

User's Guide

This guide is written for non-experts looking to use the foundational model for transfer learning for their own tasks. We will begin with the basic installation and move into an example of how to use the foundational model for Buchwald-Hartwig yield prediction. The model was trained on Linux OS and we highly recommend that users do the same.

Prerequisites

Installation of Conda:

A good rule of thumb when working with code in Python is to use virtual environments for your projects. This will allow you to install the exact packages you need without worrying about conflicts from other projects / previous installations. We will be using conda virtual environments. Official instructions for installation can be found here: <https://docs.conda.io/projects/miniconda/en/latest/>. Please follow the instructions according to your operating system.

Installation of Git (Optional):

The program used to interface with GitHub is git. Although not a requirement for getting the model from the GitHub, it is a very useful tool in managing code. Official instructions for installation can be found here: <https://github.com/git-guides/install-git>. Please follow the instructions according to your operating system.

Getting the Code

We will show you the commands used (highlighted in yellow) and the associated output (the line after the command is run) to assist users in using command line inputs and our model. Commands can be run by pressing the "return" key. Within this text, commands will be indicated with **this font**. These commands are to be implemented in your computer's terminal / console.

Create a New Directory:

A directory can also be referred to as a "folder". For the remainder of the document, we will be using the term directory but you may think of them as folders. To access our code, two methods are possible. The first uses git, the second goes through GitHub.com's web-interface. We will go through both.

Start by making a new directory called "transfer_learning" and move into the directory. This can be done by typing:

```
mkdir transfer_learning
```

There will be no output from this command.



```
emmaking-smith — ubuntu@magic: ~/user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31
...ubuntu@magic: ~/user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122
[ubuntu@magic:~/user_guide]$ mkdir transfer_learning
ubuntu@magic:~/user_guide$
```

Move into the directory with:

```
cd transfer_learning
```



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31
...user_guide/transfer_learning — ssh -i alchemist_eks ubuntu@90.202.241.122
[ubuntu@magic:~/user_guide]$ mkdir transfer_learning
[ubuntu@magic:~/user_guide]$ cd transfer_learning
[ubuntu@magic:~/user_guide/transfer_learning$
```

We are now ready to download the code for Transfer Learning for a Foundational Chemistry Model. You have two options. Option 1 requires the installation of git and Option 2 does not. Please choose **either** Option 1 or Option 2.

Option 1 - Download the GitHub Repository with Git (Git installation required):

Clone the repository. This allows you to instantly download all the information off of the repository to your new directory. Type:

```
git clone https://github.com/emmaking-smith/Modular_Latent_Space.git
```



The screenshot shows a terminal window with a light gray background and a dark gray header bar. The header bar contains the text "emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31". Below the header is a command-line interface with a light gray background. The command "git clone https://github.com/emmaking-smith/Modular_Latent_Space.git" is entered and executed. The output of the command is displayed in green text, showing the progress of the cloning process: "Cloning into 'Modular_Latent_Space'...", "remote: Enumerating objects: 277, done.", "remote: Counting objects: 100% (48/48), done.", "remote: Compressing objects: 100% (47/47), done.", "remote: Total 277 (delta 27), reused 4 (delta 1), pack-reused 229", "Receiving objects: 100% (277/277), 11.31 MiB | 5.46 MiB/s, done.", "Resolving deltas: 100% (153/153), done." The prompt "ubuntu@magic:~/user_guide/transfer_learning\$" appears at the end of the output.

Option 2 - Download the GitHub Repository from GitHub.com:

If you do not want to install git, you may download all the code from the GitHub website. Go to https://github.com/emmaking-smith/Modular_Latent_Space.

The screenshot shows a GitHub repository page for 'Modular_Latent_Space'. At the top, there's a navigation bar with links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the navigation bar, the repository name 'Modular_Latent_Space' is displayed, along with a 'Public' badge. To the right of the repository name are buttons for Pin, Unwatch (with 1 watch), Fork (with 0 forks), and Star (with 0 stars). The main content area shows a list of files and their details. A green 'Code' button is located at the top right of this list. On the far right, there's an 'About' section with a detailed description of the code, links to chemrxiv.org, machine-learning, and chemistry, and various statistics like 68 commits and 1 watching.

Click on the green "Code" button.

This screenshot is similar to the first one, showing the same GitHub repository page for 'Modular_Latent_Space'. However, the green 'Code' button has been highlighted with a red box. A dropdown menu has appeared below the button, containing options for 'Local' and 'Codespaces', followed by 'Clone' (with HTTPS, SSH, and GitHub CLI options), 'Open with GitHub Desktop', 'Download ZIP', and an AI programming offer. The rest of the page, including the file list and the 'About' section, remains the same.

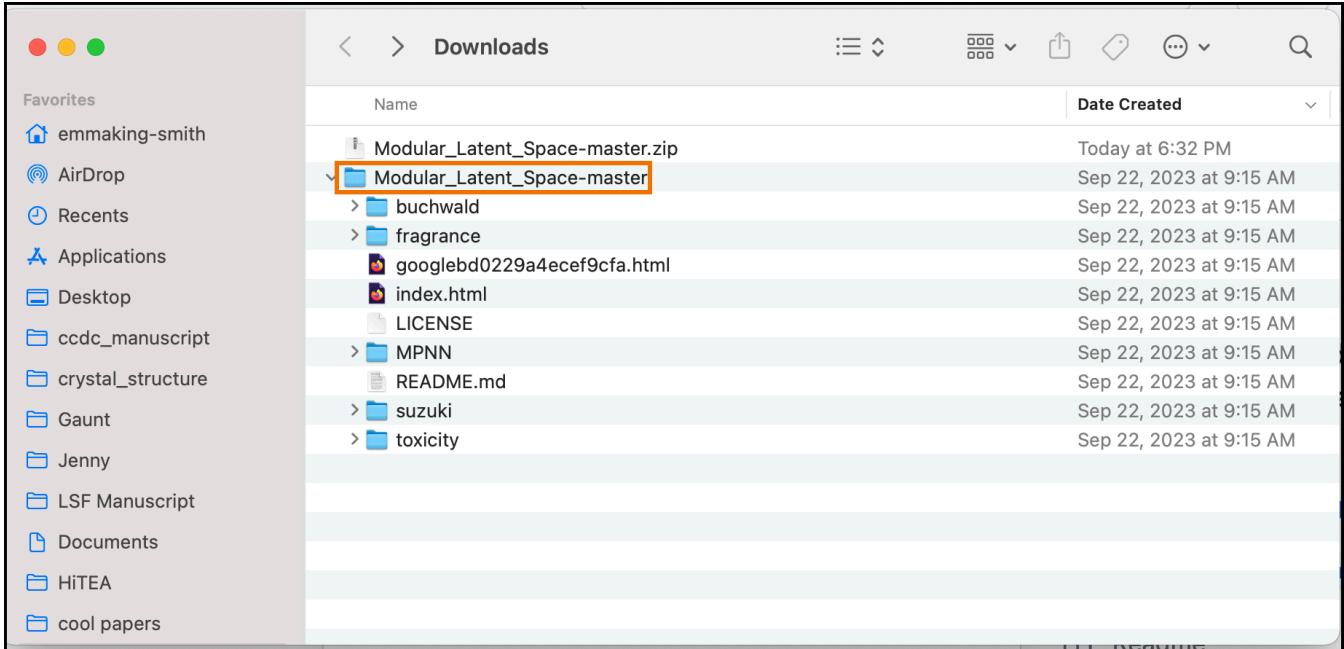
Select "Download ZIP"

The screenshot shows a GitHub repository page for 'Modular_Latent_Space'. The repository is public and has 1 branch and 0 tags. The code tab is selected. On the left, there's a sidebar with file navigation. In the center, there's a 'Clone' section with options for HTTPS, SSH, and GitHub CLI. A 'Download ZIP' button is highlighted with a red box. To the right, there's an 'About' section with a brief description of the code corresponding to Transfer Learning for a Foundational Chemistry Model, and links to chemrxiv.org and machine-learning/chemistry. There are also sections for Readme, MIT license, Activity, and releases.

Unzip the zip file. This can typically be achieved by double clicking on the file.

The screenshot shows a Mac OS X Finder window titled 'Downloads'. The sidebar on the left lists 'Favorites' including 'emmaking-smith', 'AirDrop', 'Recents', 'Applications', 'Desktop', and several document files like 'ccdc_manuscript', 'crystal_structure', 'Gaunt', 'Jenny', 'LSF Manuscript', 'Documents', 'HiTEA', and 'cool papers'. The main pane shows a single item: 'Modular_Latent_Space-master.zip' which was created 'Today at 6:32 PM'. The file is highlighted with a blue selection bar.

Move the unzipped directory, Modular_Latent_Space-master from its current directory to the directory we created at the beginning of this section (we named it transfer_learning). Drag and drop is the easiest way to do this.



Setting up the Virtual Environment

Here, we install the necessary packages to run the transfer learning. The necessary packages can be found at https://github.com/emmakin-smith/Modular_Latent_Space at the bottom of the page under "Dependencies".

First we create our virtual environment that will be used to run all of our code. Note that you must be within this virtual environment to run the code without errors. To do this, we run the following command:

```
conda create -n tl3.7 python==3.7
```

Our virtual environment's name is tl3.7 (transfer learning python version 3.7). We designate the python version with the suffix of "python==3.7". **It is critical to use the correct version of python when attempting to use any other programmer's code.**

```

emmakingsmith — ubuntu@magic: ~user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31
...buntu@magic: ~user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 ~ — zsh +]

ubuntu@magic:~/user_guide$ conda create -n tl3.7 python==3.7
Collecting package metadata (current_repodata.json): done
Solving environment: unsuccessful attempt using repodata from current_repodata.json, retrying with next repo
data source.
Collecting package metadata (repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 23.7.3
  latest version: 23.9.0

Please update conda by running

$ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

  conda install conda=23.9.0

## Package Plan ##

environment location: /home/ubuntu/miniconda3/envs/tl3.7

added / updated specs:
- python==3.7

The following NEW packages will be INSTALLED:
```

Shortly thereafter, the console will prompt you to accept the installation of new packages. Type:

y

```

emmakingsmith — ubuntu@magic: ~user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31
...buntu@magic: ~user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 ~ — zsh +]

environment location: /home/ubuntu/miniconda3/envs/tl3.7

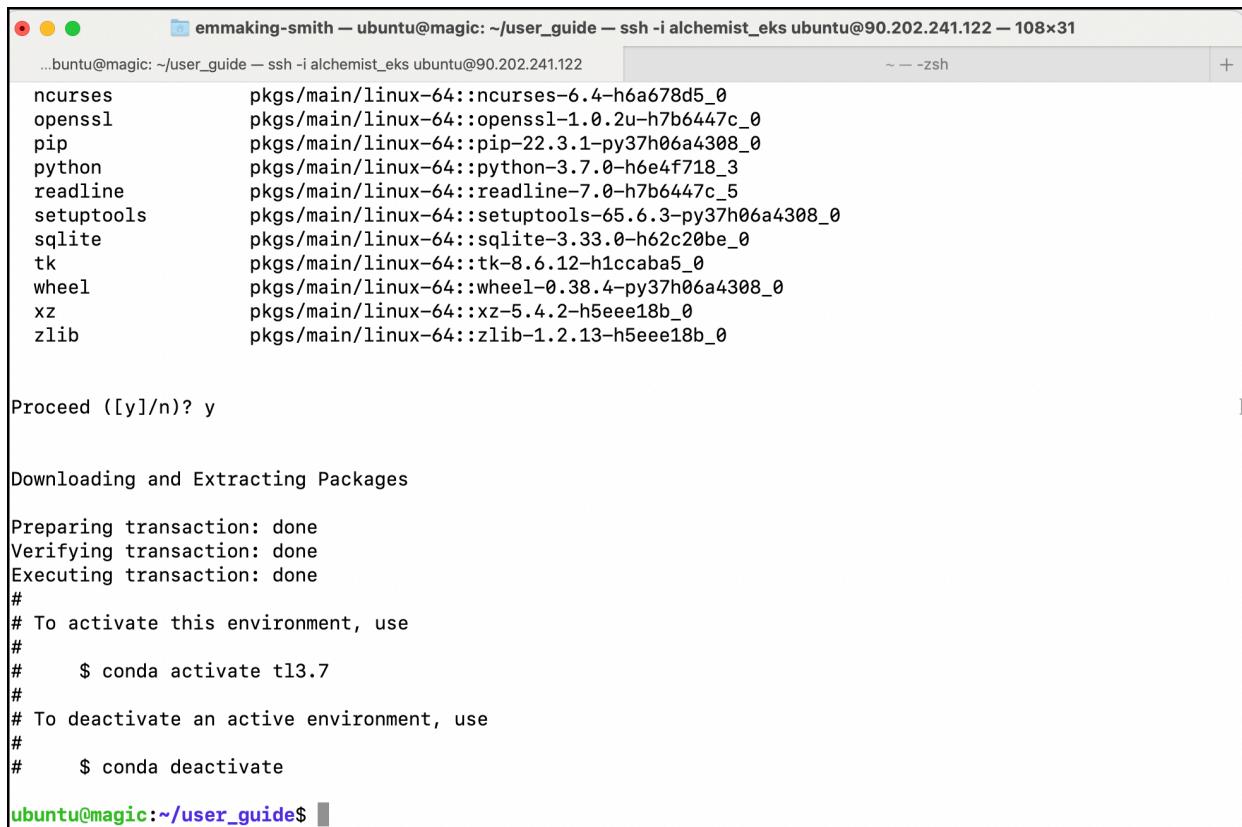
added / updated specs:
- python==3.7

The following NEW packages will be INSTALLED:

_libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main
_openmp_mutex       pkgs/main/linux-64::_openmp_mutex-5.1-1_gnu
ca-certificates    pkgs/main/linux-64::ca-certificates-2023.08.22-h06a4308_0
certifi             pkgs/main/linux-64::certifi-2022.12.7-py37h06a4308_0
libedit             pkgs/main/linux-64::libedit-3.1.20221030-h5eee18b_0
libffi              pkgs/main/linux-64::libffi-3.2.1-hf484d3e_1007
libgcc-ng           pkgs/main/linux-64::libgcc-ng-11.2.0-h1234567_1
libgomp             pkgs/main/linux-64::libgomp-11.2.0-h1234567_1
libstdcxx-ng        pkgs/main/linux-64::libstdcxx-ng-11.2.0-h1234567_1
ncurses             pkgs/main/linux-64::ncurses-6.4-h6a678d5_0
openssl             pkgs/main/linux-64::openssl-1.0.2u-h7b6447c_0
pip                 pkgs/main/linux-64::pip-22.3.1-py37h06a4308_0
python               pkgs/main/linux-64::python-3.7.0-h6e4f718_3
readline             pkgs/main/linux-64::readline-7.0-h7b6447c_5
setupuptools         pkgs/main/linux-64::setuptools-65.6.3-py37h06a4308_0
sqlite              pkgs/main/linux-64::sqlite-3.33.0-h62c20be_0
tk                  pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0
wheel               pkgs/main/linux-64::wheel-0.38.4-py37h06a4308_0
xz                  pkgs/main/linux-64::xz-5.4.2-h5eee18b_0
zlib                pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0

Proceed ([y]/n)? y
```

The output of these commands will look something like this:



emmaking-smith — ubuntu@magic: ~/user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31

```
...buntu@magic:~/user_guide$ ssh -i alchemist_eks ubuntu@90.202.241.122
~ -- zsh

ncurses      pkgs/main/linux-64::ncurses-6.4-h6a678d5_0
openssl      pkgs/main/linux-64::openssl-1.0.2u-h7b6447c_0
pip          pkgs/main/linux-64::pip-22.3.1-py37h06a4308_0
python        pkgs/main/linux-64::python-3.7.0-h6e4f718_3
readline      pkgs/main/linux-64::readline-7.0-h7b6447c_5
setuptools   pkgs/main/linux-64::setuptools-65.6.3-py37h06a4308_0
sqlite        pkgs/main/linux-64::sqlite-3.33.0-h62c20be_0
tk            pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0
wheel         pkgs/main/linux-64::wheel-0.38.4-py37h06a4308_0
xz            pkgs/main/linux-64::xz-5.4.2-h5eee18b_0
zlib          pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0

Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate tl3.7
#
# To deactivate an active environment, use
#
#     $ conda deactivate

ubuntu@magic:~/user_guide$
```

We then activate our virtual environment.

```
conda activate tl3.7
```

You can tell that you are in a virtual environment by the leftmost text, which now says the environment name (see orange box).

```

emmaking-smith — ubuntu@magic: ~/user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 — 108x31
...buntu@magic: ~/user_guide — ssh -i alchemist_eks ubuntu@90.202.241.122 ~ — zsh

openssl      pkgs/main/linux-64::openssl-1.0.2u-h7b6447c_0
pip          pkgs/main/linux-64::pip-22.3.1-py37h06a4308_0
python        pkgs/main/linux-64::python-3.7.0-h6e4f718_3
readline      pkgs/main/linux-64::readline-7.0-h7b6447c_5
setuptools    pkgs/main/linux-64::setuptools-65.6.3-py37h06a4308_0
sqlite        pkgs/main/linux-64::sqlite-3.33.0-h62c20be_0
tk            pkgs/main/linux-64::tk-8.6.12-h1ccabab_0
wheel         pkgs/main/linux-64::wheel-0.38.4-py37h06a4308_0
xz            pkgs/main/linux-64::xz-5.4.2-h5eee18b_0
zlib          pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0

Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate tl3.7
#
# To deactivate an active environment, use
#
#     $ conda deactivate

ubuntu@magic:~/user_guide$ conda activate tl3.7
(tl3.7) ubuntu@magic:~/user_guide$
```

Package Installation

Next, all the relevant packages of specific versions will be installed. We specify this with the "==" sign. RDKit will be installed first.

```
conda install -y -c rdkit rdkit==2020.09.1
```

```

emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ — zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
```

```

ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ conda activate tl3.7
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ conda install -y -c rdkit rdkit==2020.09.1
Collecting package metadata (current_repodata.json): done
Solving environment: unsuccessful initial attempt using frozen solve. Retrying with flexible solve.
Solving environment: unsuccessful attempt using repodata from current_repodata.json, retrying with next repo
data source.
Collecting package metadata (repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 23.7.3
  latest version: 23.9.0

Please update conda by running

$ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

  conda install conda=23.9.0

## Package Plan ##

environment location: /home/ubuntu/miniconda3/envs/tl3.7

added / updated specs:
- rdkit==2020.09.1

```

You will see many packages being installed. The final output will look something like this:

```

emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ — zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
```

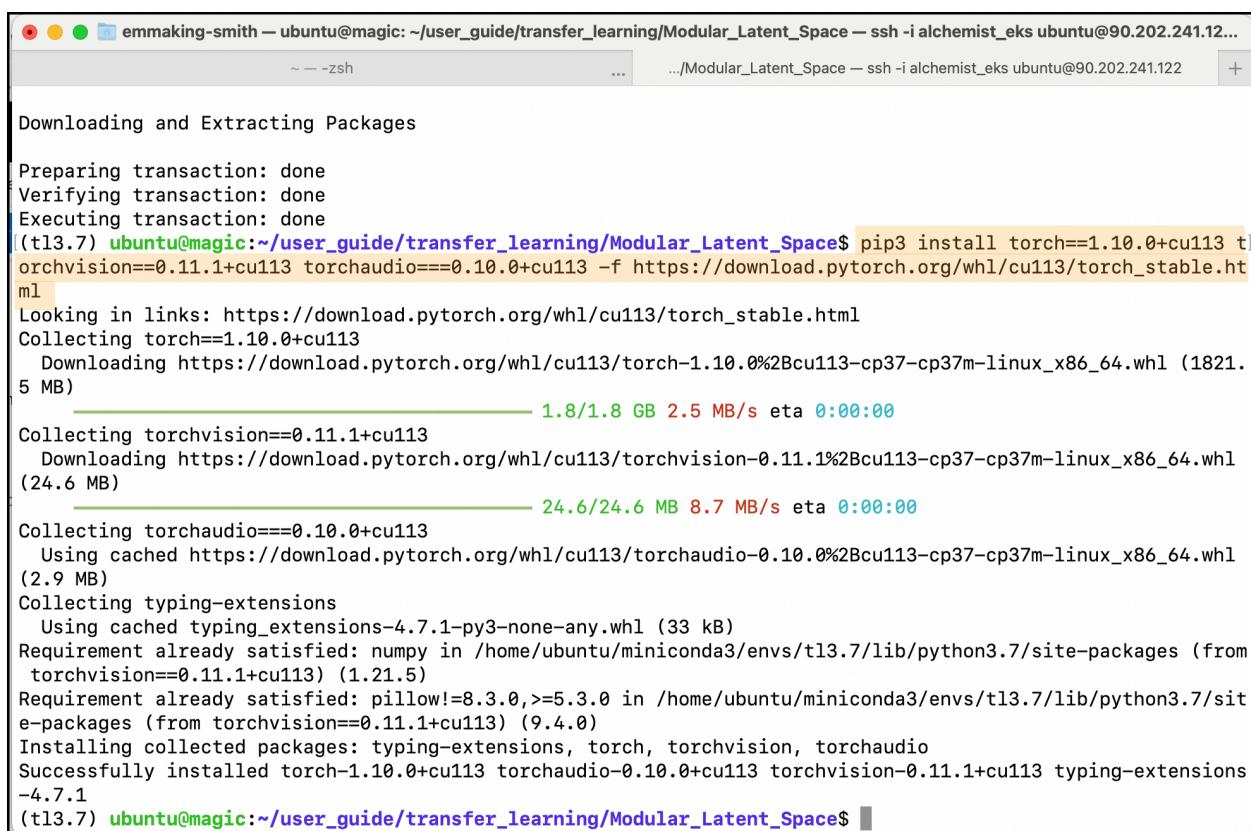
libwebp-base	pkgs/main/linux-64::libwebp-base-1.2.4-h5eee18b_1
libxcb	pkgs/main/linux-64::libxcb-1.15-h7f8727e_0
libxml2	pkgs/main/linux-64::libxml2-2.10.4-hcbfb50_0
lz4-c	pkgs/main/linux-64::lz4-c-1.9.4-h6a678d5_0
mkl	pkgs/main/linux-64::mkl-2021.4.0-h0ea4308_640
mkl-service	pkgs/main/linux-64::mkl-service-2.4.0-py37h7f8727e_0
mkl_fft	pkgs/main/linux-64::mkl_fft-1.3.1-py37hd3c417c_0
mkl_random	pkgs/main/linux-64::mkl_random-1.2.2-py37h51133e4_0
numexpr	pkgs/main/linux-64::numexpr-2.8.4-py37he184ba9_0
numpy	pkgs/main/linux-64::numpy-1.21.5-py37h6c91a56_3
numpy-base	pkgs/main/linux-64::numpy-base-1.21.5-py37ha15fc14_3
packaging	pkgs/main/linux-64::packaging-22.0-py37h06a4308_0
pandas	pkgs/main/linux-64::pandas-1.3.5-py37h8c16a72_0
pcre	pkgs/main/linux-64::pcre-8.45-h295c915_0
pillow	pkgs/main/linux-64::pillow-9.4.0-py37h6a678d5_0
pixman	pkgs/main/linux-64::pixman-0.40.0-h7f8727e_1
py-boost	pkgs/main/linux-64::py-boost-1.73.0-py37h51133e4_12
python-dateutil	pkgs/main/noarch::python-dateutil-2.8.2-pyhd3eb1b0_0
pytz	pkgs/main/linux-64::pytz-2022.7-py37h06a4308_0
rdkit	rdkit/linux-64::rdkit-2020.09.1.0-py37hd50e099_1
six	pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_0
zstd	pkgs/main/linux-64::zstd-1.5.5-hc292b87_0

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

Pytorch and other useful torch packages are installed next with:

```
pip3 install torch==1.10.0+cu113 torchvision==0.11.1+cu113
torchaudio==0.10.0+cu113 -f
https://download.pytorch.org/whl/cu113/torch_stable.html
```



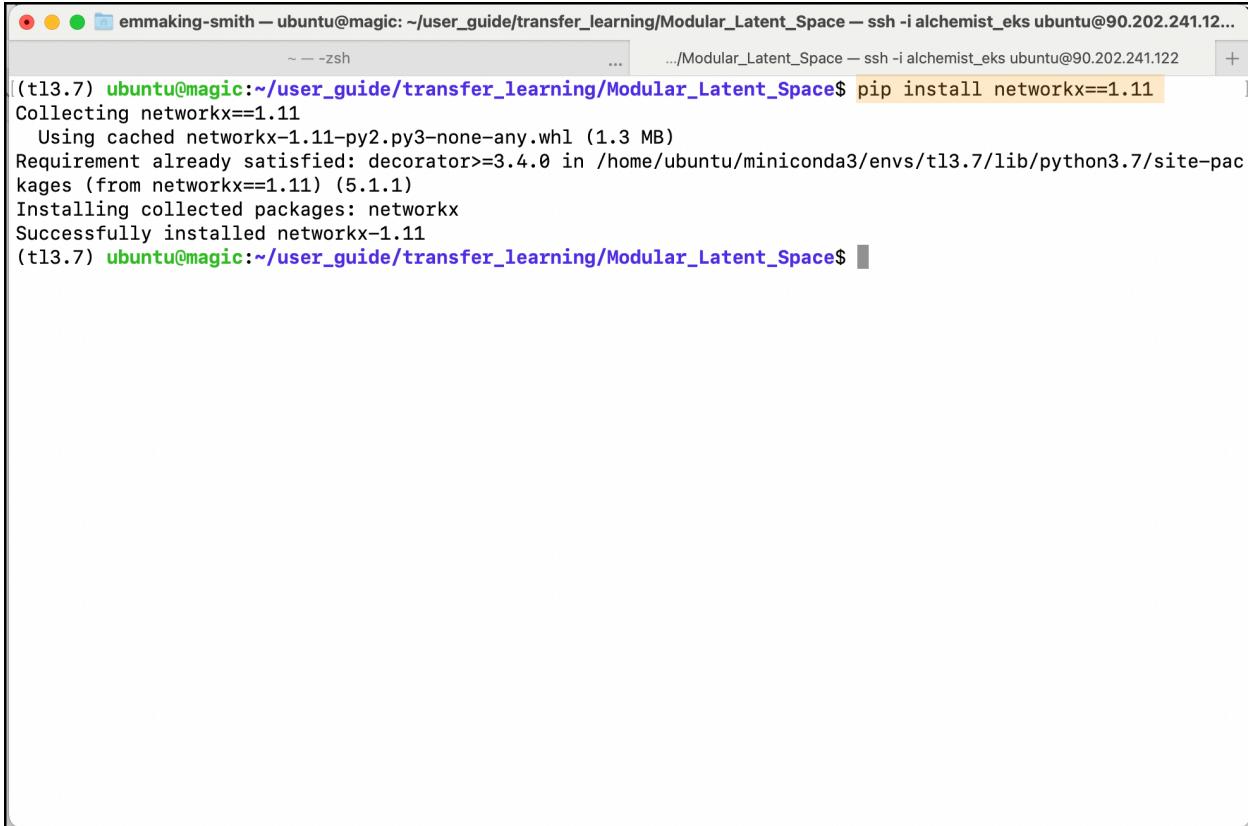
```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~/ -- zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
```

Downloading and Extracting Packages

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ pip3 install torch==1.10.0+cu113 t
orchvision==0.11.1+cu113 torchaudio==0.10.0+cu113 -f https://download.pytorch.org/whl/cu113/torch_stable.ht
ml
Looking in links: https://download.pytorch.org/whl/cu113/torch_stable.html
Collecting torch==1.10.0+cu113
  Downloading https://download.pytorch.org/whl/cu113/torch-1.10.0%2Bcu113-cp37-cp37m-linux_x86_64.whl (1821.
5 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 GB 2.5 MB/s eta 0:00:00
Collecting torchvision==0.11.1+cu113
  Downloading https://download.pytorch.org/whl/cu113/torchvision-0.11.1%2Bcu113-cp37-cp37m-linux_x86_64.whl
(24.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 8.7 MB/s eta 0:00:00
Collecting torchaudio==0.10.0+cu113
  Using cached https://download.pytorch.org/whl/cu113/torchaudio-0.10.0%2Bcu113-cp37-cp37m-linux_x86_64.whl
(2.9 MB)
Collecting typing-extensions
  Using cached typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Requirement already satisfied: numpy in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from
torchvision==0.11.1+cu113) (1.21.5)
Requirement already satisfied: pillow!=8.3.0,>=5.3.0 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-pa
ckages (from torchvision==0.11.1+cu113) (9.4.0)
Installing collected packages: typing-extensions, torch, torchvision, torchaudio
Successfully installed torch-1.10.0+cu113 torchaudio-0.10.0+cu113 torchvision-0.11.1+cu113 typing-extensions
-4.7.1
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$
```

Then networkx is installed with:

```
pip install networkx==1.11
```



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ — zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ pip install networkx==1.11
Collecting networkx==1.11
  Using cached networkx-1.11-py2.py3-none-any.whl (1.3 MB)
Requirement already satisfied: decorator>=3.4.0 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from networkx==1.11) (5.1.1)
Installing collected packages: networkx
Successfully installed networkx-1.11
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$
```

This is the most critical package to get the correct version. After version 1.11, the graph nomenclature changed and attempting to run the code with later versions will result in an error. **If you believe you are seeing a networkx error, please double check that the version you are running is 1.11.** This can easily be done using the following commands.

Start python - make sure you are in your virtual environment. Type:

```
python
```

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ -- zsh ... .../Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ python
Python 3.7.0 (default, Oct  9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

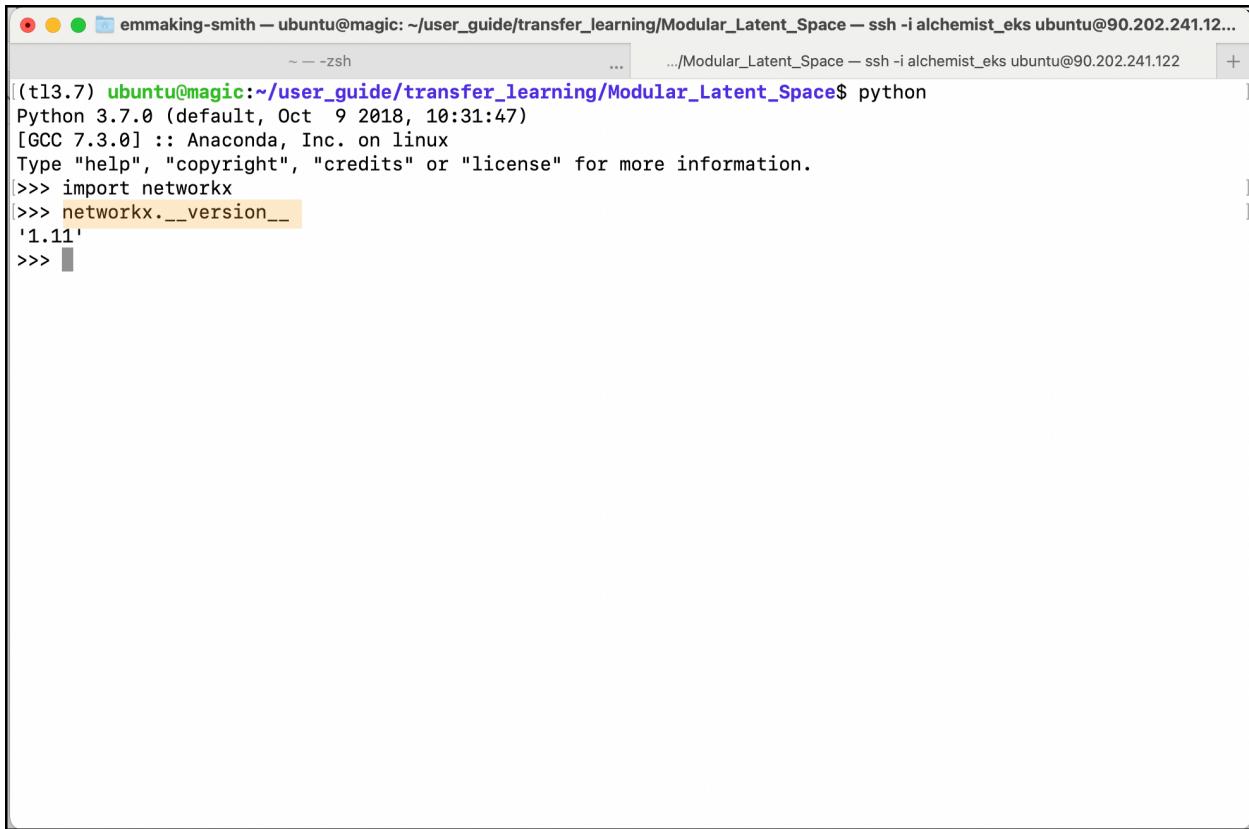
Import networkx with:

```
import networkx
```

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ -- zsh ... .../Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ python
Python 3.7.0 (default, Oct  9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import networkx
>>> 
```

Check version with:

```
networkx.__version__
```



The screenshot shows a terminal window with a dark background and light-colored text. At the top, it displays the command line information: 'emmakingsmith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...'. Below this, the Python interpreter is running. It starts with 'Python 3.7.0 (default, Oct 9 2018, 10:31:47) [GCC 7.3.0] :: Anaconda, Inc. on linux'. Then, it shows the standard help message: 'Type "help", "copyright", "credits" or "license" for more information.' Following this, the user types three commands: '|>>> import networkx', '|>>> networkx.__version__', and '|>>>'. The final output is '|>>> '1.11''. The line '|>>> networkx.__version__' is highlighted with a yellow box.

You should have an output of 1.11.

Next install numpy. This may have already been installed with a previous package.

