SNARTISI
Seminar Nasional Rekayasa Teknologi Informasi
01 – 03 November 2018
Fakultas Teknologi Informasi, Universitas Serang Raya

Brain Computer Interface untuk Menggerakkan Robot Menggunakan *Recurrent Neural Network*

Mohamad Reza Aditya Putra[1], Esmeralda C. Djamal[2], Ridwan Ilyas[3]
[1)]Jurusan Informatika, Fakultas Sains dan Informatika,Universitas Jenderal Achmad Yani
[2)]Jurusan Informatika, Fakultas Sains dan Informatika, Universitas Jenderal Achmad Yani
[3]Jurusan Informatika, Fakultas Sains dan Informatika, Universitas Jenderal Achmad Yani
Jl. Raya Serang – Cilegon Km. 05 (Taman Drangong), Serang – Banten
Email: mohrezaadityap@gmail.com[1]

ABSTRAKS

Menggerakkan sesuatu adalah salah satu aktivitas yang dilakukan dengan menggunakan otak. Bagi seorang berkebutuhan khusus terutama tuna daksa, hal tersebut sangat sulit dilakukan dengan kondisi ketidakmampuan anggota tubuh dalam melaksanakan fungsinya secara normal.Keterbatasan seorang tuna daksa dapat dibantu dengan penggunaan pikiran untuk menggerakkan suatu benda tanpa melibatkan otot dan gesture. Teknologi semacam ini disebut dengan Brain Computer Interface (BCI). Dengan menggunakan perangkat intermediet seperti Elektroensephalogram, BCI mampu menerjemahkan sinyal gelombang yang dihasilkan oleh otak menjadi perintah. Untuk mendapatkan variable pikiran seseorang, dalam menggerakkan benda, diperlukan pelatihan dan pembelajaran menggunakan Recurrent Neural Networks (RNNs). Data pelatihan tersebut didapatkan dari 20 orang naracoba dengan kondisi sehat, dilakukan perekaman sebanyak 5 kali dalam 1 menit, dan dilakukan perulangan sebanyak 5 kali. Model komputasi ini akan diterapkan dalam perangkat lunak berbasis BCI dan akan terhubung dengan microcontroller melalui perangkat tambahan yaitu modul bluetooth agar dapat lebih mudah untuk menggerakkan robot. Penelitian ini telah membangun perangkat lunak BCI untuk menggerakkan robot. Sinyal EEG yang ditangkap dan diidentifikasi menggunakan Recurrent Neural Network untuk mendapatkan lima kelas yaitu maju, mundur, kanan, kiri, dan berhenti. Pengujian dilakukan dengan 600 data dimana 480 untuk data latih dan 120 untuk data uji dan dengan learning rate 0.01 sebesar 91%, learning rate 0.02 sebesar 77%, learning rate 0.03 sebesar 64%, learning rate 0.04 sebesar 63% dan learning rate 0.05 sebesar 55%. Hasil pengujian akurasi RNN per 2 kanal didapatkan bahwa pada kanal AF3 & AF4 didapatkan akurasi sebesar 68% dan kanal T7 & T8 didapatkan akurasi sebesar 60%.

Kata Kunci: Tunadaksa, Brain Computer Interface, Elektroensephalogram, Menggerakkan Benda, Recurrent Neural Network.

1. PENDAHULUAN

1.1 Latar Belakang

Menggerakkan sesuatu adalah salah satu aktivitas yang dilakukan dengan menggunakan otak. Bagi seorang berkebutuhan khusus terutama tuna daksa, hal tersebut sangat sulit dilakukan dengan kondisi ketidakmampuan anggota tubuh dalam melaksanakan fungsinya secara normal. Penyebab tuna daksa ini dapat diakibatkan karena luka, penyakit atau pertumbuhan yang tidak sempurna (Efendi, 2006). Keterbatasan seorang tuna daksa dapat dibantu dengan penggunaan pikiran untuk menggerakkan suatu benda tanpa melibatkan otot dan gesture. Teknologi semacam ini disebut dengan Brain Computer Interface (BCI).

Brain Computer Interface (BCI) adalah sebuah teknologi yang memungkinkan manusia untuk memanfaatkan sinyal yang dibangkitkan oleh otak untuk mengirim perintah ke komputer atau mesin (Varghese, et al., 2015). Tujuan sistem ini adalah untuk membantu manusia yang memiliki kelainan fisiologi atau cacat fisik yang berhubungan dengan sistem saraf motorik. Teknologi ini menggunakan pengendali utama yaitu otak. Dengan menggunakan perangkat intermediet seperti Elektroensephalogram (EEG), BCI mampu menerjemahkan aktivitas listrik yang dihasilkan oleh otak menjadi kelas perintah tertentu.

