SKRIPSI

MEMPREDIKSI KEMACETAN DI KOTA BANDUNG MENGUNAKAN JARINGAN SYARAF TIRUAN

STEVEN DANIEL

NPM: 2012730021

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2015

UNDERGRADUATE THESIS

TRAFFIC ANALYSIS AT BANDUNG CITY USING ARTIFICIAL NEURAL NETWORK

STEVEN DANIEL

NPM: 2012730021

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2015

LEMBAR PENGESAHAN

MEMPREDIKSI KEMACETAN DI KOTA BANDUNG MENGUNAKAN JARINGAN SYARAF TIRUAN

STEVEN DANIEL

NPM: 2012730021

Bandung, «tanggal» «bulan» 2015

Menyetujui,

Pembimbing Tunggal

Pascal Alfadian, M.Com.

Ketua Tim Penguji	Anggota Tim Penguji
«penguji 1»	«penguji 2»

Mengetahui,

Ketua Program Studi

Thomas Anung Basuki, Ph.D.

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

MEMPREDIKSI KEMACETAN DI KOTA BANDUNG MENGUNAKAN JARINGAN SYARAF TIRUAN

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal «tanggal» «bulan» 2015

Meterai

Steven Daniel
NPM: 2012730021

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Kata-kata kunci: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Keywords: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»

«kepada siapa anda mempersembahkan skripsi ini. . . ?»

KATA PENGANTAR

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Bandung, «bulan» 2015

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Setiap tahun pertumbuhan jumlah kendaraan di Bandung selalu mengalami peningkatan. Ironisnya, pertumbuhan tersebut tidak diimbangi dengan pertumbuhan pembangunan jalan yang seimbang. Akibatnya, hal tersebut mengakibatkan kepadatan lalu-lintas. Kepadatan tersebut semakin diperparah dengan banyaknya pengendara yang tidak mematuhi rambu-rambu lalu-lintas.

Twitter merupakan salah satu media sosial yang populer di dunia, Melalui Twitter, netizens dapat saling bertukar informasi secara cepat, sehingga informasi dapat tersebar ke seluruh dunia dalam hitungan beberapa detik saja melalui *tweets* yang mereka kirimkan. Pada tahun 2015 tercatat pengguna Twitter di Indonesia melebihi 50 juta orang. Sebuah perusahaan analis bernama Semiocast mencatat, pengguna Twitter di kota Bandung menyumbang lebih dari 1 miliar *tweets* sepanjang bulan juni 2012.

API (Application Programming Interface) merupakan sebuah cara yang didefinisikan sebuah program untuk menyelesaikan sebuah tugas, biasanya dengan menerima atau memodifikasi data. Twitter menyediakan sebuah API yang memberikan hak akses kepada pengembang perangkat lunak untuk membaca dan menulis data dari server Twitter. Dalam pemograman berbahasa Java ada sebuah library bernama Twitter4J yang membungkus Twitter API. Library ini memudahkan programmer Java dalam mengembangkan sebuah perangkat lunak yang memanfaatkan Twitter API.

Jaringan Saraf Tiruan(JST) adalah model komputasi yang terinspirasi dari cara kerja sistem saraf biologi.Sama seperti sistem saraf biologi, JST memiliki neuron-neuron yang dapat meneruskan sinyal apabila sinyal yang dihantarkan melewati nilai tertentu. JST sendiri digunakan untuk menyelesaikan masalah yang rumit. JST juga digunakan untuk membuat sebuah kesimpulan berdasarkan informasi yang ada.

Salah satu solusi mengatasi kemacetan pada penelitian ini adalah dengan membuat sebuah perangkat lunak yang dapat menentukan tingkat kemacetan berdasarkan *tweets* dari Twitter. Perangkat lunak akan mengambil *tweets* menggunakan library Java Twitter4J. Perangkat lunak akan memproses *tweets* menggunakan Feed Forward Neural Network. Keberadaan perangkat lunak diharapkan dapat membantu netizen dalam menghindari kemacetan.

1.2 Identifikasi Masalah

Dari observasi awal yang telah dilakukan, ada beberapa masalah yang dapat diidentifikasikan, diantaranya sebagai berikut:

1. Kemacetan di kota Bandung.
2. Penumpukan jumlah kendaraan jalur tertentu.

1.3 Batasan Masalah

Karena keterbatasan waktu yang dimiliki penulis, maka ruang lingkup penelitian yang dilakukan dibatasi untuk beberapa hal berikut:

1. Peneliti hanya memprediksi tingkat kemacetan di kota Bandung
2. Peneliti ini menggunakan bahasa Java dalam pengembangan perangkat lunak
3. Peneliti hanya menggunakan data yang berasal dari Twitter dengan kriteria *tweets* berumur kurang dari 5 tahun, bahasa Indonesia, dan berlokasi bandung.

1.4 Rumusan Masalah

Berdasarkan deskripsi diatas, rumusan masalah adalah sebagai berikut:

1. Bagaimana melakukan pengambilan *tweets* dari Twitter?
2. Bagaimana memodelkan teks / kata kunci pada *tweets* di Twitter menjadi sinyal-sinyal pada Jaringan Syaraf Tiruan?
3. Apa kriteria *tweets* yang relevan untuk menjadi input JST?
4. Apa yang menjadi output dari JST?
5. Jenis JST apa yang cocok dipakai pada penelitian ini?
6. Bagaimana mengimplementasikan JST kedalam bahasa Java?
7. Apa metode pembelajaran yang akan digunakan pada JST?
8. Bagaimana cara melatih Jaringan Syaraf Tiruan?
9. Bagaimana memprediksi tingkat kemacetan di kota Bandung.

1.5 Tujuan

Karya tulis ilmiah ini bertujuan untuk:

1. Mengambil *tweets* dari Twitter secara *programmatic*.
2. Memodelkan teks / kata kunci pada *tweets* menjadi sinyal-sinyal pada JST.

3. Menentukan kriteria *tweets* yang relevan untuk menjadi input JST.
4. Menentukan output dari JST.
5. Mengimplementasikan JST kedalam bahasa Java.
6. Menentukan konfigurasi jaringan pada JST agar cocok dapat penentuan tingkat kemacetan.
7. Menentukan metode pembelajaran yang digunakan pada JST.
8. Melatih JST untuk dapat menentukan tingkat kemacetan.
9. Memprediksi tingkat kemacetan di kota Bandung.

