CSE 416: Introduction to Machine Learning
Assignment #9
Instructor: Pemi Nguyen
due: Sunday, June 5, 2022, 11:59 pm.

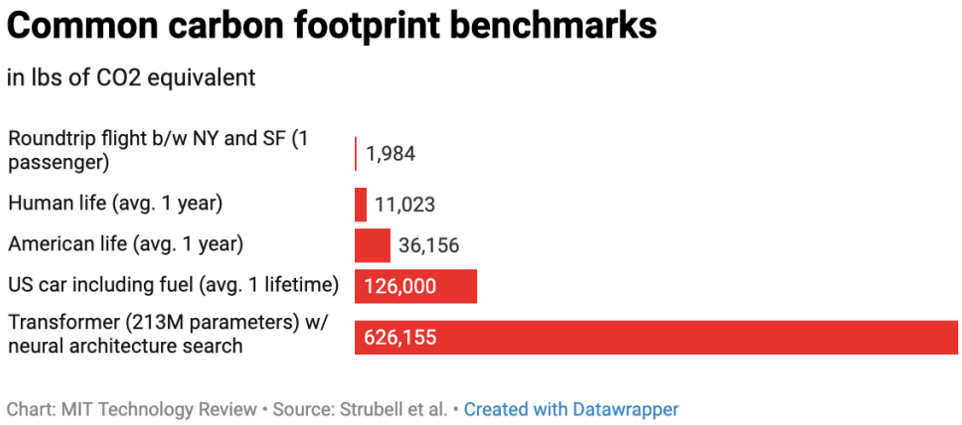
Instructions:

- **Answers:** Please provide detailed yet concise explanations for your work.
- **Turn-in:** Do not write your name on your pages (your Gradescope account will identify you to us) and do not include a copy of the exercise's question in what you turn in. You must use Gradescope to upload your written homework solutions. You will submit a single PDF file containing your solutions to all the exercises in the homework. You must follow the Gradescope prompts that have you link exercise numbers to your pages. **We do not accept handwritten solutions**, so please typeset them, such as Microsoft Word or LaTeX editor.

1. **(4 points)** In this week's lectures and guest lectures by Joseph Redmon and Ali Abdalla, you have seen the magical power of large neural networks in solving a lot of challenging problems, such as image classification, generating images from text inputs, summarizing texts, neural style transfer, named entity recognition, etc.

   As amazing as they are, there is a rising concern in the AI community about their environmental impact. Specifically, their enormous computational requirements means more energy consumption and more emitted greenhouse gases in the atmosphere consequently.

   Here's a visualization of how much energy a Transformer model, one of the state-of-the-art NLP models nowadays, consumes compared to other activities:



   Please skim this Forbes article on Deep Learning's Carbon Emissions Problem and write **about 6-8 sentences** summarizing what you learn from reading it.

   Optional: If you would like to explore more effort to create greener models, please checkout this MIT article.

2. (a) **(1 point)** In 1 sentence, explain why do we need non-linear activation functions like ReLU, tanh or sigmoid for neural networks?

   (b) **(2 points)** One common issue facing neural networks today is **vanishing gradients**. This occurs when some gradients are too small, the updates for certain weights are thus too insignificant that neural networks can no longer learn effectively. Of the following three activation functions, which one do you think can help prevent the vanishing gradients issue the most effectively? Why?