EEG adalah suatu teknik untuk merekam aktivitas listrik di bagian yang berbeda di otak dan mengubah informasi ini menjadi suatu pola. Sinyal inilah yang ditangkap dan direkam dengan bantuan komputer sehingga aktivitas otak dapat teridentifikasi. Pada EEG terkandung informasi pikiran termasuk saat membayangkan menggerakkan benda, walaupun hal ini tidaklah mudah.

ISBN : 978-602-53437-0-4

Salah satu cara untuk identikasi pikiran saat membayangkan untuk menggerakkan benda adalah dengan pembelajaran mesin. Berbagai teknik dalam pembelajaran mesin. Salah satu teknologi dalam identifikasi sinyal saat ini adalah Deep Learning. Perkembangan hardware saat ini memungkinkan perkembangan Deep Learning. Beberapa penelitian untuk identifikasi sinyal EEG menggunakan Deep Learning adalah mengenali emosi berbasis EEG dengan tingkat akurasi mencapai 53,42%. Dalam jurnal tersebut Deep Learning memiliki kinerja akurasi yang lebih baik dibandingkan dengan SVM dan pengelompokan naiveBayes. Deep Learning memiliki beberapa metode pembelajaran salah satunya Recurrent Neural Networks (RNNs). RNNs adalah salah satu bagian dari keluarga *Neural Network* untuk memproses data yang bersambung (*sequential data*).

Aplikasi BCI dapat digunakan untuk mengontrol robot. Robotika bukanlah sesuatu yang baru saat ini, sehingga pengembangan dari robot ini sudah banyak dilakukan dalam segala hal pengaplikasiannya, salah satunya adalah robot tangan yang bergerak otomatis dengan menggunakan sensor sebagai sistem kontrol (Nurmadyansyah & Arifin, 2014) ataupun robot beroda yang dikendalikan secara manual oleh manusia melalui remote kontrol (Widiyanto & Nuryanto, 2015). Dengan berkembangnya teknologi yang memanfaatkan sinyal yang dibangkitkan oleh otak untuk mengirim perintah ke komputer atau mesin (Varghese, et al., 2015), memungkinkan seseorang dapat meggerakkan robot dengan menggunakan pikiran. Hal ini memberikan pemikiran baru akan robot yang dapat dikendalikan melalui pikiran dengan Brain Computer Interface.

Penelitian ini akan membangun sebuah sistem untuk menggerakkan sebuah robot berdasarkan pikiran melalui perekaman sinyal EEG secara *real-time*. Pikiran yang digunakan adalah maju, mundur, kanan, kiri, dan stop. Sistem tersebut akan melakukan segmentasi dari sinyal EEG berdasarkan gelombang, sehingga menghasilkan beberapa layer. Kemudian dilakukan identifikasi menggunakan Recurrent Neural Networks (RNNs). Sistem tersebut akan diimplementasikan ke dalam perangkat lunak berbasis BCI yang terintegrasi dengan *wireless* EEG dan mentransmisikan hasil klasifikasi RNNs ke *microcontroller* melalui modul *bluetooth* sehingga dapat menggerakkan robot.

1.2 Rumusan Masalah

Memikirkan robot bergerak maju, mundur, kanan, kiri, dan stop biasanya melibatkan otot atau gesture. Pada penderita tuna daksa hal ini sangat menyulitkan. Walaupun demikian, dalam otak manusia, ketika ada keinginan untuk menggerakkan benda atau membayangkan gerakan, direfleksikan dari aktivitas listrik di otak yang direkam melalui sinyal EEG. Namun menerjemahkan pikiran saat membayangkan gerakan benda dari sinyal EEG tidaklah mudah mengingat bercampurnya informasi dari variabel lain dan sinyal tersebut sangat kompleks dan rentan terhadap noise.

Kemampuan menerjemahkan atau mengidentifikasi pikiran saat membayangkan gerakan benda dapat diimplementasikan dalam bentuk kendali robot.

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah membangun sebuah sistem yang dapat menggerakkan robot melalui sinyal EEG berdasarkan lima kondisi pikiran dari sinyal EEG menggunakan Recurrent Neural Netwoks secara *real-time* berbasis BCI. Hasil penelitian akan dipublikasikan dalam seminar nasional informatika.