BAB 2

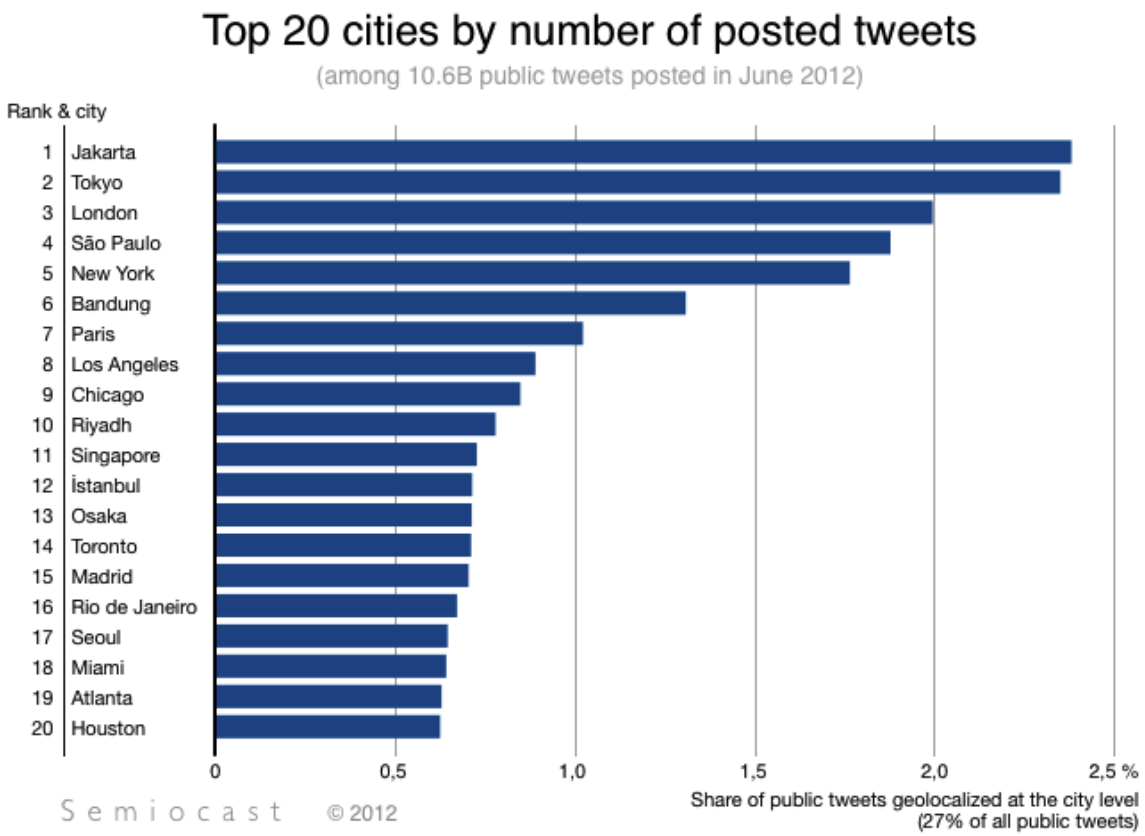
STUDI PUSTAKA

2.1 Twitter

Twitter adalah sebuah jaringan informasi yang terdiri dari pesan-pesan sepanjang 140 karakter yang disebut *Tweet* [1]. Twitter ditemukan pada tahun 2006 oleh Jack Dorsey, Evan Williams, Biz Stone, dan Noah Glass. Misi dari Twitter adalah memberikan kemampuan kepada setiap orang untuk membuat dan menyebarkan ide dan informasi secara cepat, tanpa hambatan.

Sekarang Twitter berkantor pusat di San Francisco USA. Twitter mengalami pertumbuhan yang cepat. Ditanggal 30 Juni 2015 tercatat 316 juta pengguna aktif setiap bulanya dan dalam sehari ada 500 juta tweets dikirim di seluruh dunia [2].

Pengguna Twitter di Indonesia ternyata cukup banyak. CEO Twitter Dick Costolo membeberkan bahwa jumlah pengguna Twitter di Indonesia sudah mencapai angka 50 juta dan jumlahnya makin terus bertambah [3]. Selain jumlahnya yang banyak ternyata pengguna Twitter di Indonesia cukup aktif dalam menggunakan Twitter. Dalam data yang dirilis Lembaga pemantau media sosial Semiocast "Top 20 Cities by Number of posted tweets" ada dua kota di Indonesia yang masuk 10 besar kota paling aktif dalam hal tweet. Salah satunya Kota Bandung berada di posisi 5 dengan perkiraan 1.2 persen dari 10 miliar tweets selama Juni 2012.



Tingginya popularitas Twitter dapat dimanfaatkan untuk berbagai keperluan dalam berbagai aspek, contohnya sebagai sarana komunikasi, memberikan opini, kampanye poilitik, bisnis, mendapatkan informasi, dan banyak lainya.

2.2 Twitter API

API (Application Programming Interface) merupakan sebuah cara yang didefinisikan sebuah program untuk menyelesaikan sebuah tugas, biasanya dengan menerima atau memodifikasi data. Twitter menyediakan sebuah API yang memberikan hak akses kepada pengembang perangkat lunak untuk membaca dan menulis data dari server Twitter. Programmer menggunakan Twitter API untuk membuat aplikasi, website, widgets, dan proyek lainya yang berinteraksi dengan Twitter. Program akan berkomunikasi dengan Twitter API melalui HTTP. Twitter menyediakan beberapa jenis dan fungsi API yang berbeda, diantaranya REST API, Streaming API, dan ads API.

2.2.1 REST API

REST API menyediakan akses secara program untuk membaca dan menulis data Twitter. Beberapa akses yang disediakan oleh Twitter seperti membuat tweet baru, membaca pro*file* author and data follower, dan lain-lain. REST API mengidentifikasi aplikasi dan pengguna Twitter menggunakan OAuth. Twitter menyarankan jika pengembang berniat untuk memonitor atau memproses tweets secara real-time lebih baik menggunakan Streaming API daripada REST API, dikarenakan REST API memiliki rate limits.

Rate limiting pada API versi 1.1 ditujukan per-*user* basis atau per *access token*. Rate limits pada API versi 1.1 dibagi kedalam interval 15 menit. Ada 2 buah kelompok GET *request* : 15 panggilan setiap 15 menit, dan 180 panggilan setiap 15 menit. Setiap endpoints membutuhkan authentication. Ini bertujuan agar mencegah kebiasaan yang buruk, dan juga dapat membantu Twitter mengerti lebih jauh bagaimana mengkategorikan aplikasi yang menggunakan API.

2.2.2 Streaming API

Streaming API memberikan latensi yang rendah kepada pengembang untuk mengakses Global Streaming Twitter dari data tweets. Pengimplementasian secara tepat dan benar dapat menghindari overhead. Twitter menawarkan beberapa streaming endpoints, setiap endpoints disesuaikan bergantung pada kasus tertentu.

1. Public Streams
   Public Stream menyediakan data publik yang mengalir melewati Twitter. Jenis ini cocok untuk memantau spesifik *user* atau topik, dan penambangan data.

