Northwestern

PyLogik: An open-source resource for medical image de-identification

Adrienne Kline
MD, PhD
Postdoctoral Scholar
Northwestern University
Department of Preventative Medicine

Problem Formulation

Previous work:

- Variable AI based success rates of 65-89%
- Cost - $$$
- OS specific
- Do not batch process
- Don't comment on file size

What we want (design constraints):

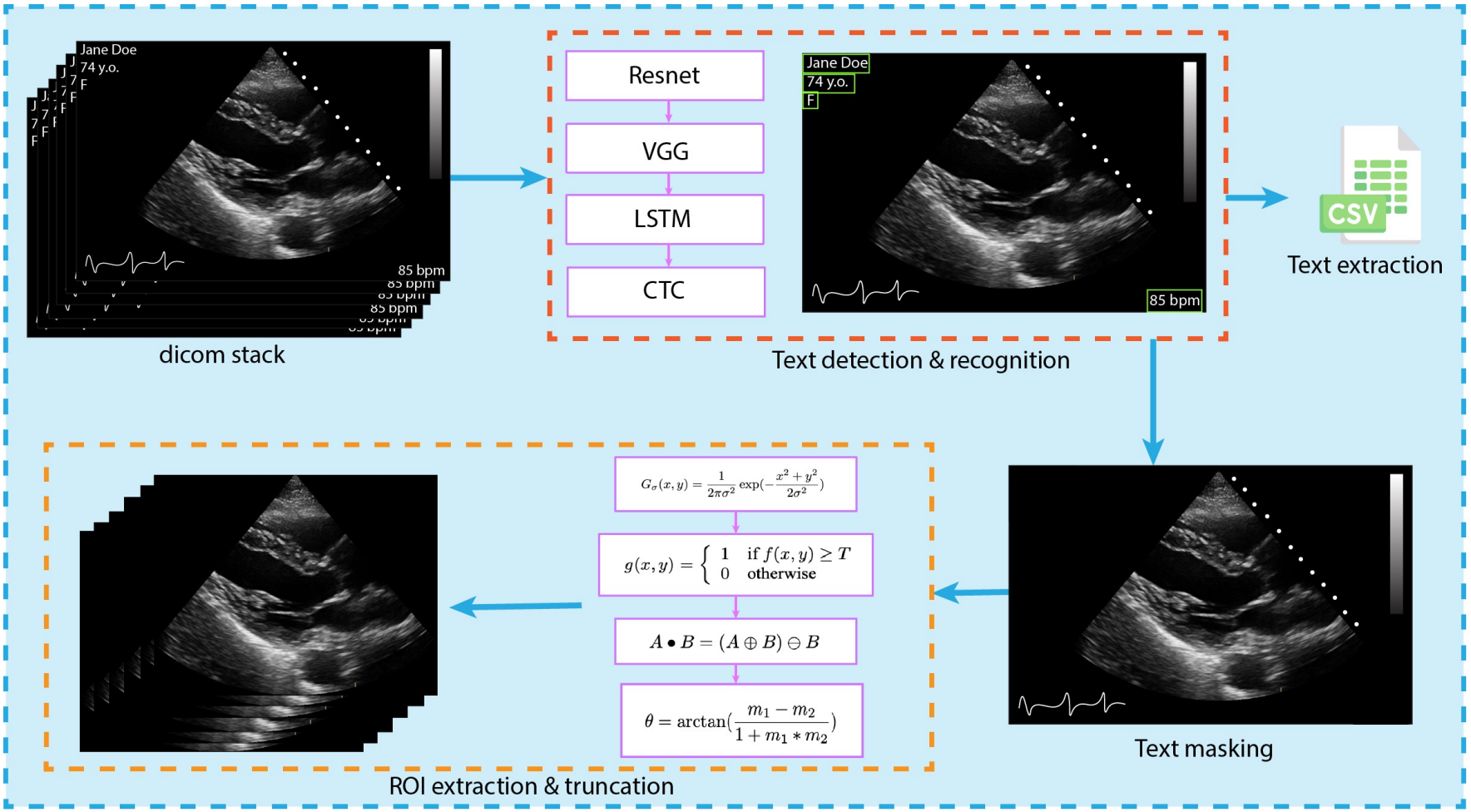
- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

Previous efforts trying to simultaneously DETECT and REMOVE ONLY PHI

Very difficult to anticipate all possible PHI formats and train your ML model to recognize ONLY that (alphanumeric, only numeric … etc.)

Solve the *simpler* problem!

