# Tugas Besar **IF3270 Pembelajaran Mesin**

**Artificial Neural Network**

**Versi**: 20 April 2024

Tugas besar ini dikerjakan berkelompok, terdiri atas 4 mahasiswa (boleh gabungan mahasiswa K1, K2, dan K3). Terdapat 2 bagian yang akan dikerjakan, dengan deadline yang berbeda.

Catatan:

1. Tugas dikumpulkan berupa hasil download notebook dalam dua format yaitu file .ipynb dan pdf, serta laporan dalam bentuk pdf.
2. Hanya **NIM terkecil** anggota kelompok saja yang mengumpulkan file tugas pada Edunex. Hal ini untuk memudahkan asisten mendata pengumpulan tugas besar. Penamaan file yang dikumpulkan: TubesX\_[NIM terkecil].zip. X ϵ {A,B}
3. Pengumpulan yang terlambat tidak diperbolehkan, perhatikan batas akhir pengumpulan setiap bagian (waktu situs kuliah ini).

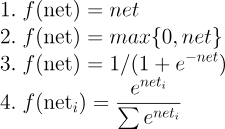
# Bagian A: Implementasi Forward Propagation untuk Feed Forward Neural Network

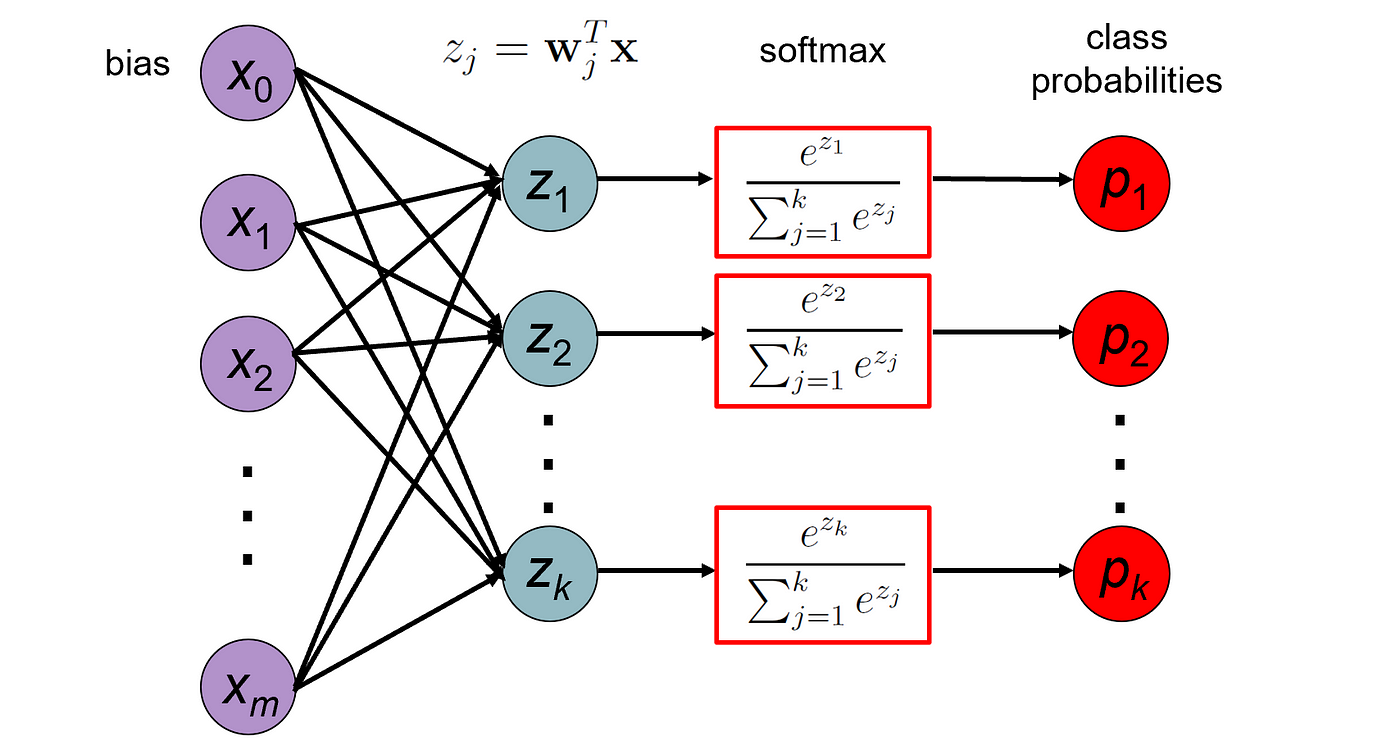
**Dikumpulkan**: 3 April 2024

Rancanglah program dalam bahasa Python untuk membuat jaringan saraf tiruan pada bagian *feed forward neural network* (FFNN).