1.4 Batasan Masalah

Terdapat beberapa batasan masalah pada penelitian ini, yaitu :

a. Perolehan data diambil dari empat kanal, yaitu AF3, AF4, T7, dan T8.
b. Variabel lainnya dianggap tetap selain 5 kondisi pikiran, yaitu maju, mundur, kanan, kiri, dan berhenti.

2. PEMBAHASAN

2.1 Brain Computer Interface

BCI adalah suatu perangkat keras ataupun perangkat lunak yang memberikan peluang untuk komunikasi antara sinyal otak dan alat kendali seperti mesin ataupun komputer. Salah satu pengaplikasinya adalah untuk membantu orang yang mempunyai masalah dengan fisiknya seperti kelumpuhan, namun otaknya masih berfungsi dengan baik. Penelitian sebelumnya, teknologi BCI banyak digunakan untuk mengendalikan perangkat keras ataupun perangkat lunak. Teknologi BCI dilakukan dengan pengukuran sinyal otak, dan kemudian dilakukan sistem pengolahan sinyal otak tersebut untuk mendeteksi pola-pola unik yang akan diterjemahkan menjadi perintah, seperti pola otak saat memikirkan benda bergerak maju, mundur, kanan, kiri, dan berhenti.

Pada penelitian lain, BCI digunakan sebagai sistem kendali robot, pada penelitian tersebut menggunakan *microcontroller* untuk memproses sinyal EEG dan mengendalikan perangkat

ISBN : 978-602-53437-0-4

(Varghese, et al., 2015). Terdapat beberapa penelitian lain yang dilakukan dengan teknologi BCI menggunakan modul bluetooth untuk pengendalian robot. Ada juga yang menggunakan BCI untuk mengendalikan gerak mobile robot menggunakan Emotiv Epoc+ sebagai penangkap sinyal EEG dengan ditambah rangkain pendukung Bluetooth HC-5 dan rangkaina driver, mobile robot mampu mengikuti perintah secara realtime dengan lima kelas (Rizal & Wijaya, 2016) dan ada juga penelitian lain yang menggunakan teknologi BCI untuk menggerakkan karakter game (Abdullah, Djamal, & Renaldi, 2016).

2.2 Sinyal Elektroensephalogram (EEG)

EEG merupakan teknik untuk merekam aktivitas listrik di bagian yang berbeda di otak dan dan mengubah informasi ini menjadi suatu pola. Sinyal inilah yang ditangkap dan direkam dengan bantuan komputer sehingga aktivitas otak dapat teridentifikasi.

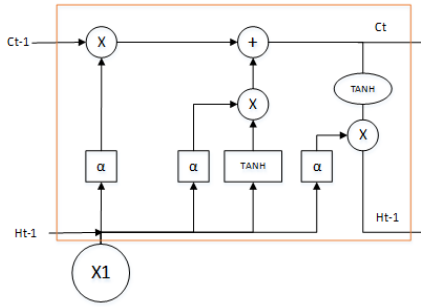
Pada penelitian ini akan menggunakan metode Recurrents Neural Networks (RNNs) untuk menggerakkan robot dan menggunakan 4 kanal sinyal EEG, yaitu AF3, AF4, T7, dan T8. Kanal digunakan untuk mengidentifikasi sinyal otak dengan sensor yang ada pada kanal tersebut.

2.3 Recurrent Neural Network

Deep learning merupakan pengembangan dari Jaringan Saraf Tiruan (Artificial Neural Network) yang memiliki lebih banyak lapisan (layer). Dengan Lapisan yang lebih banyak, Deep learning diharapkan untuk dapat mengenali proses yang lebih kompleks. Dengan kata lain, Deep Learning merepresentasikan pengetahuan yang lebih detil.

Pada umunya, manusia membuat keputusan secara tunggal setiap saat. Dalam membuat suatu keputusan biasanya kita memperhitungkan masa lalu. Cara berpikir seperti ini adalah dasar dari pengembangan *Recurrent Neural Networks.*

Recurrent Neural Networks (RNNs) merupakan model yang meniru cara berpikir dalam pengambilan keputusan tersebut dimana RNNs tidak membuang begitu saja informasi dari masa lalu dalam proses pembelajarannya.