Methods – Overview



Methods – Overview

De-identification (AI based)

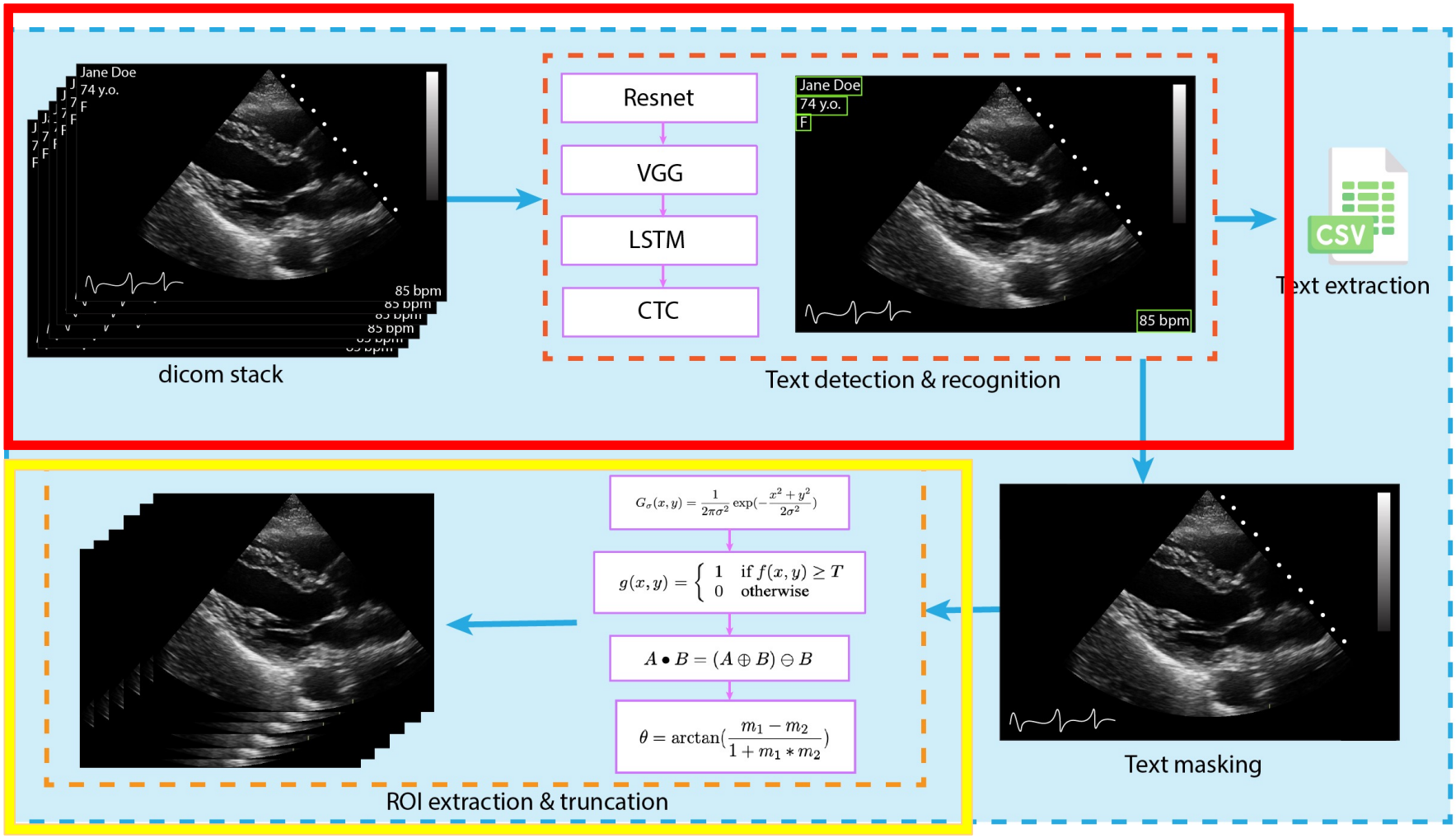
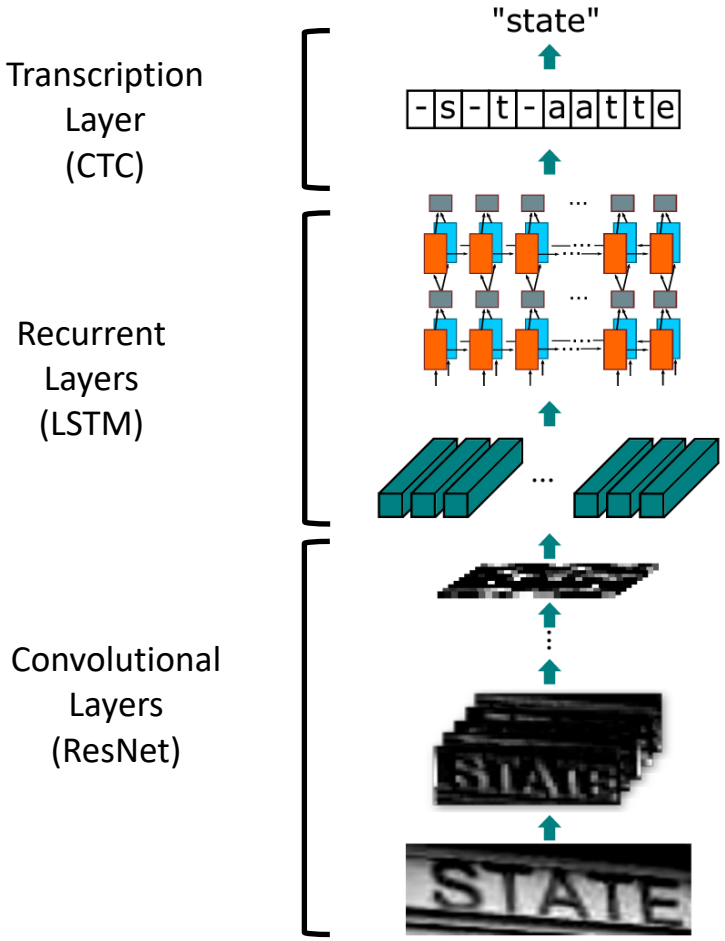