**Spesifikasi**

1. Pada bagian A, program berfungsi sebagai *neural network* yang dapat melakukan *feed forward* dari *input* yang diberikan.
2. Program dapat menerima masukan dari file JSON.
3. Implementasi pada fungsi aktivasi berikut:
4. Linear
5. ReLU
6. Sigmoid
7. Softmax





Gambar 1. Fungsi softmax

1. Program dapat menyimpan bobot (*weights*) dan struktur model.
2. Implementasi forward propagation untuk FFNN dengan kemampuan:
3. Menampilkan model berupa struktur jaringan dan bobotnya, formatnya bebas.
4. Memberikan output untuk input 1 instance.
5. Memberikan output untuk input berupa batch.
6. Direkomendasikan membuat layer dan fungsi aktivasi secara modular.

**Batasan**

1. Satu layer memiliki fungsi aktivasi yang sama; layer yang berbeda dapat memiliki fungsi aktivasi yang berbeda.

**Kasus Uji**

1. Uji implementasi FFNN kelompok Anda dengan satu kasus sebagai berikut: [relu.json](https://drive.google.com/file/d/16lDA6ar_p_nA68cTBFO_bKTAi1s9oAHA/view?usp=drive_link)
2. Kasus uji lain akan diberikan menyusul oleh asisten.

**Deliverables**

1. Source code dalam .ipynb
2. Laporan dalam .pdf berisi:
   1. Penjelasan implementasi
   2. Hasil pengujian
   3. Perbandingan dengan hasil perhitungan manual
   4. Pembagian tugas setiap anggota kelompok

# Bagian B: Implementasi Mini-batch Gradient Descent

**Dikumpulkan:** 22 April 2024

**Spesifikasi**

1. Lakukan implementasi *backpropagation* dari modul FFNN yang sebelumnya telah dibuat berdasarkan materi kuliah.
2. Backpropagation dapat melakukan update weight saat training secara mini-batch, yang diatur melalui parameter **batch\_size**.
3. Algoritma fungsi aktivasi yang diimplementasikan adalah ReLU, sigmoid, linear, dan Softmax.
   1. Turunan fungsi aktivasi:

| **Nama Fungsi Aktivasi** | **Turunan** |
| --- | --- |
| ReLU |  |
| Sigmoid |  |
| Linear |  |
| Softmax\* |  |

\*baca appendix untuk penggunaan turunan softmax

* 1. Fungsi loss:

| **Fungsi Aktivasi** | **Loss** |
| --- | --- |
| ReLU, sigmoid, dan linear |  |
| Softmax |  |

1. Update weight pada gradient descent dilakukan dengan aturan rantai.
2. Persamaan untuk update weight



1. Persamaan gradient pada hidden layer berbeda dengan output layer
2. Hidden layer



1. Output layer



Pengecualian pada **softmax** karena langsung berbentuk dE/dnet sehingga pada softmax dE/dw = dE/dnet \* dnet/dw

1. Kondisi berhenti training terjadi ketika **error kumulatif <= threshold** atau **iterasi maksimum** tercapai. Error threshold dan iterasi maksimum merupakan parameter yang dapat diubah.
2. Definisikanlah parameter apa saja yang bisa ditangani dalam implementasi kelompok Anda. Parameter yang wajib ada adalah: struktur jaringan (jumlah layer, jumlah neuron setiap layer, fungsi aktivasi setiap layer), learning\_rate, error\_threshold, max\_iter, batch\_size.

**Batasan**

1. Satu layer memiliki fungsi aktivasi yang sama; layer yang berbeda dapat memiliki fungsi aktivasi yang berbeda.

**Kasus Uji**

1. Gunakan [**dataset Iris**](https://drive.google.com/file/d/1yIG2WVsoEum49iIkCKNoKKBcb2QzJYSq/view?usp=sharing)untuk menguji implementasi modul neural network. Arsitektur yang digunakan bebas.
2. Simpan model yang telah di-training pada dataset, lalu uji implementasi neural network dengan load model dan berikan input pada model yang dimuat.
3. Kasus uji lain akan diberikan menyusul oleh asisten.

**Deliverables**

1. Source code dalam .ipynb
2. Laporan dalam .pdf berisi:
   1. Penjelasan implementasi
   2. Hasil pengujian
   3. Perbandingan dengan penggunaan library MLP sklearn atau library lain yang mendukung penggunaan neural network
   4. Pembagian tugas setiap anggota kelompok

# 

# Appendix

Penggunaan dE/dnet pada softmax:

