

Deep Learning for Medical Imaging School 2023

Autoencoders

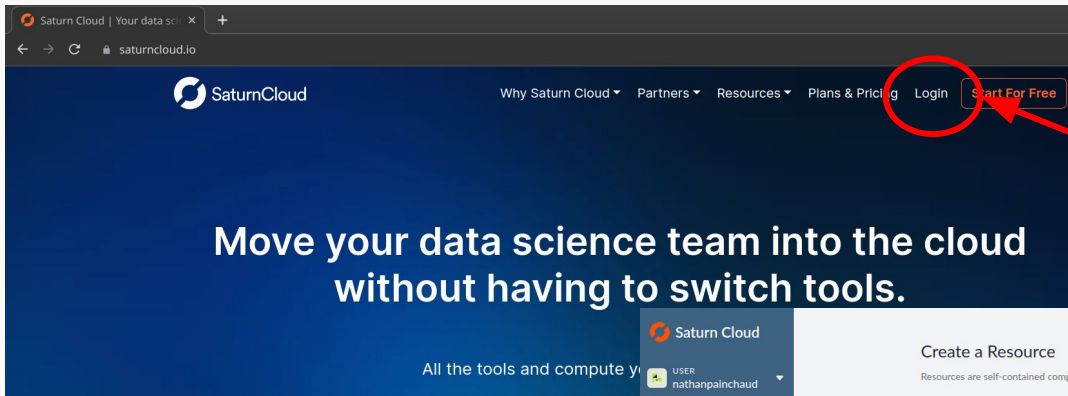
by Nathan Painchaud and Pierre-Marc Jodoin

with the help of Thomas Grenier and Olivier Bernard



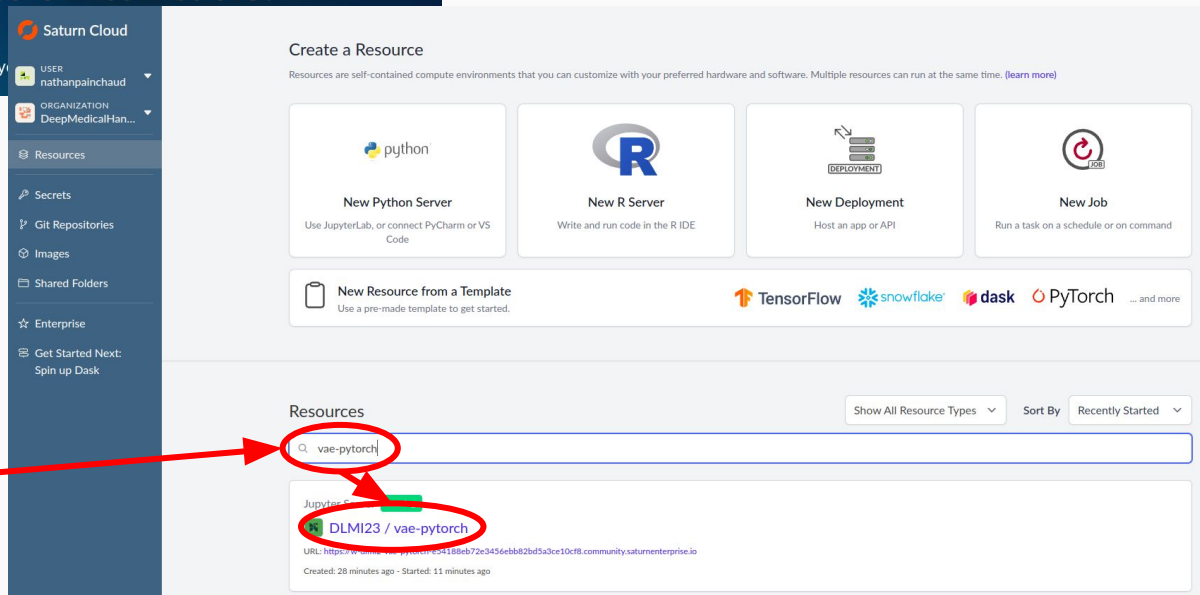
Set-up a SaturnCloud Server

SaturnCloud Environment Setup



Login to saturncloud.io

Search for the pre-configured Python server for the VAE hands-on and select it.



SaturnCloud Environment Setup

The screenshot displays the Saturn Cloud interface. At the top, the breadcrumb path is 'DeepMedicalHandsOn / Resources / DLM123 / vae-pytorch'. The main header shows 'JUPYTER SERVER' and 'DLM123 / vae-pytorch' with an ID 'e54188eb72e3456ebb82bd5a3ce10cf8'. A navigation bar includes 'Overview', 'Environment', 'Secrets and Roles', 'Git Repos', 'Shared Folders', 'Logs', 'Metrics', and 'Manage'. The 'Manage' button is circled in red. A callout box points to it with the text: 'Navigate to the Manage menu of the pre-configured server'. Below the navigation bar, a 'Jupyter Server' card shows a 'running' status, a 'Stop' button, and hardware specifications: 'T4-XLarge - 4 cores - 16 GB RAM - 1 GPU - 261 Disk'. An 'Auto Shutoff: 1 hour' warning is also present. A 'New Dask Cluster' button is visible. A 'Clone Resource' dialog is open, showing three options: 'Clone as a Python Server' (circled in red), 'Clone as a Deployment', and 'Clone as a Job'. A callout box points to the 'Clone as a Python Server' option with the text: 'Scroll down and select the Clone as a Python Server option'. Below the dialog, an 'API Token' section shows a masked token and a 'Replace Key' button.