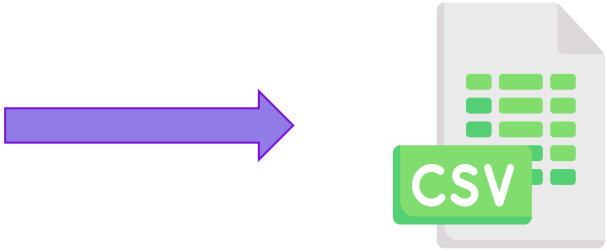
Image cleaning/compression

Text Detector/Interpreter

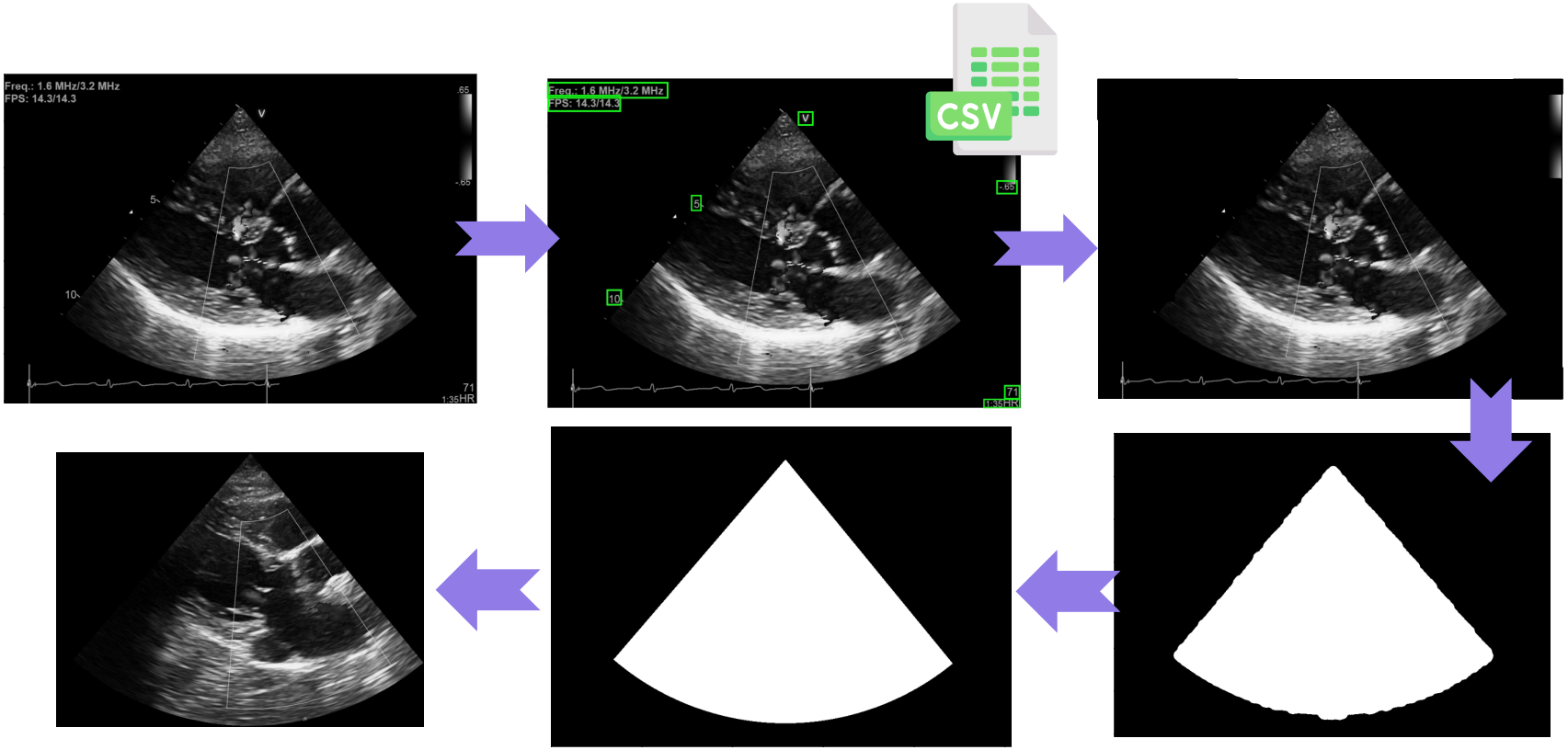


Free text (labeled) data used for training:

- IC13
- IIIT5k
- SVT



Methods



- Applied to all frames of the image and saved as a lossless .jpeg
- Runs in a loop
- Walks through all subfolders in directory
- Output: stack of .jpegs and a .csv
- Logs available to check skipped & processed files

Solve the *simpler* problem!

- Can employ regular expressions (regex) on a per-site to filter CSV on format of PHI (for sharing) and/or information fusion
- Or destroy the CSVs outright

Information Compression

- Same sample of 50 echos processed by expert user (cardiology fellow) analyzed for compression
- Saved as a lossless JPEG stack

Table 1: File Size Before and After Algorithm

	After			
Before	Image Data	MetaData	Total	Compression
969 MB (100%)	273.8 MB	209 KB	~ 274 MB (28.3%)	71.7%

72% more files can be stored (HDD/SSD) or held in working memory (RAM)

- Implications for developing ML algorithms (working memory)
- Implications for longer term storage on servers

Using this…

- Code is designed to be modular can run in **PIECES** or **TOGETHER**
