**TUGAS JARINGAN SYARAF TIRUAN**

**BAB 3**

**Pengenalan Pola Huruf dengan Backpropagation**



**Disusun oleh:**

**Adrian Maulana Muhammad** **06111540000099**

**M. Kevin Adnan Murbiantoro** **06111640000045**

**Departemen Matematika**

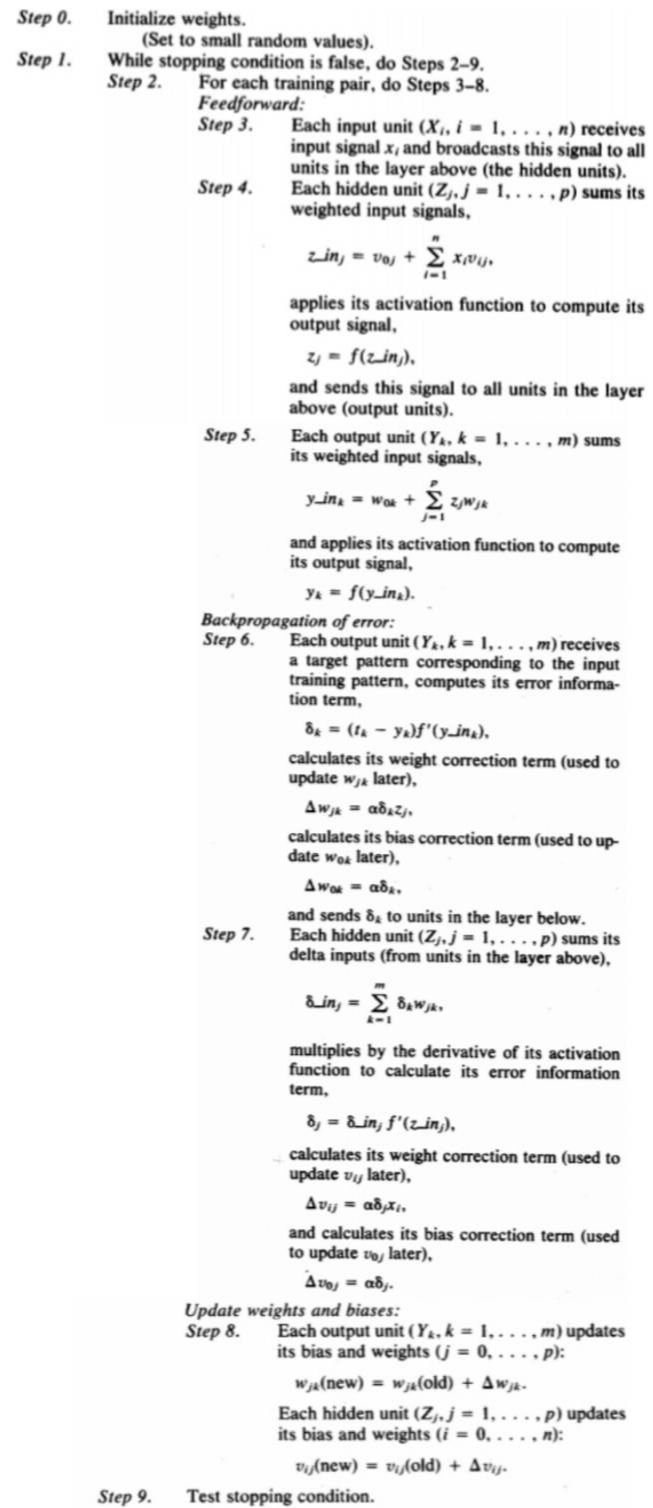
**Fakultas Sains dan Analitika Data**

**Institut Teknologi Sepuluh Nopember**

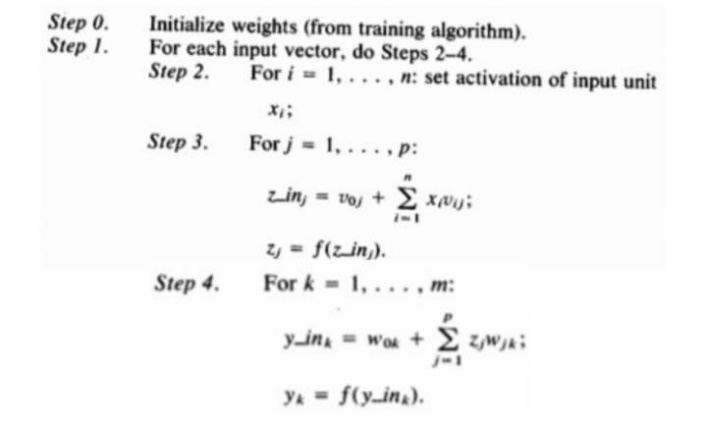
**Surabaya**

**2020**

1. **Algoritma Backpropagation**
   1. **Algoritma Training**

****

* 1. **Algortima Testing**



1. **Implementasi Algoritma Backpropagation pada Program**
2. **Implementasi Algoritma Training**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

