, motion corrupted MR images cause substantial problems in AI algorithms for diagnosis or segmentation [10]; whereas they are useful for motion-correction AI models.

3 Cloud usage

The National Institute of Standards and Technology (NIST) defines cloud computing as follows

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [11].
In the NIST special publication on cloud computing, five essential characteristics, three service models, and four deployment models are defined (figure 2) [11] [9].

3.1 Essential Characteristics

he first part of the NIST cloud model are the Essential Characteristics, which are on-demand self-service, broad network access, resource pooling, rapid elasticity and measured service. A cloud implementation should include all of these characteristics.

On-demand self-service is about the ability of a consumer or user to provision the required computing capabilities (such as server time or network storage) by themselves on-demand and automatically without requiring human intervention from the server provider side. Next, broad network access means that easy access should be provided through network

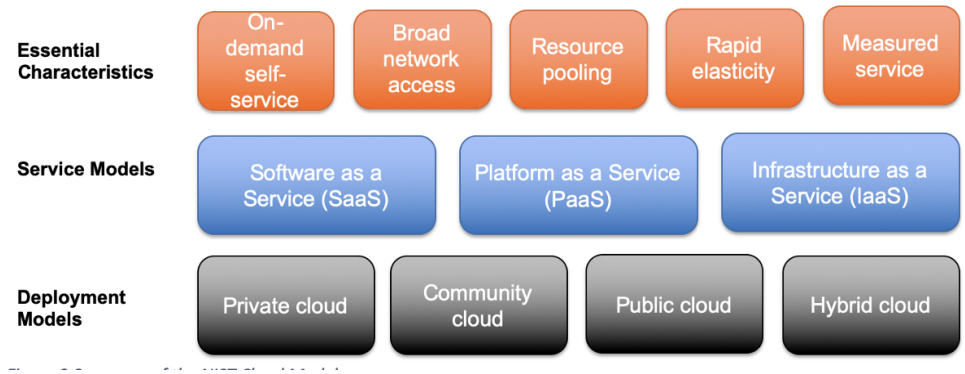

Figure 2: Summary of the NIST Cloud Model

access by using standard mechanisms. This broad access also refers to the requirement that it should be accessible through a wide variety of different thin or thick client platforms ranging from smart-phones and tablets to laptops and workstations.

The characteristic of resources pooling refers to the fact that the provider is running pooled computing resources that can be used to serve multiple customers using a so-called multi-tenant model. The assignment of virtual resources is dynamically executed based on the requirements of the consumer. This also means that the user is not knowledgeable about the exact location of the provided resources (e.g. which server or rack is providing the resources). However, because of legal restrictions that might occur most providers do allow a user to select a designated location on a higher abstraction level to limit the resource location to for example a certain country, state or datacenter. The resources that can be provided and dynamically assigned by the cloud provider are multiple and include storage capacity, CPU and GPU processing, memory, and network bandwidth.

The term rapid elasticity is used to describe the characteristic of the cloud that rapid on-demand scaling of resources is possible within the virtual environment of the consumer. This scaling could even be done automatically based on certain triggers when additional resources are needed. This kind of scaling gives a sense of unlimited resources available to the user.

3.2 Service Models

The service models describe three different variations of cloud services that provide software, platform or infrastructure through cloud access.

Software as a Service (SaaS) In short, Software as a Services, or SaaS, deals with running specific applications through a cloud service. This means that a provider provides access to a single application through the cloud service. Al management and control of the underlying infrastructure such as the servers, operating system, storage, etcetera is fully controlled by the provider of the service without any influence on this by the user. A typical example is a web-based e-mail service such as Gmail.

Platform as a Service (PaaS) Expanding on the SaaS, a Platform as a Service (PaaS), provides a suite of applications, programming languages and other user tools through a cloud service. This means that a user is able to deploy required applications either own or acquired to a cloud service. The underlying cloud infrastructure is still fully controlled by the cloud provider, but the user has influence on what applications should be deployed in the environment.

Infrastructure as a Service (IaaS) Finally, Infrastructure as a Services (IaaS) provides the most flexible solution by providing access to fundamental computing resources. In this service model the underlying cloud infrastructure is provided by the cloud provider, but the user can control and configure the operating system, storage, and application deployment. Also limited control over selected networking component, for example hosting a firewall, could be granted to the user.

3.3 Deployment Models

The Deployment Models proposed by NIST range from Private, via Community to Public cloud and also allow for Hybrid solutions combining private and public cloud technology. The private cloud is the most closed – and assumed most secure – deployment model. In this model a single organization will be granted exclusive use for multiple

consumers belonging to that organization. This does not mean it needs to be owned, managed, and operated by that organization and all hardware must exist on-premise. Third parties could be involved in one or multiple of these points.