  - **PIECES :** de-identification procedure can be separate from the cleaning/compression process
  - **TOGETHER:** de-identification and cleaning/compression all occur together as a pipeline for various tasks

https://pypi.org/project/pylogik/

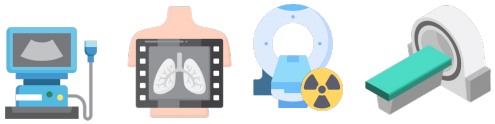
Install:

```
$ pip install pylogik
```

Using this...

1. De-identify only

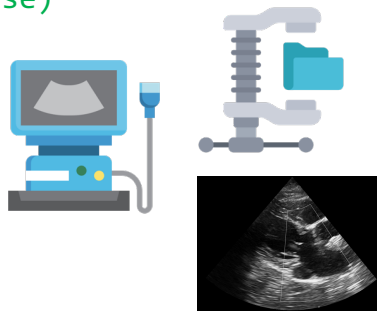
```
deid.deid(input_path,
output_path, rename_files =
False)
```



- All modalities (MRI, CT, ultrasound)
- Outputs csvs and images

2. De-identify & compress (ultrasound specific)

```
deid.deid_us(input_path,
output_path, rename_files =
False)
```



- Tailored to Ultrasound

3. De-identify & compress (other image modalities) retains largest ROI in image

```
deid.deid_one(input_path,
output_path)
```

- All modalities (MRI, CT, ultrasound)
- Outputs csvs and images

4. De-identify and clean (remove) small objects from image

```
deid.deid_clean(input_path,
output_path,
rename_files=False
```

- All modalities (MRI, CT, ultrasound)
- Outputs csvs and images

5. Finds text in image and write to CSVs (does not output images)

```
find_txt(input_path =
"path_to_files", output_path
= "path_to_save_files")
```