USER nathanpainchaud
ORGANIZATION DeepMedicalHan...

Resources
Secrets
Git Repositories
Images
Shared Folders
Enterprise
Get Started Next: Spin up Dask

DeepMedicalHandsOn / Resources / DLM123 / vae-pytorch

JUPYTER SERVER
DLM123 / vae-pytorch
e54188eb72e3456ebb82bd5a3ce10cf8

Overview Environment Secrets and Roles Git Repos Shared Folders Logs Metrics Manage

Jupyter Server running Stop

T4-XLarge - 4 cores - 16 GB RAM - 1 GPU - 261 Disk

Auto Shutoff: 1 hour
Spot Instance: No
SSH URL: (not enabled) (?)
App URL: (not enabled)

New Dask Cluster

Clone Resource
Create a new resource with the same settings as this one.

python

Clone as a Python Server
Use JupyterLab, or connect PyCharm or VS Code

Clone as a Deployment
Host an app or API

Clone as a Job
Run a task on a schedule or on command

API Token
The API Token for the current resource

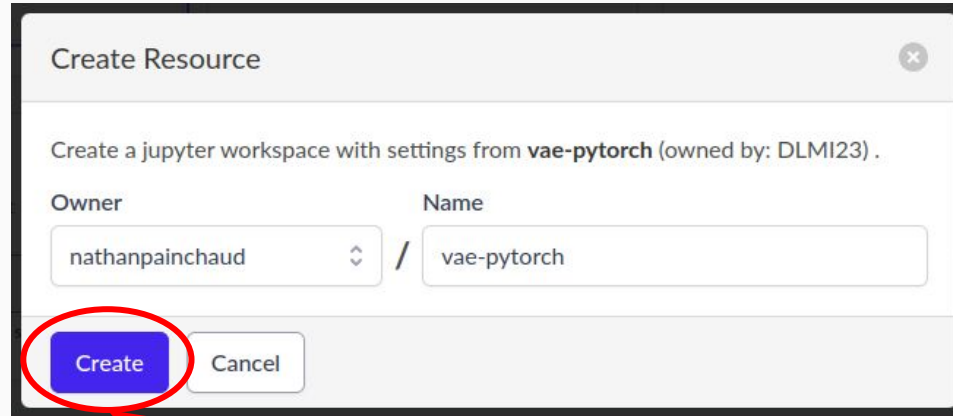
Replace Key

This is the resource token. Do not share this token with anyone.

Navigate to the Manage menu of the pre-configured server

Scroll down and select the Clone as a Python Server option

SaturnCloud Environment Setup



Create Resource

Create a jupyter workspace with settings from **vae-pytorch** (owned by: DLM123) .

Owner: nathanpainchaud / Name: vae-pytorch

Create Cancel

In the pop-up that appears, keep the default options (to create a clone of the resource that you own) and simply **click Create**

SaturnCloud Environment Setup

The image shows two screenshots of the Saturn Cloud interface. The top screenshot shows a Jupyter Server in a 'stopped' state. A red circle highlights the 'Start' button, and a red arrow points from a blue box labeled 'Start the server' to this button. The bottom screenshot shows the same Jupyter Server in a 'pending' state. A red circle highlights a progress bar with five stages: 'Provisioned Hardware', 'Pulling Image', 'Set Up Environment', 'Execute Start Script', and 'Ready'. A red arrow points from a blue box labeled 'Wait (possibly for a few minutes) while the progress bar indicates the progression of the set up of the environment' to this progress bar.

Start the server

Wait (possibly for a few minutes) while the progress bar indicates the progression of the set up of the environment

DeepMedicalHandsOn / Resources / nathanpainchaud / vae-pytorch

JUPYTER SERVER

nathanpainchaud / vae-pytorch
389788dd47f64945b81085bcc6121115a

Overview Environment Secrets and Roles Git Repos Shared Folders Logs Metrics Manage

Jupyter Server stopped

T4-XLarge - 4 cores - 16 GB RAM - 1 GPU - 261 Disk

Auto Shutoff: 1 hour
Spot Instance: No
SSH URL: (not enabled) (?)
App URL: (not enabled)

New Dask Cluster

DeepMedicalHandsOn / Resources / nathanpainchaud / vae-pytorch

JUPYTER SERVER

nathanpainchaud / vae-pytorch
389788dd47f64945b81085bcc6121115a

Overview Environment Secrets and Roles Git Repos Shared Folders Logs Metrics Manage

Jupyter Server pending

T4-XLarge - 4 cores - 16 GB RAM - 1 GPU - 261 Disk

Auto Shutoff: 1 hour
Spot Instance: No
SSH URL: (not enabled) (?)
App URL: (not enabled)

