**Міністерство освіти і науки**

**України Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет інформатики та обчислювальної техніки Кафедра обчислювальної техніки**

**ЗВІТ**

лабораторної роботи №1

з курсу «Програмні засоби проєктування і реалізації неромережевих систем»

Тема: «Перцептрон»

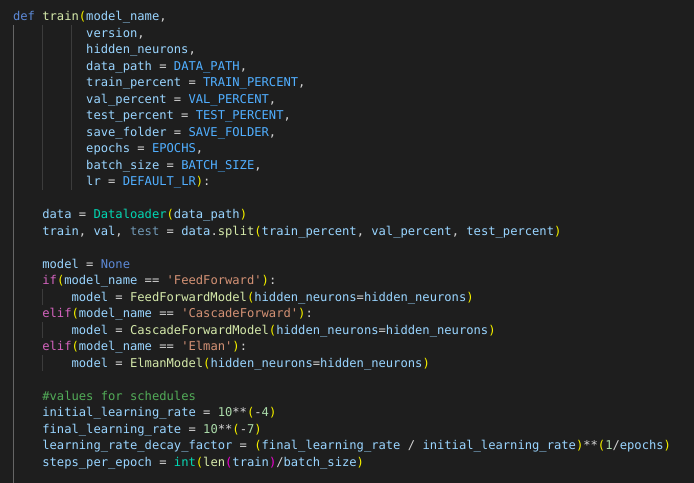
| Перевірив:  Шимкович В. М. | Виконав:  Студент Гр. ІП-01 Шпилька В.С. |
| --- | --- |

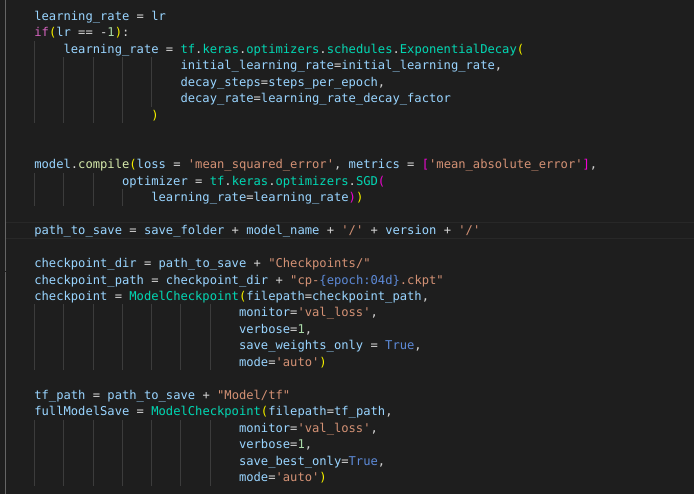
Київ 2023

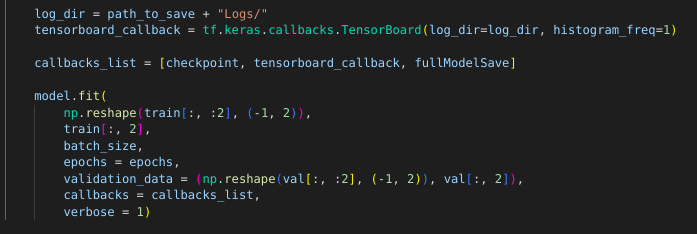
**Завдання:** Написати програму, що реалізує нейронні мережі для моделювання функції двох змінних. Функцію двох змінних, типу f(x+y) = x2+y2, обрати самостійно. Промоделювати на невеликому відрізку, скажімо від 0 до 10.

Дослідити вплив кількості внутрішніх шарів та кількості нейронів на середню відносну помилку моделювання для різних типів мереж (feed forward backprop, cascade - forward backprop, elman backprop):

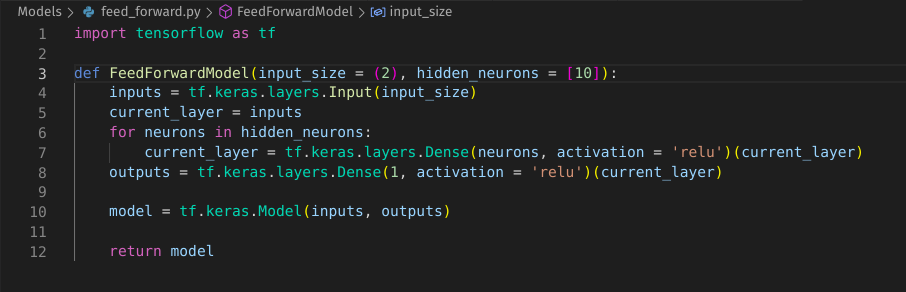
Функція тренування:





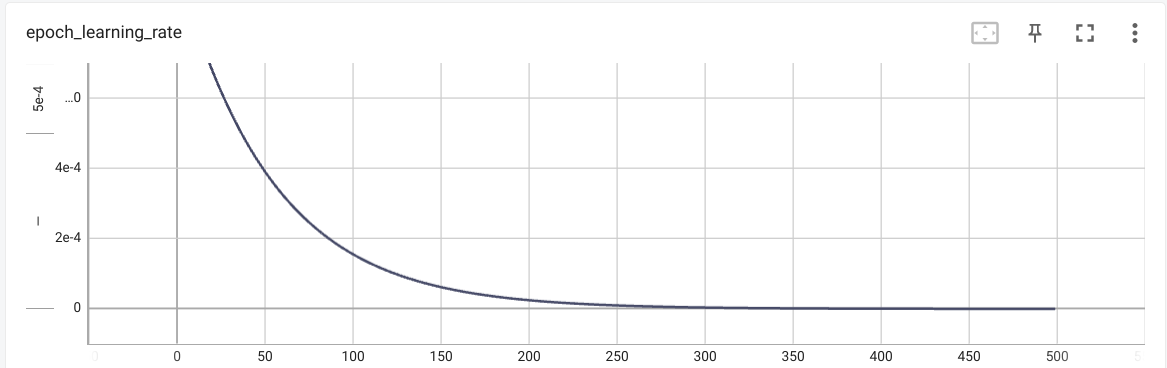


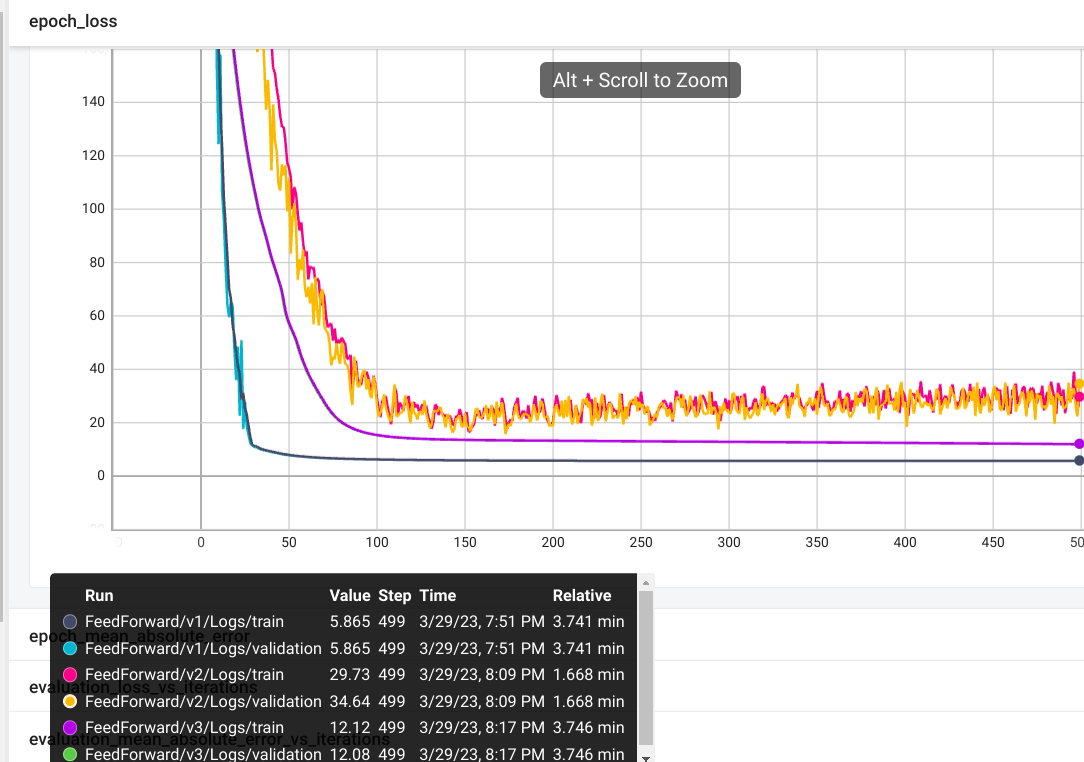
В функція передається назва моделі, версію під якої зберегти та кількість нейронів в кожному шарі. Функція створює модель, компілює її, створює функції для зберігання чекпоінтів, найкращої моделі та логів. Та саме тренування за допомогою model.fit

1. Тип мережі: feed forward backprop:

a) 1 внутрішній шар з 10 нейронами;

Одним з питанням яке було вирішено на даном кроці стало, що краще використати крок навчання константне число, чи ExponentialDecay, який зменшує learning rate під час навчання:

Функція зміни learning rate:

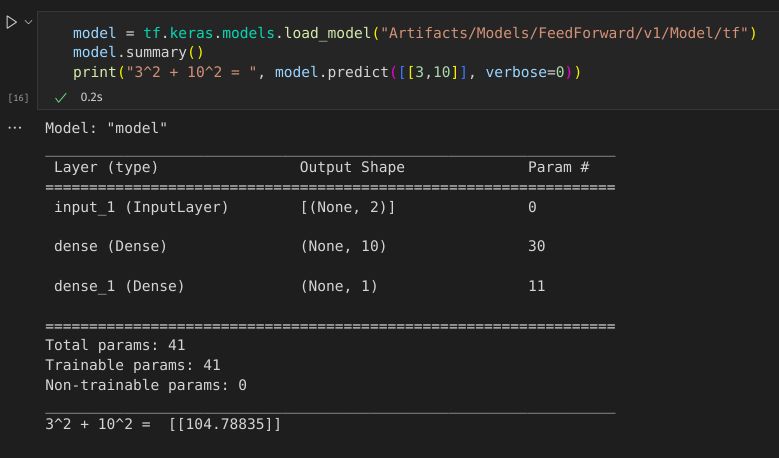
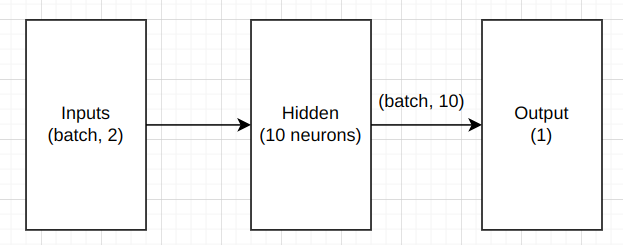
На наступному рисунку, показано як зменшується loss функції при різних learning rate:

v1 – використання ExponentialDecay в межах 10е-3 до 10е-7

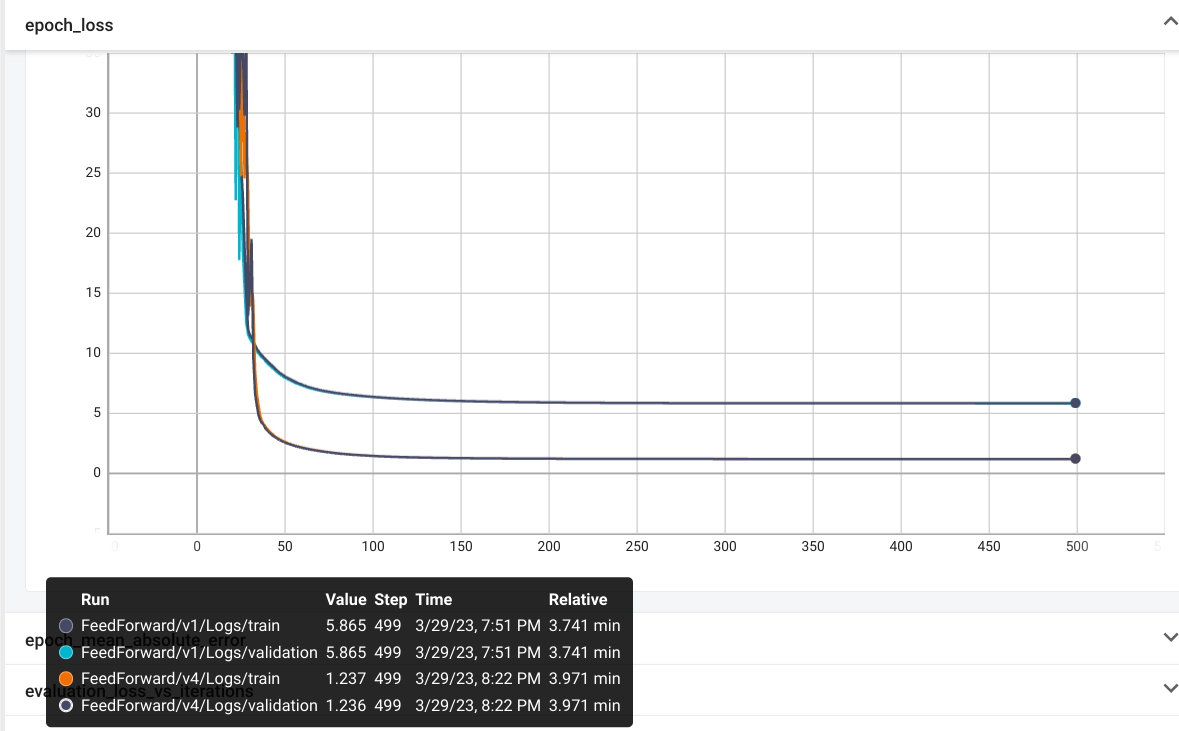
v2 – константне значення 10е-3

v2 – константне значення 10е-4

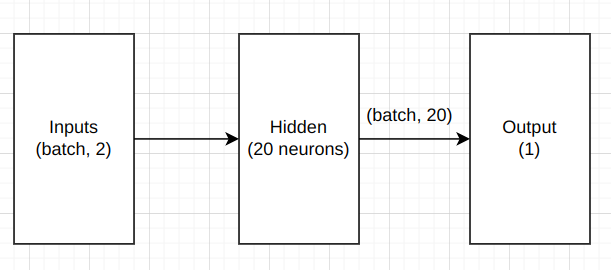
Як бачимо за допомогою ExponentialDecay ми швидше і ліпше отримали loss значення функції. Тому в майбутніх тренуваннях будемо використовувати його.