In case of a community cloud the use is not exclusive to one organization but to multiple organizations that are part of a community. In most cases, such a community will exist of a group of organizations with shared concerns such as, for example, a group of hospitals in a region or country. Again, ownership, management and operation can be shared with a third party and the actual hardware may be on or off-premises.

A public cloud is a cloud infrastructure that is open for use by the general public. This will, in most cases, physically exist on-premise with the cloud provider which can be single or combination of a number of companies, universities and/or government organizations.

In a hybrid cloud the cloud infrastructure is deployed consisting of a combination of cloud models. The different environments are unique entities but have the ability to share and exchange data and applications.

4 AI pipeline

Because of the complex nature of the process of AI and the different steps involved, research is ongoing to develop more automated and integrated AI pipelines that support the whole process from data collection to deployment [8]. Depending of the situation, each step in such an AI pipeline is prone to automation thus eliminating as many of the manual steps as possible. In most cases, there is a distinction between a development and a deployment pipeline. The development pipeline consists of the steps of data collection, data annotation or labelling, data cleansing, training and validation. In a deployment pipeline one or more trained networks will be included to receive (imaging) data and produce a final prediction which again needs to be integrated into the (clinical) database [8].

4.1 Local implementation

When implementing AI locally it can be part of the current PACS or EMR environment or run as a private cloud or on-premise service.

4.1.1 Networking

Local networks consist of data storage systems, processors and deployment systems. These components are interdependent, i.e. each system is unable to operate unless it receives response from other systems. This interdependency makes communication a crucial factor in developing AI pipelines. The ultimate goal of a local network should be to minimize latency, and to synchronize the processing among different nodes. If communication in a network is sub-optimal, the network with its costly hardware resources becomes under-utilized and inefficient. Several techniques can be considered to enhance communication efficiency in a locally implemented network. A first effective technique in high-performance computing is reducing model precision. In many cases, data and model parameters are stored with double precision, which could be converted to single floating point or less. Since medical data are not required to have double floating-point precision, this conversion does not hurt the model performance. Removing additional integers could help saving bandwidth, reduces model size and network burden. [12] Another useful technique to improve data exchange speed is compression of data transferred between different nodes. This is especially important in (medical) infrastructures where network bandwidths are limited. Some compression techniques utilize lossless methods, i.e. data is fully recoverable after decompression, but have limited capacity to save bandwidth usage. Instead, lossy methods are utilized more frequently. One method in deep learning based AI applications to perform compression is to limit the values of gradient updates to binary values, this method ensures gradient updates in the correct direction and transferred data will be as low as just one bit. Another technique is to neglect insignificant, small gradient update values, and convey only significant gradients. Tao et al implemented this method for federated systems, which is applicable in medical AI pipelines [13]. An AI pipeline consisting of multiple components requires each component to participate actively, exchange data and allow access to other components when necessary. But most PACS systems are unable to operate within an active environment. They might for example fail when changing IP, hostname and DICOM attributes, since these are hardcoded to work on a pre-defined situation with limited options and minimum flexibility [14].

4.1.2 GPU and CPU

Early AI pipelines and deep learning deployments used CPUs to perform computations. AI pipelines were dependent on clusters of CPUs to parallelize computing and improve model efficiency. However, state-of-the-art deep learning pipelines are nowadays relying on GPUs instead of CPUs. For many medical institutes acquiring GPUs is a crucial step in deploying AI pipelines, for both local and cloud-based implementations. To build an AI pipeline, both CPU and

GPU nodes are needed, and tasks are divided between them to reach the optimal point of cost-efficiency. Since GPUs maximize performance, not latency, they have worse latency than CPUs. Hence, GPU nodes are mostly used in the training phase, and CPU nodes are mostly used in the deployment phase when the latency becomes an important factor. Thus, when building large local sites for AI in healthcare, a combination of GPU and CPU devices is preferred.

4.1.3 Data Management

Having high volumes of data requires careful data preparation, storage, processing and exchange between systems when developing local AI pipelines. Data should also be well integrated into existing hospital IT systems in both development and deployment stages.

Developing DL models on existing hospital image archiving systems requires a good integration. This interoperability may be hampered by the co-existence of multiple imaging databases (PACS) in the hospital environment. During development, the DL model can be trained on a data dump of DICOM images to avoid using PACSs in the process. After deployment, it could be needed to ensure all systems within a local/cloud network have the same PACS version, or that PACSs are connected with an intermediary web-based protocol which allows to integrate data from different PACS systems.