Provisioned Hardware Pulling Image Set Up Environment Execute Start Script Ready

New Dask Cluster

SaturnCloud Environment Setup

The screenshot displays the Saturn Cloud management console. On the left is a navigation sidebar with options like Resources, Secrets, and Git Repositories. The main area shows a 'JUPYTER SERVER' for user 'nathanpainchaud' in environment 'vae-pytorch'. The server is in a 'running' state. A red circle highlights the 'Jupyter Lab' button in the server's control panel. A red arrow points from this button to a blue callout box on the right. Below the server details, a 'New Dask Cluster' button is visible. In the foreground, a Jupyter Lab interface is shown, with a red circle highlighting the 'deep-learning-tutorials' folder in the left-hand file explorer. A second red arrow points from this folder to a blue callout box at the bottom left.

Once the server is ready, **click on the *Jupyter Lab* button** to launch it

In *Jupyter Lab*, navigate through the file explorer in the left panel to **get to the code**

SaturnCloud Environment Setup

The screenshot displays the SaturnCloud interface. On the left, a file explorer shows the directory structure of a workspace named 'deep-learning-tutorials'. The 'tutorials' folder is highlighted in blue and circled in red. A red arrow points from this folder to a blue callout box with white text that reads: "Inside the repo, **open the tutorials folder** to access the notebooks for the hands-on".

The main workspace area shows the contents of the 'tutorials' folder, including a 'Notebook' section with two 'saturn (Python 3)' icons, a 'Console' section with one 'saturn (Python 3)' icon, and an 'Other' section with icons for Terminal, Text File, Markdown File, Python File, and Show Contextual Help.

SaturnCloud Environment Setup

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

Name	Last Modified
.ipynb_checkpoints	seconds ago
cardiac-mri-autoencode...	6 minutes ago
mnist-autoencoders.ipynb	6 minutes ago

Hands-on session 2.1: Variational Autoencoder

Building Autoencoders in PyTorch

Made by **Nathan Painchaud** and **Pierre-Marc Jodoin** from the Université de Sherbrooke, Canada.

Inspired by a [similar tutorial](#) for Keras by François Chollet.

```
[ ]: %load_ext autoreload
      %autoreload 2

[ ]: %%capture data_download

# Make sure the data is downloaded and extracted where it should be
!wget -N http://info.usherbrooke.ca/pmjodoin/projects/data.tar.gz --directory-prefix=../
!tar -xvzf ../data.tar.gz -C ../

[ ]: %%capture project_path_setup

import sys

if '..' in sys.path:
    print(sys.path)
else:
    sys.path.append('../')
    print(sys.path)

[ ]: %%capture packages_install

# Make sure the repo's package and its dependencies are installed
!pip install -e ..
```

Simple 0 1 saturn (Python 3) | Idle Mode: Command Ln 1, Col 1 mnist-autoencoders.ipynb

Start with the **mnist** notebook, and move on to the **cardiac-mri** notebook afterwards

Autoencoders Recap

Summary

Note: If you are familiar with AEs and VAEs, you may skip the rest of the slides

- What are autoencoders
- How are they implemented
- How do they apply to MNIST (grayscale images)
- How do they apply to ACDC (cardiac segmentation maps)

What are autoencoders?

Problem: Learn the distribution of a set of data

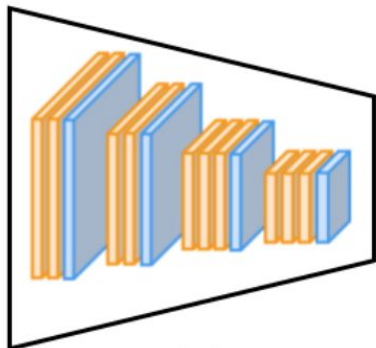
Method: Train a neural network to output... its own input!

