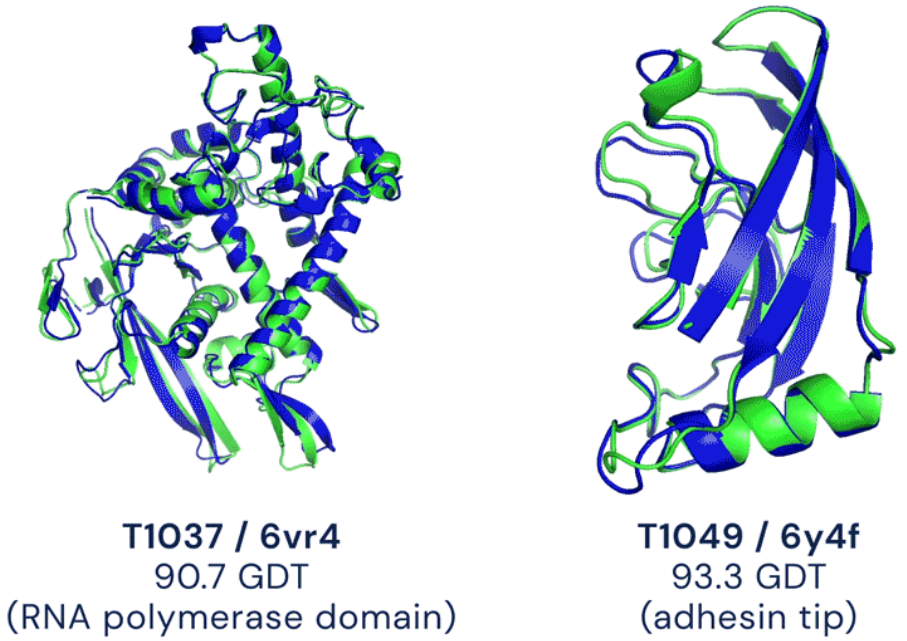
540.414/614: Protein Structure Prediction and Design

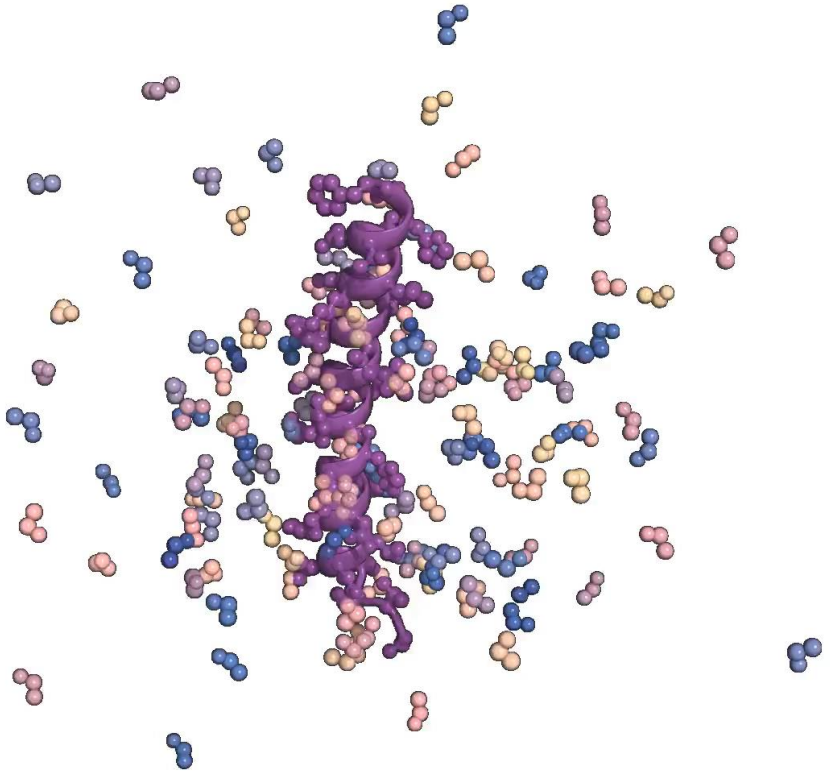
Recent Breakthroughs

AlphaFold2 (Google DeepMind)