//-------------------------- Algoritma Training --------------------------

//---------- Step 0 ----------

for (int i=0; i< hidden; i++) {

for(int j=0; j<7; j++) {

// small random values

w[i][j] = Math.random()-0.5;

}}

//---------- Step 1 ----------

// While stopping condition is false

while (it < iterasi){

//---------- Step 2 ----------

// For each training pair

for(int k=0; k<p; k++){

// Feedforward:

//---------- Step 3 ----------

// The hidden units

for(int j=0; j<hidden; j++){

z\_inJ[j]=0.0;

//---------- Step 4 ----------

for(int i=0; i<63; i++){

// z\_in

z\_inJ[j]=z\_inJ[j]+(x[i][k]\*v[i][j]);

}

z\_inJ[j] = z\_inJ[j]+biasZ[j];

}

// Penggunaan Sigmoid (aktivasi)

for (int j=0; j<hidden; j++) {

if(radio\_biner.isSelected()) {

z[j] = Sigmoid(z\_inJ[j]);

} else if(radio\_bipolar.isSelected()){

z[j] = Sigmoid2(z\_inJ[j]);

}

}

//---------- Step 5 ----------

for( int r=0; r<7; r++){

y\_inK[r]= 0.0;

for(int j=0; j<hidden; j++){

// y\_in

y\_inK[r]=y\_inK[r]+(z[j]\*w[j][r]);

}

y\_inK[r]= y\_inK[r] + biasY[r];

}

// Penggunaan Sigmoid (aktivasi)

for (int j=0; j<7; j++) {

if(radio\_biner.isSelected()) {

y[j] = Sigmoid(y\_inK[j]);

} else if(radio\_bipolar.isSelected()){

y[j] = Sigmoid2(y\_inK[j]);

}

}

//---------- Step 6 ----------

// Backpropagation of Error:

for (int i=0; i<7; i++) {

if(radio\_biner.isSelected()) {

faktorK [i] = (t[i][k]-y[i])\*DiffSigmoid(y\_inK[i]);

} else if(radio\_bipolar.isSelected()){

faktorK [i] = (t[i][k]-y[i])\*DiffSigmoid2(y\_inK[i]);

}

}

for (int i=0; i<7; i++) {

for (int j=0; j< hidden; j++) {

// delta w

delta\_w[j][i] = alpha\*faktorK[i]\*z[j];

}

// delta bias

delta\_biasY[i] = alpha\*faktorK[i];

}

//---------- Step 7 ----------

for(int j=0; j<hidden; j++){

faktor\_inJ[j]=0.0;

for( int r=0; r<7; r++){

faktor\_inJ[j]=faktor\_inJ[j]+(faktorK[r]\*w[j][r]);

}

}

for(int j=0; j<hidden; j++){

if(radio\_biner.isSelected()) {

faktorJ[j]=faktor\_inJ[j]\*DiffSigmoid(z\_inJ[j]);

} else if(radio\_bipolar.isSelected()) {

faktorJ[j]=faktor\_inJ[j]\*DiffSigmoid2(z\_inJ[j]);

}

}

for(int j=0; j<hidden; j++){

for(int i=0; i<63; i++){

delta\_v[i][j]=alpha\*faktorJ[j]\*x[i][k];

}

delta\_biasZ[j]=alpha\*faktorJ[j];

}

//---------- Step 8 ----------

for(int j=0; j<hidden; j++){

for(int r=0; r<7; r++){

// updating weight

w[j][r]=w[j][r]+delta\_w[j][r];

}

}

for(int j=0; j<63; j++){

for(int r=0; r<hidden; r++){

v[j][r]=v[j][r]+delta\_v[j][r];

}

}

for(int j=0; j<hidden; j++){

// updating bias

biasZ[j]=biasZ[j]+delta\_biasZ[j];