Autoencoder Framework



Input space

$$x_i \in \mathbb{R}^{N \times M}$$

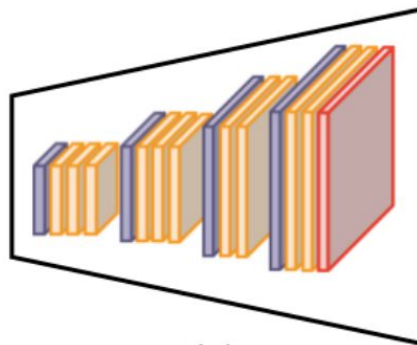


$e(x)$

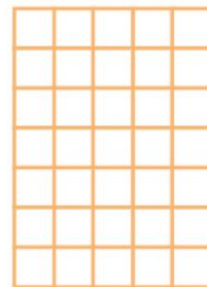


Latent space

$$z_i \in \mathbb{R}^K$$



$f(z)$

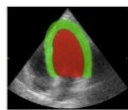


Output space

$$\hat{x}_i \in \mathbb{R}^{N \times M}$$

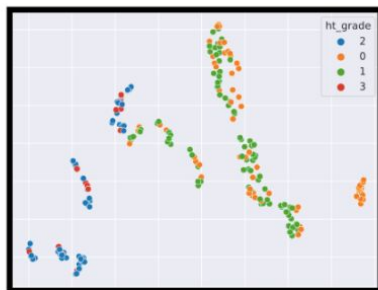
$$\text{loss} = \|x - f(e(x))\|^2$$

→ Data representation



Patients

encoder



Population representation

→ Generative model

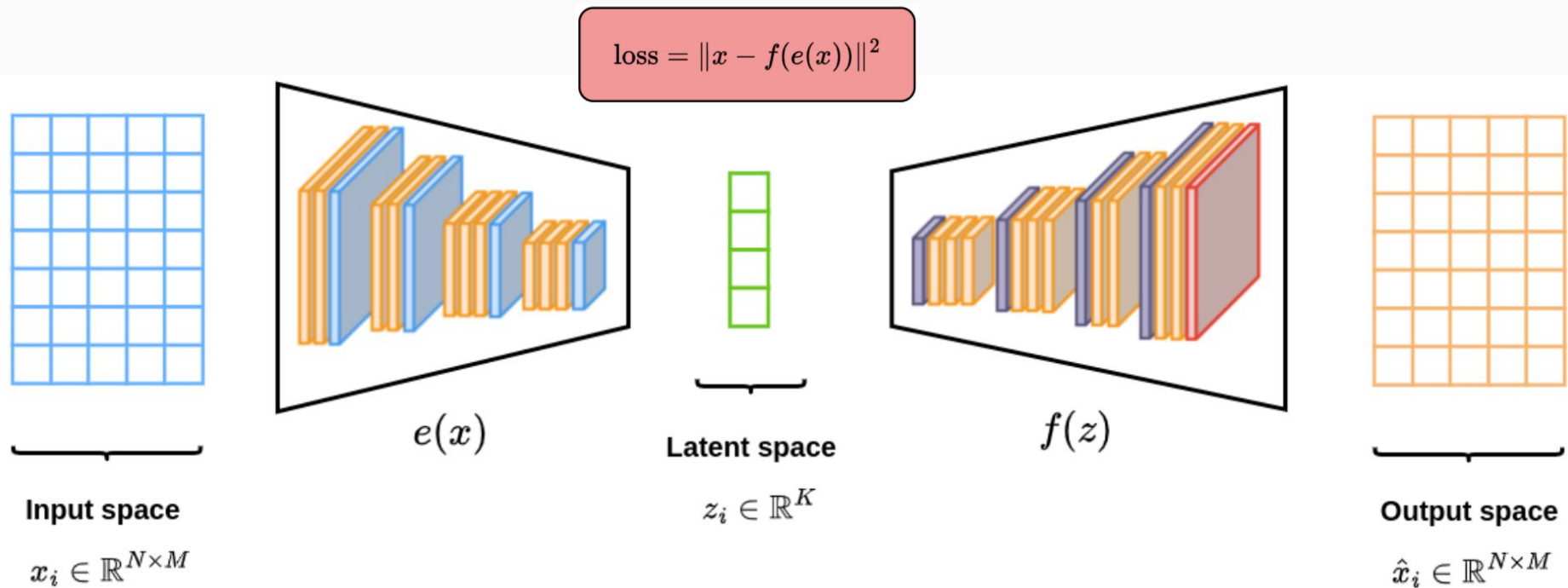


decoder

0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9

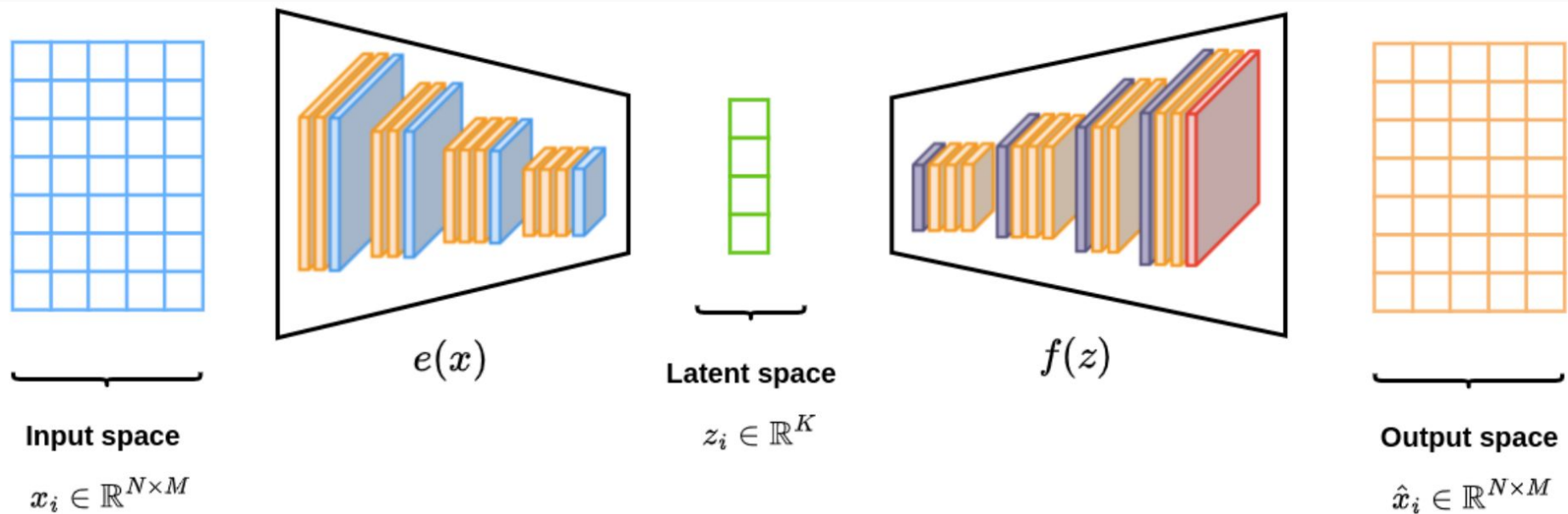
Key Concepts

- Loss minimizes reconstruction error of the output, e.g.