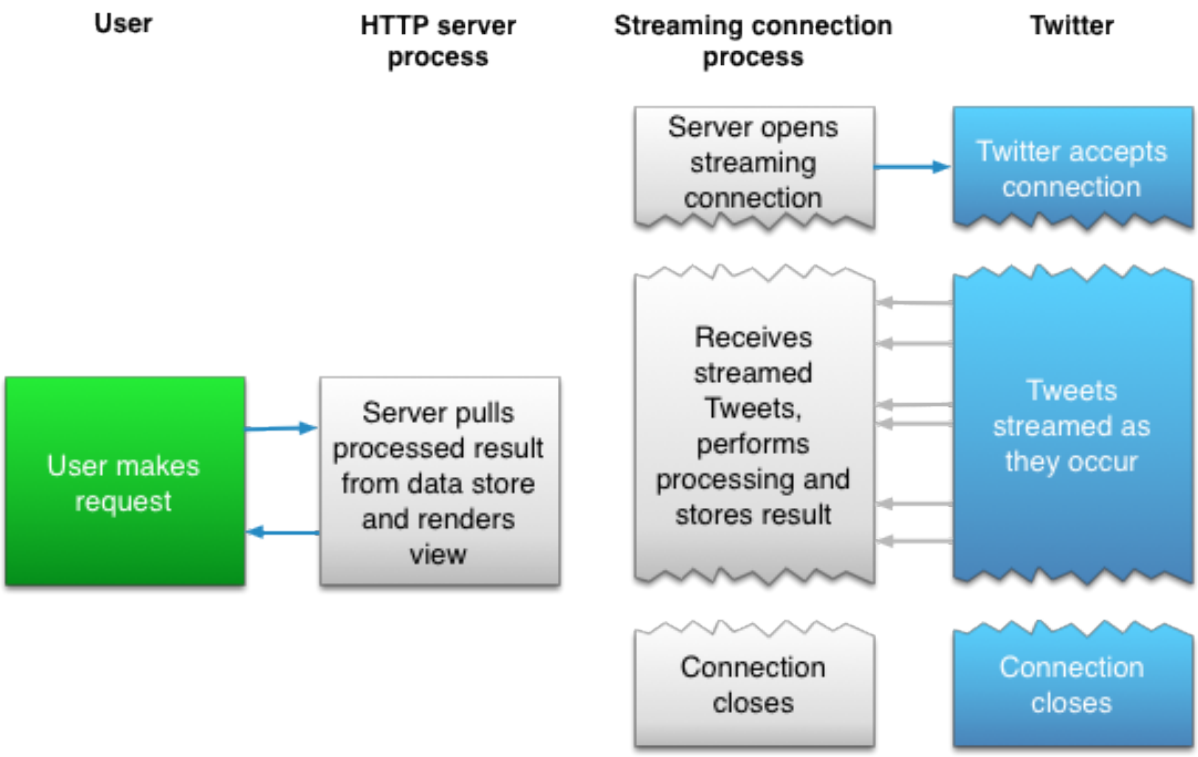
2. *user* Streams
   user Streams menyediakan aliran data dan spesifik kejadian untuk *user* yang terautentikasi. Single-*user* streams, mengandung semua data yang sesuai dengan view yang dimiliki *user* tunggal dari Twitter.

3. Site Streams
   Site Streams adalah versi multi-*user* dari *user* streams. Site stream ditujukan untuk server yang terhubung ke Twitter atas nama banyak *user*. Site streams berguna untuk menerima udates secara real-times dari jumlah *user* yang banyak.

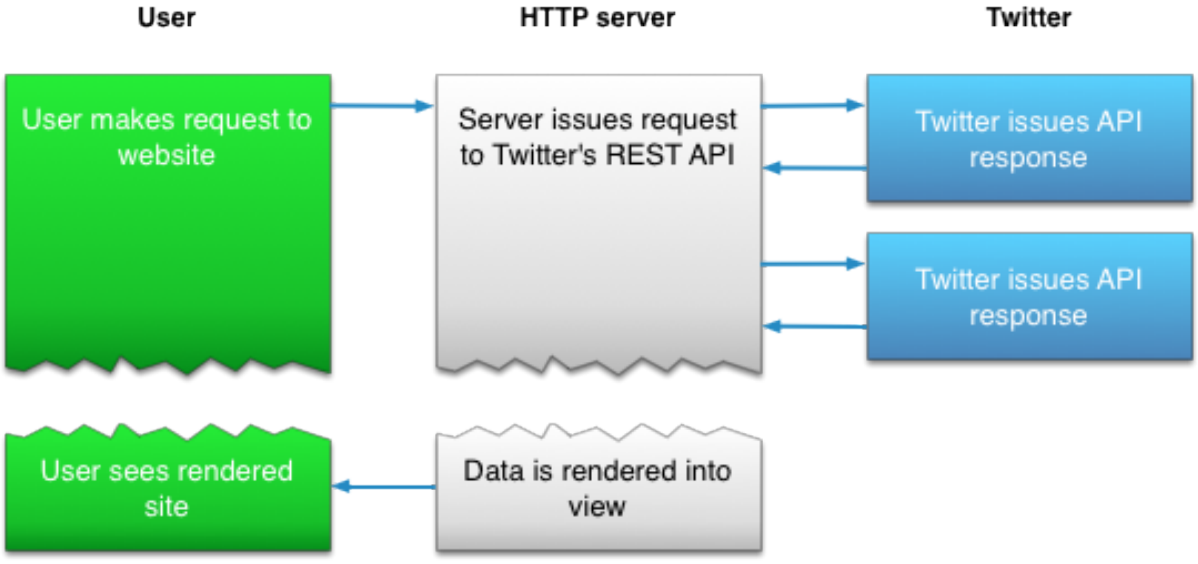
2.2.3 Perbedaaan antara Streaming dan REST

Streaming API dan REST API memiliki beberapa perbedaan. Karena adanya perbedaan ini, pengembang harus memikirkan jenis API mana yang cocok digunakan dalam aplikasinya. Setiap API memiliki karakteristik yang berbeda ada kelebihan yang dimiliki masing-masing API.

Ketika menggunakan Streaming API, aplikasi memerlukan koneksi HTTP yang tetap terbuka antara streaming process dengan server Twitter. Streaming API tidak dapat merespon *user request* secara langsung, namun *user request* harus ditangani oleh HTTP server yang dimiliki oleh aplikasi. Pada gambar dibawah, aplikasi memiliki 2 buah penanganan yang terdiri dari server yang menangani *user request*, dan server yang menangani streaming process.



Dalam menggunakan REST API aplikasi idealnya memiliki 2 buah koneksi HTTP. Koneksi *user* dengan HTTP server dan Twitter server dengan HTTP server. *user request* akan diteruskan oleh HTTP server melalui REST API ke server Twitter. Respon dari server Twitter akan diproses oleh HTTP server dan diteruskan kepada *user*. Gambar dibawah menggambarkan cara kerja dari REST API.



2.3 OAuth

OAuth 2.0 authorization adalah sebuah framework yang memungkinkan sebuah aplikasi pihak ketiga untuk mendapatkan akses terbatas pada layanan HTTP, tanpa memberikan data autentikasi.

