BAB III

LANDASAN TEORI

3.1 Gaya Belajar

Gaya belajar dapat didefinisikan sebagai *attitude* dan *behaviour* yang menentukan sebuah cara yang lebih baik dalam belajar secara individual (Honey & Mumford, 1992). Gaya belajar adalah salah satu faktor yang berpengaruh pada taraf/tingkatan belajar. Dalam suatu lingkungan belajar, mahasiswa memiliki kebutuhan dan karakteristik yang berbeda secara individual (Felder, 2005). Pada penelitian ini dipilih model gaya belajar VARK untuk menentukan jenis gaya belajar mahasiswa. Klasifikasi gaya belajar berdasarkan model gaya belajar VARK yaitu *visual, aural, read/write, kinaesthetic* (Fleming, 1995). Gaya belajar VARK memiliki kuisioner dalam menentukan jenis gaya belajar seseorang. Kuisioner tersebut berisi pertanyaan-pertanyaan yang merepresentasikan cara belajar yang lebih baik. Setiap pertanyaan memiliki 4 pilihan jawaban yang menunjukkan *preference* seseorang dalam melakukan sesuatu. Berdasarkan hasil dari penggisian kuisioner tersebut, kemudian dapat diketahui jenis gaya belajar yang dimiliki seseorang. Peneliti menggunakan kuisioner ini untuk menentukan jenis gaya belajar dari mahasiswa (Zapalska & Brozik, 2006).

3.2 Analisis Sentimen

Analisis Sentimen dapat dikategorikan kedalam tiga *task*, yaitu *informative text detection, information extraction* dan *sentiment interestingness*

classification (*emotional, polarity indentification*). *Sentiment classification* (negatif atau positif) digunakan untuk memprediksi *sentiment polarity* berdasarkan data sentimen dari pengguna (Pan, et al., 2010).

Sentiment analysis adalah metode untuk menganalisis sebagian data untuk mengetahui emosi manusia. Emosi merupakan hal yang mendasar untuk *human experience, influencing cognition, perception*, kegiatan sehari-hari seperti *learning*, *communication*, dan bahkan *rational decision-making* (Goleman, 1999). Menurut Plutchik (2001) ada 8 klasifikasi emosi yaitu *joy, acceptance, fear, surprise, sadness, disgust, anger* dan *anticipation*. Dalam penelitian yang dilakukan oleh Nakamura (2003) ada 10 sentimen yaitu *Joy, Anger, Sorrow, Fear, Shame, Liking, Dislike, Excitement, Relief, Surprise*. Saat ini *textual sentiment analysis* telah banyak digunakan, penggunaannya tidak sebatas dalam area penelitian ilmiah tetapi juga untuk kebutuhan *business marketing* dan teknologi (Chintala, 2012). Menurut Go (2009), *sentiment analysis* adalah sebuah area penelitian yang menonjol dan berkembang aktif, hal itu dipengaruhi oleh pertumbuhan teknologi media sosial yang cepat dan peluang untuk mengakses opini yang berharga dari sejumlah orang pada berbagai jenis bisnis, isu dunia dan isu sosial.

3.3 Twitter

Twitter adalah salah satu layanan *microblogging* yang dirilis secara resmi pada 13 Juli 2006 (Mostafa, 2013). Twitter dianggap sebagai *microblog* karena aktifitas utamanya adalah mem-*posting* sesuatu yang pendek (disebut *tweet*) melalui *web* atau *mobile*. Panjang maksimal dari *tweet* adalah 140 karakter,

kira-kira seperti panjang karakter dari judul koran. Twitter menjadi sumber yang hampir tak terbatas yang digunakan pada *text classification*. Menurut Go (2009), terdapat banyak karakteristik pada *tweets* twitter. Pesan pada twitter memiliki banyak *attribute* yang unik, yang membedakan dari media sosial lainnya :

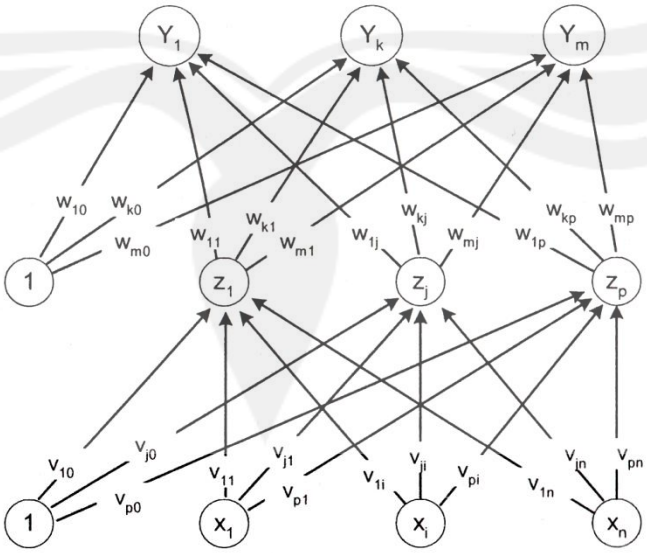
1. Twitter memiliki maksimal panjang karakter yaitu 140 karakter.
2. Twitter menyediakan data yang bisa diakses secara bebas dengan menggunakan Twitter API, mempermudah saat proses pengumpulan *tweets* dalam jumlah yang sangat banyak.
3. Model bahasa – pengguna twitter mem-*posting* pesan melalui banyak media berbeda, termasuk dari telepon seluler. Frekuensi dari salah ejaan, bahasa gaul dan singkatan lebih tinggi daripada media sosial lainnya.
4. Pengguna twitter mengirim pesan singkat tentang berbagai topik yang disesuaikan dengan topik tertentu dan itu berlaku secara global.