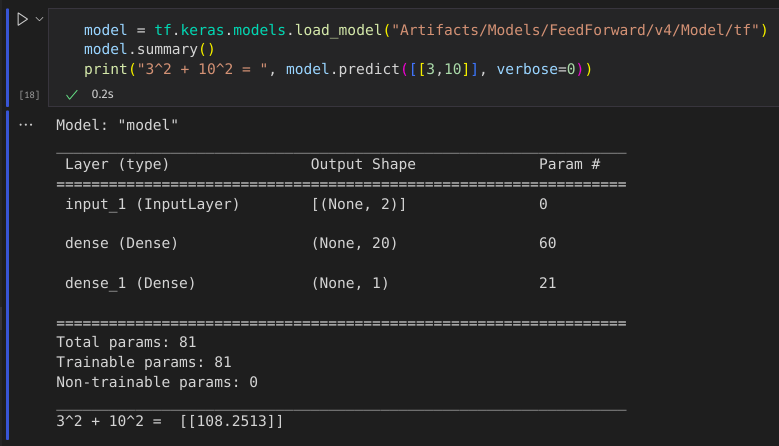
На наступному рисунку зображено архітектуру і приклад використання нейронної мережі:

б) 1 внутрішній шар з 20 нейронами;

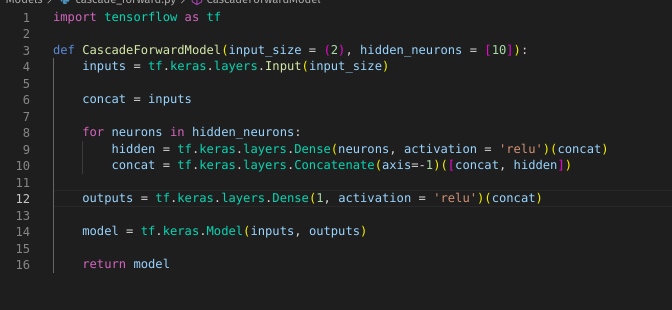
Різниця з поперднім випадком наведена на наступному рисунку:

Як бачимо додавання ще 10 нейронів значно покращило ситуацію.

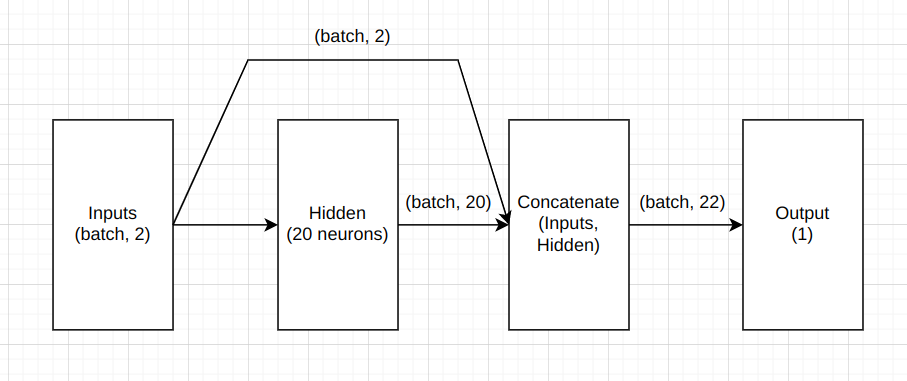
Архітектура і приклад використання наведено нижче:

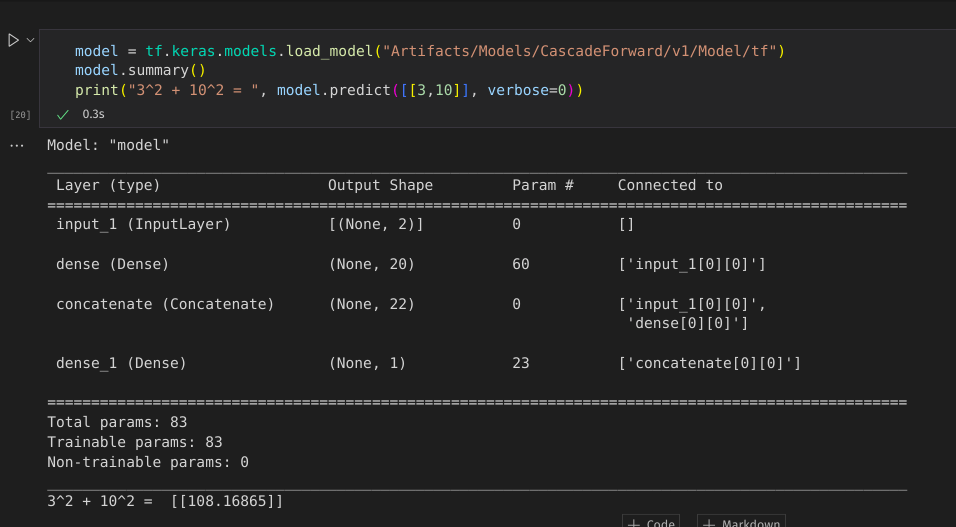


2. Тип мережі CascadeForward:

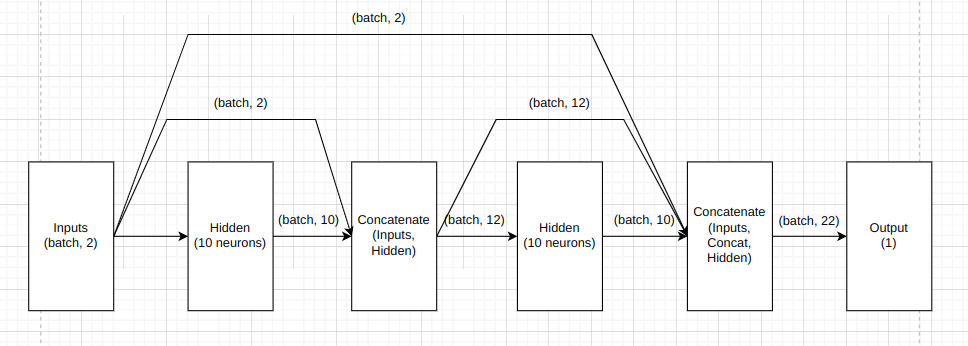


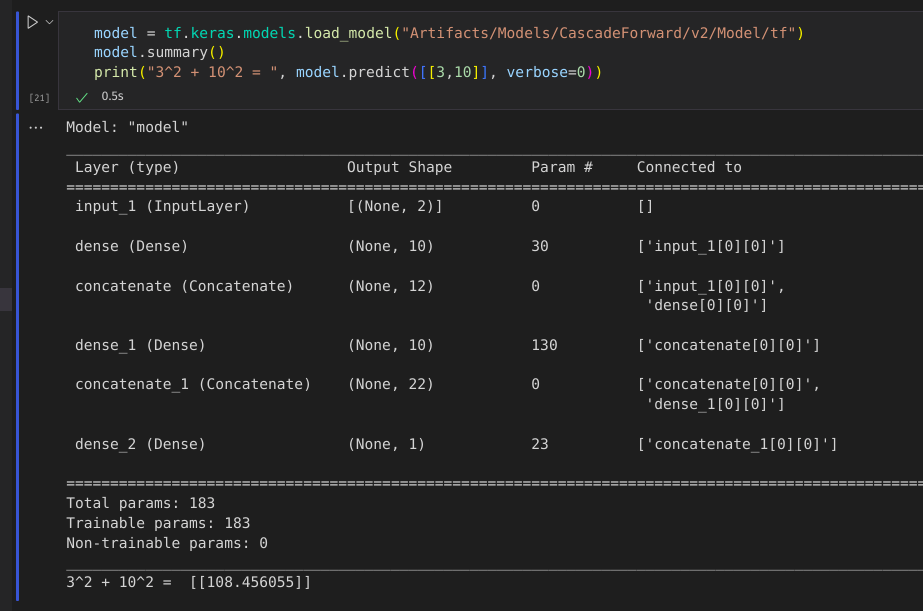
a) 1 внутрішній шар з 20 нейронами;

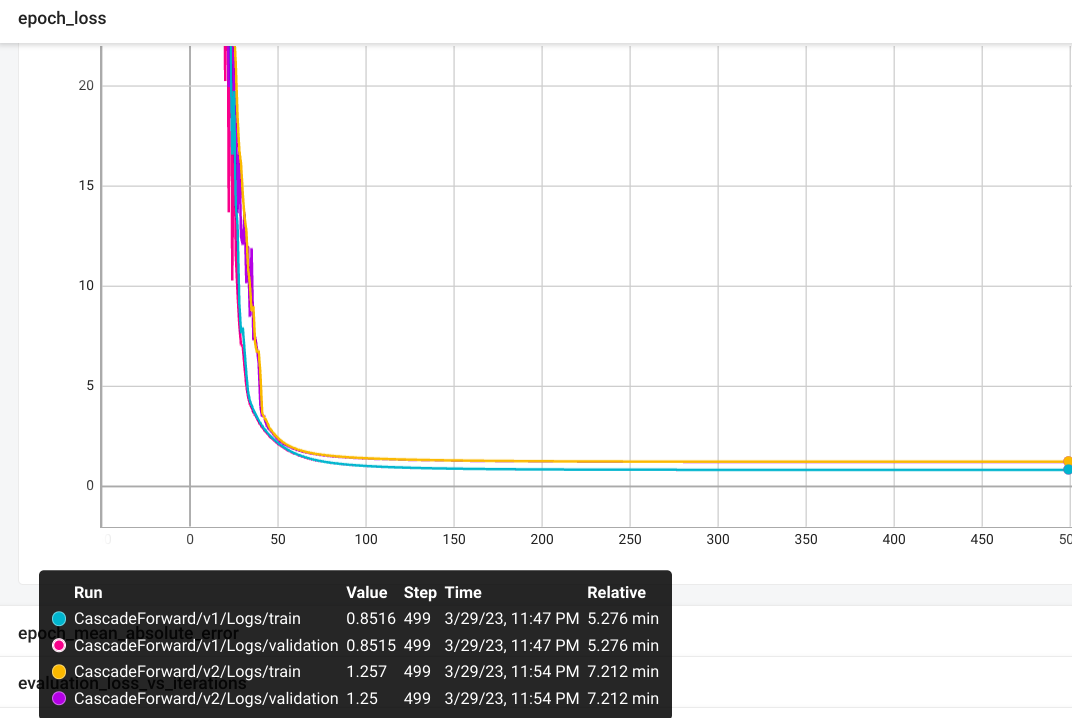
Архітектура і приклад використання:



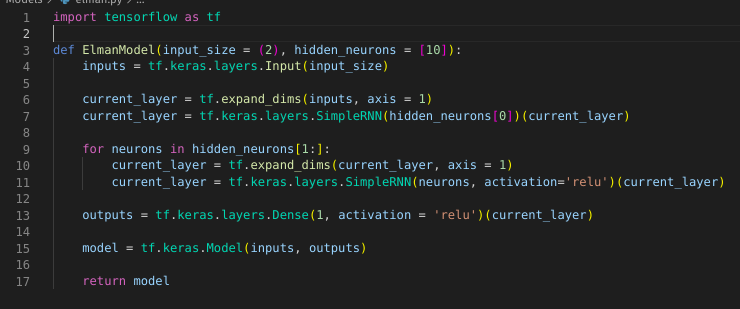
б) 2 внутрішніх шари по 10 нейронів у кожному;

Архітектура і приклад використання:

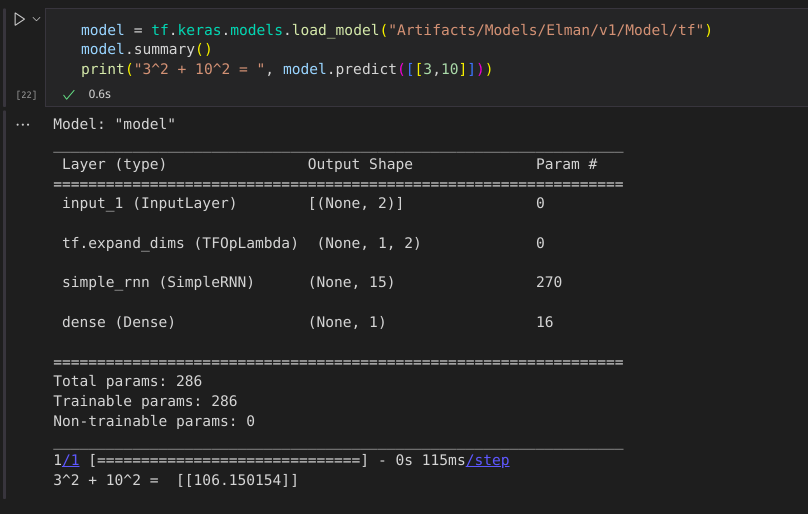
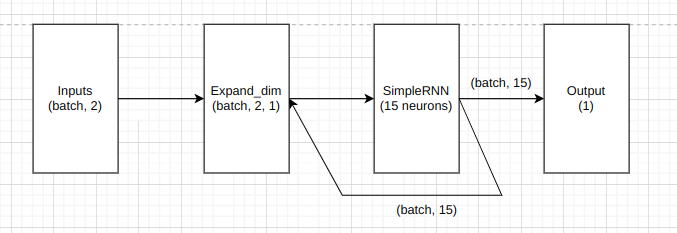


Порівняння:

Нейрона мережа з одним прихованим шаром, але більшою кількістю нейронів показала трохи краще результати, ніж 2 шари, але менше нейронів.

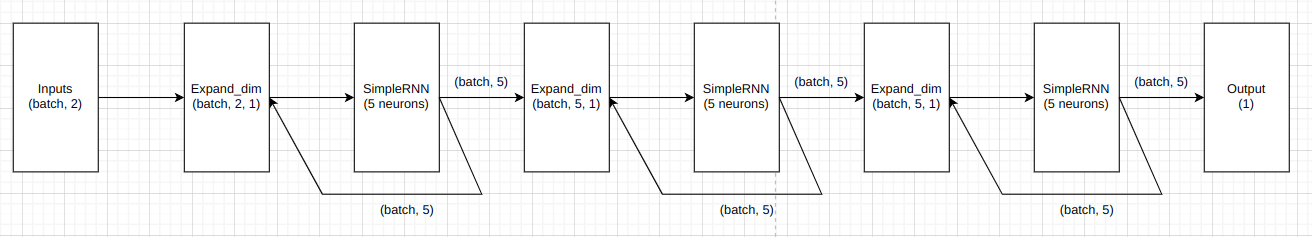
3. Тип мережі Elman:

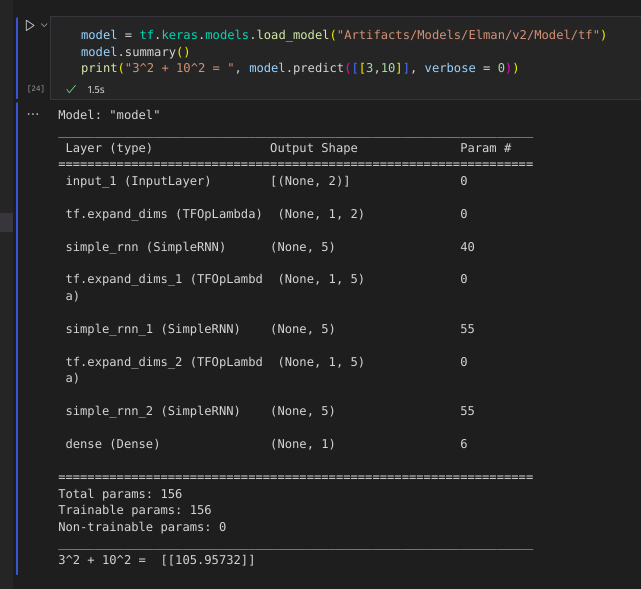
a) 1 внутрішній шар з 15 нейронами;

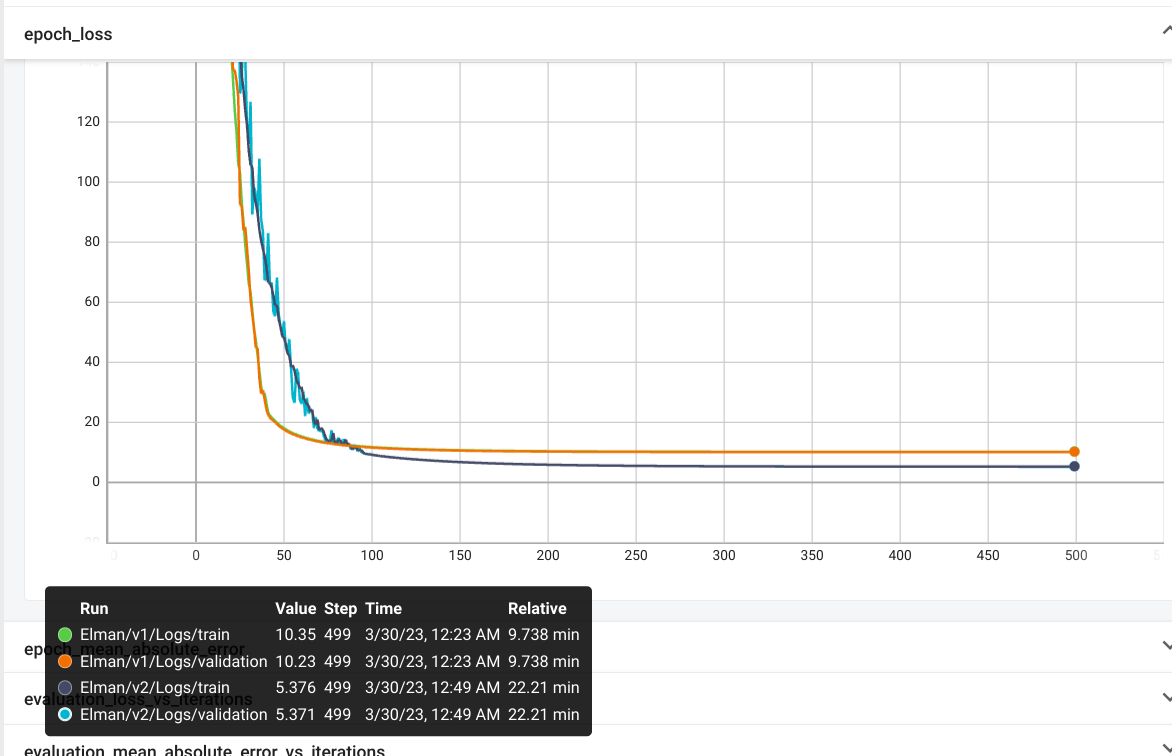
Архітектура і приклад використання:

b) 3 внутрішніх шари по 5 нейронів у кожному;

Архітектура і приклад використання:

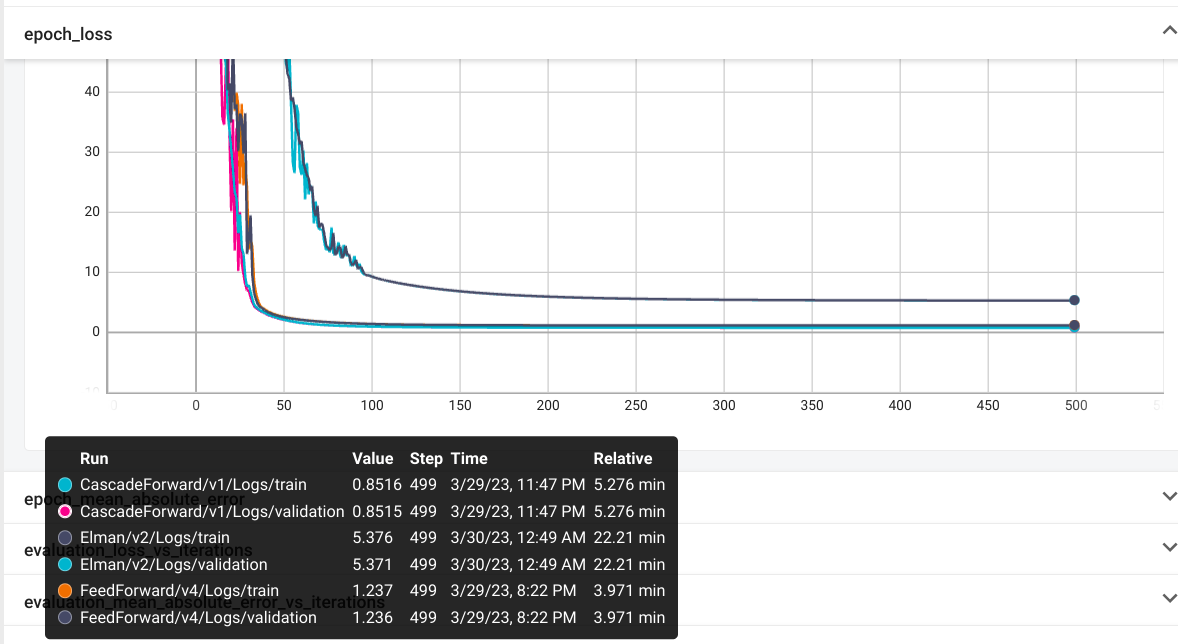




Порівняння:

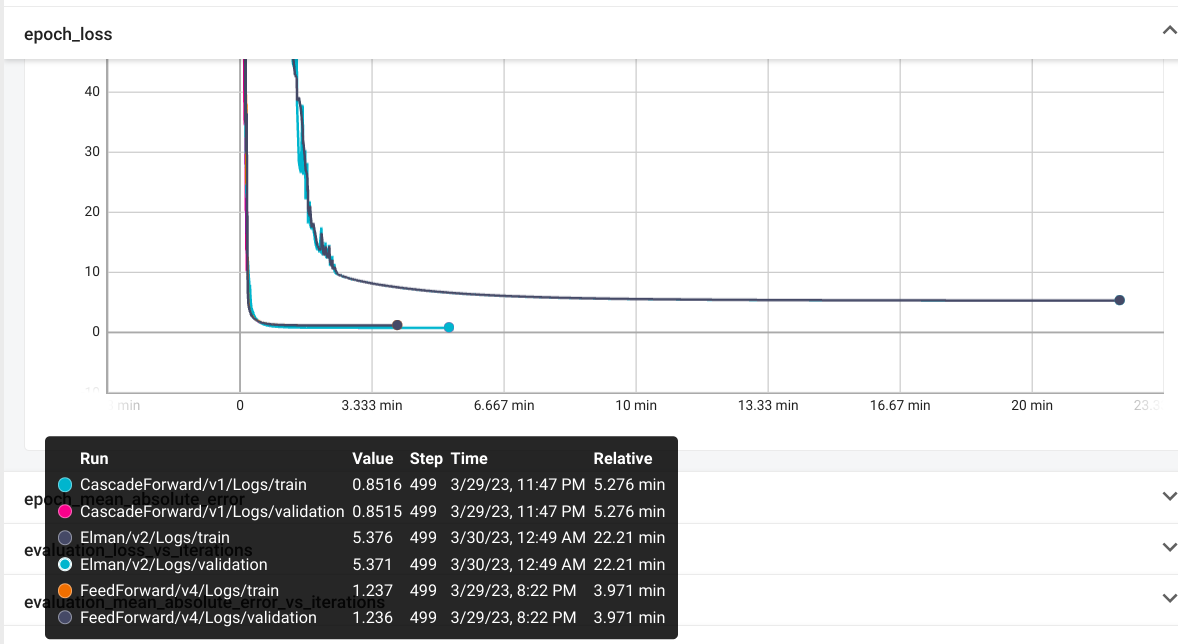
Нейрона мережа з 3 прихованими шарами працює в загальному краще ніж з 1 шаром, але більшою кількістю нейронів.

Загальне порівняння:



З графіку видно, що найкраще працює саме CascadeForward з одним прихованим шаром з 20 нейронами. Нейронна мережа Елмана показала гірші результати, адже рекурентні нейронні мережі, скоріше потрібні для знаходження закономірностей в часових рядах, а не звичайних функціях.

Що стосується часу тренування:



FeedForward тренується найшвидше, CascadeForward трохи повільніше, Elman дуже повільно.

**Висновки:**

В даній лабораторній роботі було розглянуто 3 види нейронних мереж. Було розглянуто користь від використання ExponentionalDecay для зміни кроку навчання та показано перевагу даного підходу перед константним значенням. Було розглянуто зміну результату від зміни кількості нейронів у шару. Можна однозначно сказати, чим більше нейронів в одному шарі тим краще, але якщо збільшувати кількість шарів жертвуючи кількістю нейронів, то результат може бути неочікуваним і залежить від конкретних даних та ситуації. Також було зроблено висновок, що мережа Елмана не підходить для задачі регресії, і скоріше за все її варто використовувати для прогнозування в часових рядах.