- Outputs csvs

6. Other functions

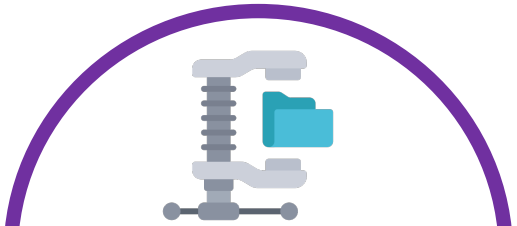
```
imshowpair(pred, true, color1 =
(124,252,0), color2 =
(255,0,252),
show_fig=True)

dice_score(pred, true, k=1)

color_select(img)
```

Smaller file size (up to ~8x) = faster reading and writing time

- Long term: servers
- Working mem: RAM

What we want (design constraints):

- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size



Smaller file size (up to ~8x) = faster reading and writing time

- Long term: servers
- Working mem: RAM

Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.

What we want (design constraints):

- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

Smaller file size (up to ~8x) = faster reading and writing time
- Long term: servers
- Working mem: RAM

Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.

Data cleaning: preparing data for machine learning experiments

What we want (design constraints):
- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

Smaller file size (up to ~8x) = faster reading and writing time
- Long term: servers
- Working mem: RAM

Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.

Data cleaning: preparing data for machine learning experiments

What we want (design constraints):
- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

Can run behind institutional firewalls (or locally)

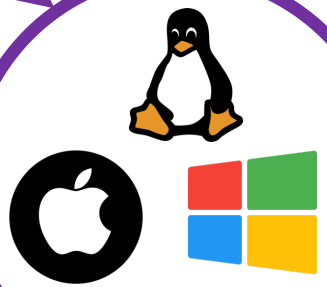
Smaller file size (up to ~8x) = faster reading and writing time
- Long term: servers
- Working mem: RAM

Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.

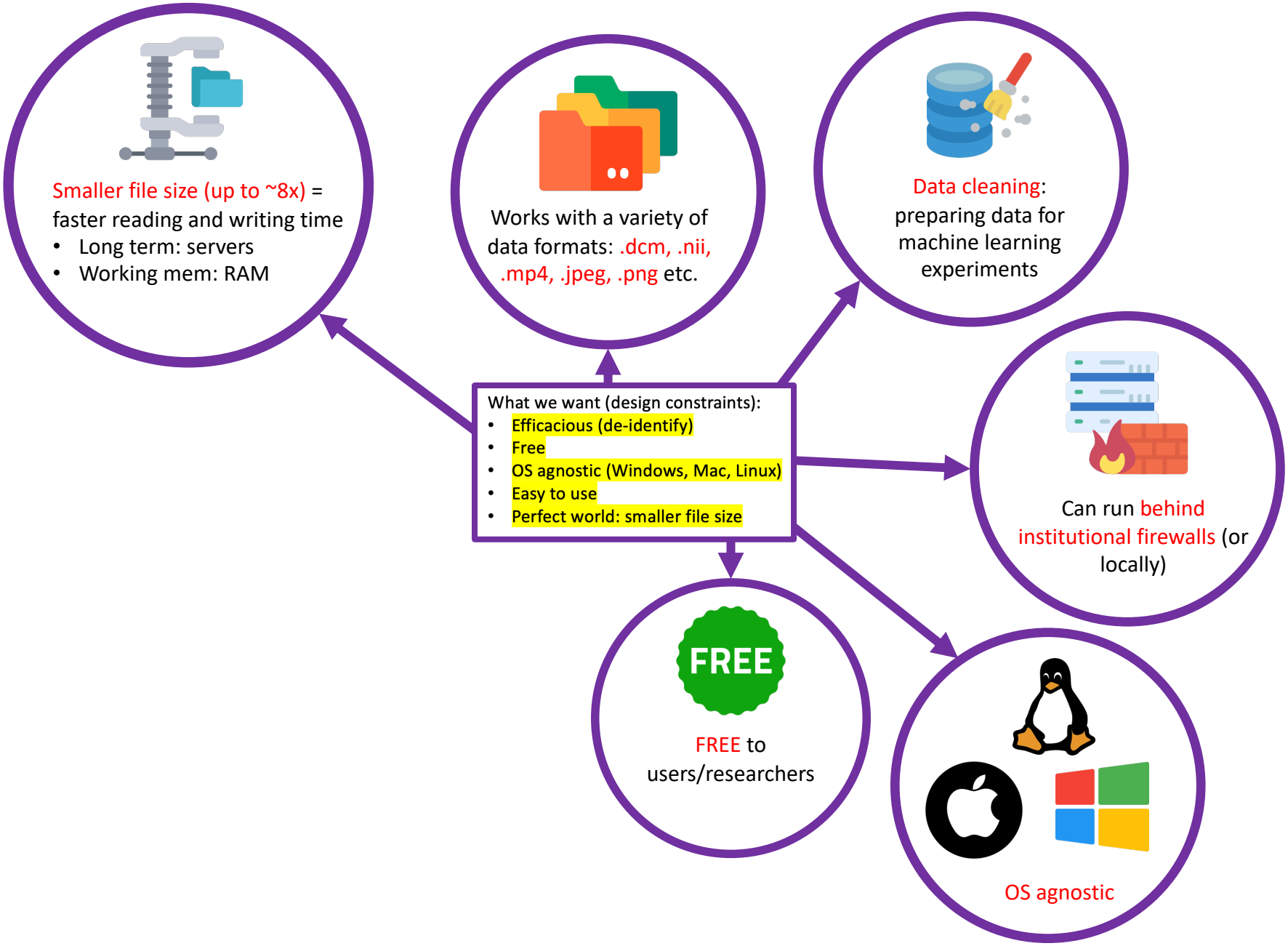
Data cleaning: preparing data for machine learning experiments