Dalam arsitektur RNNs tersebutterdapat *cells* dengan beberapa *inputx*$_1$ dan *output* dari *cell* tersebut merupakan isi dari h_t. Perulangan pada *cells* memungkinkan informasi untuk dilalui dari satu langkah dalam jaringan tersebut ke langkah berikutnya. Perulangan tersebut membuat RNNs terlihat berbeda dari Neural Network biasanya. RNNs mengikuti beberapa neuron dari jaringan yang sama dimana masing-masing neuron menyampaikan informasi kepada neuron selanjutnya yang dihubungkan dengan bobot. Koneksi dari satu *cell* ke *cell* selanjutnya membuat jaringan dapat menyimpan informasi pada memori sementara yang membawa dampak dalam cara *input* untuk menghadirkan kembali nilai masa lalu ke dalam jaringan tersebut. Dilihat dari penjabaran tersebut setiap jaringan mengambil nilai *inputx* kemudian dimasukkan ke dalam *cell* yang berisi suatu *gate* yang akan diperbarui setiap kali jaringan tersebut membaca masukan baru sehingga menghasilkan *output* pada setiap waktu.

Proses pelatihan pada RNNs sangat mirip dengan pelatihan pada Neural Network menggunakan algoritma Backpropagation tetapi dengan sedikit putaran. Karena parameter yang dibagikan secara merata pada setiap *time step*, maka *gradient* untuk setiap *output* tergantung tidak hanya pada kalkulasi dari *time step* saat ini, tetapi juga pada *time step* sebelumnya. Pada RNNs terdapat beberapa unit *gate* seperti Gate Recurrent Unit (GRU), Backpropagtaion Through Time (BPTT) dan Long Short Term Memory (LSTM). Penelitian ini menggunakan jenis *gate* LSTM yang merupakan jenis istimewa dari RNNs. LSTM digunakan karena dapat mengatasi ketergantungan proses dalam jangka panjang yang merupakan suatu fenomena yang timbul dalam pengolahan data sekuensial.
Kunci dari LSTM adalah *cellstate* dengan arsitektur yang ditandai dengan garis horizontal yang mengalir dari C_{t-1} sampai C_t. LSTM memiliki kemampuan untuk mengahapus atau menambahkan informasi ke *cell state* diatur oleh struktur yang disebut *gate*. Dimana *gate* merupakan suatu cara untuk melewatkan informasi yang terdiri dari fungsi sigmoid biner

ISBN : 978-602-53437-0-4

(σ) dan operasi perkalian dengan *x*. Fungsi sigmoid biner merupakan fungsi aktivasi yang digunakan agar data memiliki rentang antara 0 dan 1 yang terlihat pada Persamaan 1. Langkah pertama pada LSTM adalah dengan memutuskan informasi apa yang akan dibuang dari *cell state* yang disebut *forget gate* menggunakan Persamaan 2.

$$f(x) = \frac{1}{1+e^{-x}} \quad (1)$$

$$f_t = \sigma(W_f . [h_{t-1}, x_t] + b_f) \quad (2)$$

Dimana nilai h_{t-1} dan x_t adalah rentang 0 sampai 1 untuk setiap *cell*. Nilai 1 mewakili informasi tersebut disimpan dan 0 mewakili informasi tersebut dihapus. Langkah kedua adalah memutuskan informasi baru yang akan disimpan pada *cell*. Pada langkah ini terdapat dua bagian, yang pertama adalah *input gate* yang memntukan nilai yang akan diperbarui menggunakan Persamaan 3 dan perhitungan untuk kandidat *cell* baru ($\tilde{C}_t$) yang menghasilkan vektor nilai menggunakan fungsi aktivasi tanh yang terlihat pada Persamaan 4.

$$i_t = \sigma(W_f . [h_{t-1}, x_t] + b_i) \quad (3)$$

$$\tilde{C}_t = \tanh(W_c . [h_{t-1}, x_t] + b_c) \quad (4)$$

Untuk memperbarui *cell state* dilakukan dengan menjumlahkan *cell state* (C_t) yang lama dengan kandidat *cell* ($\tilde{C}_t$) menggunakan Persamaan 5. Dimana *cell state* lama dikalikan dengan *forget state* dan kandidat *cell* dikalikan dengan *input gate*.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (5)$$

Langkah terakhir adalah *output gate* yang digunakan untuk menentukan *output* yang akan dihasilkan berdasarkan *cell state* yang hasil dari Persamaan 5 dan melakukan kalkulasi dengan fungsi sigmoid biner terlihat seperti pada Persamaan 6. Kemudian hasil dari Persamaan 6 dikalikan dengan fungsi aktivasi tanh dari *cell state* yang telah diperbarui menggunakan Persamaan 7 sehingga dapat menghasilkan informasi yang ingin dikeluarkan.

$$o_t = \sigma(W_o . [h_{t-1}, x_t] + b_o) \quad (6)$$

$$h_t = o_t * \tanh(C_t) \quad (7)$$

Keterangan:
σ = fungsi aktivasi sigmoid dengan rentang (0, 1)
tanh = fungsi aktivasi tangent dengan rentang (-1, 1)
W_f, W_t,W_c, W_o = matriks bobot hasil *random*
h_t =*hidden state* saat ini
h_{t-1} =*hidden state* sebelumnya
b_f, b_i, b_c, b_o = vektor bias
c_t =*cell state* saat ini
c_{t-1} =*cell state* sebelumnya

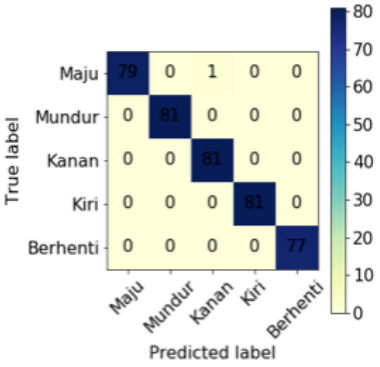
2.4 Perolehan Data

Sinyal direkam dalam format .csv yang terdiri dari 20 kolom. Namun, karena elektroda yang digunakan terdapat 4 kanal yaitu kanal AF3, AF4, T7, dan T8 sehingga hanya kolom ke 3,4,6, dan 7 yang berisi sinyal EEG.

Perekaman untuk data latih secara offline dilakukan sepanjang 5 detik. Sementara membayangkan gerakan sebagai variabel dalam penelitian ini diasumsikan selama 1 detik. Oleh karena itu, setiap set data latih dibagi atas 5 segmen. Apabila sinyal EEG direkam dengan frekuensi sampling 128 Hz, sehingga dalam satu kali perekaman dalam 4 kanal menghasilkan data sebanyak 2560 titik atau 128 x 5 detik x 4 kanal. Mengingat setiap perekaman dibagi atas 1 detik maka sinyal EEG yang diolah dari 4 kanal sebanyak 512 titik.

2.5 Hasil Pengujian

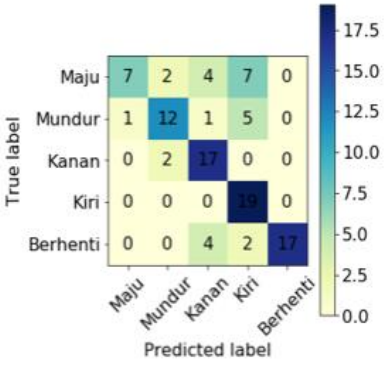
Pada tahap ini dilakukan pengujian terhadap kinerja RNN pada setiap kelas, pengaruh jumlah data, pengujian akurasi data latih dan data baru. Hasil uji pengaruh kelas untuk data latih dapat dilihat pada Gambar 1



Gambar 1. *Confusion Matrix* uji pengaruh

Kelas Untuk Data Latih

Sedangkan hasil uji pengaruh kelas untuk data uji dapat dilihat pada Gambar 2.



Gambar 2.*Confusion Matrix* Uji Pengaruh Kelas Untuk Data Uji

ISBN : 978-602-53437-0-4

Pada Gambar 1, uji pengaruh kelas terhadap data latih didapatkan akurasi sebesar 94%, sedangkan pada Gambar 2, uji pengaruh kelas terhadap data uji didapatkan akurasi sebesar 72%.

2.6 Algoritma Segmentasi

Algoritma segementasi dilakukan untuk memotong data dari 5 detik menjadi 1 detik. Contoh algoritma dapat dilihat dalam Gambar 3.

```
Function segmen(filename :array[], detik_awal, detik_akhir, path :string) □ array[] of string
{I.S :data belum disegemntasi}
{F.S :data berhasil disegmentasi dengan data berbentuk array}
{kamus}
Kolom3,Kolom4,Kolom6,Kolom7,Kanal3,Kanal4,kanal6,kanal7 : array[] of real
i,j : integer
dataframe, datakanal : array[] of real
{Algoritma}
detik_awal  = 1
detik_akhir = 12
For i = 0 to filename.lenght do
  Dataframe = read(path)
  Kolom3 = dataframe.kolom(3)
  Kolom4 = dataframe.kolom(4)
  Kolom6 = dataframe.kolom(6)
  Kolom7 = dataframe.kolom(7)
  For detik_awal to detik_akhir do
      Kanal3 = Kolom3(detik_awal * 128, (detik_awal + 1) * 128)
      Kanal4 = Kolom4(detik_awal * 128, (detik_awal + 1) * 128)
      Kanal6 = Kolom6(detik_awal * 128, (detik_awal + 1) * 128)
     Kanal7 = Kolom7(detik_awal * 128, (detik_awal + 1) * 128)
      Datakanal = Flatten(Kanal3, Kanal4, Kanal6, Kanal7)
  Endfor
Endfor
End Function
```

Gambar 3. Algoritma Segmentasi

2.7 Algoritma RNN

Algoritma Recurrent Neural digunakan untuk melakukan generalisai data latih. Tahap pertama adalah algoritma untuk set bobot awal. Dapat dilihat pada Gambar 4.

```
Procedure setBobotAwal()
{I.S :Bobot awal terdefinisi dan mungkin kosong}
{F.S :Bobot awal tidak kosong)}
{kamus}
i,j : integer
{Algoritma}
For i □ 1 to panjang(v) do
  For j □ 1 to Panjang(v[0]) do
    v[i][j] □ Random (-0.05 - 0.5)
  Endfor
Endfor
For i □ 1 to panjang(v0) do
  v0[i] □ Random (-0.05 - 0.5)
Endfor
For i □ 1 to panjang(w) do
  For j □ 1 to Panjang(w[0]) do
    w[i][j] □ Random (-0.05 - 0.5)
  Endfor
Endfor
End Procedure
```

Gambar 4. Algoritma Set Bobot Awal

Selanjutnya adalah algoritma aktivasi ReLu. Dapat dilihat pada Gambar 5.

```
Function relu(data_EEG : Data_EEG) □ array[]
{I.S :Data_EEG terdefinisi dan tidak kosong}
{F.S :Mengirim Data_EEG hasil relu dari eeg input)}
{kamus}
panjang,lebar : integer
data_EEG : real
Hasil_relu: real
{Algoritma}
panjang □ panjang data_EEG
lebar □ lebar data_EEG
For i □ 0 to panjang do
  For j □ 0 to lebar do
     Hasil_relu □ data_EEG[i][j]
     If Hasil_relu < 0 then
        Hasil_relu □ 0
     End if
  Data_EEG[i][j]□ Hasil_relu
  Endfor
Endfor
□ data_EEG
End Function
```

Gambar 5. Algoritma Aktivasi ReLu

Tahap berkutnya adalah algoritma feedforward. Dapat dilihat pada Gambar 6

```
Procedure hitungLSTM1()
{I.S :Cell terdefinisi dan mungkin kosong}
{F.S :Cell terisi nilai dan tidak kosong)}
{kamus}
i,j : integer
{Algoritma}
For i □ 0 to panjang(z) do
  Cell1_in[i] □ v0[i]
  For j □ 0 to panjang(x) do
    Cell1_in[i] □ (x[j] * v[j][i] + Cell1_in[i]
  Endfor
  Cell1[i] □ relu(Cell1_in[i])
  For j □ 0 to panjang(x) do
    Cell2_in[i] □ (x[j] * v[j][i] + Cell1[i]
  Endfor
  Cell2[i] □ relu(Cell2_in[i])
  For j □ 0 to panjang(x) do
    Cell3_in[i] □ (x[j] * v[j][i] + Cell2[i]
  Endfor
  Cell3[i] □ relu(Cell3_in[i])
Endfor
End Procedure
Procedure hitungOutput()
{I.S :Y terdefinisi dan mungkin kosong}
{F.S :Y terisi nilai dan tidak kosong)}
{kamus}
i,j : integer
{Algoritma}
For i □ 0 to panjang(y) do
  y_in[i] □ v0[i]
  For j □ 0 to panjang(z) do
    y_in[i] □ (Cell3[j] * w[j][i] + y_in[i]
  Endfor
  y[i] □ softmax(y_in[i])
Endfor
End Procedure
```

Gambar 6. Algoritma Feedforward

Tahap berikutnya yaitu algoritma backpropagation. Dapat dilihat pada Gambar 7.