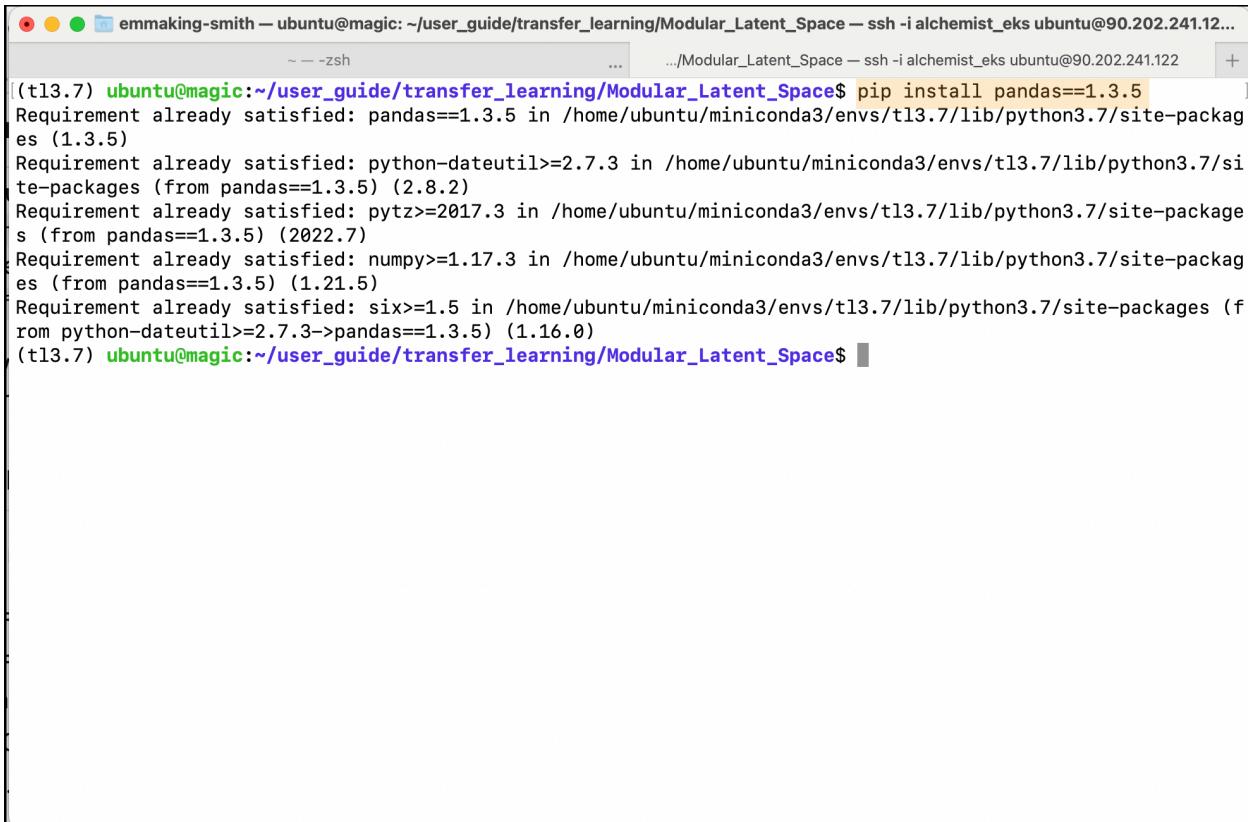
```
pip install numpy==1.21.5
```



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ — zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ pip install numpy==1.21.5
Requirement already satisfied: numpy==1.21.5 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (1.21.5)
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$
```

Then install pandas. Similar to numpy, this may already have been accomplished with a previous package installation.

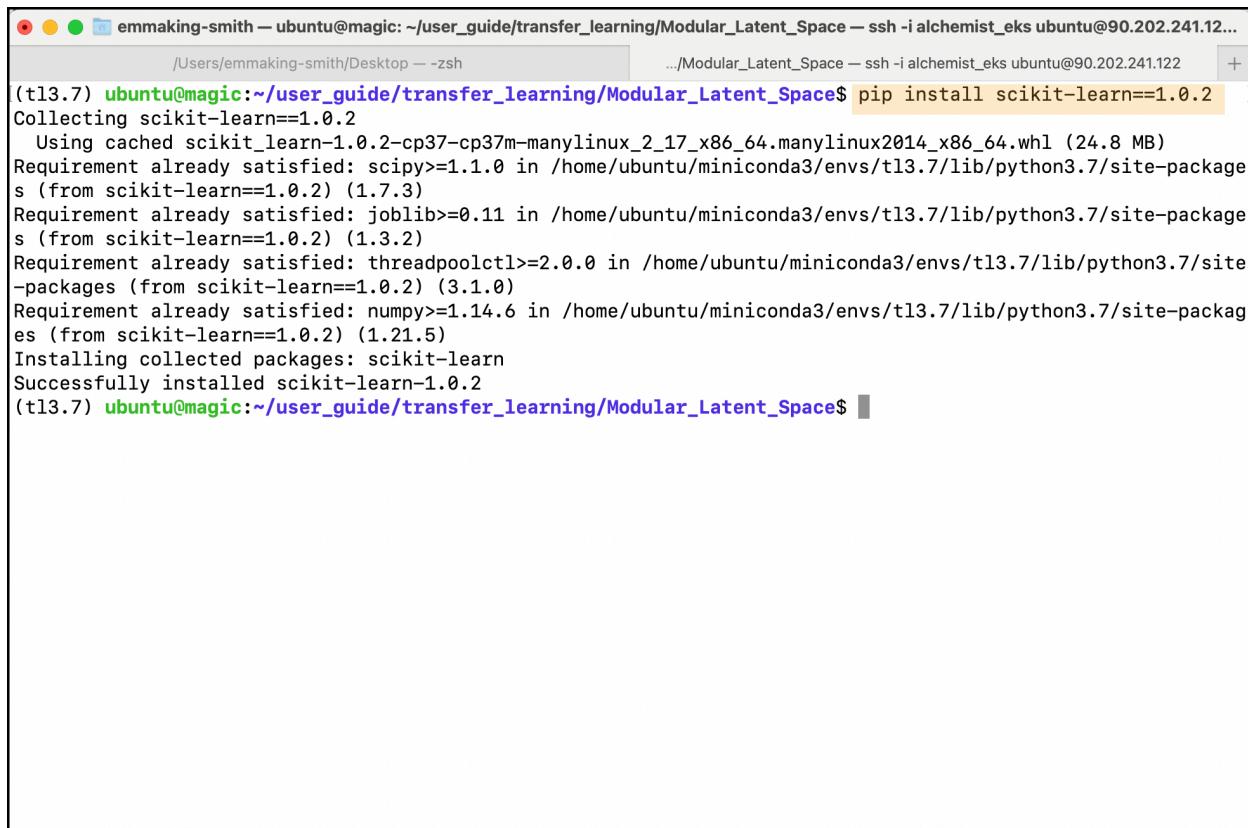
```
pip install pandas==1.3.5
```



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
~ — zsh ... ./Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ pip install pandas==1.3.5
Requirement already satisfied: pandas==1.3.5 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (1.3.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from pandas==1.3.5) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from pandas==1.3.5) (2022.7)
Requirement already satisfied: numpy>=1.17.3 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from pandas==1.3.5) (1.21.5)
Requirement already satisfied: six>=1.5 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from python-dateutil>=2.7.3->pandas==1.3.5) (1.16.0)
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$
```

Finally, install sklearn.

```
pip install scikit-learn==1.0.2
```



The screenshot shows a terminal window with two tabs. The active tab is titled 'Modular_Latent_Space' and contains the command 'pip install scikit-learn==1.0.2'. The output of the command is displayed below the command line, showing the progress of the download and installation of various dependencies. The terminal is running on an Ubuntu system via SSH, with the path '/user_guide/transfer_learning/Modular_Latent_Space' visible at the top.

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.12...
/Users/emmaking-smith/Desktop — zsh .../Modular_Latent_Space — ssh -i alchemist_eks ubuntu@90.202.241.122 +  
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$ pip install scikit-learn==1.0.2  
Collecting scikit-learn==1.0.2  
  Using cached scikit_learn-1.0.2-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (24.8 MB)  
Requirement already satisfied: scipy>=1.1.0 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from scikit-learn==1.0.2) (1.7.3)  
Requirement already satisfied: joblib>=0.11 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from scikit-learn==1.0.2) (1.3.2)  
Requirement already satisfied: threadpoolctl>=2.0.0 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from scikit-learn==1.0.2) (3.1.0)  
Requirement already satisfied: numpy>=1.14.6 in /home/ubuntu/miniconda3/envs/tl3.7/lib/python3.7/site-packages (from scikit-learn==1.0.2) (1.21.5)  
Installing collected packages: scikit-learn  
Successfully installed scikit-learn-1.0.2  
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space$
```

We are now all set up!

Run the Transfer Learning

We will be using the Buchwald-Hartwig dataset as our example transfer learning. Note that the pretrained layers from the crystal structure information will be frozen. This has been done for you and can be observed in Modular_Latent_Space/buchwald/buchwald_yield_mpnn lines 29-30 (see orange box in the figure below).

```

13     device = 'cuda' if torch.cuda.is_available() else 'cpu'
14
15     class Buchwald_MPNN(nn.Module):
16         def __init__(self, message_size, message_passes, atom_list, pretrained_mpnn_path):
17             super(Buchwald_MPNN, self).__init__()
18
19             self.message_size = message_size
20             self.message_passes = message_passes
21             self.atom_list = atom_list
22             self.pretrained_mpnn_path = pretrained_mpnn_path
23
24             self.mpnn = Big_MPNN(self.message_size, self.message_passes, self.atom_list)
25             mpnn_trained_state_dict = self.gen_states(self.pretrained_mpnn_path)
26             self.mpnn.load_state_dict(mpnn_trained_state_dict)
27
28             # Turning off the params.
29             for param in self.mpnn.parameters():
30                 param.requires_grad = False
31
32             self.yield_predictor = nn.Sequential(
33                 nn.Linear(self.message_size * 4, self.message_size * 4),
34                 nn.ReLU(),
35                 # nn.Linear(self.message_size * 4, self.message_size * 4),
36                 # nn.ReLU(),
37                 # nn.Linear(self.message_size * 4, self.message_size * 4),
38                 # nn.ReLU(),
39                 # nn.Linear(self.message_size * 4, self.message_size * 4).

```

To run the transfer learning, we run a variation of following command:

```
python py_file_that_makes_predictions.py --option1 choice --option2 choice ...
```

The "--" indicates a flag. We are telling the model what arguments we are inputting. The flag lets the program know that a user-defined selection will occur, and the words after it detail the selection.

Understanding the Buchwald-Hartwig Flags:

For the Buchwald-Hartwig transfer learning, we have 3 flags of importance: split, test_mol_idx, and save_path. The split refers to what type of molecule should be left out for model validation. The options are "halide", "base", "ligand", and "additive". They are case sensitive.

The test_mol_idx stands for test molecule index and refers to which halide / base / ligand / additive molecules should be left out for model validation. Each index will yield a different set of molecules.

The final flag, save_path, is the place we wish to save our model and predictions. If we want to save it in a new directory called "predictions" to our Desktop our save_path flag would look like:

```
--save_path ~/Desktop/predictions.
```

To summarize our flags:

Flag name	Expected Value(s)	Example Flag Value	What to type
split	one of the following: halide,	ligand	--split ligand

	base, ligand, additive. NOTE: Case sensitive!		
test_mol_idx	An integer between 0 and 3 if your split is NOT base. An integer between 0 and 2 if your split IS base.	3* *Must not have --split base flag.	--test_mol_idx 3
save_path	A path to a directory.	A new directory in the Modular_Latent_Space directory called transfer_learning_test.	--save_path transfer_learning_test NOTE: Assuming your current location is the Modular_Latent_Space directory.

For more flags, please refer to Modular_Latent_Space/buchwald/buchwald_yield_prediction.py lines 25 - 37. For a basic transfer learning, feel free to use the default options.

```

19     from buchwald_yield_mpnn import Buchwald_MPNN
20
21     class Initialization:
22         def __init__(self):
23             pass
24
25     def init_args(self):
26         parser = argparse.ArgumentParser()
27         parser.add_argument('--path_to_buchwald_data', type=str, help='The path to the Buchwald csv file.',
28                             default='doyle_buchwald_data.csv')
29         parser.add_argument('--epochs', type=int, default=100)
30         parser.add_argument('--learning_rate', type=float, default=1e-4)
31         parser.add_argument('--batch_size', type=int, default=64)
32         parser.add_argument('--message_size', type=int, default=128)
33         parser.add_argument('--message_passes', type=int, default=3)
34         parser.add_argument('--pretrained_mpnn_path', type=str, default='../MPNN/big_mpnn_no_delocalised_no_unknown_model')
35         parser.add_argument('--save_path', '-s', type=str)
36         parser.add_argument('--split', type=str, help='Options are: halide, ligand, base, additive.')
37         parser.add_argument('--test_mol_idx', type=int, help='What molecule to leave out for testing.')
38
39         return parser.parse_args()

```

Run the Transfer Learning (Finally!):

We will then run the transfer learning on the Buchwald-Hartwig dataset. The module to do so is called buchwald_yield_prediction.py and is located in the buchwald directory. Move yourself into the buchwald directory using the cd command. If you are currently in the Modular_Latent_Space directory, this can be achieved with:

```
cd buchwald
```

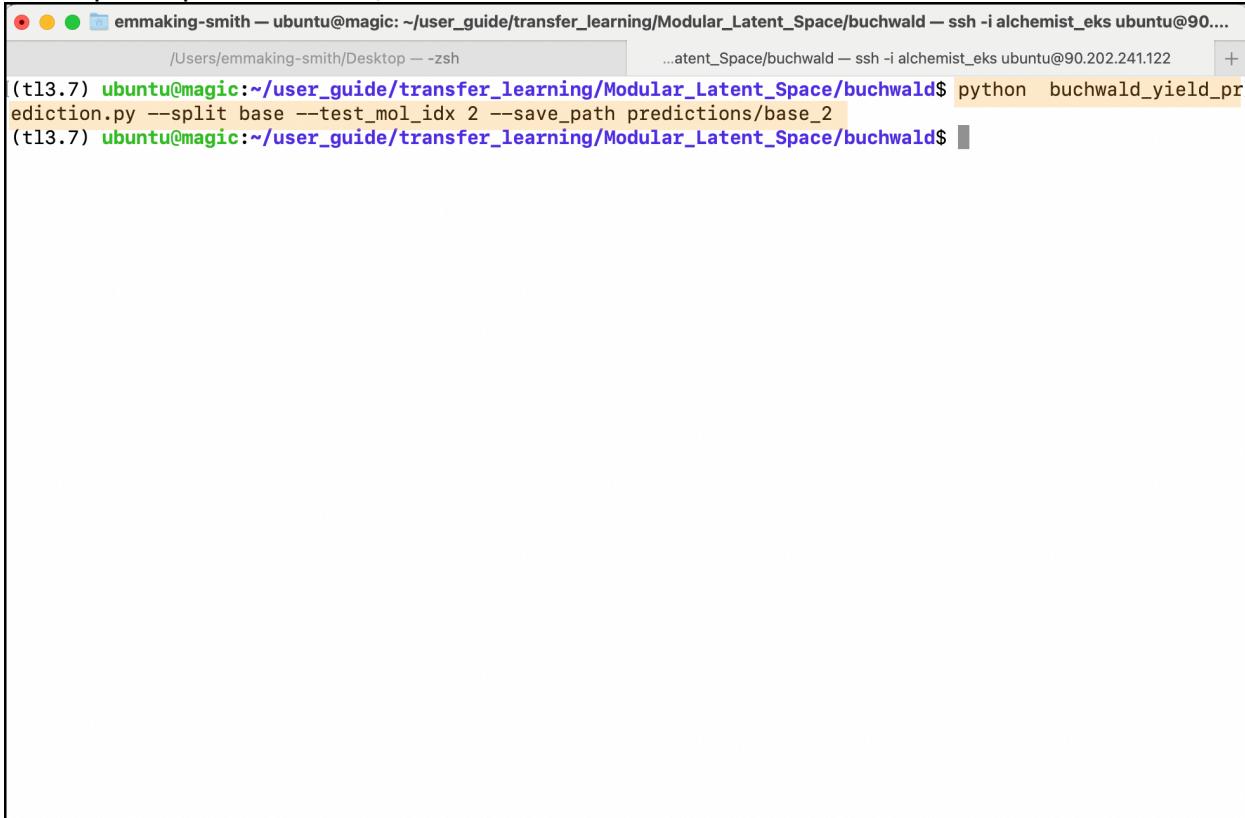
You can easily tell you are now in the buchwald directory by looking at the path in blue. The final name will be "buchwald" (see orange box in figure below).



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald — ssh -i alchemist_eks ubuntu@90....  
/Users/emmaking-smith/Desktop — zsh ...atent_Space/buchwald — ssh -i alchemist_eks ubuntu@90.202.241.122 +  
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ cd buchwald  
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$
```

To run the Buchwald-Hartwig predictions, using a base split, with split index 2, and saving the predictions to a new directory called predictions/base_2, type:

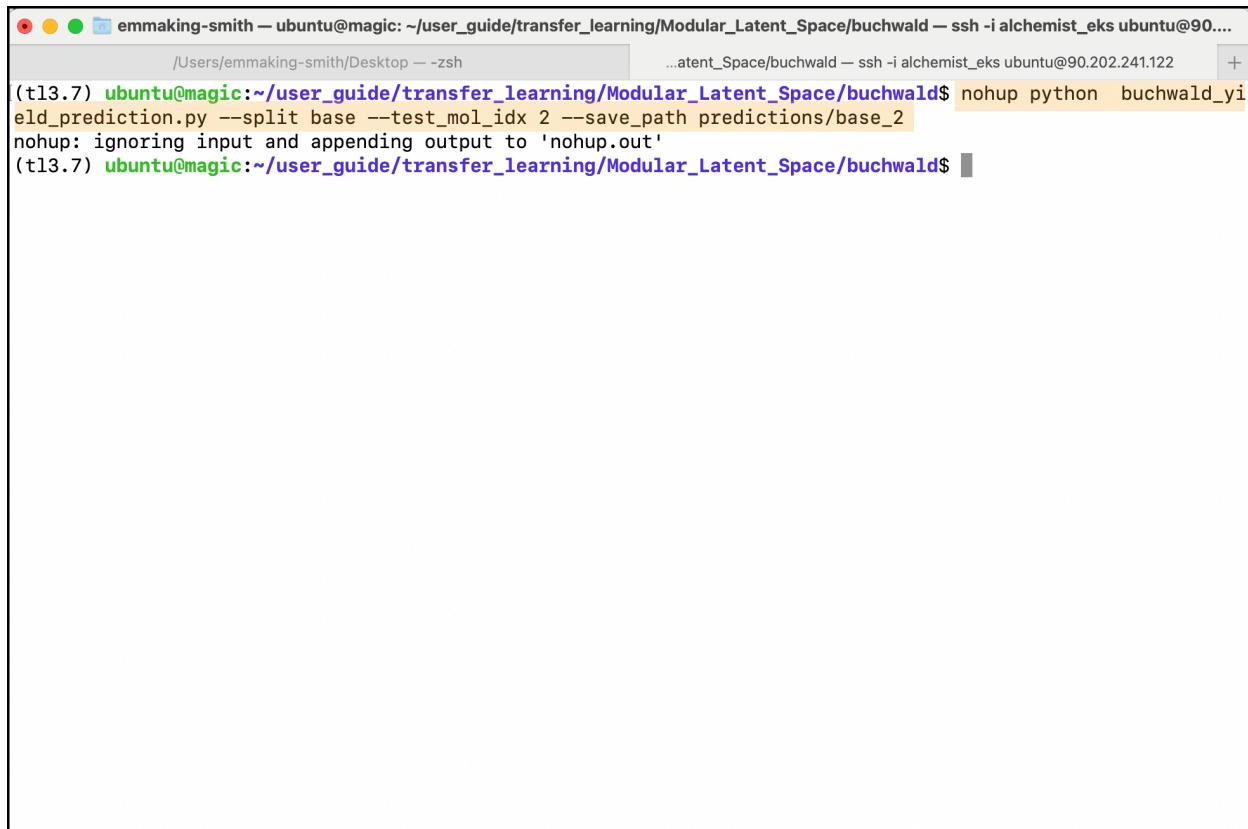
```
python buchwald_yield_prediction.py --split base --test_mol_idx 2 --  
save_path predictions/base_2
```



```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald — ssh -i alchemist_eks ubuntu@90....  
/Users/emmaking-smith/Desktop — zsh ...atent_Space/buchwald — ssh -i alchemist_eks ubuntu@90.202.241.122 +  
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ python buchwald_yield_pr  
ediction.py --split base --test_mol_idx 2 --save_path predictions/base_2  
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$
```