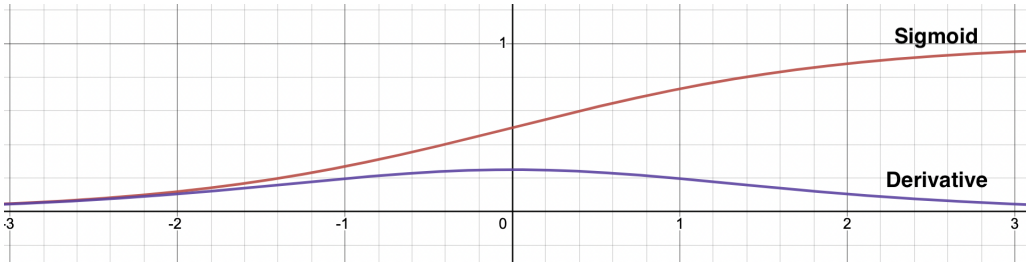


Figure 1: Sigmoid and its derivative

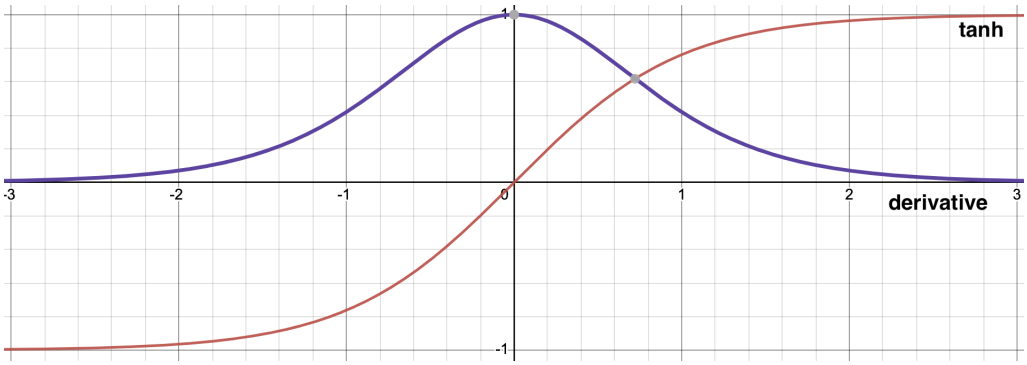


Figure 2: Tanh and its derivative

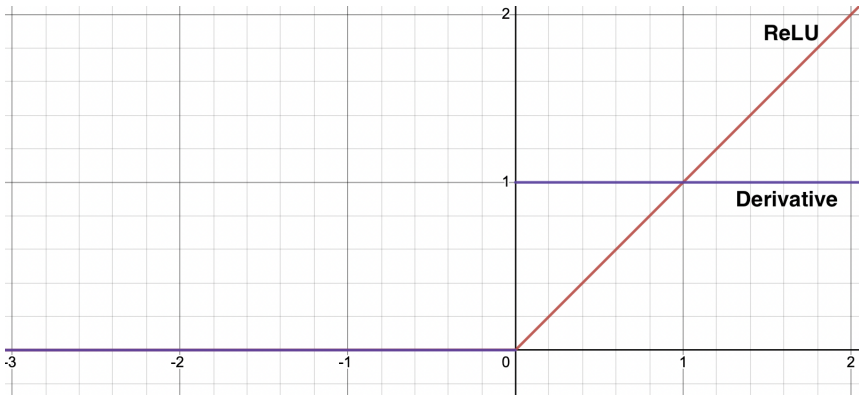


Figure 3: ReLU and its derivative

3. In lecture, we have introduced the concept of transfer learning. In this Colab link, we're going to take a pretrained ResNet18 model and apply transfer learning on it to classify the CIFAR-10 images.

   ResNet was trained on the ImageNet dataset, designed by academics intended for computer vision research. There are more than 14 million images in the dataset, 21 thousand groups or classes, and a bit more than 1 million images that have bounding box annotations