ISBN : 978-602-53437-0-4

```
Procedure hitungDeltaY()
{I.S :deltaY terdefinisi dan mungkin kosong}
{F.S :deltaY terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 1 to panjang(y) do
  deltaY[i] □ (t[i] - y[i]) * (y[i]) * (1 - y[i])
Endfor
End Procedure
Procedure hitungLSTMDeltaW()
{I.S :deltaW terdefinisi dan mungkin kosong}
{F.S :deltaW terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 1 to panjang(z) do
  for j □ 1 to panjang(y) do
    dW[i][j] □ alpha * Cell3[i] * deltaY[j]
  Endfor
Endfor
End Procedure
Procedure hitungDZ_in()
{I.S :dZ_in terdefinisi dan tidak kosong}
{F.S :dZ_in terisi nilai dan tidak kosong)}
{kamus}
i,j : integer
{Algoritma}
For i □ 0 to panjang(z) do
  deltaZ_in[i] □ 0
  For j □ 0 to panjang(y) do
    deltaZ_in[i] □ deltaZ_in[i] + (deltaY[j] * w[i][j])
  Endfor
Endfor
End Procedure
Procedure hitungDZ()
{I.S :dZ terdefinisi dan tidak kosong}
{F.S :dZ terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 0 to panjang(z) do
  deltaZ[i] □ (deltaZ_in[i]) * (z[i]) * (1 - z[i])
Endfor
End Procedure
 Procedure hitungDV()
 {I.S :dV terdefinisi dan tidak kosong}
 {F.S :dV terisi nilai dan tidak kosong)}
 {kamus}
 i,j : integer
 {Algoritma}
 For i □ 0 to panjang(x) do
   For j □ 0 to panjang(z) do
     dv[i][j] □ alpha * x[i] * deltaZ[j]
   Endfor
 Endfor
 End Procedure
 Procedure hitungDV0()
 {I.S :dV0 terdefinisi dan tidak kosong}
 {F.S :dV0 terisi nilai dan tidak kosong)}
 {kamus}
 i : integer
 {Algoritma}
 For i □ 0 to panjang(z) do
   dv0[i] □ alpha * deltaZ[i]Endfor
 End Procedure
```

Gambar 7. Algoritma Backpropagation

Selanjutnya adalah tahap modifikasi bobot. Tahap ini mengkoreksi bobot antara *input layer, hidden layer,* dan *output layer.* Dapat dilihat pada Gambar 8.

```
Procedure perbaikiBobotV()
{I.S :bobotV terdefinisi dan tidak kosong}
{F.S :bobotV terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 0 to panjang(z) do
  For j □ 0 to panjang(z) do
    v[i][j] □ v[i][j] +dV[i][j]
  Endfor
Endfor
End Procedure
Procedure perbaikiBobotV0()
{I.S :bobotV0 terdefinisi dan tidak kosong}
{F.S :bobotV0 terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 0 to panjang(z) do
  v0[i] □ v0[i] + dv0[i]
Endfor
End Procedure
Procedure perbaikiBobotW()
{I.S :bobotW terdefinisi dan tidak kosong}
{F.S :bobotW terisi nilai dan tidak kosong)}
{kamus}
i,j : integer
{Algoritma}
For i □ 0 to panjang(z) do
  For j □ 0 to panjang(y) do
    w[i][j] □ w[i][j] +dW[i][j]
  Endfor
Endfor
End Procedure
Procedure perbaikanBobotW()
{I.S :bobotW terdefinisi dan tidak kosong}
{F.S :bobotW terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 0 to panjang(z) do
  w0[i] □ w0[i] + dw0[i]
Endfor
End Procedure
```

Gambar 8. Algoritma Modifikasi Bobot

Tahap terakhir yaitu algoritma loss. Algoritma ini digunakan mengoreksi bobot antara lapisan masukan keluaran dan lapisan tersembunyi. t, y dan Cross Entropy merupakan variabel global dengan tipe data real. t dan y merupakan variabel bertipe array satu dimensi yang memiliki panjang sebanyak lapisan keluaran (y). Dapat dilihat pada Gambar 9 .

```
Procedure Loss()
{I.S :loss terdefinisi dan mungkin kosong}
{F.S :loss terisi nilai dan tidak kosong)}
{kamus}
i : integer
{Algoritma}
For i □ 0 to panjang(y) do
  loss □ loss + (t[i]*log(y[i]))*-1
Endfor
End Procedure
```

Gambar 9. Algoritma Loss

3. KESIMPULAN

Penelitian ini telah menghasilkan sebuah sistem berbasis Brain Computer Interface (BCI) menggunakan Recurrent Neural Network. Pengujian dalam sistem BCI ini dilakukan dengan menguji akurasi sistem. Pengujian tersebut dilakukan dengan 600 data dimana 480 untuk data latih dan 120 untuk data uji dan

ISBN : 978-602-53437-0-4

dengan nilai *learning rate* yang berbeda yaitu 0.01, 0.10, 0.2, dan 0.5 dengan epoch masing-masing 150. Hasil dari pengujian akurasi RNN terhadap pengaruh leraning rate, pada sistem ini didapatkan bahwa data dengan *learning rate* 0.01 sebesar 91%, data dengan *learning rate* 0.02 sebesar 77%, data dengan *learning rate* 0.03 sebesar 64%, data dengan *learning rate* 0.04 sebesar 63% dan data dengan *learning rate* 0.05 sebesar 55%. Hasil pengujian per kanal didapat bahwa pada kanal AF3 didapatkan akurasi sebesar 44%, kanal AF4 didapatkan akurasi sebesar 51%, kanal T7 didapatkan akurasi sebesar 43%, dan pada kanal T8 didapatkan akurasi sebesar 46%. Hasil pengujian akurasi RNN per 2 kanal didapatkan bahwa pada kanal AF3 & AF4 didapatkan akurasi sebesar 68% dan kanal T7 & T8 didapatkan akurasi sebesar 60%.