Dalam model tradisional autentikasi *client*-server, ketika *client* meminta *request*s sebuah akses terhadap sumber daya yang dilindungi, pemilik dari sumber daya harus memberikan surat kepercayaan kepada *client*. Dalam hal mendukung aplikasi pihak ke-3, pemilik sumber daya harus membagi surat kepercayaanya kepada pihak ke-3. Hal ini menyebabkan beberapa masalah dan batasan :

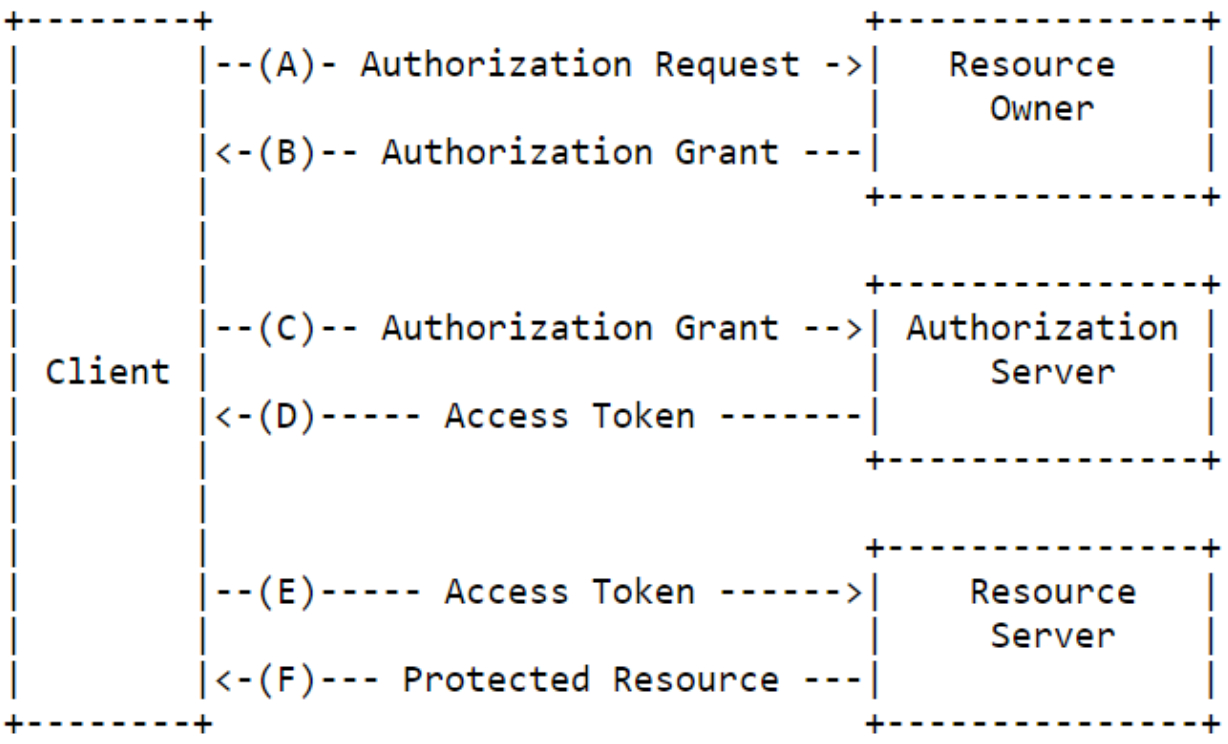
1. Aplikasi pihak ke-3 perlu menyimpan surat kepercayaan pemilik sumber daya untuk penggunaan dimasa depan, biasanya adalah sebuah password dalam teks.
2. Server perlu mendukung autentikasi password, walaupun kelemahan keamanan melekat pada passwords.
3. Aplikasi pihak ke-3 mendapatkan seluruh akses pada sumber daya yang dilindungi. Pemilik sumber daya tidak bisa memberikan batasan akses kepada pihak ke-3.
4. Pemilik sumber daya tidak bisa mencabut akses aplikasi pihak ke-3 tanpa mencabut seluruh akses pihak ke-3, dan harus mengganti password.

OAuth mengatasi masalah ini dengan memperkenalkan sebuah authorization *layer* dan membagi peran *client* untuk mengakses sumber daya. Sebagai ganti menggunakan surat kepercayaan pemilik untuk mengakses sumber daya yang dilindungi. *client* mendapatkan *access token* (sebuah string yang berisi cakupan yang spesifik, waktu akses, dan attribut akses lainya.) *access token* diberikan kepada *client* pihak ketiga oleh authorization server dengan persetujuan pemiliki sumber daya. *client* menggunakan *access token* untuk mengakses sumber daya yang dilindungi oleh resource server.

Dalam OAuth didefinisikan ada 4 peran:

1. Resource Owner
   Sebuah entitas yang dapat memberika akses kepada sumber daya yang dilindungi. Ketika resource owner seorang manusia, ini merujuk pada end-*user*.
2. Resource Server
   Server yang menyediakan sumber daya yang dilindungi, dapat menerima dan merespon *request* sumber daya yang dilindungi menggunakan *access token*s.
3. *client*
   Sebuah aplikasi yang meminta hak akses kepada sumber daya yang dilindungi.
4. Authorization Server
   Server yang memberikan *access token*s kepada *client* ketika proses autentikasi berhasil.

Protocol Flow



1. *client* melakukan meminta authorization grant kepada pemiliki sumber daya. *request* bisa dilakukan secara langsung kepada pemilik sumber daya, atau secara tidak langsung melewati authorization server sebagai penengah.
2. *client* mendapatkan authorization grant.
3. Authorization Grant digunakan untuk meminta *access token* kepada Authorization Server.
4. Authorization Server mengautentikasi apakah authorization grant yang dimiliki *client* valid, dan jika valid, *client* diberikan sebuah *access token*.
5. *client* meminta sumber daya yang dilindungi dari resource Server dan melakukan autentikasi dengan memperlihatkan acces token.
6. Resource server memvalidasi *access token*, dan jika valid, server akan melayani *request*.

2.4 Twitter4J

Twitter4j adalah sebuah Java library untuk Twitter API bersifat *open sourced* dan gratis. Dengan Twitter4j, *user* dapat dengan mudah mengintegrasikan aplikasi Java dengan Twitter service. Twitter4j dapat di unduh pada situs twitter4j.org. Fitur yang dimiliki oleh Twitter4j antar lain :

1. 100% Java murni - bekerja pada semua platform Java versi 5 atau lebih baru.
2. *Zero dependencey* - tidak membutuhkan jars tambahan
3. Mendukung Built-in Oauth

4. Mendukung 100% Twitter API 1.1

5. Dokumentasi dengan Javadoc

Untuk menggunakan Twitter4j disarankan mengikuti persyaratan sistem yang ditetapkan oleh pengembang Twitter4j :

1. OS: Windows atau semua jenis Unix yang mendukung Java.

2. JVM: Java 5 atau lebih baru

Cara menggunakan Twitter4j dengan cara mengunduh jar *file* Twitter4j dan menambahkan Twitter4j-core-4.0.4.jar dalam *classpath* aplikasi kita. Di dalam twitter4j-core-4.0.4.jar terdapat 8 *package*.