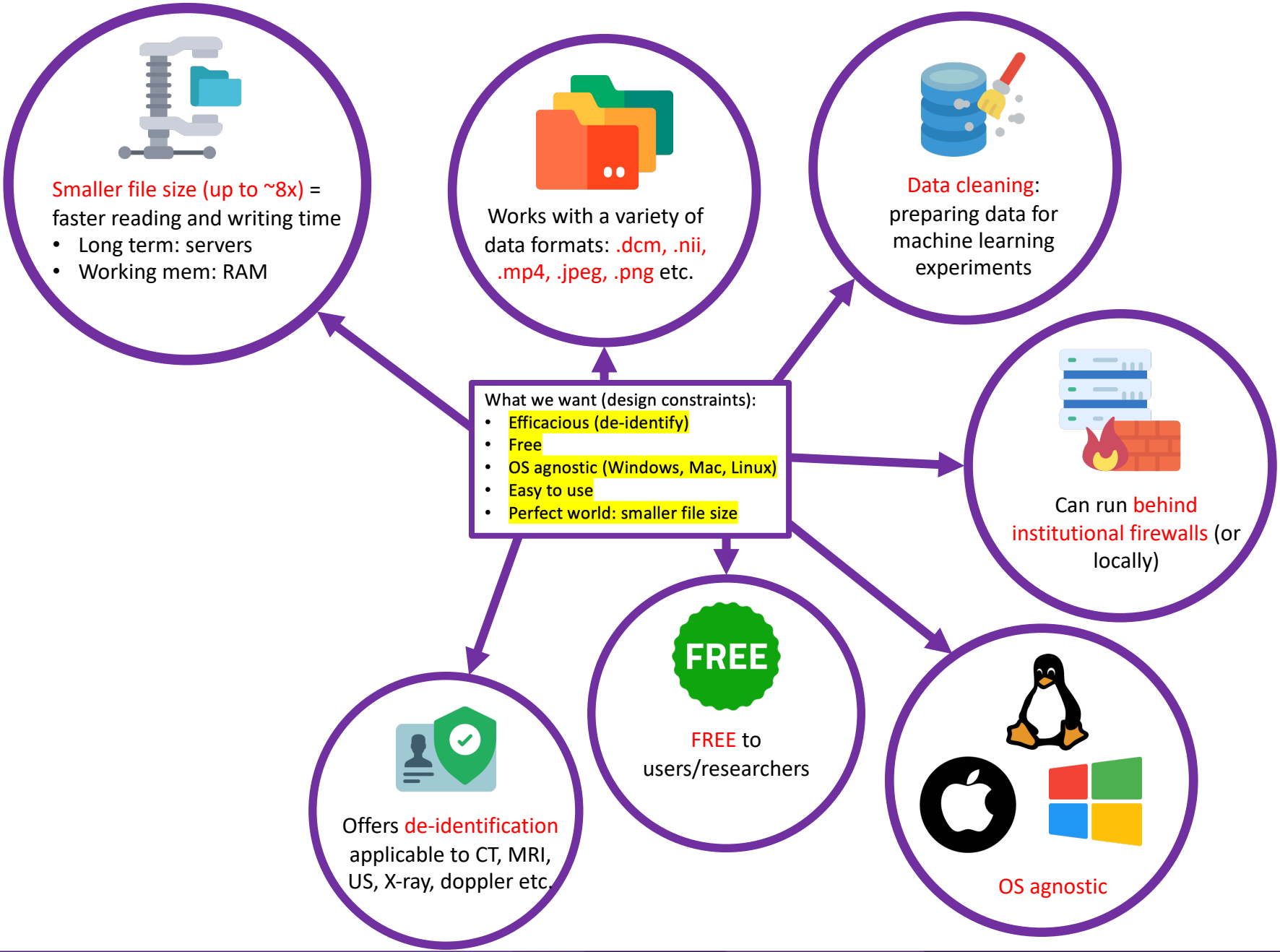
What we want (design constraints):
- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