PUSTAKA

Abdullah, M. Y., Djamal, E. C., & Renaldi, F. (2016). Aksi Game Arcade Berdasarkan Pikiran Menggunakan Filter Fast Fourier Transform dan Learning Vector Quantization. *Seminar Nasional Aplikasi Teknologi Informasi (SNATi)* , 17 - 22.

Alshbatat, A. I., Vial, P. J., Premaratne, P., & Tran, L. C. (2014). EEG-based brain-computer interface for automating home appliances . *Journal of Computers, 9*(9), 2159 - 2166.

Efendi, M. (2006). *Penyesuaian Diri pada Remaja Tunadaksa Bawaan.* Jakarta: Bumi Aksara.

Guler, N. F., Ubeyli, E. D., & Guler, I. (2005). Recurrent Neural Networks Employing Lyapunov Exponents for EEG Signals Classification. *Expert Systems with Applications* (29), 506-514.

Hindarto, Hariadi, M., & Purnomo, M. H. (2011). Identifikasi Sinyal Elektrode Enchepalo Graph Untuk Menggerakkan Kursor Menggunakan Teknik Sampling Dan Jaringan Syaraf Tiruan . *Jurnal Ilmiah Kursor , 6*(1), 11-18.

Mahardhika, W. A., Husni, M., & Pratomo, B. A. (2014). Pengendalian Robot Berbasis IP Melalui Jaringan Wi-Fi Menggunakan Perangkat Mobile Android . *JURNAL TEKNIK POMITS , 2*(1), 1-5.

Nurmadyansyah, R. F., & Arifin, A. (2014). Perancangan dan Implementasi Sistem Kendali Robot Tangan Prensilia. *JURNAL TEKNIK POMITS , 3*(1), F1-F6.

Rizal, A. A., & Wijaya, S. K. (2016). PENGENDALIAN GERAK MOBILE ROBOT BERBASIS BCI (BRAIN COMPUTER INTERFACE). *Prosiding Seminar Nasional Fisika (E-Journal) SNF2016 , V*, 1-6.

Shanmugapriya, T., & Senthilkumar, S. (2014). Robotics and the Brain-Computer Interface System: Critical Review for Manufacturing Application. *International Journal of Innovative Research in Computer and Communication Engineering, 2*(2), 2929-2935.

SuwichaJirayucharoensak, SethaPan-Ngum, & PasinIsrasena. (2014). EEG-Based Emotion Recognition Using Deep Learning Network with Principal Component Based Covariate Shift Adaptation. *The Scientific World Journal , 2*(1), 1-10.

Utami, E. R. (2012). Antibiotika, Resistensi, Dan Rasionalitas Terapi. *Saintis, 1*(1), 124-138.

Varghese, G., James, J., Joseph, L., John, M. K., Mathew, S., & Ramachandhran, S. (2015). Human Robot Cooperative System Based on Non-invasive Brain Computer Interface. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, 4*(1), 118-123.

Vokorokos, L., Ádám, N., & Madoš, B. (2014). Non-Invasive Brain Imaging Technique for Playing Chess with Brain-Computer Interface. *International Journal of Computer and Information Technology, 03*(05), 877-882.

Wibisono, M. R., & Findawati, Y. (2010). Game Motorik Untuk Pendidikan Anak Berkebutuhan Khusus (Tuna Grahita) Berbasis Android. *Informatika, I*(6), 20-29.

Widiyanto, A., & Nuryanto. (2015). Rancang Bangun Mobil Remote Control Android dengan Arduino. *Citec Journal, 3*(1), 50-61.

Yu, D., & Deng, L. (2015). *Automatic Speech Recognition A Deep Learning Approach.* USA: Springer.

Yulianto, E., Susanto, A., T. S., & Wibowo, S. (2013). Spektrum Frekuensi Sinyal EEG Terhadap Pergerakan Motorik dan Imajinasi Pergerakan Motorik . *Forum Teknik, 35*(1), 21-32.