}

for(int j=0; j<7; j++){

biasY[j]=biasY[j]+delta\_biasY[j];

}

for(int i=0; i<63;i++){

for(int j=0; j<hidden; j++){

vtest[i][j]=v[i][j];

}

}

for(int j=0; j<hidden;j++){

for(int r=0; r<7; r++){

wtest[j][r]=w[j][r];

}

}

for(int j=0; j<hidden;j++){

biastestZ[j]=biasZ[j];

}

for(int r=0; r<7;r++){

biastestY[r]=biasY[r];

}

}

//---------- Step 9 ----------

// Iterasi bertambah

it++;

}

//------------------------------------------------------------------------------

* 1. **Implementasi Algoritma Testing**

//--------------Algoritma Testing -------------------------------------------------

//--------- Step 1 ---------

// for each input vector

for(int i=0;i<63; i++){

if(radio\_biner.isSelected()) {

//--------- Step 2 ---------

// set activation of input unit

if(inputS[i]!=null) {

S[i]=1;

} else {

S[i]=0;

}

} else if (radio\_bipolar.isSelected()){

if(inputS[i]!=null) {

S[i]=1;

} else {

S[i]=-1;

}}}

//--------- Step 3 ---------

// z\_in

for(int i=0; i<hidden; i++){

z\_inJ[i]=0.0;

for(int j=0; j<63; j++){

z\_inJ[i]=z\_inJ[i]+(S[j]\*v[j][i]);

}

z\_inJ[i] = z\_inJ[i]+biastestZ[i];

}

for(int j=0; j<hidden; j++) {

if(radio\_biner.isSelected()) {

// zj

z[j] =Sigmoid(z\_inJ[j]);

} else if(radio\_bipolar.isSelected()){

z[j] =Sigmoid2(z\_inJ[j]);

}

}

//--------- Step 4 ---------

for(int i=0; i<7; i++){

// y\_in

y\_inK[i]=0.0;

for(int j=0; j<hidden; j++){

y\_inK[i]=y\_inK[i]+(z[j]\*W[j][i]);

}

y\_inK[i] = y\_inK[i]+B[i];

}

for(int j=0; j<7; j++) {

if(radio\_biner.isSelected()) {

y[j] = Sigmoid(y\_inK[j]);

} else if(radio\_bipolar.isSelected()){

y[j] = Sigmoid2(y\_inK[j]);

}

}

for ( int k=0; k<7; k++) {

if(radio\_biner.isSelected()) {

// yk

t[k] = (int)(Math.round(y[k]));

} else if(radio\_bipolar.isSelected()) {

if(y[k] > 0) {

t[k] = (int)(Math.ceil(y[k]));

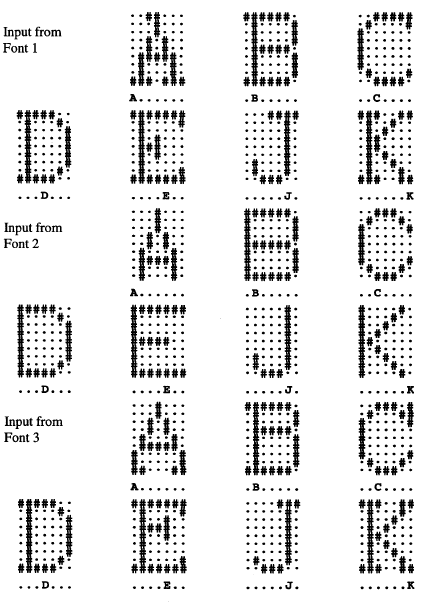
} else {

t[k] = (int)(Math.floor(y[k]));

