Tugas Besar B IF3270 Pembelajaran Mesin
Implementasi Mini-Batch Gradient Descent

Disusun oleh:

Zaidan Naufal Sudrajat	13518021
Delisha Azza Naadira	13519133
Siti Iedrania Azzariyat Akbar	13519137
Karina Imani	13519166

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika - Institut Teknologi Bandung
Jl. Ganesha 10, Bandung 40132

2022

A. Penjelasan Implementasi

Secara garis besar, program dibagi menjadi tiga, yakni kelas Model, kelas Layer, dan kelas Perceptron. Sebuah model terdiri dari beberapa layer, dan sebuah layer terdiri dari beberapa perceptron. Untuk membangun model, digunakan file .txt yang memiliki format sebagai berikut:

Format	Contoh
[ukuran batch] [jumlah layer] [jumlah input] [jumlah perceptron layer 1] [fungsi aktivasi layer 1] [bobot perceptron 1] ... [jumlah perceptron layer 2] [fungsi aktivasi layer 2] [bobot perceptron1] ...	5 3 2 2 sigmoid -0.5 0.5 0.5 0.5 -0.5 -0.5 1 sigmoid -0.5 0.5 0.5 ...

Angka pada baris pertama adalah ukuran dari satu *batch*, angka pada baris kedua adalah jumlah layer dari model, dan dan dilanjutkan dengan jumlah input (dimensi x). Selanjutnya, akan dijabarkan keterangan setiap layer, yakni jumlah perceptron, fungsi aktivasi, dan bobot dari tiap perceptron. Adapun, string yang valid untuk fungsi aktivasi adalah: "`sigmoid`", "`relu`", "`linear`", "`softmax`".

1) Model

a) Atribut

i) `total_layer : Integer`
Jumlah layer yang terdapat pada model.

ii) `total_input : Integer`
Jumlah input yang diminta oleh model.

iii) `arr_act_func : List[String]`
Daftar tipe fungsi aktivasi untuk setiap layer.

iv) `arr_act_perc : List[List[Perceptron]]`
Daftar perceptron yang dimiliki oleh setiap layer.

v) `model : List[Layer]`
Daftar layer yang dibuat berdasarkan keseluruhan atribut lainnya.

b) Fungsi

i) `__str__`
Memperlihatkan model dalam bentuk string format tertentu, yang menampilkan fungsi aktivasi dan perceptron (dan bobotnya) untuk setiap layer.

ii) `readModel(String)`
Melakukan parsing dari file yang dimasukkan ke parameter.

iii) `buildModel : List[Layer]`

Membangun model dari file yang sudah di-parsing.

iv) `doffnn(List[List[Float]]) : List[List[Float]]`
Memprediksi nilai dari input yang menjadi parameter fungsi berdasarkan model yang telah dibuat menggunakan FFNN.

v) `dobackwardpropagation(Float, List[Float])`
Melakukan backpropagation untuk memperbaiki weight dari model yang digunakan.

vi) `fit(List[List[Float]], List[Float], Float, Float, Integer)`
Melakukan iterasi dari backpropagation sampai error di bawah *error threshold* atau jumlah iterasi melebihi *max iteration*.

vii) `sumsquarederror(List[Float], List[Float]) : Float`
Menghitung loss dari hasil yang diperoleh FFNN untuk tipe output layer linear, sigmoid, dan ReLU.

viii) `crossentropy(List[List[Float]]) : Float`
Menghitung loss dari hasil yang diperoleh FFNN untuk tipe output layer softmax.

ix) `printw()`
Mencetak nilai-nilai weight dari setiap perceptron yang ada di setiap layer yang terkandung dalam model.