Jumper et al / DeepMind.com (2020)

RFDiffusion (U. Washington)



Watson & Juergens et al. Nature 2023

540.414/614: Protein Structure Prediction and Design

1. Amino acids – *PyMOL*
2. Protein geometry – *PyRosetta*
3. Protein energies and forces – *PyRosetta*
4. Intro to deep neural networks – *NumPy*
5. Convolutional neural networks (CNNs) – *PyTorch*
6. Language models (GPT) – *ESM*, *ProGen*
7. Multi-track equivariant attention models – *AlphaFold*, *RosettaFold*
8. Graph neural networks (GNNs) – *ProteinMPNN*
9. Denoising diffusion probabilistic models (DDPM) – *RFDiffusion*

Teaching Challenges

- Diversity of student backgrounds
  - Engineers, biologists, bioinformatics MS, (CS/applied math)
  - *Strategy: Specifications grading*
- Fundamental depth vs application
  - *Strategy: Lecture + Workshop format (Tues/Thurs)*
- Workshops: pre-built to run vs. code-on-your-own
  - *Strategy: make it up as I go* 🤷

We are using *specifications grading* this semester. Problems are divided into four levels:

Problem Level	Problem Type Description	Estimated Time	Number for "A" Grade	Grading System	Gradescope Encoding
L1	L1 problems focus on your conceptual or quantitative understanding of a given topic.	5-10 mins	28	S/U	1
L2	L2 problems focus on using your conceptual, qualitative understanding to find/apply a next-obvious step in a topic.	10-30 mins	15	EMRN	10
L3	L3 problems focus on applying course knowledge to topics within the field with a medium-large level of effort. Could involve writing a short computer program or doing a long set of calculations.	30-60 mins	4	EMRN	100
L4	L4 problems require the application of course and self-taught knowledge to challenging, time-intensive problems. Could involve longer coding problems requiring creativity and experimentation.	1-4 hours	4	EMRN	1000
L2 – E	To achieve an "Excellent/Exemplary" on a problem requires a combination of: • Clear documented code following PEP8 guidelines • Exploration around periphery of assigned calculations/code/topic that shows in depth understanding		For 540.614, 50% of problems		0.001
L3 – E					0.01
L4 – E					0.1

ChemBE 414/614 **Homework #4** **Due: 2/20/2023**

Tasks to complete (ungraded):

1. Complete *Neural Networks from Scratch Workshop*.
2. Review papers by Higham & Higham (2019) and Gao *et al.* (2020).
3. For next week, review papers for trRosetta (2020), RaptorX (2019), and/or DeepAb (2022).

Homework problems you may submit:

1. [L1] Consider a neural net with 3 inputs, 4 nodes in the first layer, 4 nodes in the second layer, 3 nodes in the third layer that produce three outputs. How many parameters will be in this model? Draw a picture and show your work.

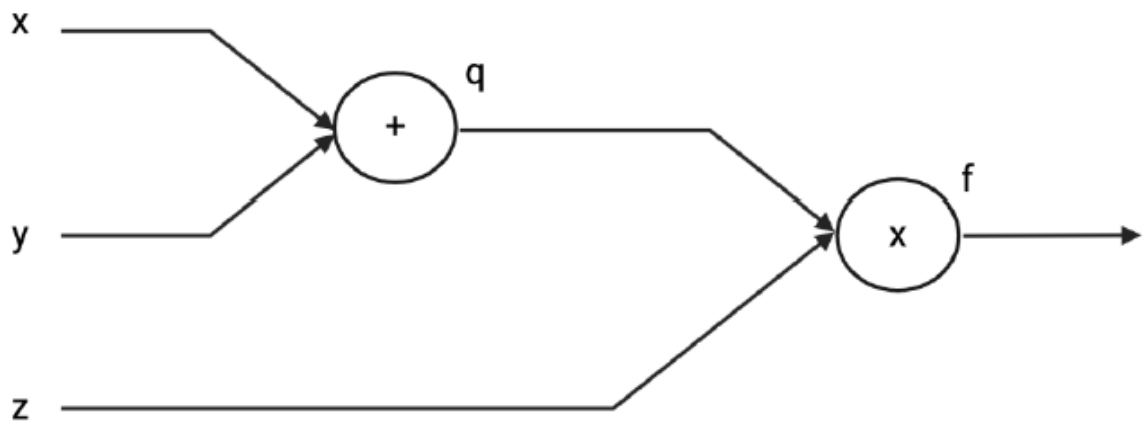
2. [L1] Why is back propagation so critical in the development of neural networks?

3. [L1] Explain stochastic gradient descent.

4. [L2] Let us compute gradients using backpropagation in a tiny neural network model. Consider the function f, which operates on three scalar variables x, y, z :

$$f(x, y, z) = (x + y)z$$

We can represent this visually with a computational graph:



Where $q = x + y$ and $f = qz$. Suppose we want to evaluate this function at a particular point of the input space:

$$x = -4, y = 2, z = -8$$

a. First, perform a **forward pass** through the model to compute the output values q and z, given the input values. Label your intermediate numbers along the graph in the figure.

b. Now in our **backward pass**, our goal is to compute the derivative of the output with respect to each of the inputs. Compute and simplify the following derivatives: $\frac{df}{dx}$, $\frac{df}{dy}$, and $\frac{df}{dz}$. Label intermediate derivatives along the graph in the figure. You may need the chain rule.

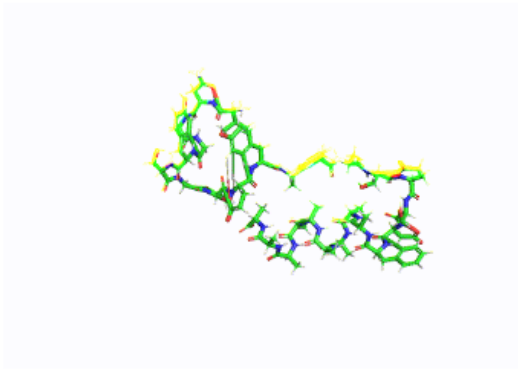
3. [L3] **Amino Acid Pictograph Classifier.** Use PyTorch to create a classifier for this set of pictographs of amino acids. For all 20 amino acids, the set contains 500 28x28-pixel images in the style of MNIST training datasets and feature colors (green carbon, red oxygen, blue nitrogen, yellow sulfur, and white covalent bonds) on a black background.

    a. For this first model, one dense layer of 128 outputs and a second with 20 outputs (corresponding to the 20 amino acid types). Use a sparse categorical cross entropy loss and the Adam optimizer with learning rate of 0.001. Train your network and record the performance on the training and test sets.
    b. Compare performance with a hidden dimension of 64 vs 128. Compare performance with two hidden layers instead of one. Explain any changes.
    c. Which amino acid is most often classified correctly? From the test set, find one correctly predicted image and one incorrectly predicted image. Show them and speculate why this amino acid might be easy to classify and why the incorrect image was a failure case.
    d. Which amino acid is most often classified incorrectly? For amino acid, what is the most common amino acids it is misclassified as? Again, find one correctly predicted image and one incorrectly predicted image. find one correctly predicted image and one incorrectly predicted image.

Workshop – Live Links!

WS10 RFDiffusion and Chroma binder design	Apr 11, 2024
WS9 Denoising diffusion probabilistic model (DDPM)	Apr 4, 2024
WS8_ProteinMPNN	Mar 28, 2024
WS7_intro_to_AF2.ipynb	Mar 7, 2024
WS6_GPT_for_shakespeare_and_proteins.ipynb	Feb 29, 2024
WS5_part2_Convolutions.ipynb	Feb 22, 2024
WS5_part1_IntroToNNs_PyTorch.ipynb	Feb 22, 2024
WS4_IntroToNNs_NumPy.ipynb	Feb 15, 2024
PyRosetta-student-notebooks.zip	Feb 1, 2024
Pymol_Reference_Sheet.pdf	Jan 25, 2024
PyRosetta4_Workshop1_PyMOL_2024.pdf	Jan 25, 2024

PyRosetta.notebooks



Jupyter Notebooks for learning the PyRosetta platform for biomolecular structure prediction and design

View the Project on GitHub
RosettaCommons/PyRosetta.notebooks

PyRosetta Workshops

Welcome to PyRosetta!