AI systems, similar to clinical experts, require various types of patient data to obtain comprehensive knowledge about a medical situation. For example, in radiology, additional patient EMR data might be required, and AI models cannot be trained properly until they integrate all relevant data. This requires data exchange between different departments within the same hospital. Protocols and data type standardizations should thus be developed for all the relevant departments to facilitate information exchange.

4.1.4 AI Models

In addition to training datasets, trained AI models should also be stored in the medical IT environment. AI researchers might need future access to the trained models for testing, deployment, enhancement, and migration to a new framework. Thus, models have to be accessed, queried and analysed easily through hospital systems. In larger networks, model interoperability is another concern. Each system might have its own DL framework, which is a burden to develop an AI pipeline. Research is being done to facilitate the interoperability of DL models trained on different frameworks, by introducing standardized exchange formats. One popular framework is ONNX [15], which enables various DL frameworks, including Pytorch, Tensorflow, Caffe, MXNET, and CNTK to share their model properties and co-operate within the same network.

For AI pipelines implemented locally, there is a risk for AI models to be prone to various kinds of bias, including demographic bias (gender, age) or data bias (annotation, equipment, acquisition), thus having limited capacity for generalization. Solutions include using distributed datasets, and transfer learning. Current DL models might be designed to work with limited data types (e.g. one modality or annotation protocol). However, with current pace in imaging techniques, any change has to be expected in model design. Models are required to handle new data formats, with minimal interruption. Thus, it is better to avoid static AI models for medical usages, and design flexible models.

4.2 Cloud implementation

4.2.1 Development phase

In the development phase of the AI pipeline, the cloud can be employed to provide compute power (cloud-based learning), data science workspaces, crowd-sourced annotation, or distributed learning. The process of annotation or labelling of imaging data can be quite time-consuming and difficult to organize because of the low availability of busy healthcare professionals that are needed for expert annotation. Cloud implementations could help in such cases to provide an online annotation environment where experts from all over the world can provide their annotations on the imaging data [7]. Important in such a process is to have a transparent annotation procedure in place which is well documented and uses guidelines to ensure the annotation from different users is uniform [1].

With cloud-based learning, the cloud is used to train the deep learning network by utilizing cloud-provisioned CPU or GPU computation power. The programming and implementation of the deep learning network is done locally and the High-Performance Computing (HPC) environment in the cloud is used to perform the iterative learning process.

Data science workspaces provide access to a full data science environment and allow all development steps to be performed in the cloud. In this case the full programming environment is provided through a virtual machine in the cloud and access to the HPC cloud facilities is also provided. Because of the move to the cloud it has also become feasible to not limit the learning phase in the development to a single location but to distribute the work over multiple

locations and even with different databases at those different locations. This distributed or federated learning is a novel area that could solve issues concerning the limited sharing of data and data privacy, it will be described in more detail separately in the next section.

4.2.2 Deployment Phase

The deployment phase introduces possibilities such as a single AI algorithm in the cloud, compute power in the cloud, federative deployment of AI and AI platforms [16].

Many AI applications are provided by the vendor as a single AI algorithm in the cloud. This means that the imaging data will be sent to a cloud-based DICOM receiver from the PACS or an upload through a web interface is done. After processing of the data, a report or other generated output (e.g. segmentation files, processed imaging data, etc) will be returned to the user uploading the data. In a 2020 report, Mehrizi et al. showed that of 269 AI applications on the market 32 % were offered as a cloud- only solution and 46% offer a choice to be either cloud-based or on-premise [17]. Besides commercial implementations, these kind of cloud-based solutions for online AI applications are also available from research institutions in order to support research. The RECOMIA platform is an example of such a research-based AI application provider that also aims to expand and function as a research marketplace for AI applications developed by other research groups [7]. In the deployment phase, compute power in the cloud could also be used if the required HPC facilities are not available on-premise.

Several federative deployments of AI can be envisioned. One possible solution is the one proposed by the Early Lung Imaging Confederation (ELIC) [6]. They proposed and tested a federative environment where imaging data is distributed over the world at the locations where the data were obtained. A central database is updated on the content of these different systems. Software tools (AI or other) that perform tasks on the decentralized images will be distributed to the relevant nodes in the federative network where automated image analysis is performed. All resulting quantitative measurements are collected on the central HUB where further analysis can be done on the aggregated database.