Helpful hint: If you are running these computations through an ssh, adding the nohup command to the beginning will keep your training running even if you lose the ssh connection:

```
nohup python buchwald_yield_prediction.py --split base --test_mol_idx 2 --  
save_path predictions/base_2
```



The screenshot shows a terminal window with two tabs. The active tab displays the command: `nohup python buchwald_yield_prediction.py --split base --test_mol_idx 2 --save_path predictions/base_2`. The output of the command is shown below the command line, indicating that nohup is ignoring input and appending output to 'nohup.out'.

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald — ssh -i alchemist_eks ubuntu@90...  
/Users/emmaking-smith/Desktop — zsh  
...atent_Space/buchwald — ssh -i alchemist_eks ubuntu@90.202.241.122 +  
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ nohup python buchwald_yield_prediction.py --split base --test_mol_idx 2 --save_path predictions/base_2  
nohup: ignoring input and appending output to 'nohup.out'  
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$
```

You can track your progress with the `model_log.log` (saved to the `save_path` directory) and the nohup output file (default `nohup.out`).

Voila! Your predictions will be saved to `predictions/base_2`.

The screenshot shows a terminal window with two tabs. The left tab is titled '/Users/emmaking-smith/Desktop — zsh' and contains the command 'ls'. The right tab is titled '...atent_Space/buchwald — ssh -i alchemist_eks ubuntu@90.202.241.122' and contains the following command-line session:

```
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ python buchwald_yield_prediction.py --split base --test_mol_idx 2 --save_path predictions/base_2
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ ls predictions/base_2/
model  model_log.log  preds.pickle
(tl3.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$
```

The preds.pickle are the predicted values, the model file is the final trained model, and the model_log.log is the log file that gives more information regarding the training loss and model parameters.

Viewing the Predictions

Ensure that you are in the correct directory (your save_path directory). If you are in the buchwald directory after completing training and have been following the naming convention of this document, you can move to the save_path directory by typing:

```
cd predictions/base_2/
```

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemi...
/Users/emmaking-smith/Desktop — zsh ...ald/predictions/base_2 — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald$ cd predictions/base_2/
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2$
```

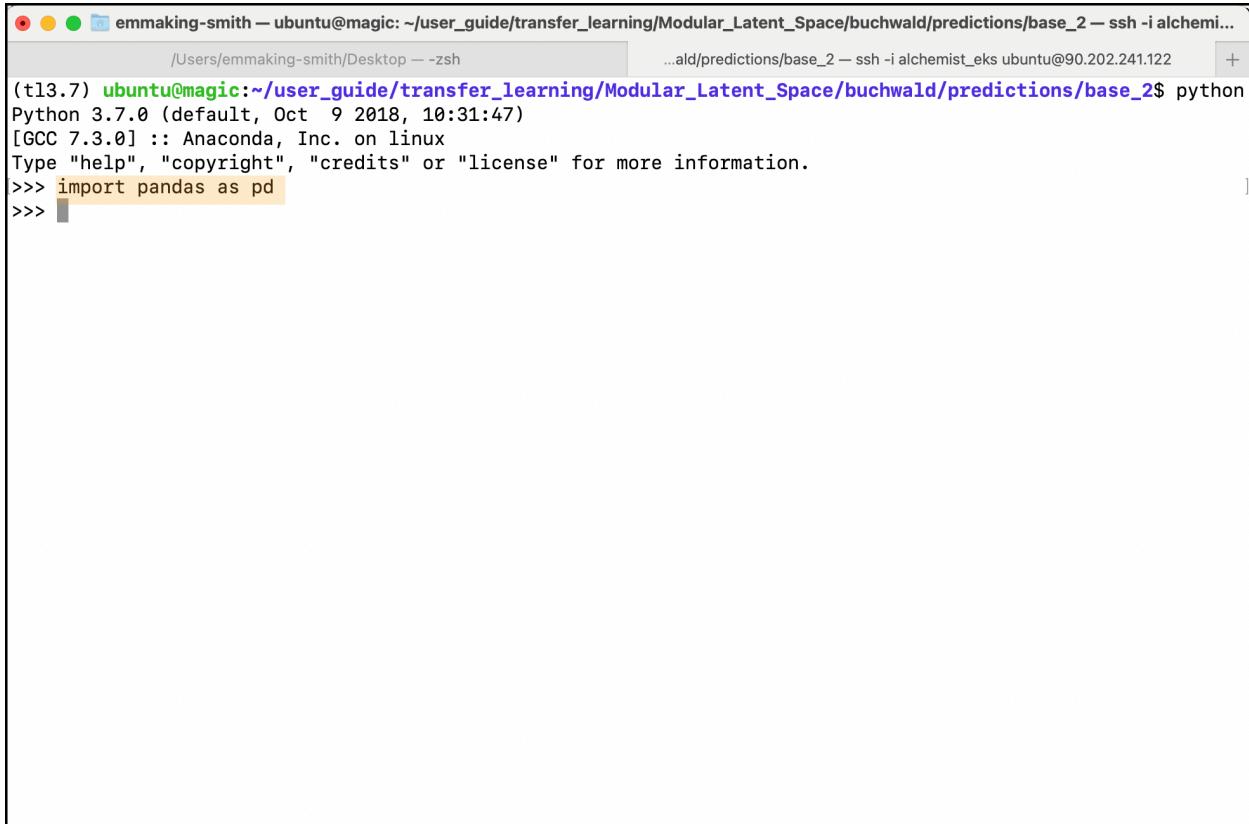
To view pickle files, go into python by typing:

python

```
emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemi...
/Users/emmaking-smith/Desktop — zsh ...ald/predictions/base_2 — ssh -i alchemist_eks ubuntu@90.202.241.122 +
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2$ python
Python 3.7.0 (default, Oct  9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Import pandas with:

```
import pandas as pd
```

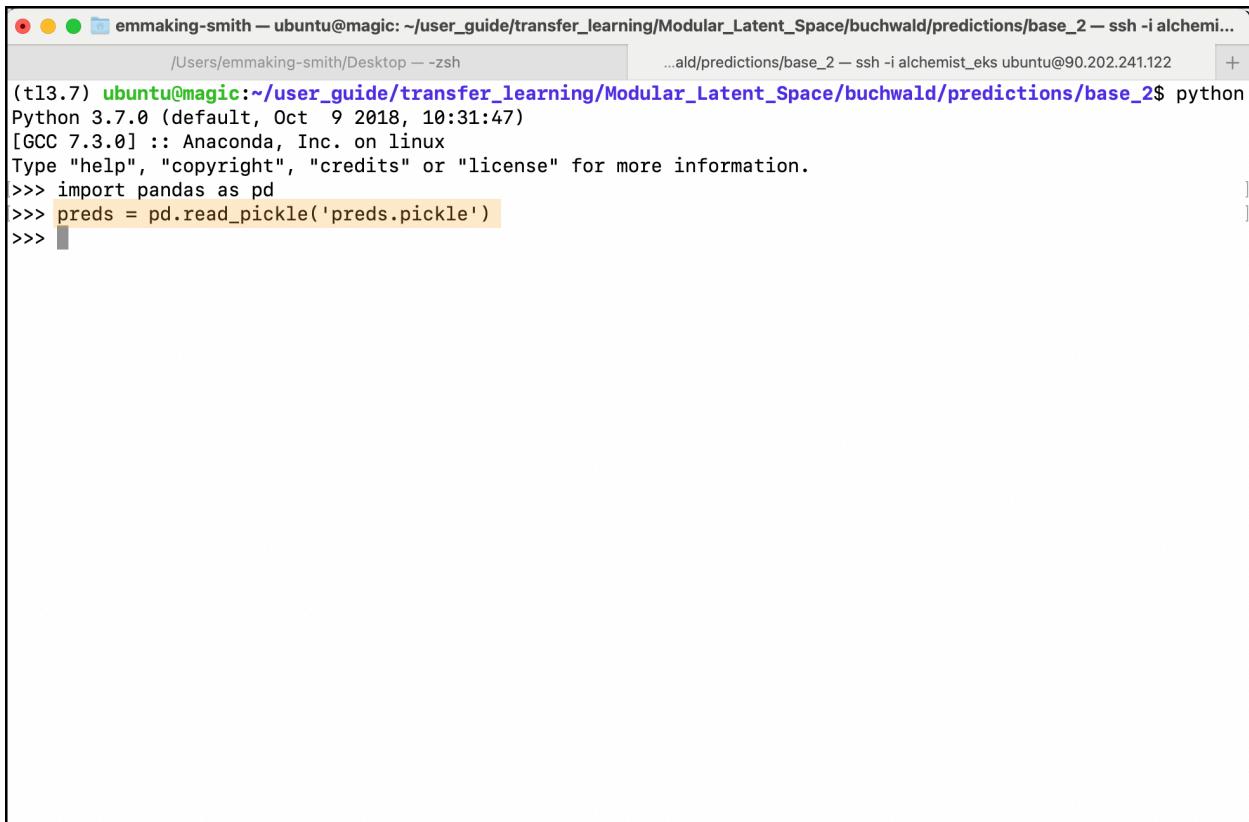


A screenshot of a terminal window titled "emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemi...". The window shows a command-line interface with the following text:

```
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2$ python
Python 3.7.0 (default, Oct  9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pandas as pd
>>>
```

Import your predictions file with:

```
preds = pd.read_pickle('preds.pickle')
```

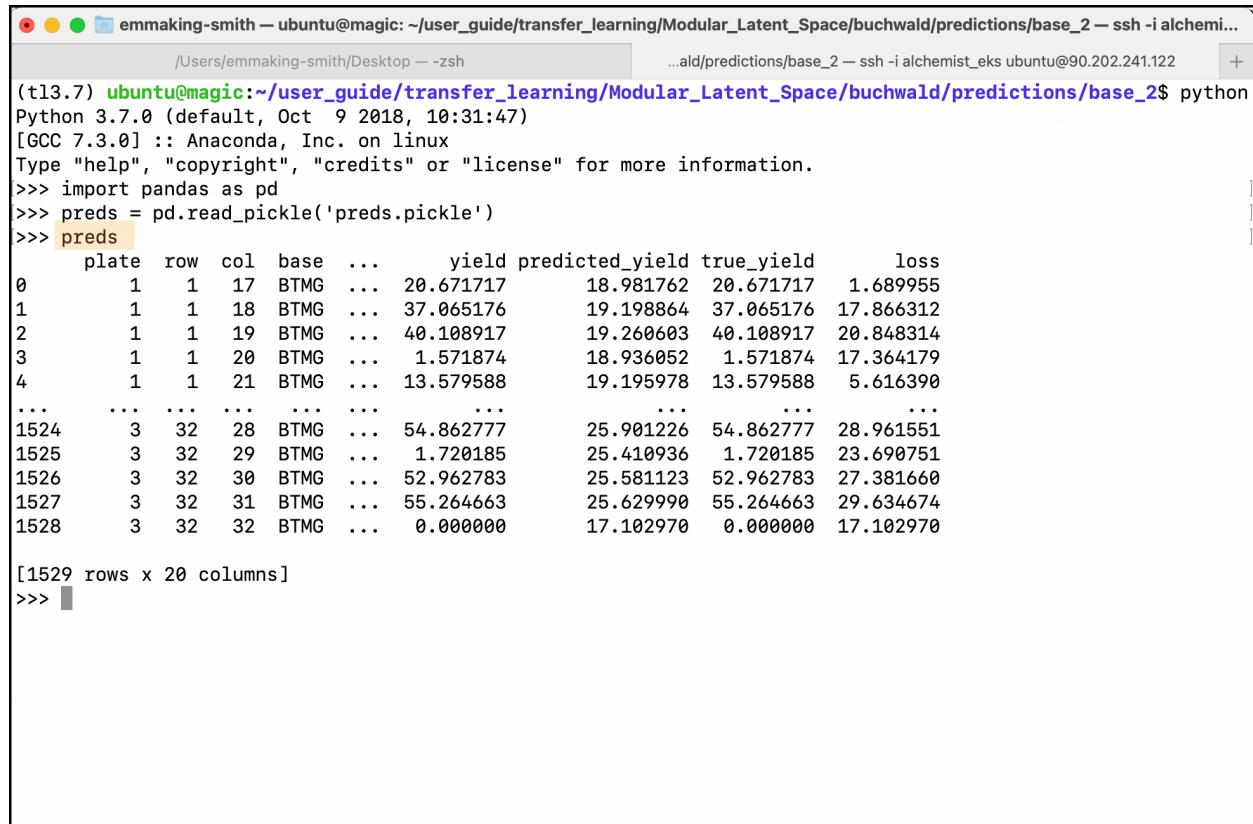


A screenshot of a terminal window titled "emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemi...". The window shows a command-line interface with the following text:

```
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2$ python
Python 3.7.0 (default, Oct  9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import pandas as pd
>>> preds = pd.read_pickle('preds.pickle')
>>>
```

View your predictions file with:

```
preds
```



The screenshot shows a terminal window with the following content:

```
emmaking-smith — ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemist_eks ubuntu@90.202.241.122 +  
/Users/emmaking-smith/Desktop — zsh  
(t13.7) ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2$ python  
Python 3.7.0 (default, Oct  9 2018, 10:31:47)  
[GCC 7.3.0] :: Anaconda, Inc. on linux  
Type "help", "copyright", "credits" or "license" for more information.  
>>> import pandas as pd  
>>> preds = pd.read_pickle('preds.pickle')  
>>> preds  
   plate  row  col  base  ...      yield predicted_yield  true_yield      loss  
0       1    1   17  BTMG  ...  20.671717    18.981762  20.671717  1.689955  
1       1    1   18  BTMG  ...  37.065176    19.198864  37.065176  17.866312  
2       1    1   19  BTMG  ...  40.108917    19.260603  40.108917  20.848314  
3       1    1   20  BTMG  ...  1.571874    18.936052  1.571874  17.364179  
4       1    1   21  BTMG  ...  13.579588    19.195978  13.579588  5.616390  
...     ...  ...  ...  ...      ...        ...        ...        ...  
1524    3    32   28  BTMG  ...  54.862777    25.901226  54.862777  28.961551  
1525    3    32   29  BTMG  ...  1.720185    25.410936  1.720185  23.690751  
1526    3    32   30  BTMG  ...  52.962783    25.581123  52.962783  27.381660  
1527    3    32   31  BTMG  ...  55.264663    25.629990  55.264663  29.634674  
1528    3    32   32  BTMG  ...  0.000000    17.102970  0.000000  17.102970  
[1529 rows x 20 columns]  
>>>
```

This will give you a snapshot of the prediction file. You can save it out as a csv file which is openable with Microsoft Excel if that is more convenient for you with:

```
preds.to_csv('preds.csv')
```

```

emmaking-smith — ubuntu@magic: ~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2 — ssh -i alchemi...
/Users/emmaking-smith/Desktop — zsh ...ald/predictions/base_2 — ssh -i alchemist_eks ubuntu@90.202.241.122 +
```

(t13.7) **ubuntu@magic:~/user_guide/transfer_learning/Modular_Latent_Space/buchwald/predictions/base_2\$** python

Python 3.7.0 (default, Oct 9 2018, 10:31:47)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.

```

>>> import pandas as pd
>>> preds = pd.read_pickle('preds.pickle')
>>> preds
```

	plate	row	col	base	...	yield	predicted_yield	true_yield	loss
0	1	1	17	BTMG	...	20.671717	18.981762	20.671717	1.689955
1	1	1	18	BTMG	...	37.065176	19.198864	37.065176	17.866312
2	1	1	19	BTMG	...	40.108917	19.260603	40.108917	20.848314
3	1	1	20	BTMG	...	1.571874	18.936052	1.571874	17.364179
4	1	1	21	BTMG	...	13.579588	19.195978	13.579588	5.616390
...
1524	3	32	28	BTMG	...	54.862777	25.901226	54.862777	28.961551
1525	3	32	29	BTMG	...	1.720185	25.410936	1.720185	23.690751
1526	3	32	30	BTMG	...	52.962783	25.581123	52.962783	27.381660
1527	3	32	31	BTMG	...	55.264663	25.629990	55.264663	29.634674
1528	3	32	32	BTMG	...	0.000000	17.102970	0.000000	17.102970

[1529 rows x 20 columns]

```

>>> preds.to_csv('preds.csv')
>>>
```

This will generate a csv file in your working directory called preds.

Thank you for reading to the end of this guide!

We hope it has been a helpful resource for you.