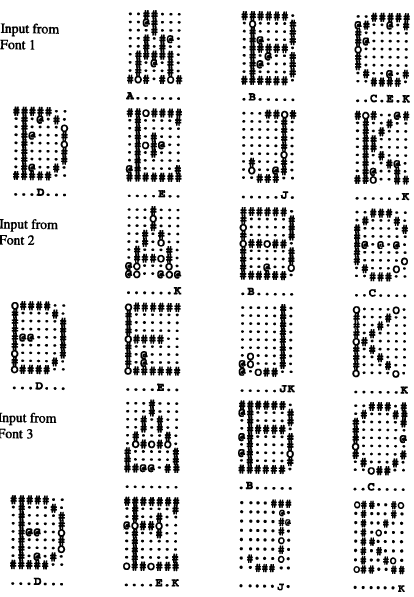
}}}

//----------------------------------------------------------------------------

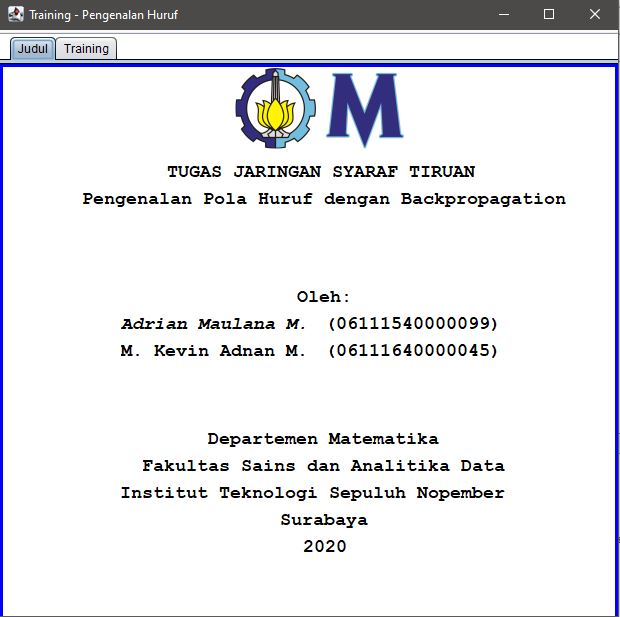
1. **Implementasi Data pada Program**
   1. **Data Input Training**

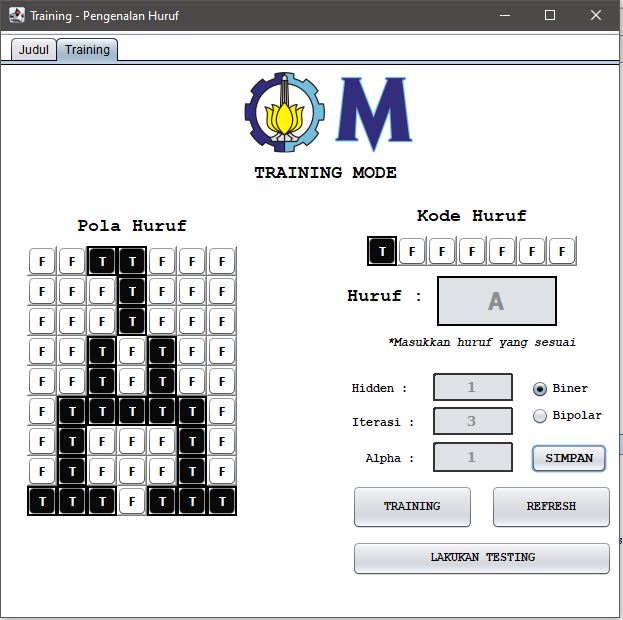


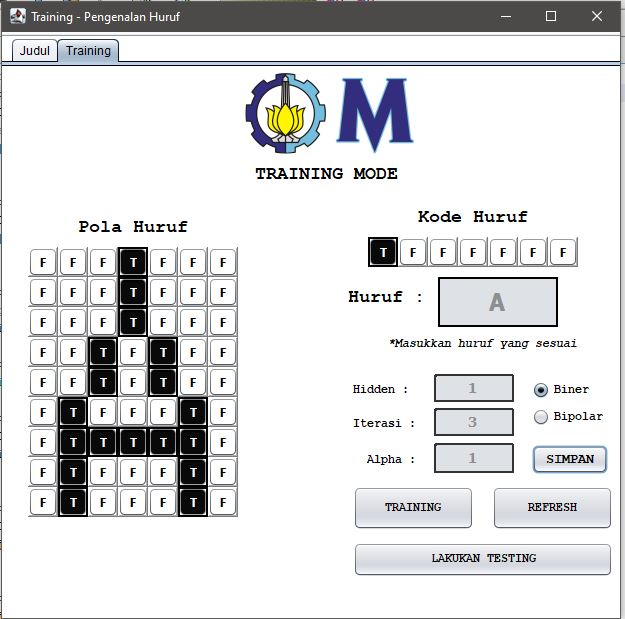
* 1. **Data Input Testing**

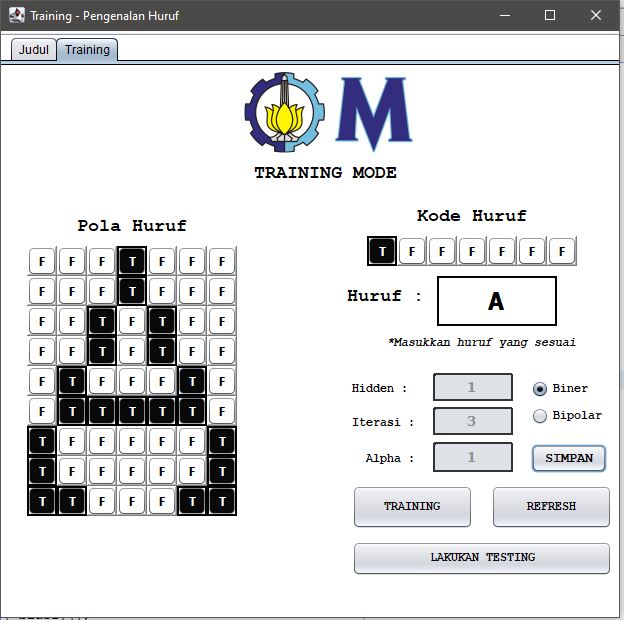


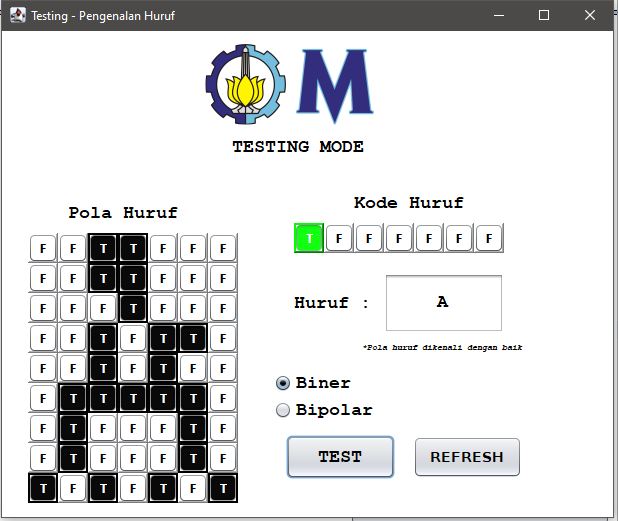
1. **Output dari Program**

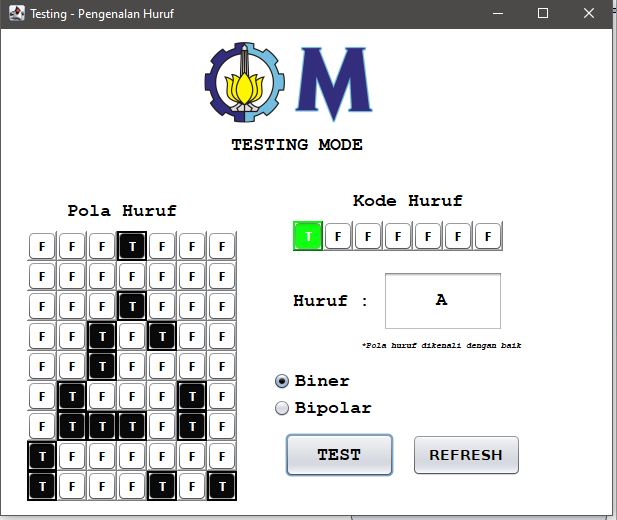
****

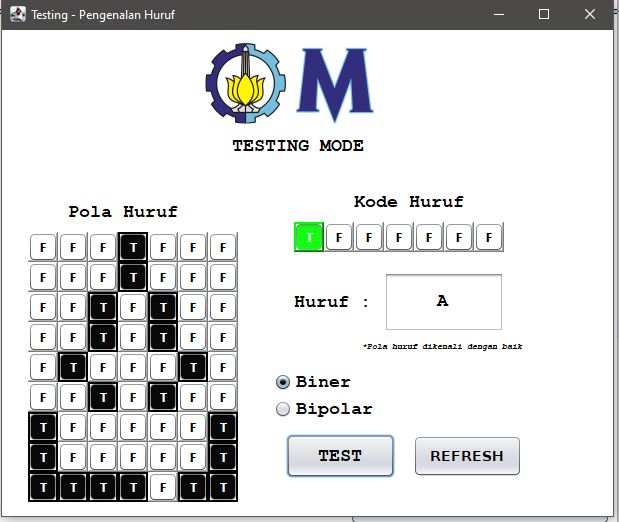
****

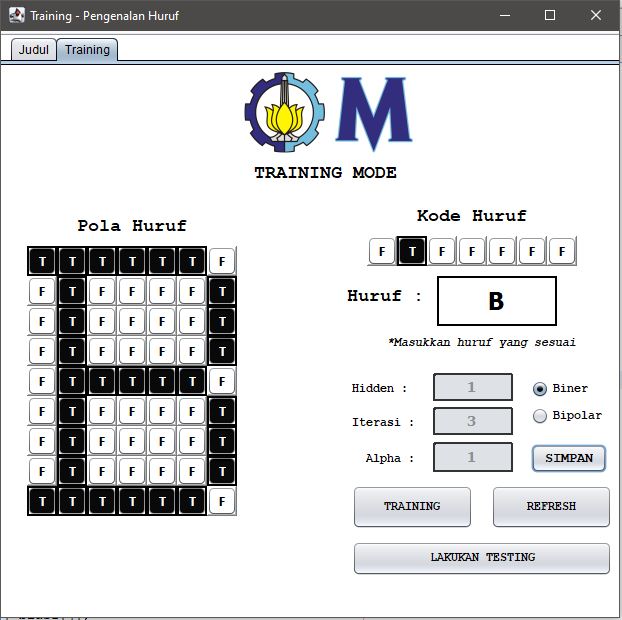
****

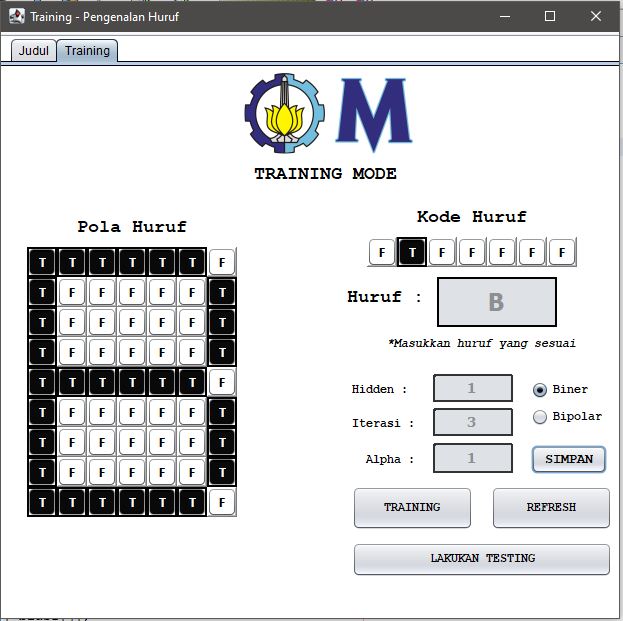
****

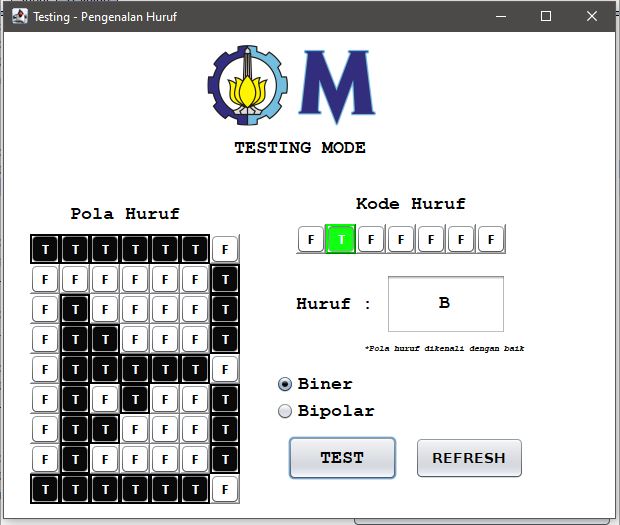
****

****

****

****

****

****

1. **Referensi**

# [Laurene Fausett](https://www.google.co.id/search?tbo=p&tbm=bks&q=inauthor:%22Laurene+Fausett%22), [Laurene V. Fausett](https://www.google.co.id/search?tbo=p&tbm=bks&q=inauthor:%22Laurene+V.+Fausett%22), 1994, “*Fundamentals of Neural Networks: Architectures, Algorithms, and Applications”.*