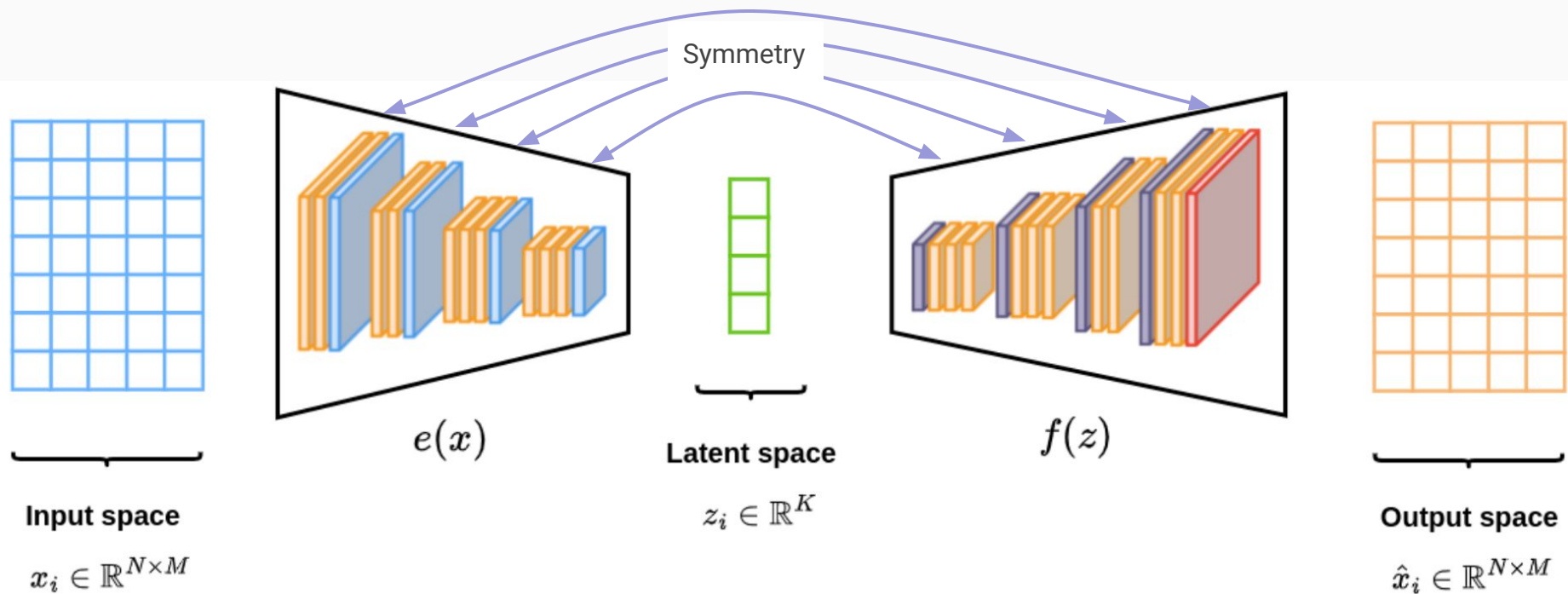
Key Concepts

- Encoder-decoder architecture to compress input, with $K \ll N \times M$



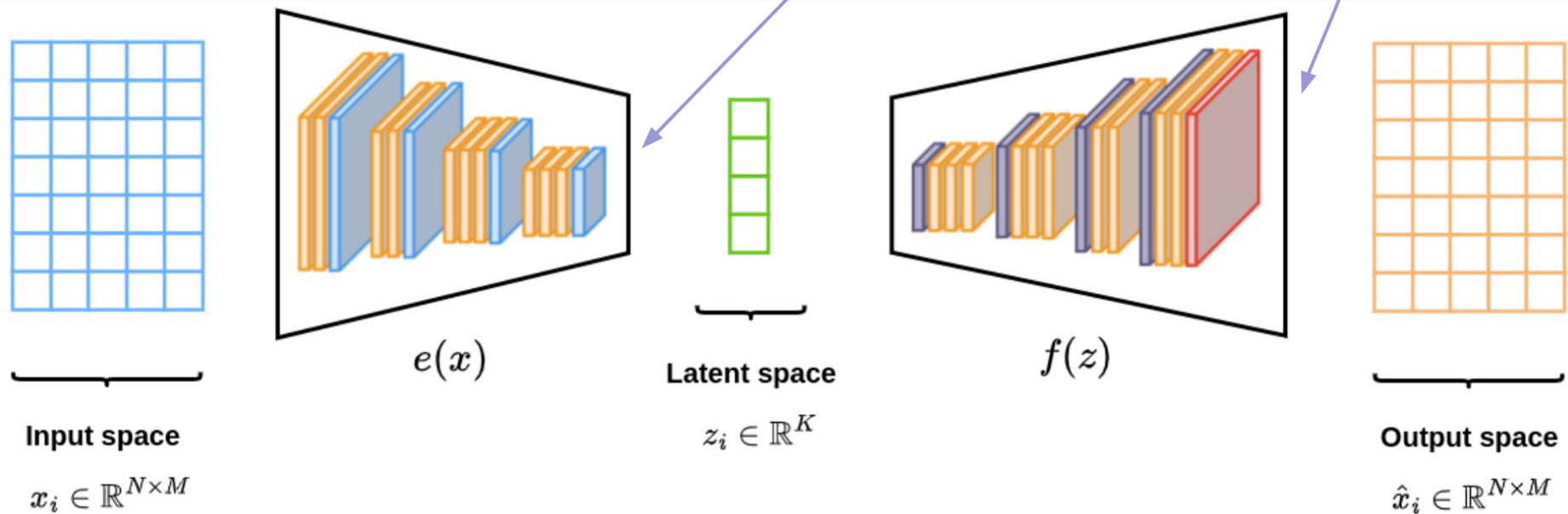
Key Concepts