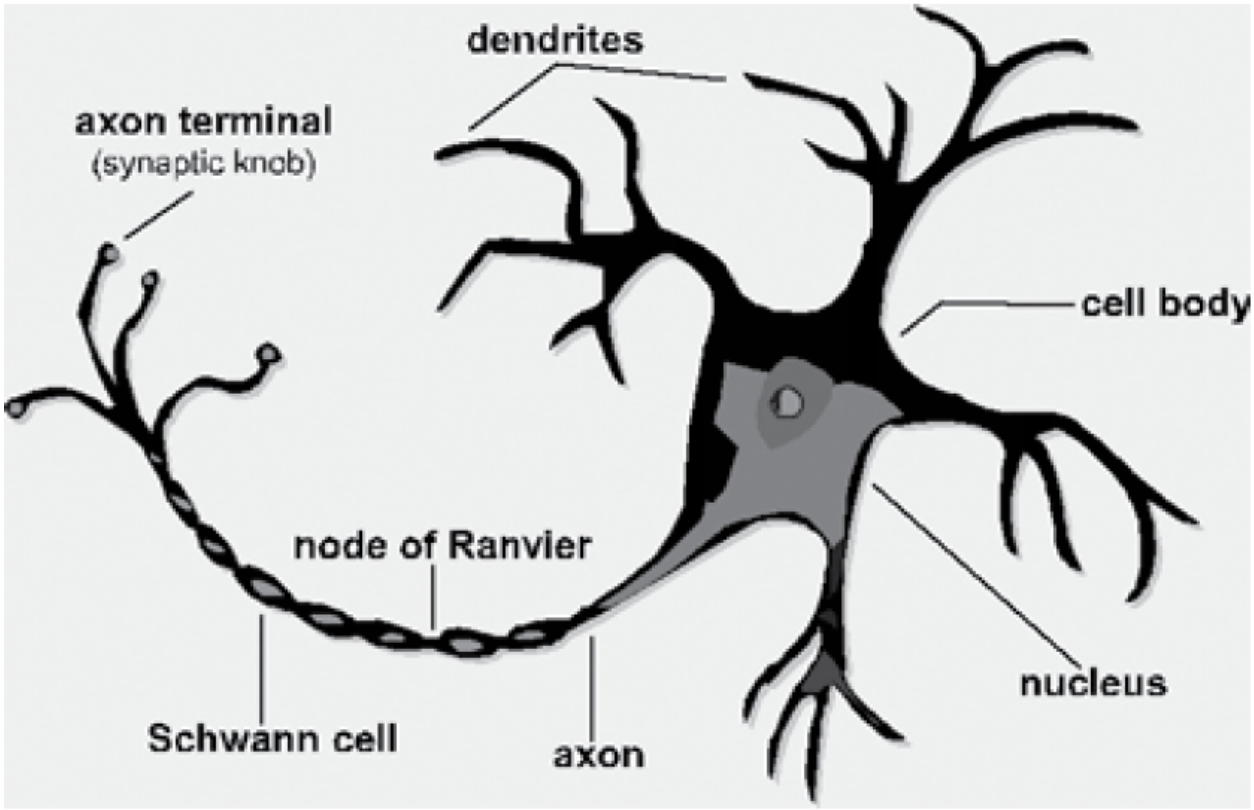
2.5 Jaringan Syaraf Tiruan

Jaringan Saraf Tiruan (JST) adalah paradigma memproses sebuah informasi yang terinspirasi dari cara kerja sistem saraf biologi, seperti otak, memproses informasi. Kunci utama dari paradigma ini adalah struktur dari sistem proses informasi. Sistem ini terdiri dari dari banyak element proses yang saling terhubung (neurons) bekerja secara serempak untuk menyelesaikan suatu masalah. JST dikonfigurasi untuk sebuah penerapan yang spesifik, seperti pengenalan pola atau pengklasifikasian data, melalui proses belajar. Di dalam sistem biologi belajar melibatkan penyesuaian hubungan sinaptic yang berada diantara neuron.

2.5.1 Cara Kerja Jaringan Saraf Biologi

Untuk membangun sebuah komputer yang dapat berfikir seperti manusia, para peneiliti harus memodelkanya seperti otak manusia. Komposisi utama otak manusia adalah sel neuron. JST harus mencoba mensimulasikan sifat-sifat dari sel neuron itu sendiri.

Sebuah sel neuron, seperti pada gambar dibawah, menerima sinyal dari dendrit. Ketika neuron menerima sebuah sinyal, ada kemungkinan neuron tersebut meneruskan sinyal tersebut. Ketika neuron meneruskanya, sinyal tersebut ditransimisikan melewati axon neuron. Sinyal tersebut akan melewati terminal axon, dan ditransmisikan ke neuron lain.



2.5.2 Menyelesaikan masalah dengan JST

JST dapat memproses kumpulan informasi yang akan menghasilkan output. Output JST nantinya akan digunakan untuk membantu pengambilan sebuah keputusan. Faktanya tidak semua masalah cocok diselesaikan dengan JST. Ada beberapa jenis masalah yang kurang cocok diselesaikan oleh JST :

1. Masalah yang dapat diselesaikan dengan program yang mudah untuk di tuliskan kedalam flowcharts.
2. Masalah yang dapat diselesaikan dengan program yang langkahnya dapat didefnisikan dengan terperinci.
3. Masalah yang harus diketahui cara solusinya diturunkan.
4. Algoritma yang digunakan untuk menyelesaikan sebuah masalah tidak berubah-rubah(Statik).

Walau tidak semua masalah dapat diselesaikan oleh JST namun ada masalah yang dapat diselesaikan secara efisien dan efektif oleh JST daripada program tradsional.

2.5.3 Masalah yang dapat diselesaikan dengan JST

Ada banyak hal yang dapat diselesaikan oleh JST. Jenis masalah yang sering diselesaikan oleh JST sebagai berikut :

1. **Classification** adalah proses pengklasifikasian informasi menjadi beberapa jenis kelompok. Contohnya, perusahaan asuransi ingin mengklasifikasikan permohononan asuransi menjadi beberapa kategori risiko yang berbeda, atau sebuah organisasi online ingin membuat sistem email mereka dapat mengklasifikasikan pesan masuk menjadi kelompok spam dan bukan spam. Untuk mencapai hal tersebut JST harus dilatih menggunakan beberapa contoh kelompok data dan instruksi. Setiap kelompok data diklasifikasikan menjadi anggota himpunan tertentu. JST dapat belajar dari contoh kelompok data tersebut. Setelah proses pembelajaran diharapkan JST dapat mengindikasi anggota kelompok data yang baru.

2. **Prediction** adalah penerapan lain yang sering digunakan untuk JST. Dengan memberikan serangkaian input-output data berdasarkan basis waktu, sebuah JST digunakan untuk memprediksi masa depan. Akurasi prediksi akan bergantung pada banyak faktor, seperti quantiti dan relevansi dari input data. JST biasanya diterapkan pada masalah yang melibatkan prediksi pergerakan dalam pasar finansial.