AI platforms can also provide a centralized environment to manage and run multiple applications that can be accessed by the user through some sort of online marketplace [16]. In such a cloud-based marketplace of AI tools, developers can distribute their tools (after approval by the marketplace holder) to a large customer base without the requirement of their own sales and deployment team [16]. The deployment only needs to be done in the environment of the marketplace, saving the costly and tedious in-hospital connectivity challenges. Also, all workflow and user-interaction is covered through the platform, ensuring that the AI tool developer only needs to focus on the core technology and the API connection to the platform.

For the user, such a platform removes the burden of installing a multitude of different AI applications all performing their own task, application selection and implementation of AI is also simplified to a large extent. The platform will handle the data transfer and ensures the right data is sent to the right application and ensures the data transfer security. The organisation using the platform only has the concern itself with the configuration and implementation of the required data connections once. When connecting to the platform, all AI tools will rely on that single data connection to obtain data. Furthermore, maintenance and future changes will also be limited to that single platform connection (e.g. when migrating to a new PACS or changes in imaging modalities) and updates and upgrades of the AI tools themselves will be handled by the platform provider. The AI tools provided can either be from one specific vendor or from a variety of suppliers including scientific institutions that provide access to research outcome. The platform provider will select AI tools based on pre-defined criteria concerning the legal status of the tools and the quality of the results. A variety of these marketplaces already exist, in their technography study Mehrizi et al. reported 10 of such marketplaces that provide access to other applications in 2019 [17].

5 Distributed / Federated Learning

Training deep learning models requires finding optimal values for millions of parameters, and this is time-consuming. Sometimes it is important to distribute the processing on parallel machines, to reduce the training time and improve network efficiency. Besides, there might be privacy considerations and challenges regarding data governance. In such cases, distributed/federated deep learning allows researchers to develop an AI pipeline, without having direct access to data from other institutions.

5.1 Parallelization methods

Data parallelization In data parallelization, each worker (e.g. GPU machine) is given an identical copy of a DL model, and part of the data. To avoid duplication of tasks, data should be split into non-overlapping batches (figure 3). After

each model finished its processing, the DL parameters for all workers will be updated. Hence, a strong synchronization among workers is required.

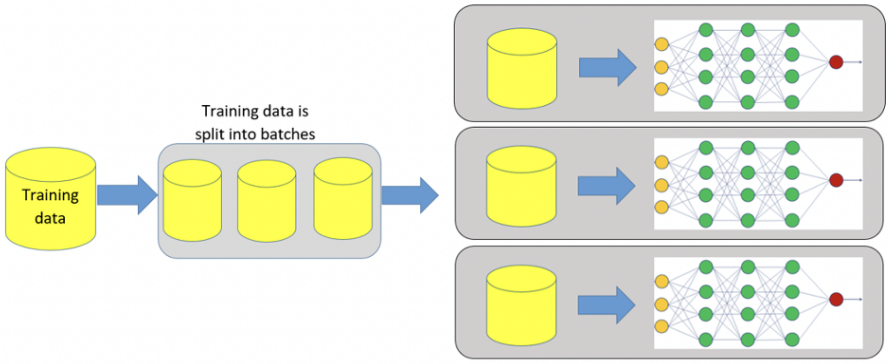

Figure 3: Data parallelism

Model parallelism In model parallelism, processors are parallelized to perform operations of a certain part of a deep neural network (figure 4). For example, a neural network can be divided into multiple sections, each processor assigned to one section, and processors exchange information in both forward and backward propagation. A major concern in model parallelism is how to split the network into multiple sections so that the processors work together properly.

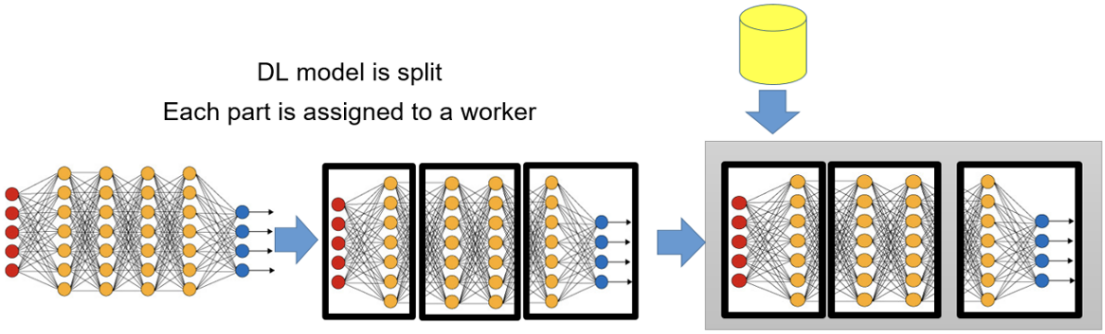

Figure 4: Model parallelism

Pipeline parallelism is a combination of model and data parallelism and has the benefits of both methods. In pipeline parallelism, data is divided into non-overlapping mini-batches, and neural network is divided into sections and each section is assigned to a processor (figure 5). In backpropagation, the gradient of a certain mini-batch is propagated through layers and gradient processing is performed on GPUs, associated to those layers.