2) Layer

a) Atribut

i) `type : String`
Tipe layer, berupa string “output” atau “hidden”.

ii) `activation_function_type : String`
Fungsi aktivasi yang digunakan untuk perceptron pada layer.

iii) `array_of_perceptron : List[Perceptron]`
List perceptron yang terdapat pada layer.

b) Fungsi

i) `calculate_all(List[List[Float]]) : List[List[Float]]`
Menghitung array input ke perceptron pada `array_of_perceptron`, kemudian masing-masing hasilnya dimasukkan ke suatu list output. Fungsi me-*return* list output tersebut.

ii) `calculate_back_hidden(Float, List[List[Float]], List[List[Float]]) : List[List[Float]], List[List[Float]]`
Menghitung update weight dari setiap perceptron pada layer hidden. Fungsi me-*return* list error dan weight dari layer yang dimaksud untuk membantu perhitungan layer di belakangnya.

iii) `calculate_back_output(Float, List[Float]) : List[List[Float]], List[List[Float]]`
Menghitung update weight dari setiap perceptron pada layer output. Fungsi me-*return* list error dan weight dari layer yang dimaksud untuk membantu perhitungan layer di belakangnya.

3) Perceptron

a) Atribut

i) `weights : List[Integer]`
Daftar bobot dari perceptron.

ii) `outputs : List[Float]`
Daftar output yang sudah diaktivasi dari perceptron.

iii) `inputs : List[Float]`
Daftar input yang diterima oleh perceptron.

iv) `errors : List[Float]`
Daftar error perceptron, berupa list yang berisi float target - output untuk perceptron pada output layer dan list berisi sum perkalian error dengan weight perceptron yang berkaitan.

b) Fungsi

i) `calculate(List[List[Integer]], String)`
Menghitung array input pada perceptron dengan bobot sesuai dengan atribut `weights`. Fungsi akan memanggil fungsi aktivasi sesuai parameter yang diberikan dan mengembalikan hasilnya (list output).

ii) `sigmoid(List[List[Float]]) : List[List[Float]]`
Menghitung dengan fungsi aktivasi Sigmoid.

iii) `relu(List[List[Float]]) : List[List[Float]]`
Menghitung dengan fungsi aktivasi ReLU.

iv) `softmax(List[List[Float]]) : List[List[Float]]`
Menghitung dengan fungsi aktivasi Softmax.

v) `error_softmax(Float) : List[Float]`
Menghitung error softmax dengan fungsi cross entropy.

vi) `d_linear() : List[List[Float]]`
Menghitung dengan fungsi turunan aktivasi Linear.

vii) `d_relu() : List[List[Float]]`
Menghitung dengan fungsi turunan aktivasi ReLU.

viii) `d_sigmoid() : List[List[Float]]`
Menghitung dengan fungsi turunan aktivasi Sigmoid.

ix) `d_softmax() : List[List[Float]]`
Menghitung dengan fungsi turunan aktivasi SoftMax.

x) `update_weight_hidden(Float,String,List[Float],List[Float])`
Melakukan update bobot perceptron untuk hidden layer.

xi) `update_weight_output(Float, String, List[Float])`
Melakukan update bobot perceptron untuk output layer yang menggunakan fungsi aktivasi selain softmax.

xii) `update_weight_output_softmax(Float, List[List[Float]])`
Melakukan update bobot perceptron untuk output layer yang menggunakan fungsi aktivasi softmax.

xiii) `calculate_gradient(String) : List[Float]`
Menghitung gradien perceptron menggunakan fungsi turunan aktivasi linear, sigmoid, atau ReLU.

xiv) `calculate_gradient_softmax(List[Float]) : List[Float]`
Menghitung gradien perceptron menggunakan fungsi turunan aktivasi softmax.

B. Hasil Eksekusi

Berikut adalah hasil eksekusi dengan dataset Iris dan model sebagai berikut:

C. Perbandingan dengan Hasil MLP Sklearn

Berikut ini adalah perbandingan dengan hasil MLP Sklearn.

D. Pembagian Tugas

No.	NIM - Nama	Bagian Tugas
1	13518021 - Zaidan Naufal Sudrajat	Backpropagation pada model, turunan softmax, calculate backpropagation output dan hidden layer
2	13519133 - Delisha Azza Naadira	Calculate gradient, cross entropy, divide input into mini batches
3	13519137 - Siti Iedrania Azzariyat Akbar	Turunan linear, sigmoid, ReLU, update build model, sum squared error
4	13519166 - Karina Imani	Update weight output dan hidden, calculate backpropagation output dan hidden layer

*) Sebagian besar tugas dikerjakan secara bersamaan melalui LiveShare.

E. Lampiran

- **GitHub dari Source Code : Link**