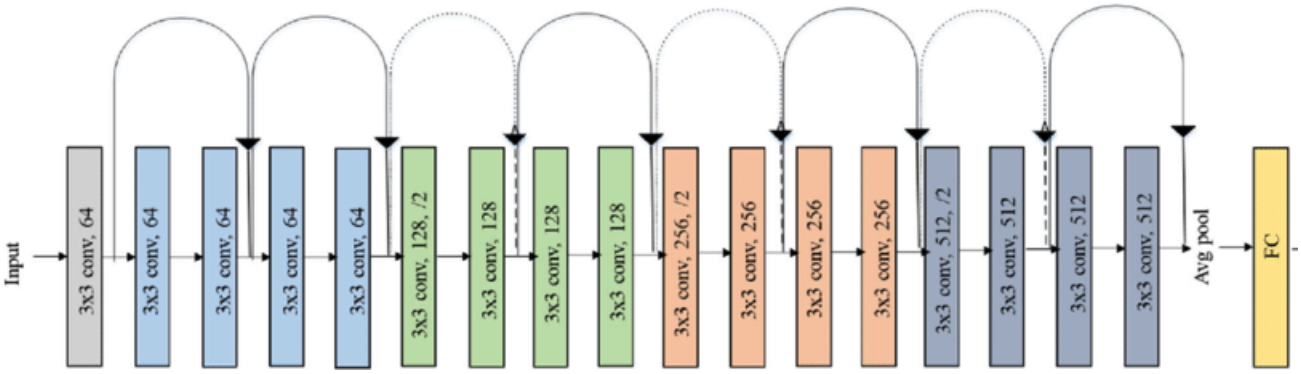


Figure 4: ResNet18 architecture

The CIFAR-10 dataset (Canadian Institute For Advanced Research) is a collection of images is one of the most widely used datasets for machine learning research. It contains 60,000 32x32 color images in 10 different classes: airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks. There are 6,000 images of each class.

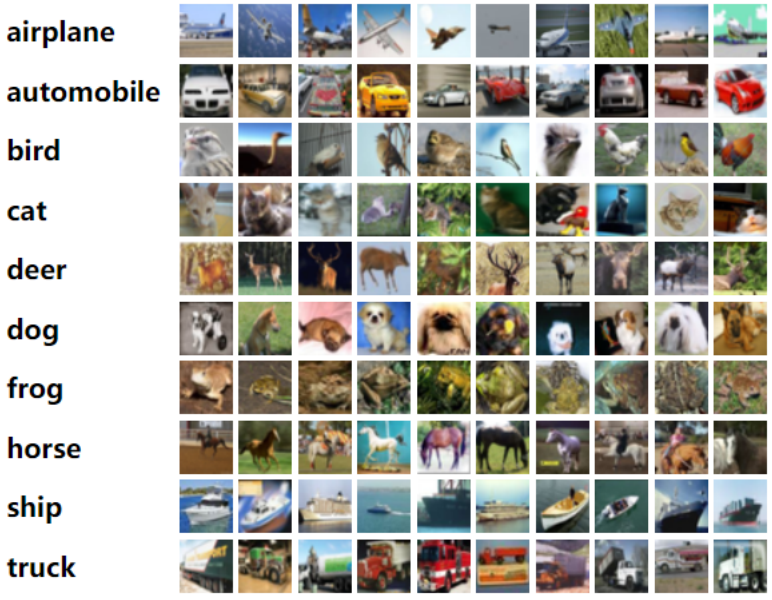


Figure 5: Example images from 10 classes in the CIFAR-10 dataset

There are two approaches in fine-tuning pretrained CNN models:

- **Finetuning the ConvNet**: Instead of random weight initialization, we initialize the network with a pretrained network with saved weights. Rest of the training looks as usual.
- **Using ConvNet as fixed feature extractor**: Here, we will freeze the weights for all of the network except that of the final fully connected layer. This last fully connected layer is replaced with a new one with random weights and only this layer is trained.

Run the Colab and answer the following questions:

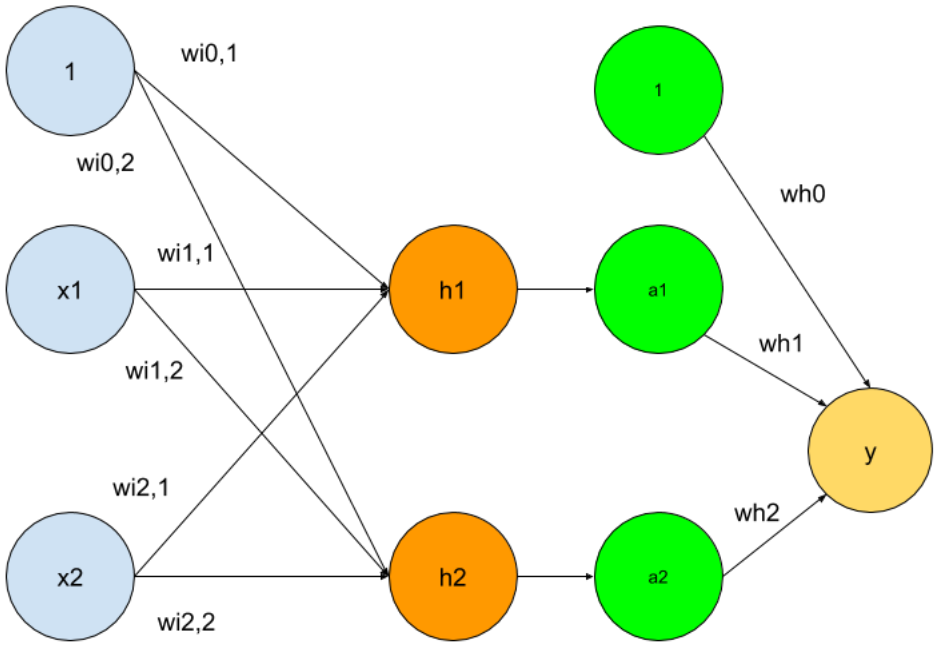
(a) **(2 points)** Which approach takes less time to train? Why?

(b) **(2 points)** Which approach has better validation accuracy? Propose a hypothesis on why this is the case.

4. **(Extra credit)**

In this problem, you're going to understand why random weight initialization is important in neural networks (and proper initialization is also a research question in the ML community).

Here is the architecture of an one-hidden-layer neural network. a_1 and a_2 refer to a single non-linear activation function. The neural network accepts 2-dimensional inputs $x = (x_1, x_2)$, outputs value y, and uses a differentiable loss function $\mathcal{L}$. Nodes labelled 1 refer to biases.



(a) **(2 points)** Let all initial weights in the neural networks be 0, specifically:

$$w_{i_{0,1}} = w_{i_{0,2}} = w_{i_{1,1}} = w_{i_{1,2}} = w_{i_{2,1}} = w_{i_{2,2}} = w_{h_0} = w_{h_1} = w_{h_2} = 0$$

.

Using ReLU for a_1 and a_2, prove mathematically that no learning can occur with this initialization.

Hint: You should be able to prove that:

$$\nabla w_{i_{0,1}} = \nabla w_{i_{0,2}} = \nabla w_{i_{1,1}} = \nabla w_{i_{1,2}} = \nabla w_{i_{2,1}} = \nabla w_{i_{2,2}} = \nabla w_{h_0} = \nabla w_{h_1} = \nabla w_{h_2} = 0$$

Without loss of generalization, you can just prove $\nabla w_{i_{1,1}} = \frac{\partial \mathcal{L}}{\partial w_{i_{1,1}}} = 0$.

(b) **(4 points)** In a more general form, let all initial weights in the neural networks be:

$$w_{i_{0,1}} = w_{i_{0,2}} = \alpha_B$$
$$w_{i_{1,1}} = w_{i_{1,2}} = \alpha_1$$
$$w_{i_{2,1}} = w_{i_{2,2}} = \alpha_2$$
$$w_{h_1} = w_{h_2} = \beta$$
$$w_{h_0} = \beta_B$$

Using any non-linear activation functions for a_1 and a_2, prove mathematically that no learning can occur with this initialization.

Hint: This is more difficult than the previous one. You should be able to prove that:

$$\nabla w_{h_1} = \nabla w_{h_2}$$

After that, you can prove after the first backpropagation from the initialization, the updated weights will be stuck in the same initial form, resulting in no learning:

$$w'_{i_{0,1}} = w'_{i_{0,2}} = \alpha'_B$$
$$w'_{i_{1,1}} = w'_{i_{1,2}} = \alpha'_1$$
$$w'_{i_{2,1}} = w'_{i_{2,2}} = \alpha'_2$$
$$w'_{h_1} = w'_{h_2} = \beta'$$
$$w'_{h_0} = \beta'_B$$