PyRosetta is an interactive Python-based interface to the powerful Rosetta molecular modeling suite. It enables users to design their own custom molecular modeling algorithms using Rosetta sampling methods and energy functions.

The Jupyter Notebooks below provide an introduction to the fundmamental principles and tools for using PyRosetta. The Notebooks can be viewed directly in nbviewer. To execute any of the notebooks either locally or in Google Colaboratory, please see Chapter 1 for setup instructions.

Many of the workshops have been adapted from the book *The PyRosetta Interactive Platform for Protein Structure Prediction and Design: PyRosetta4 Update* by Jeffrey Gray, Sidhartha Chaudhury, Sergey Lyskov, and Jason Labonte (Amazon). Other Rosetta developers have also lent their various areas of expertise to help create workshops for this series. Additional contributions to our open-source repository are always welcomed. To learn more about this project, check out our preprint.

Table of Contents

Keyword Index

Chapter 1.0 How to Get Started

- 1.1 How to Get PyRosetta on Your Personal Computer
- 1.2 Jupyter Notebooks, Python, and Google Colaboratory
- 1.3 Frequently Asked Questions/Troubleshooting Tips

Chapter 2.0 Introduction to PyRosetta

- 2.1 Pose Basics

By CER Tech Fellows!

```
[ ] import numpy as np
```

+ Code + Text

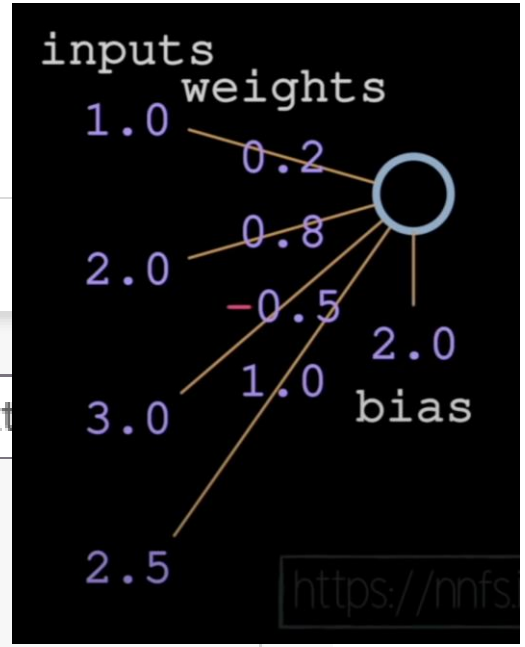
```
# the outputs from the previous layer (which had 3 neurons)
# this will be the input to our current neuron
inputs = np.array([1, 2, 3])

# every unique input will have a unique weight associated with it
# since we have three inputs, we have 3 weights
weights = np.array([0.2, 0.8, -0.5])

# every unique neuron has a unique bias
bias = 2

# output from our neuron is the input*weight + bias
output = inputs[0]*weights[0] + inputs[1]*weights[1] + inputs[2]*weights[2] + bias
print(output)
```

```
2.3
```



RfDisffusion will only give you the backone, so you can use next steps to design the sequence for your backbones. Ask your instructor would need to run them.

run **ProteinMPNN** to generate a sequence and **AlphaFold** to validate

num_seqs: 8

initial_guess: ☐

num_recycles: 1

use_multimer: ☐

rm_aa: C

mpnn_sampling_temp: 0.1

- for **binder** design, we recommend `initial_guess=True num_recycles=3`

Show code

Teaching Challenges

- Diversity of student backgrounds
  - Engineers, biologists, bioinformatics MS, (CS/applied math)
  - *Strategy: Specifications grading*
- Fundamental depth vs application
  - *Strategy: Lecture + Workshop format (Tues/Thurs)*
- Workshops: pre-built to run vs. code-on-your-own
  - *Strategy: make it up as I go*