For more complex networks that consist of many layers and complex network architectures, dividing the network or data to distinct parts might not be feasible or reasonable. Thus, large distributed deep learning projects which deal with complex networks, mostly apply Hybrid parallelism - a combination of model, data and pipeline parallelism.

5.2 Synchronization

One of the most important questions in distributed deep learning is when to synchronize parameters among workers. Synchronization strategies exert a trade-off between network speed and performance and is challenging.

Synchronous training, workers share their updates after they processed one batch (i.e. after each iteration). This method is widely used in distributed deep learning platforms and has better convergence that other synchronization paradigms. However, synchronous training has the straggler problem; the network speed will be determined by the speed of the slowest worker, and other workers have to wait until the last worker finishes its processing.

Bounded synchronous training, workers train on stale parameters, and parameters resulting from bounded-synchronous training are an approximation of the parameters of synchronous training. However, to prevent that the approximation

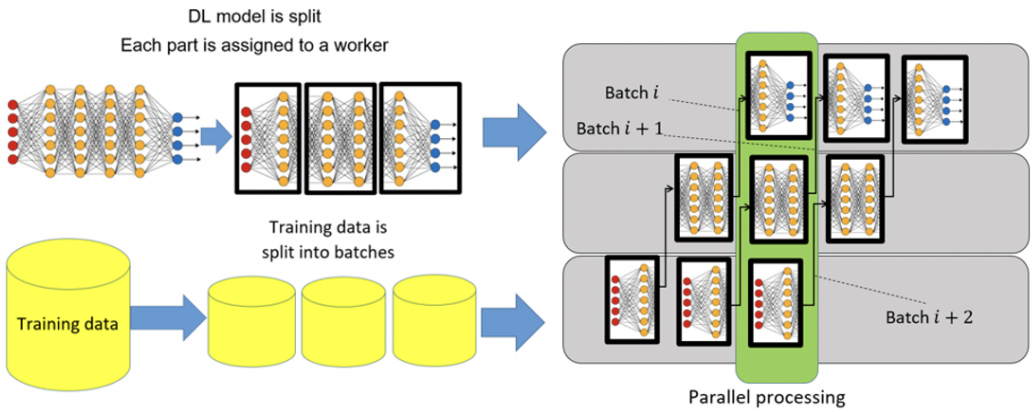

Figure 5: Pipeline parallelism

harms the model accuracy and convergence, the staleness should be bounded. This allows more freedom of the model, and better efficiency, while avoiding the straggler problem. [19] Another way to update a model is by updating parameters independently or asynchronous, i.e., the model is updated once one worker finishes its batch and sends the gradients to the network, regardless of other workers. This approach might cause problems in convergence. However, the model will be considerably more flexible and training proceeds faster.

5.3 System architectures

Having a large number of workers, another major challenge in designing a distributed deep learning architecture is how to synchronize parameters among different workers. Many inter-connected workers exchange considerable amounts of information with each other. The performance of each processors is dependent on the parameters it receives from other processors. Thus, if the system architecture is not designed properly, failure in one worker causes interruption of the whole system. Different system architectures will be discussed in this section.

Centralized architectures are architectures in which every node or worker reports its parameters to a central node, (or nodes), called parameter server (PS). One approach is to divide the model parameters into a few chunks and send parameters belonging to each chunk to a PS. Thus, letting model processors and PSs work in parallel. Notable systems using PS architecture are GeePS [20], DistBelief [21], and Tensorflow [22].

Decentralized architectures work without a parameter server. Processors in decentralized networks, communicate directly with each other (in a fully connected network), or communicate through other processors. In a fully connected network, in which every single worker is connected to all other workers, communication is a major concern. However, alternative topologies exist. For example, a ring topology is implemented in the widely used Horovod [23] framework from Uber. The major drawback of all topologies other than fully connected topology is that communication between nodes might require involvement of the other nodes and processing time is substantially increased.