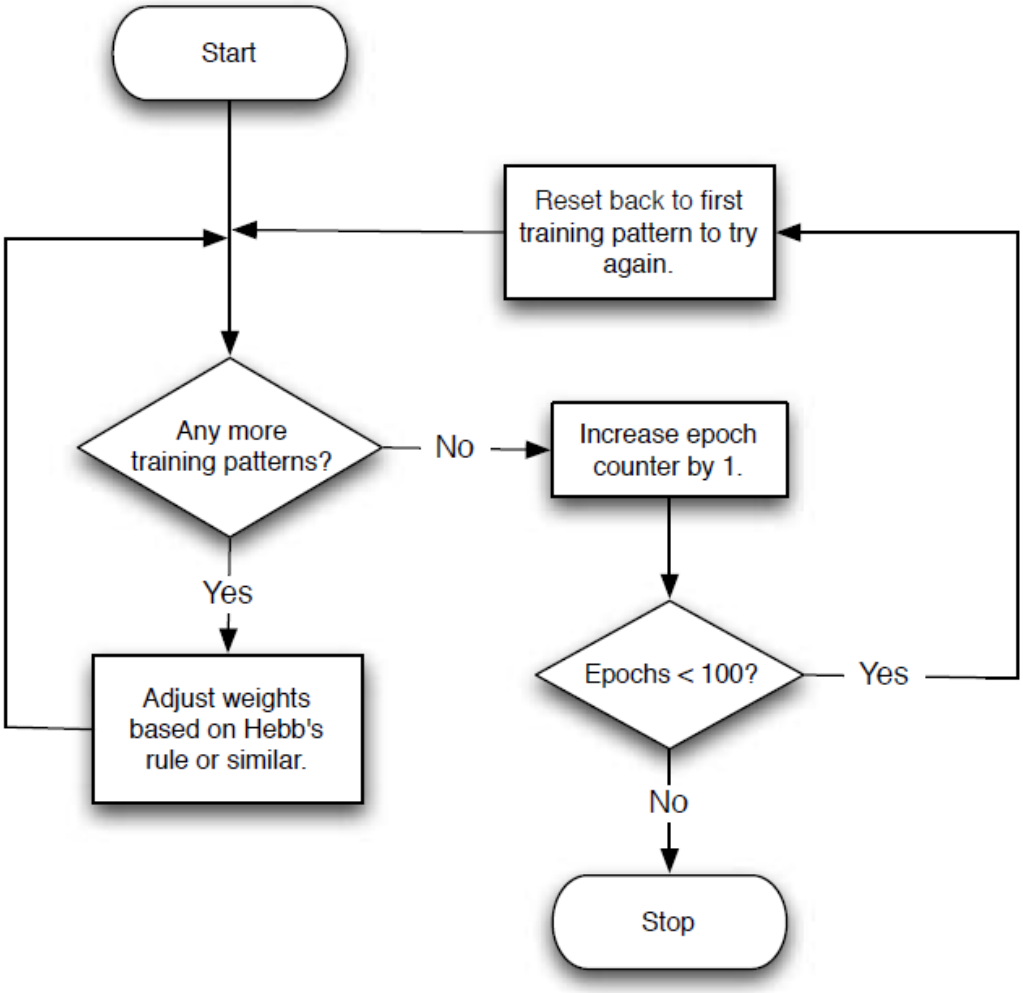
3. **Pattern Recognition** adalah sebuah bentuk pengklasifikasian. Pattern recognition adalah kemampuan untuk mengenali pola. Pola harus bisa dikenali bahkan ketika datanya berubah. Sebagai contoh setiap pengemudi harus dapat dengan tepat mengidentifikasi lampu lalu lintas. Walaupun tidak semua lampu stopan bentuknya sama, pengemudi tetap dapat mengenalinya. Hal ini juga harus dicapai oleh JST agar komputer dapat melakukan pengenalan pola.

4. **Optimization** suatu kemampuan JST untuk mencari solusi yang optimal. Biasanya digunakan ketika suatu masalah memiliki state space yang yang sangat besar. JST mungkin tidak selalu menemukan solusi optimal, namun JST dapat mencari solusi yang dapat diterima. Salah satu masalah optimasi yang paling terkenal ialah Traveling Sales Problem.

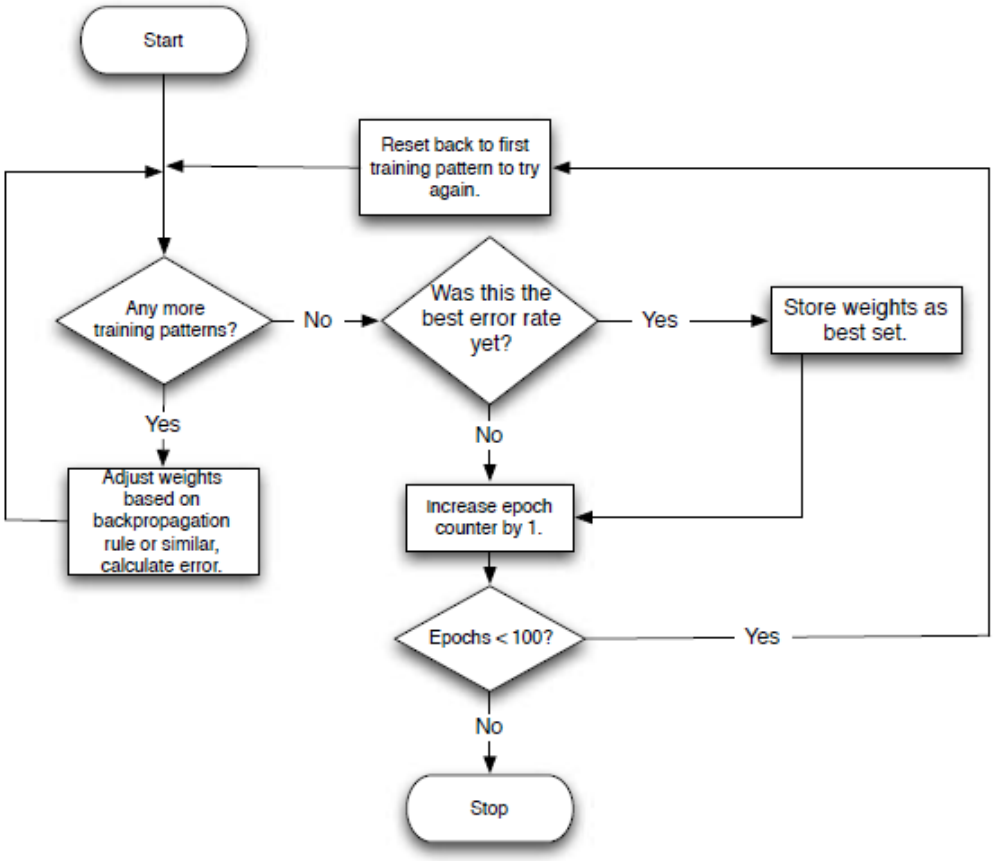
2.5.4 Metode Pembelajaran

Ada banyak cara untuk membuat JST dapat belajar. Setiap algoritma pembelajaran nantinya akan melibatkan perubahan bobot setiap penghubung neuron. Proses pelatihan sangatlah penting bagi JST. Ada dua bentuk dari pelatihan yang dapat digunakan, supervised dan unsupervised. Pelatihan supervised melibatkan JST dengan serangkaian input-output yang diinginkan. Pelatihan unsupervised dibutuhkan juga kumpulan pelatihan, namun tidak disertai output.