- Generally, decoder is a **mirror** of the encoder



Key Concepts

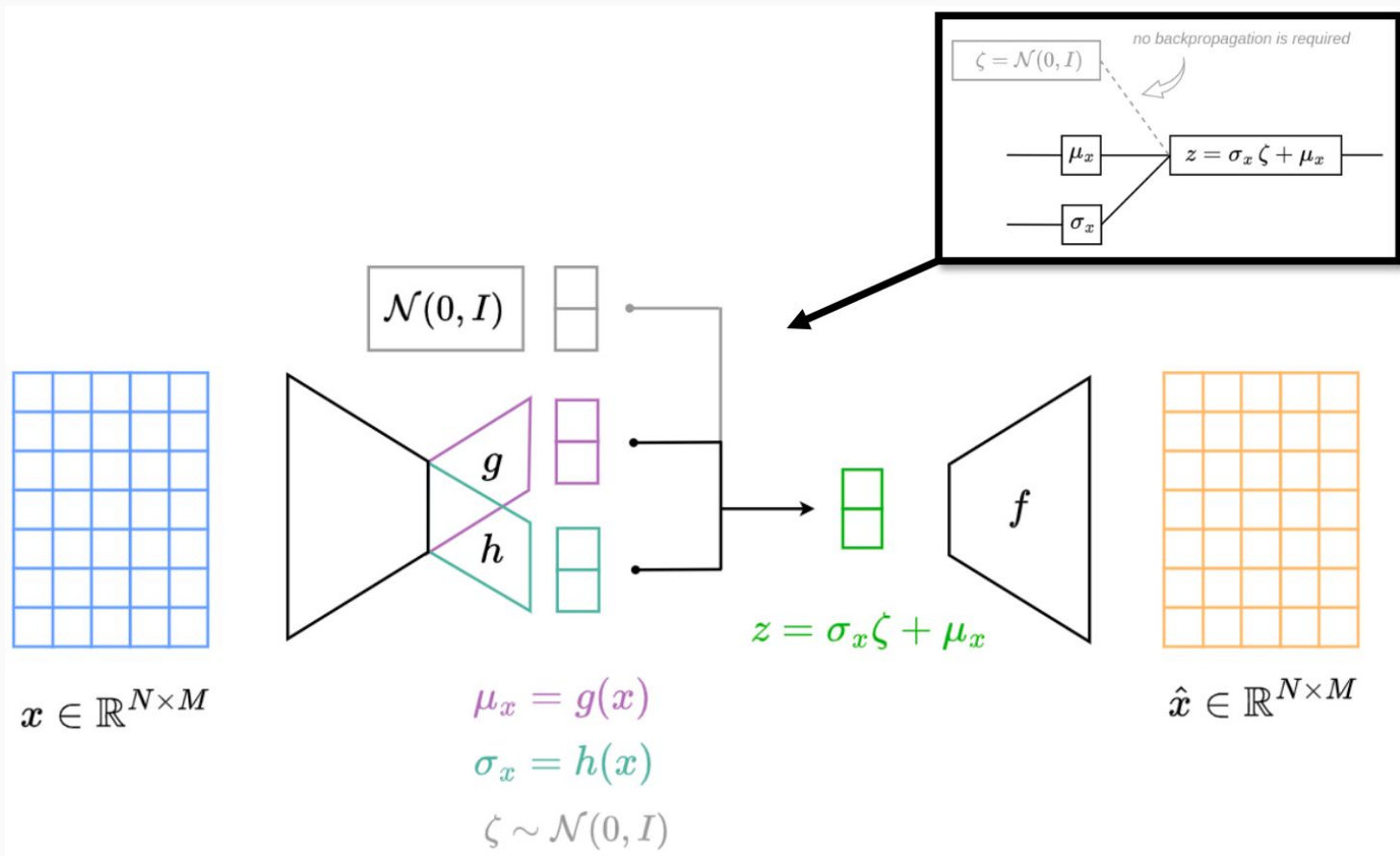
- Mostly, **sigmoid/relu** at the output of the **decoder**
+ **no activations** at the output of the **encoder**