Decentralized topologies are much easier to deploy than centralized topologies. There is no need to set-up a PS and complexities of PS planning are avoided. Another advantage is that decentralized topologies are more robust to failure since there is no central hub on which the existence of the network depends. Instead, in decentralized networks, other workers can easily take over other failed workers' duties, and the network goes on without interruption. Decentralized architectures also have disadvantages. Communication is a major problem in decentralized networks. Changing topology from fully connected also introduces other complexities and trade-offs.

Both centralized and decentralized topologies assume that data is available for all workers, and the network data and model parameters are being controlled, either by a central server or by individual nodes. However, in some scenarios, especially in the medical domain, data and model parameters might not be visible to other workers or PS.

Federated learning topology is used in such cases, which is commonly the only available option of distributed deep learning in the medical domains. In federated learning, data is kept locally on each worker and a global network is trained based on the data stored in each server. Each worker reports the updated model parameters based on its dataset to the whole network. This topology is also beneficial for networks with limited bandwidth, because in federated learning, heavy training data is not exchanged in network and only parameters are propagated.

Common topologies of Federated learning are the aggregation server metho, peer-to-peer method, and sequential methods. In the aggregation server method, a server initializes the model between different workers, each worker computes its gradients and sends them back to the central server. After all of the workers computed their gradients, the model is aggregated and updated from the aggregation server, and the new model is sent back to continue training the next iteration.

Peer-to-peer topology avoids using a server to save the whole model. Instead, a model is initialized from one node, each worker starts training based on its own data. After each node computed its gradients, it reports the calculated parameters to all other nodes. The model is then updated after all of the workers reported their own updates.

Sequential methods, a model is trained on data from one institution and is then adapted to new institutions, with different scanners and protocols. Two forms of sequential methods are domain adaptation and lifelong learning. In domain adaptation, a model is trained on a source domain with rich data. It then will be trained again (fine-tuned) on new samples from the target domain to learn the new distribution. Since the goal of domain adaptation is maximizing performance on the new domain, the model might not be able to preserve its performance on the source domain. Lifelong learning is another variant of sequential methods, aiming to solve this issue. In lifelong learning, a pre-trained model sees a few samples from the new domain, learns the new distribution, while maintains the model's capability on previous data. This can be done by learning batch normalization parameters for different domains and sharing the convolutional filters [24].

5.4 Discussion on Distributed Learning

Deep learning models trained on large-scale datasets ensure better clinical performance, generalizability and allows to make less biased decisions. Federated learning helps to provide access to high-quality diagnosis and treatment even in remote locations and the benefit of using data from other institutions. This advantage is significant, especially when dealing with rare diseases. Furthermore, hospitals retain control over their own data, limiting concerns about data leakage or data misuse by third parties although some concerns remain such as what one might learn from the updates to the Parameter Server. For researchers, federated learning helps them to train their models based on a vast amount of diverse data from multiple institutions. This is especially important since even the strongest DL models fail when trained on inadequate data.

However, heterogeneity of data might also have major consequences. For example, the global optimal point might not be the optimal point for each institution, since each institution has a particular group of patients and has some bias in favour of that group. But if an FL framework is trained at multiple institutions, it converges to an optimal point for all parties. And some institutions might not be interested in a global optimal point since they want to serve a specific group. Federated Learning training ensures some levels of privacy, but it does not guarantee full security of the model. Some techniques can be applied, but they all add their own complexities to the model and cause trade-offs. Hence, when designing a federated learning paradigm, the desired level of security and performance should always be considered, and an optimal trade-off should be designed based on the level of trust in other institutions, legal limitations, etc.

FL models are also prone to information leakage, even though they do not share direct information from training data. For example, techniques like model inversion, gradient leakage, and adversarial attacks can extract confidential data from the model. If adversaries can track the gradient updates, for example, the values of gradients of a single institute, data from that institute can be stolen. A high level of safety, reproducibility, and traceability is critical in federated networks. In centralized and local DL models, all the data and systems are in control of a master system. This means that model history, hardware configurations, and hyperparameters are all accessible. As a consequence, faults can be debugged and tracked easily. However, in large scale federated set-ups, not all of the above properties are trackable. Each institution has its own settings, software, network, and infrastructure. One solution to ensure model traceability and integrity is to compel all parties to declare their hyperparameters and training settings.

Finally, in federated topologies, secure data transmission is a necessity. For most strict cases, an additional operator might be required as a 'trusted broker' to transmit data between nodes. It exerts an extra cost on the system, which sometimes is undesirable. Encryption of transmitted data is also crucial. Secure data transmission prevents third-parties to access network parameters.

