Meeting

Po Hsun Wu

April 11, 2022

Progress report

- Target function:

$$f(x) = x^2$$

- Database:
  - $x = \{0, 1, ..., 9\}$ adding noise with normal distribution($\mu = 0, \sigma = 0.2$).
  - 100 random data for each point(total of 1,000 data).
  - Train for 1000 times.

Result

- Using two hidden layer, each layer with 50 neuros.

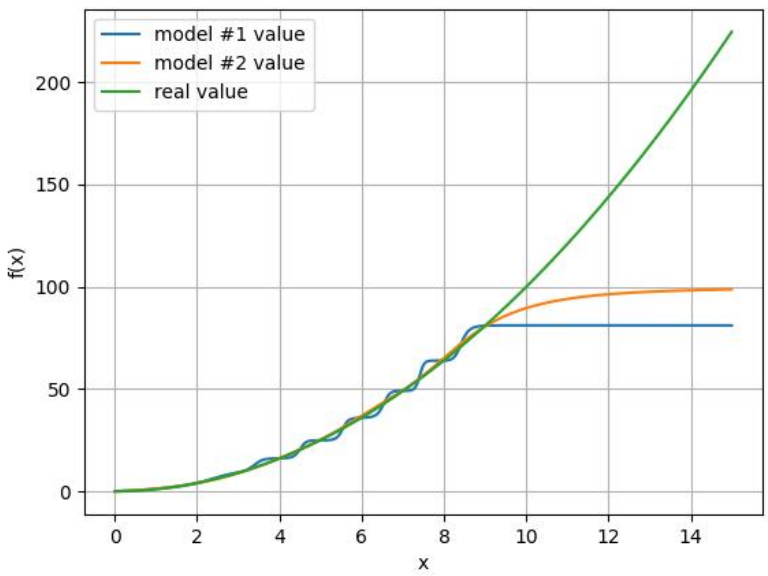

Figure 1: f(x) vs x

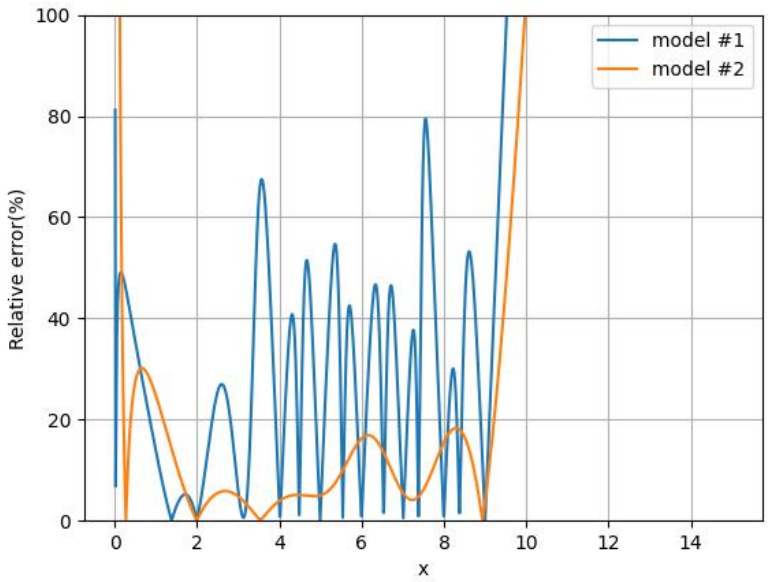

Figure 2: Relative error

Result

- Using three hidden layer, each layer with 50 neuros.