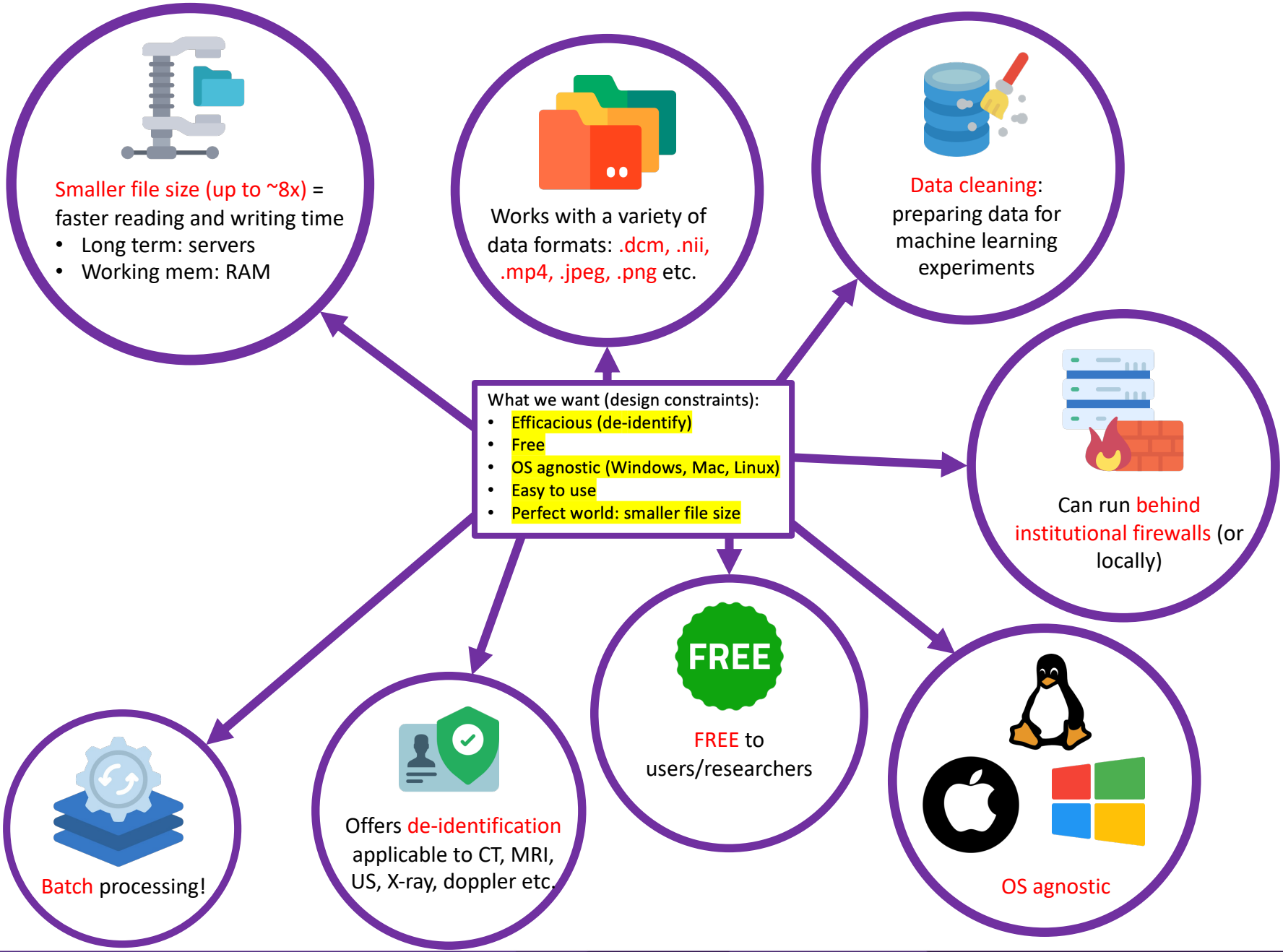
Can run behind institutional firewalls (or locally)