1. *Unsupervised Training* adalah salah satu metode pembelajaran yang disediakan input data namun tidak disediakan antisipasi output. Unsupervised training biasanya digunakan untuk melatih JST klasifikasi. Penerapan lainya digunakan untuk data mining. Unsupervised training juga biasa digunakan untuk self-organizing maps (SOM). Unsupervised training dapat diterapkan kedalam banyak situasi.



2. *Supervised Training* adalah metode pembelajaran, yang memiliki sekumpulan pelatihan. Perbedaan utama antara supervised Training dan unsupervised training adalah bahwa supervised training disediakan outputs harapan. Hal ini memungkinkan JST untuk menyesuaikan nilai dari bobot matrix berdasarkan perbedaan antara output yang diharapkan dengan output yang sesunguhnya.



2.5.5 Perhitungan Galat

Perhitungan galat adalah salah satu aspek penting dari setiap JST. Apakah JST itu adalah supervised atau unsupervised, sebuah rata-rata galat harus dihitung. Tujuan dari setiap algoritma pelatihan ialah untuk meminimalisasi rata-rata galat. Disini hanya akan membahasa perhitungan galat untuk supervised training.

Perhitungan Galat dan Supervised Training

Ada dua nilai yang harus dipertimbangkan dalam menentukan rata-rata galat untuk supervised training. Pertama kita harus menghitung galat untuk setiap element pelatihan. Kedua, kita harus menghitung rata-rata galat untuk semua element pelatihan untuk setiap sampel. (Root Mean Square) RMS adalah salah satu metode untuk menghitung rata-rata galat untuk sebuah pelatihan. Metode RMS efektive dalam perhitungan rata-rata galat tanpa memperhatikan apakah hasil yang sebenarnya lebih tinggi atau lebih kecil daripada hasil yang diharapkan. Persamaan ini digunakan untuk menghitung RMS

$$x_{rms} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(actual_i - ideal_i)^2}$$

2.6 Feedforward Neural Network

Feedforward adalah sebuah arsitektur JST yang paling popular dan paling banyak digunakan sebagai model dalam banyak aplikasi. feedforward dikenal dengan banyak nama, salah satunya "multi-layer perceptron." Dalam JST feedforward, setiap layer dari JST mengandung hubungan ke layer berikutnya (contohnya dari input dihubungkan ke layer tersembunyi). Pda feedforward tidak ada hubungan kembali, jadi hubungan antar neuron hanya satu arah. Feedfoward selalu dimulai dari layer input. Jika input terhubung dengan sebuah layer tersembunyi, layer tersembunyi dapat terhubung dengan layer tersembunyi lainya atau dapat langsung terhubung dengan output. Bisa ada banyak jumlah dari layer tersembunyi. Kebanyakan JST biasanya akan memiliki satu buah layer tersembunyi, dan akan sangat jarang JST memiliki lebih dari dua buah layer tersembunyi.

Memilih Struktur Jaringan Ada banyak cara untuk membangun JST feedfowrward. kita harus menentukan berapa jumlah neuron pada layer input dan layer output. Selain itu juga layer tersembunyi juga harus ditentukan. Ada banyak teknik untuk memilih parameter tersebut. Namun untuk menentukan struktur yang optimal pada feedforward dibutuhkan pengalaman dan percobaan.

- **Layer Input**
  Jumlah neuron pada layer input dapat ditentukan bergantung pada data yang kita miliki. Parameter ini biasanya ditentukan secara unik ketika kita mengetahui data pelatihan kita. Secara spesifik, jumlah dari neuron setara dengan banyaknya kolum pada data kita. Biasanya kita dapat menambahkan satu buah titik bias.

- **Layer Output**
  Seperti input layer, setiap JST memiliki satu buah layer output. Untuk jumlah neuron di dalamanya ditentukan pada model apa yang kita gunakan. Apakah JST kita Machine Mode atau Regression Mode. Machine Mode mengembalikan label kelas, sedangkan Regression Mode mengembalikan sebuah nilai. bila JST menggunakan sebuah regressor, maka outputnya akan memiliki 1 neuron. Bila outputnya sebuah pengklasifikasian, maka neuronya akan satu namun bisa lebih.

- **Layer Tersembunyi**
  Dalam layer tersembunyi ada dua buah keputusan yang harus diperhatikan yang pertama berapa jumlah layer tersembunyi yang dibutuhkan, dan yang kedua berapa jumlah neuron pada setiap layer. Pertama kita akan membahas jumlah layer tersembunyi yang akan digunakan pada JST. Masalah yang membutuhkan dua layer tersembunyi sangat jarang ditemukan. JST dengan dua layer tersembunyi dapat di representasikan sebagai fungsi dengan setiap jenis potongan. Tidak ada alasan secara teori untuk menggunakan layer tersembunyi lebih dari

dua. Faktanya pada masalah yang ada pada kehidupa sehari-hari, cukup diselesaikan dengan 1 buah layer tersembunyi. Tabel berikut merupakan kegunaan JST berdasarkan banyaknya layer tersembunyi

Number of Hidden Layer	Result
none	Only capable of representing linear separable functions or decisions.
1	Can approximate any function that contains a continuous mapping from one finite space to another.
2	Can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy.

Untuk menentukan jumlah neuron dalam layer tersembunyi adalah bagian yang sangat penting dalam menentukan keseluruhan arsitektur JST. Walaupun layer ini tidak secara langsung berinteraksi dengan lingkungan luar, layer ini memiliki pengaruh yang sangat besar pada akhir output. Jumlah layer tersembunyi dan jumlah neuronya sangat penting diperhatikan. Jika kita terlalu sedikit menggunakan neuron dalam layer tersembunyi ini akan mengakibatkan sesuatu yang disebut underfitting. Namun bila terlalu banyak menggunakan neuron pada layer tersembunyi ini akan mengakibatkan overfitting. Overfitting terjadi ketika JST terlalu banyak memiliki informasi yang diproses daripada jumlah batas dari informasi yang terkandung dalam sejumlah pelatihan. Ada beberapa tips untuk menentukan jumlah neuron: Pertama jumlah neuron harus berada diantara jumlah input dan output. Kedua jumlah neuron seharusnya 2/3 ukuran dari layer input, ditambah dengan layer output. Ketiga jumlah neuron seharusnya kurang dari dua kali dari layer input.

- **Fungsi Aktivasi** Kebanyakakn JST mengeluarkan output dari layernya menggunakan fungsi aktivasi. Fungsi aktivasi ini menskalakan output dari JST dalam jangkauan tertentu. Fungsi aktivasi dapat kita buat sendiri namun umumnya menggunakan fungsi yang sudah sering digunakan. Ada beberapa jenis fungsi aktivasi yang sering digunakan diantara sebagai berikut: **Sigmoid** adalah fungsi aktivasi yang menggunakan fungsi sigmoid untuk menentukan aktivasi. fungsi sigmoid di definisikan sebagai berikut:

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

  Kurva dibawah menggambarkan fungsi sigmoid

  Hal penting yang perlu diperhatikan dalam menggunakan fungsi sigmoid. Fungsi ini hanya akan menghasilkan nilai positif. **Hyperbolic Tangent** berbeda dengan sigmoid bila pada sigmoid selalu menghasilkan nilai positif. Dengan menggunakan fungsi Hyperbolic Tangent akan menghasilkan nilai negatif. Bila kita menginginkan output dapat bernilai negatif dan positif kita dapat menggunakan fungsi yang di definisikan sebagai berikut:

$f(x) = \frac{e^{2x}-1}{e^{2x}+1}$

Kurva dibawah menggambarkan fungsi Hyperbolic Tangent

DAFTAR REFERENSI

[1] Twitter, "Faq twitter," September 2015.