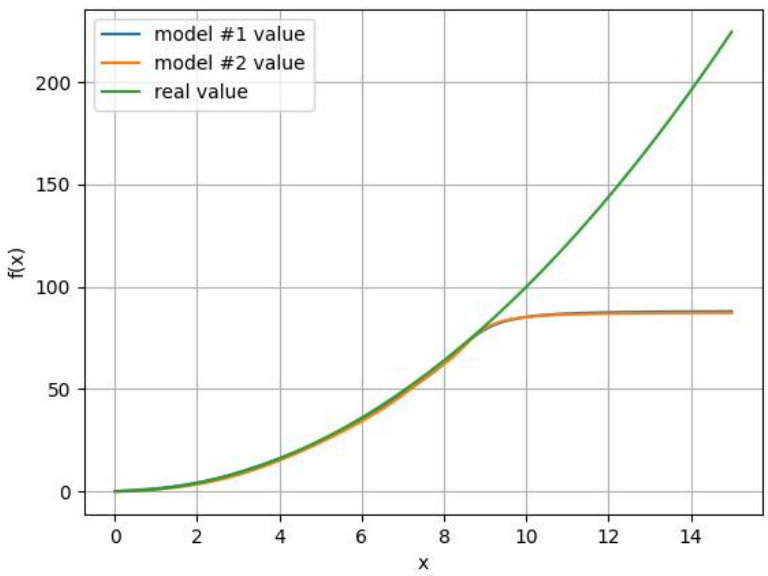

Figure 3: f(x) vs x

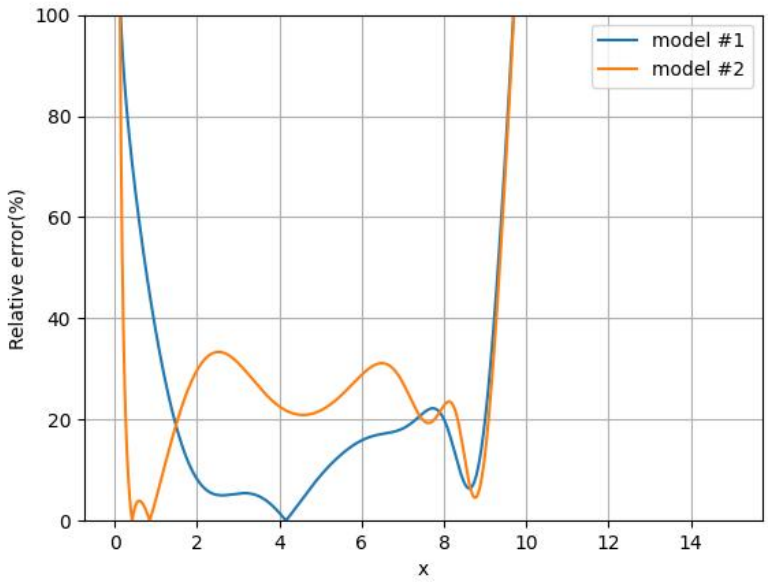

Figure 4: Relative error

Result

- Compare between 2 hidden layers and 3 hidden layers.