Variational Autoencoders

- Encoder output is a $\mathcal{N}(g(x), h(x))$ distribution instead of a precise point
How does this affect the implementation?
 - 2 heads g and h at the end of the encoder (shared weights in previous layers)
 - Reparameterization trick (see [next slide](#))
- $\mathcal{N}(0, I)$ prior on the encoder's predictions
How does this affect the implementation?
 - Add a KL divergence term to the total loss

Reparameterization Trick



MNIST

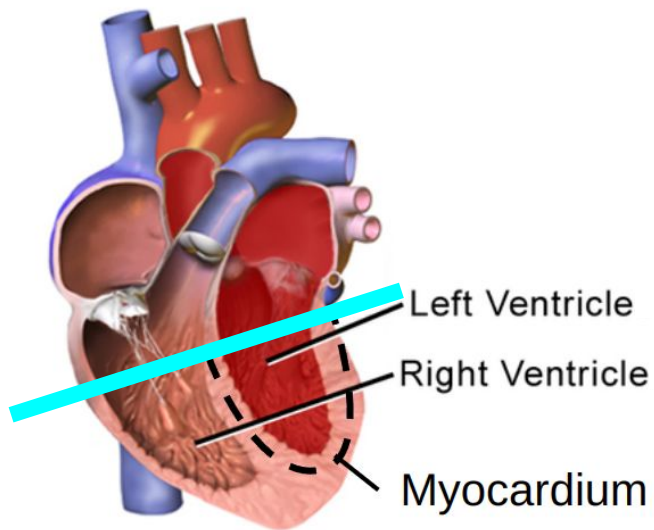
- Handwritten digits
 - 60,000 images
 - 32x32 pixels
 - [Website](#)
- Simple images/distribution ->
 - Fully-connected AE
 - Interactive visualization of 2D latent space
- Test autoencoder vs. variational autoencoder



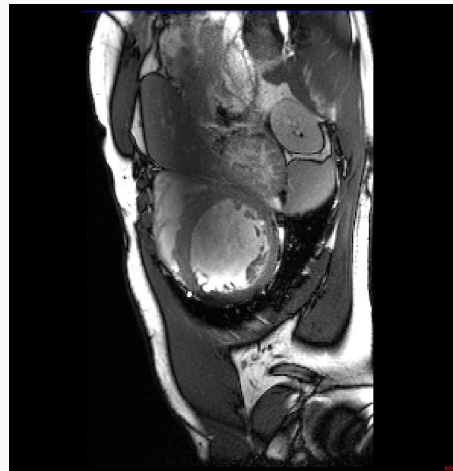


- Cardiac short-axis cine-MRI
 - 150 patients
 - 5 clinical groups
 - 256x256 pixels
 - [Website](#)
- Complex images/distribution ->
 - Convolutional AE/VAE
- Showcase AE/VAE on real-world problems

Normal Heart



Chambers relax and fill,
then contract and pump.



Right ventricle

Myocardium

Left ventricle

Background

Create SaturnCloud Environment from Scratch

Create a Jupyter Server

Jupyter servers let you interactively write code via JupyterLab, a terminal, or via SSH.

Start From a Recipe

Recipes are pre-configured Saturn Cloud resources. Choose from an existing recipe or upload your own.

[Use a Recipe](#)

Overview [Show Advanced Options](#)

Owner: nathanpainchaud

Name: dlmi2023-ho3

Give a descriptive name to the server

Hardware [Show Advanced Options](#)

The hardware your Jupyter server will run on.

Hardware

CPU
An instance with only CPU processors.

GPU
An instance with both CPU and GPU processors.

Size

T4-XLarge - 4 cores - 16 GB RAM - 1 GPU

Disabled options are not supported due to your account limit. To increase the limit, please contact your administrator.

Select the GPU hardware

Environment [Show Advanced Options](#)

The software your Jupyter server will use. This includes libraries, packages, environment variables, and other attributes.

Image

saturncloud/saturn-python-pytorch

Version

2022.03.01

Select the pytorch image

Select the 2022-03-01 version (NOT the most recent one)

Create SaturnCloud Environment from Scratch

Environment
The software your Jupyter server will use. This includes libraries, packages, environment variables, and other attributes. [Show Advanced Options](#)

Image: saturncloud/saturn-python-pytorch : Version: 2022.03.01

Extra Packages
Extra packages are installed every time the resource starts up - right before the start script. Use spaces to separate packages. If you find yourself adding the same packages to lots of resources, you may want to permanently add packages to a custom image instead. (?)