6 Discussion

Although very promising in healthcare data exchange, cloud solutions still have to cope with fear and unease about the technology by hospital IT leadership. On the level of performance this fear is about image latency because of low bandwidth, delay in image access and cloud service disruption by downtime. But also, security issues such as DDoS or hacking attacks and confidential data leakage are major concerns.

That this is still an issue in the current cloud environments of medical imaging was shown by the Health IT Security report in November 2019: about 1.19 billion images were accessible through a very large number of PACS systems that were connected to the cloud and easily accessed by unauthorized users [25].

Security issues prominent in the cloud are secure transfer – including for example data encryption – privacy, confidentiality and integrity of the (imaging) data. The cloud design and implementation should be such that it takes these four points into careful consideration.

Besides security, cloud implementations should also consider safety by providing backup or redundant storage to avoid data loss, high availability with low downtime, and restricted and tracked access by implementation of role-based access and extensive user activity tracking and audit trails. Although the concerns about security and safety are valid and important, the need to establish imaging biobanks and integrate them with existing biobanks is of utmost importance to facilitate the development of AI-based tools and to increase quantitative, patient centred healthcare. Cloud-based imaging biobanks will help drive AI and deep learning development and the developed tools can be used to further annotate and label other imaging biobanks.

The implementation of data sharing using imaging biobanking and cloud solutions still has to cope with technical, legal and societal challenges. To overcome these, a number of conditions have to be met. One of these is the standardization of communication, data format, and lexicons or terminologies which is essential to achieve environments with a high level of interoperability. Interchanging data between cloud solutions and standardization of APIs is still limited [9]. In medical imaging common data standards like DICOM and HL-7 could help in this, together with workflow definitions as provided by the IHE.

AI solutions are a step towards solving many existing problems in the area of healthcare. They provide a strong tool for healthcare experts. Distributed AI solutions, especially federated learning have the capability to provide accurate, safe, privacy-preserving and unbiased models for clinical usage. However, when designing distributed AI pipelines, many considerations should be taken into account. Proper infrastructure, data preparation, and AI model design are necessary for accurate training. How to perform parallelization, and how to enable efficient and secure communication among workers is another challenge which should be addressed.

Other issues to cope with are originating from the legal frameworks as defined by the HIPAA in the United States and the General Data Protection Regulation (GDPR) in the European Union. As an example, the GDPR has a large impact on safety and security of data stored in Imaging Biobanks and Biobanks and requires explicit informed consents of the data subjects, includes the right to be forgotten and requires cross border data transmission and sharing.

7 Conclusion

We need to enable sharing within an institution, between institutions and with patients. To realize this, the cloud plays an important role in fostering better models for fluent exchange of images and information and cloud based medical image sharing and multicentre databases has many advantages. It can be cost effective because of the economy of scale and can improve performance. Furthermore, with online patient health records, it is easier to access and share data between patients and doctors and between doctors. Cloud also offers an increased and more flexible storage capacity. PACS independent storage in the cloud could remove the necessity of tedious data migration projects when moving to another PACS vendor. And finally, consolidating and storing medical image information in single centralized repository in the cloud instead of multiple PACS in different sites means health care providers can quickly access and share images across various departments and organizations resulting in a more patient-centric environment.

References

[1] J. He, S. L. Baxter, J. Xu, J. Xu, X. Zhou and K. Zhang, "The practical implementation of artificial intelligence technologies in medicine," *Nat Med,* vol. 25, no. 1, pp. 30-36, 2019.

[2] Z. Shi, I. Zhovannik, A. Traverso, F. J. Dankers, T. M. Deist, K. Petros, M. René, B. Johan, F. Rianne, H. J, A. L, A. Dekker and L. Wee, "Distributed radiomics as a signature validation study using the Personal Health Train infrastructure," *Scientific Data volume,* vol. 218, 2019.

[3] M. A. Levy, J. B. Freymann, J. S. Kirby, A. Fedorov, F. M. Fennessy, S. A. Eschrich, A. E. Berglund, D. A. Fenstermacher, Y. Tan, X. Guo, T. L. Casavant, B. J. Brown, T. A. Braun, A. Dekker and Roelofs, "Informatics methods to enable sharing of quantitative imaging research data," *Magnetic Resonance Imaging,* vol. 30, pp. 1249-1256, 2012.

[4] European Society of Radiology, "ESR Position Paper on Imaging Biobanks," *Insights Imaging,* vol. 6, no. 4, pp. 403-410, 2015.