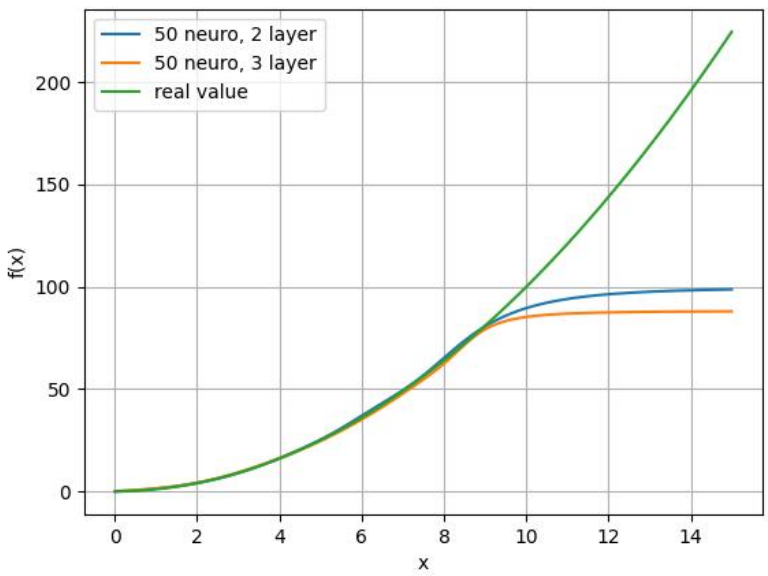

Figure 5: f(x) vs x

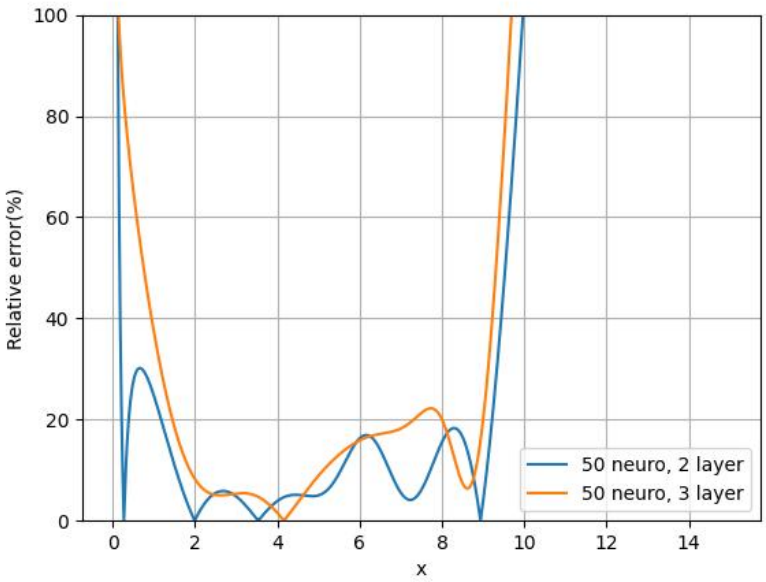

Figure 6: Relative error

Result

- Compare between 30 neuros and 50 neuros.

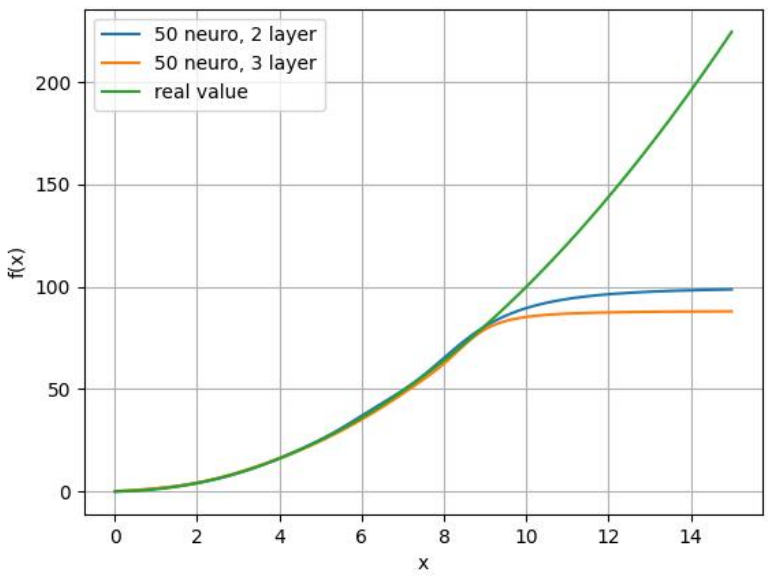

Figure 7: f(x) vs x

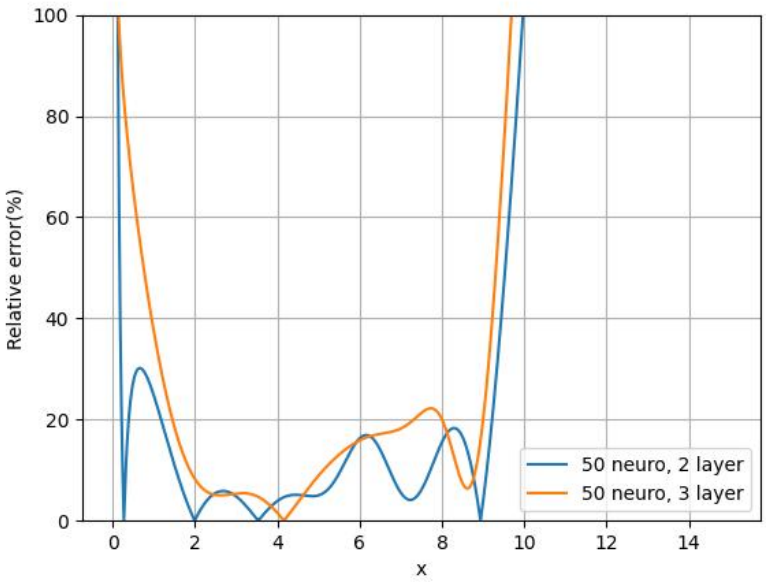

Figure 8: Relative error

Result

- Compare different learning rate.