OS agnostic


Smaller file size (up to ~8x) = faster reading and writing time
• Long term: servers
• Working mem: RAM
Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.
Data cleaning: preparing data for machine learning experiments
What we want (design constraints):
• Efficacious (de-identify)
• Free
• OS agnostic (Windows, Mac, Linux)
• Easy to use
• Perfect world: smaller file size
Can run behind institutional firewalls (or locally)
FREE
FREE to users/researchers
OS agnostic


Smaller file size (up to ~8x) = faster reading and writing time
• Long term: servers
• Working mem: RAM
Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.
Data cleaning: preparing data for machine learning experiments
What we want (design constraints):
• Efficacious (de-identify)
• Free
• OS agnostic (Windows, Mac, Linux)
• Easy to use
• Perfect world: smaller file size
Can run behind institutional firewalls (or locally)
FREE
FREE to users/researchers
Offers de-identification applicable to CT, MRI, US, X-ray, doppler etc.
OS agnostic


Smaller file size (up to ~8x) = faster reading and writing time
• Long term: servers
• Working mem: RAM
Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.
Data cleaning: preparing data for machine learning experiments
What we want (design constraints):
• Efficacious (de-identify)
• Free
• OS agnostic (Windows, Mac, Linux)
• Easy to use
• Perfect world: smaller file size
Can run behind institutional firewalls (or locally)
Batch processing!
Offers de-identification applicable to CT, MRI, US, X-ray, doppler etc.
FREE
FREE to users/researchers
OS agnostic

Smaller file size (up to ~8x) = faster reading and writing time
- Long term: servers
- Working mem: RAM

Works with a variety of data formats: .dcm, .nii, .mp4, .jpeg, .png etc.

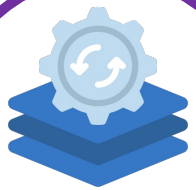
Data cleaning: preparing data for machine learning experiments

What we want (design constraints):
- Efficacious (de-identify)
- Free
- OS agnostic (Windows, Mac, Linux)
- Easy to use
- Perfect world: smaller file size

Can run behind institutional firewalls (or locally)

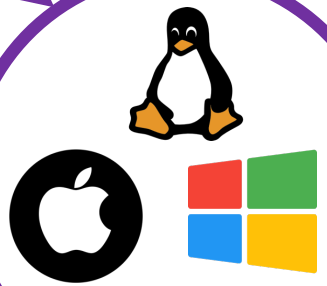
Easy installation/use:
`pip install pylogik`

FREE to users/researchers



Batch processing!

Offers de-identification applicable to CT, MRI, US, X-ray, doppler etc.



OS agnostic

Want more info?

https://arxiv.org/abs/2304.12322

Contact info: askline1@gmail.com

Medical Image Deidentification, Cleaning and Compression Using Pylogik

Adrienne Kline, Vinesh Appadurai, Yuan Luo, Sanjiv Shah

Leveraging medical record information in the era of big data and machine learning comes with the caveat that data must be cleaned and de-identified. Facilitating data sharing and harmonization for multi-center collaborations are particularly difficult when protected health information (PHI) is contained or embedded in image meta-data. We propose a novel library in the Python framework, called PyLogik, to help alleviate this issue for ultrasound images, which are particularly challenging because of the frequent inclusion of PHI directly on the images. PyLogik processes the image volumes through a series of text detection/extraction, filtering, thresholding, morphological and contour comparisons. This methodology de-identifies the images, reduces file sizes, and prepares image volumes for applications in deep learning and data sharing. To evaluate its effectiveness in processing ultrasound data, a random sample of 50 cardiac ultrasounds (echocardiograms) were processed through PyLogik, and the outputs were compared with the manual segmentations by an expert user. The Dice coefficient of the two approaches achieved an average value of 0.976. Next, an investigation was conducted to ascertain the degree of information compression achieved using the algorithm. Resultant data was found to be on average ~72% smaller after processing by PyLogik. Our results suggest that PyLogik is a viable methodology for data cleaning and de-identification, determining ROI, and file compression which will facilitate efficient storage, use, and dissemination of ultrasound data. Variants of the pipeline have also been created for use with other medical imaging data types.

Thanks to the team!

Dr. Vinesh Appadurai

Dr. Yuan Luo

Dr. Sanjiv Shah

Preventive Medicine

HEARTSHARE