Conda Pip **Apt**

htop zip unzip python3-opencv

The packages together will run the following script:

```
apt-get install htop zip unzip python3-opencv
pip install torchvision==0.10.1
```

apt-install the packages listed here

pip install the packages listed here (including the versions)

Environment
The software your Jupyter server will use. This includes libraries, packages, environment variables, and other attributes. [Show Advanced Options](#)

Image: saturncloud/saturn-python-pytorch : Version: 2022.03.01

Extra Packages
Extra packages are installed every time the resource starts up - right before the start script. Use spaces to separate packages. If you find yourself adding the same packages to lots of resources, you may want to permanently add packages to a custom image instead. (?)

Conda Pip **Apt**

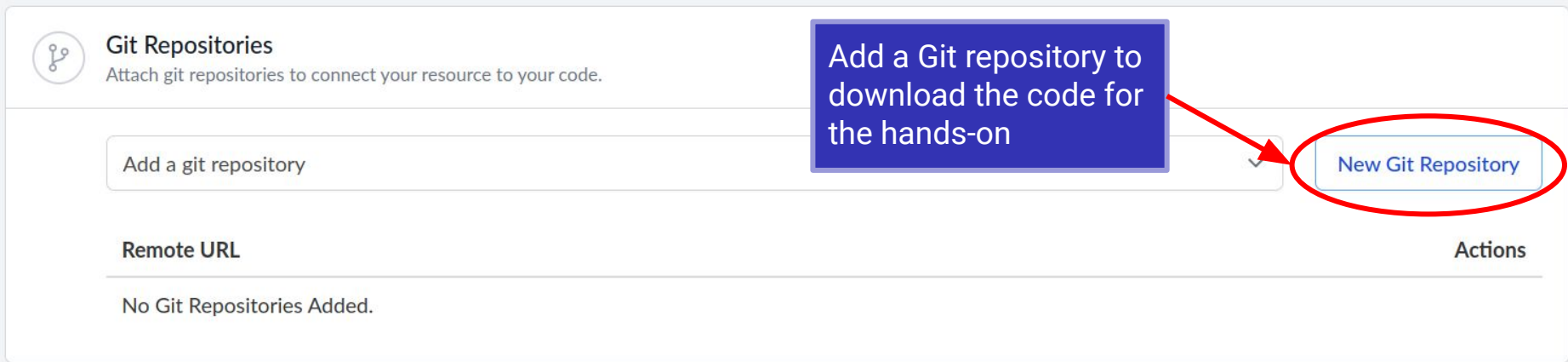
torchvision==0.10.1

This is a requirements.txt file
If enabled, this should be a valid requirements file describing all packages to be installed by pip, such as the content produced by pip freeze. See the pip documentation for details.

The packages together will run the following script:

```
pip install torchvision==0.10.1
```


Create SaturnCloud Environment from Scratch



Git Repositories
Attach git repositories to connect your resource to your code.

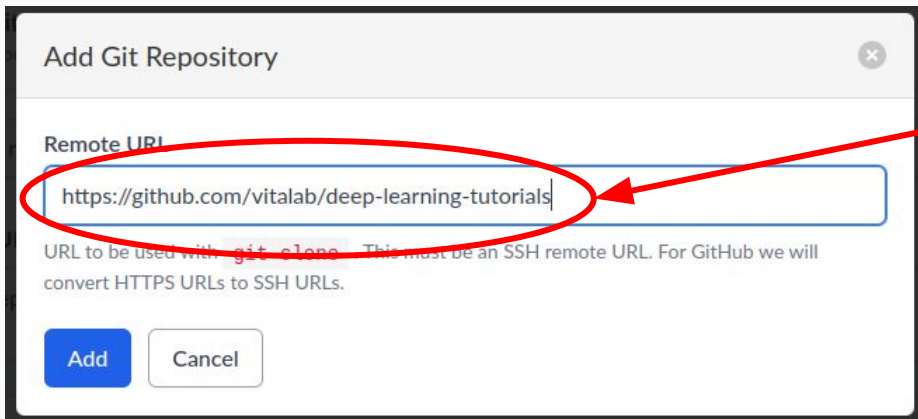
Add a git repository

New Git Repository

Remote URL Actions

No Git Repositories Added.

Add a Git repository to download the code for the hands-on



Add Git Repository

Remote URL

`https://github.com/vitalab/deep-learning-tutorials`

URL to be used with `git clone`. This must be an SSH remote URL. For GitHub we will convert HTTPS URLs to SSH URLs.

Add Cancel

In the pop-up that appears, paste the following URL:
<https://github.com/vitalab/deep-learning-tutorials>

Create SaturnCloud Environment from Scratch



Additional features

Optional settings for your Jupyter server.

[Show Advanced Options](#)

Allow SSH Connections

Use SSH to directly connect to the server, including through VSCode, PyCharm, and other tools (?)

Shutoff After

1 hour



Create

Cancel

Create the server