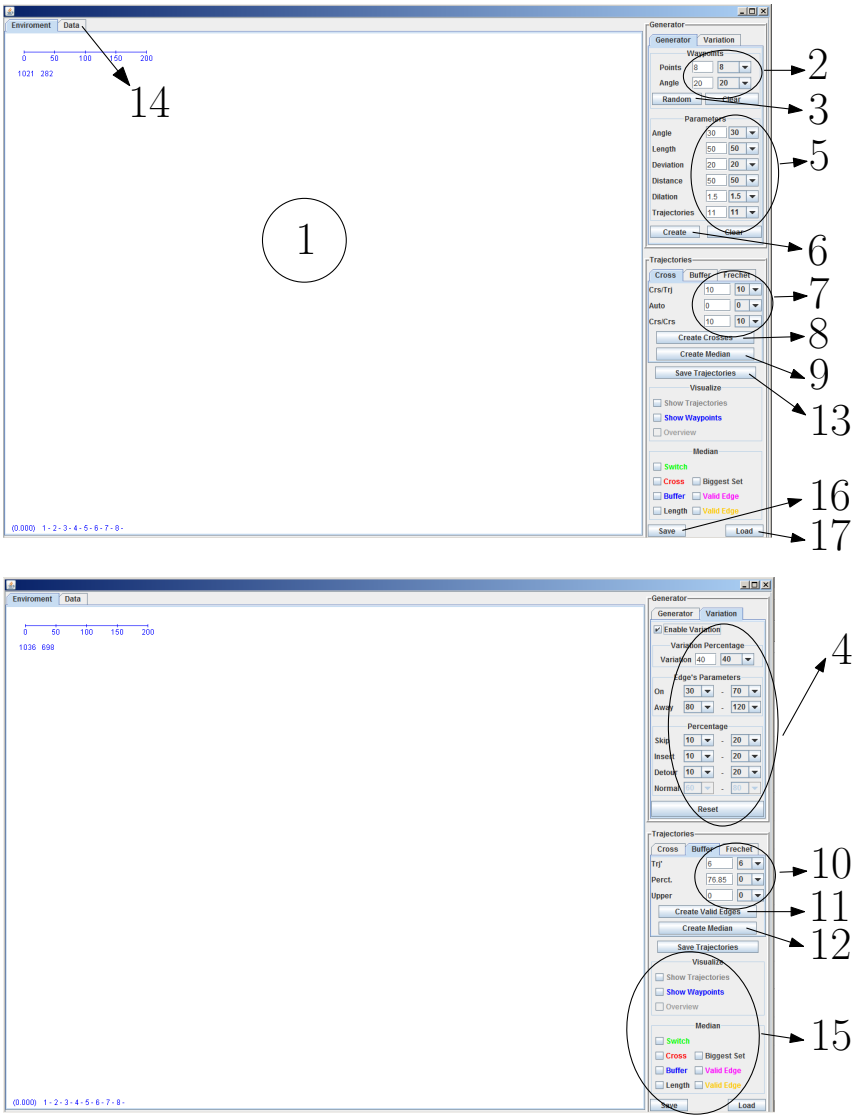
[2] Twitter, "Faq twitter," September 2015.

[3] C. I. Noor Aspasia Hasibuan, "Jumlah pengguna twitter di indonesia akhirnya terungkap," March 2015.

LAMPIRAN A

THE PROGRAM

The interface of the program is shown in Figure A.1:



Gambar A.1: Interface of the program

Step by step to compute the median trajectory using the program:

1. Create several waypoints. Click anywhere in the "Environment" area(1) or create them automatically by setting the parameters for waypoint(2) or clicking the button "Random"(3).

2. The "Variation" tab could be used to create variations by providing values needed to make them(4).

3. Create a set of trajectories by setting all parameters(5) and clicking the button "Create"(6).

4. Compute the median using the homotopic algorithm:

   - Define all parameters needed for the homotopic algorithm(7).
   - Create crosses by clicking the "Create Crosses" button(8).
   - Compute the median by clicking the "Compute Median" button(9).

5. Compute the median using the switching method and the buffer algorithm:

   - Define all parameters needed for the buffer algorithm(10).
   - Create valid edges by clicking the "Create Valid Edges"button(11).
   - Compute the median by clicking the "Compute Median"button(12).

6. Save the resulting median by clicking the "Save Trajectories" button(13). The result is saved in the computer memory and can be seen in "Data" tab(14)

7. The set of trajectories and its median trajectories will appear in the "Environment" area(1) and the user can change what to display by selecting various choices in "Visualize" and "Median" area(15).

8. To save all data to the disk, click the "Save"(16) button. A file dialog menu will appear.

9. To load data from the disk, click the "Load"(17) button.

LAMPIRAN B

THE SOURCE CODE

Listing B.1: MyFurSet.java

```
1
2  import java.util.ArrayList;
3  import java.util.Collections;
4  import java.util.HashSet;
5
6  /**
7   *
8   * @author Lionov
9   */
10
11 //class for set of vertices close to furthest edge
12 public class MyFurSet {
13     protected int id;                                    //id of the set
14     protected MyEdge FurthestEdge;                       //the furthest edge
15     protected HashSet<MyVertex> set;                     //set of vertices close to furthest edge
16     protected ArrayList<ArrayList<Integer>> ordered;     //list of all vertices in the set for each
           trajectory
17     protected ArrayList<Integer> closeID;                //store the ID of all vertices
18     protected ArrayList<Double> closeDist;               //store the distance of all vertices
19     protected int totaltrj;                              //total trajectories in the set
20
21     /**
22      * Constructor
23      * @param id : id of the set
24      * @param totaltrj : total number of trajectories in the set
25      * @param FurthestEdge : the furthest edge
26      */
27     public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
28         this.id = id;
29         this.totaltrj = totaltrj;
30         this.FurthestEdge = FurthestEdge;
31         set = new HashSet<MyVertex>();
32         ordered = new ArrayList<ArrayList<Integer>>();
33         for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
34         closeID = new ArrayList<Integer>(totaltrj);
35         closeDist = new ArrayList<Double>(totaltrj);
36         for (int i = 0;i <totaltrj;i++) {
37             closeID.add(-1);
38             closeDist.add(Double.MAX_VALUE);
39         }
40     }
41
42     /**
43      * set a vertex into the set
44      * @param v : vertex to be added to the set
45      */
46     public void add(MyVertex v) {
47         set.add(v);
48     }
49
50     /**
51      * check whether vertex v is a member of the set
52      * @param v : vertex to be checked
53      * @return true if v is a member of the set, false otherwise
54      */
55     public boolean contains(MyVertex v) {
56         return this.set.contains(v);
57     }
58 }
```