[6] J. L. Mulshine, R. S. Avila, E. Conley, A. Devraj, L. F. Ambrose, T. Flanagan, C. I. Henschke, F. R. Hirsch, R. Janz, R. Kakinuma, S. Lam, A. McWilliams, P. M. van Ooijen, M. Oudkerk and Pastorin, "The International Association for the Study of Lung Cancer Early Lung Imaging Confederation," *JCO Clinical Cancer Informatics,* vol. 4, pp. 89-99, 2020.

[7] E. Trägårdh, P. Borrelli, R. Kaboteh, T. Gillberg, J. Ulén, O. Enqvist and L. Edenbrandt, "RECOMIA - a cloud-based platform for artificial intelligence research in nuclear medicine and radiology," *EJNMMI Physics,* vol. 7, no. 51, 2020.

[8] G. González and C. L. Evans, "Biomedical Image Processing with Containers and Deep Learning: An Automated Analysis Pipeline," *BioEssays,* vol. 41, no. 1900004, 2019.

[9] G. C. Kagadis, C. Kloukinas, K. Moore, J. Philbin, P. Papadimitroulas, C. Alexakos, P. G. Nagy, D. Visvikis and W. R. Hendee, "Cloud computing in medical imaging," *Medical Physics,* vol. 40, no. 7, 2013.

[10] T. S. Mathai, Y. Wang and N. Cross, "Assessing Lesion Segmentation Bias of Neural Networks on Motion Corrupted Brain MRI," in *MICCAI BrainLes* , 2020.

[11] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," *NIST Special Publication ,* no. 800-145, 2011.

[12] S. Gupta, A. Agrawal, K. Gopalakrishnan and Narayanan, "Deep learning with limited numerical precision.," in *Proceedings of the 32Nd International Conference on International Conference on Machine Learning*, 2015.

[13] Z. Tao and Q. Li, "Communication efficient distributed deep learning on the edge computing," in *USENIX Workshop on Hot Topics in Edge Computing*, Boston , MA, 2018.

[14] M. Alhajeri, S. Ghulam and S. Shah, "Limitations in and Solutions for Improving the Functionality of Picture Archiving and Communication System: an Exploratory Study of PACS Professionals' Perspectives," *Journal of Digital Imaging,* vol. 67, p. 32:54, 2019.

[16] Blackford Analysis Ltd., "Adopting a Platform strategy: Simplify the deployment and management of medical imaging applications and AI algorithms," Blackford Analysis Ltd., Edinburgh, 2019.

[17] M. H. R. Mehrizi, P. van Ooijen and M. Homan, "Applications of artificial Intelligence (AI) in diagnostic radiology: a technography study," *European Radiology,* 2020.

[18] R. Mayer and H.-A. Jacobsen, "Scalable Deep Learning on Distributed Infrastructures: Challenges, Techniques and Tools," *ACM Computing Surveys,* 2019.

[19] J. Cipar, Q. Ho, J. Kim, S. Lee, G. Ganger, G. Gibson, K. Keeton and E. Xing, "Solving the straggler problem with bounded staleness," in *14th Workshop on Hot Topics in Operating Systems*, Santa Ana, NM, 2013.

[20] H. Cui, H. Zhang, G. Ganger, P. Gibbons and E. Xing, "Scalable deep learning on distributed gpus with a gpu-specialized parameter server," *n Proceedings of the Eleventh European Conference on Computer Systems,* pp. 4:1-4:16, 2016.

[21] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang and Q. Le, "Large scale distributed deep networks," *Advances in neural information processing systems,* pp. 1223-1231, 2012.

[22] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving and M. Isard, "Tensorflow: A system for large-scale machine learning.," *USENIX Symposium on Operating Systems Design and Implementation,* pp. 265-283, Savannah, GA, 2016.

[23] A. Sergeev and M. Balso, "Horovod: fast and easy distributed deep learning in tensorflow," *CoRR abs/1802.05799,* 2018.

[24] N. Karani, K. Chaitanya, C. Baumgartner and E. Konukoglu, " A Lifelong Learning Approach to Brain MR Segmentation Across Scanners and Protocols," in *MICCAI: Medical Image Computing and Computer Assisted Intervention*, 2018.

[25] J. Davis, "Health IT Security," Xtelligent Healthcare Media, 19 November 2019.[Online]. Available: https://healthitsecurity.com/news/number-of-exposed-pacs-medical-images-increasing-us-biggest-culprit. [Accessed 27 October 2020].

[26] "Data Management Tailored for Machine Learning Workloads in Kubernetes.," Google,[Online]. Available: Kubernetes.io.

[27] I. Chen, F. D. Johansson and D. Sontag , "Why Is My Classifier Discriminatory?," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.