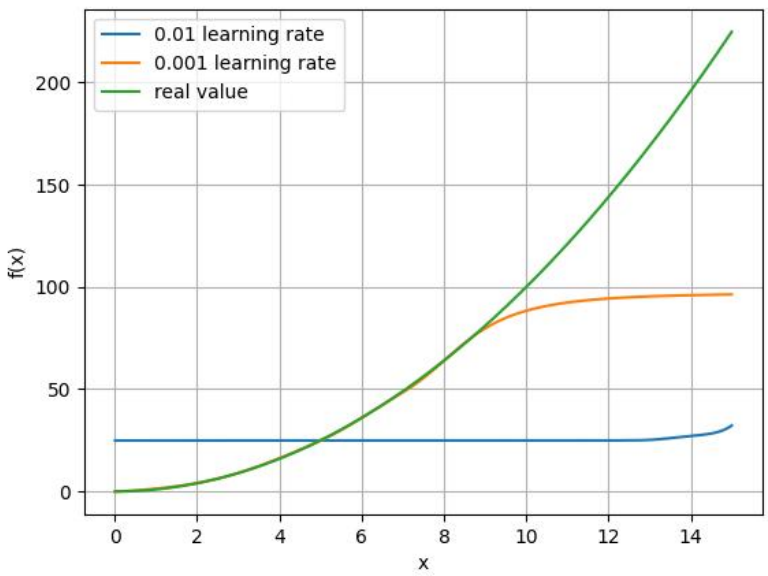

Figure 9: f(x) vs x

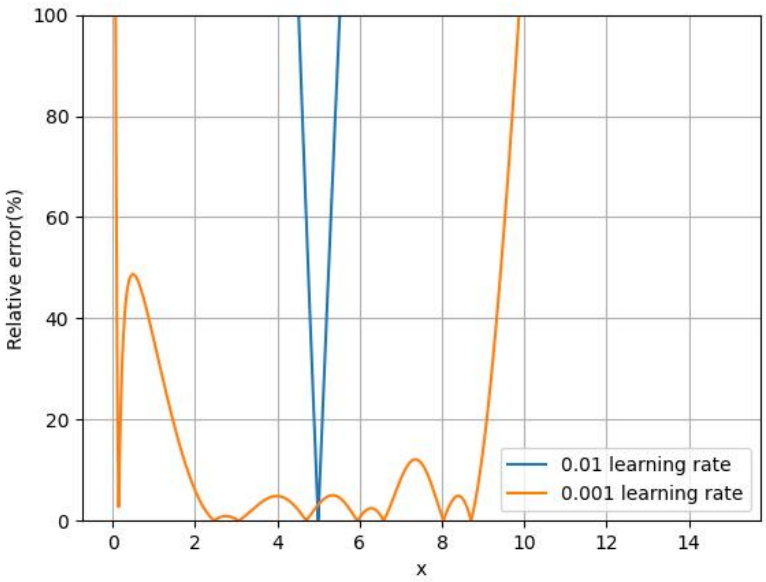

Figure 10: Relative error

```
import numpy as np
from tensorflow import keras

target_fun = lambda x: x * x

x = np.arange(0, 10, 1)
x = x.reshape((1, x.size))
x_ = np.repeat(x, 100, axis=0)
y = target_fun(x_).reshape((x_.size,))

init = initializer = keras.initializers.glorot_uniform()
model = keras.models.Sequential(
    [
        keras.layers.InputLayer(input_shape=(1,)),
        keras.layers.Dense(units=50, activation="sigmoid", kernel_i
        keras.layers.Dense(units=50, activation="sigmoid", kernel_i
        keras.layers.Dense(units=1, activation="linear", kernel_ini
    ]
)

noise = np.random.normal(0, 0.2, size=x_.shape)
x_noise = x_ + noise
x_noise = x_noise.reshape((x_noise.size,))

callback = [
    keras.callbacks.EarlyStopping(
        monitor="loss",
        patience=100,
        min_delta=0.001
    )
]
opt = keras.optimizers.SGD(
    learning_rate=0.001,
    momentum=0.005
)

model.compile(optimizer=opt, loss="MSE")
history = model.fit(
    x_noise, y,
    batch_size=10,
    epochs=1000,
    callbacks=callback,
    use_multiprocessing=True
)
```