Selama beberapa tahun terakhir, twitter menjadi sangat populer. Jumlah pengguna twitter telah naik menjadi 190 juta dan jumlah *tweet* yang dipublikasikan di twitter setiap hari adalah lebih dari 65 juta (Ravichandran & Kulanthaivel, 2014).

3.4 Backpropagation

Backpropagation adalah salah satu jenis *neural network* yang memiliki struktur jaringan *layered feed-forward* yang terdiri dari banyak neuron dengan kemampuan *nonlinear mapping*. Backpropagation juga merupakan salah satu metode yang sederhana dan umum digunakan untuk *supervised training* pada

multilayer neural networks. Backpropagation bekerja dengan memperkirakan hubungan *nonlinear* antara *input* dan *output* dengan menyesuaikan nilai bobotnya sesuai dengannilai minimum dari *error function* sehingga memungkinkan jaringan untuk berpusat pada keadaan stabil dan memberikan *output* yang sesuai ketika menerima masukan yang tidak termasuk ke dalam pola data *training*. Secara umum, jaringan backpropagation memiliki dua tahap, *training* dan *testing.* Pada tahap *training*, *network* melakukan proses dengan pola input dan klasifikasi yang benar. Pada tahap *testing*, input yang digunakan adalah input baru yang tidak termasuk ke dalam pola input *training*, sehingga akan diketahui hasil klasifikasi berdasarkan tahap *training*. Gambar 3.1 menunjukkan arsitektur dari backpropagation *neural network*, terdapat bagian *input layer*, *hidden layer*, dan *output layer*. Pada jaringan backpropagation dapat memiliki lebih dari satu *hidden layer* (Dang, et al., 2010)(G.Vinodhini & RM.Chandrasekaran, 2013).



Gambar 3.1 Arsitektur jaringan Backpropagation

Arsitektur backpropagation dengan n buah masukan (ditambah sebuah bias), *layer* tersembunyi yang terdiri dari p unit (ditambah sebuah bias) dan *layer* ini bisa lebih dari 1 *layer*, serta m buah unit keluaran.v_{ji}adalah bobot garis dari unit masukan $x_i$ke unit layer tersembunyi z_j (v_{j0}adalah bobot garis yang menghubungkan bias di unit masukan ke unit layar tersembunyi z_j). w_{kj} adalah bobot dari unit *layer* tersembunyi z_j ke unit keluaran y_k (w_{k0}adalah bobot dari bias di layar tersembunyi ke unit keluaran z_k).Pelatihan backpropagation meliputi tiga fase (Hermawan, 2006)(Sutojo, et al., 2010):

a. Fase I : Propagasi maju

   Selama propagasi maju, sinyal masukan (=x_i) dipropagasikan ke *layer* tersembunyi menggunakan fungsi aktivasi yang ditentukan. Keluaran dari setiap unit layar tersembunyi (=z_j) selanjutnya dipropagasikan maju lagi ke *layer* tersembunyi di atasnya menggunakan fungsi aktivasi yang ditentukan. Demikian seterusnya hingga menghasilkan keluaran jaringan (=y_k).Berikutnya, keluaran jaringan (=y_k) dibandingkan dengan target yang harus dicapai (=t_k). Selisih t_k-y_k adalah kesalahan yang terjadi. Jika kesalahan ini lebih kecil dari batas toleransi yang ditentukan, maka iterasi dihentikan. Akan tetapi apabila kesalahan masih lebih besar dari batas toleransinya, maka bobot setiap garis dalam jaringan akan diubah untuk mengurangi kesalahan yang terjadi.

b. Fase II : Propagasi mundur

   Berdasarkan kesalahan t_k - y_k, dihitung faktor δ_k (k = 1,2 , ... , m) yang dipakai untuk mendistribusikan kesalahan di unit y_k ke semua unit

tersembunyi yang terhubung langsung dengan y_k. δ_k juga dipakai untuk mengubah bobot garis yang berhubungan langsung dengan unit keluaran.Dengan cara yang sama, dihitung faktor δ_j di setiap unit di *layer* tersembunyi sebagai dasar perubahan bobot semua garis yang berasal dari unit tersembunyi di *layer* di bawahnya. Demikian seterusnya hingga semua faktor δ di unit tersembunyi yang berhubungan langsung dengan unit masukan dihitung.

c. Fase III : Perubahan bobot

Setelah semua faktor δ dihitung, bobot semua garis diubah bersamaan. Perubahan bobot suatu garis didasarkan atas faktor δ *neuron* di *layer* atasnya. Sebagai contoh, perubahan bobot garis yang menuju ke *layer* keluaran didasarkan atas δ_k yang ada di unit keluaran.

Ketiga fase tersebut diulang terus sampai kondisi penghentian terpenuhi. Kondisi penghentian yang sering dipakai adalah jumlah iterasi atau kesalahan. Iterasi akan dihentikan jika jumlah iterasi yang dilakukan sudah melebihi jumlah maksimum iterasi yang ditetapkan, atau jika kesalahan yang terjadi sudah lebih kecil dari batas toleransi yang diijinkan.

Dibawah ini adalah langkah dari algoritma backpropagation:

1. Inisialisasi bobot (bobot awal dengan nilai *random* yang kecil).
2. Kerjakan langkah berikut selama kondisi berhenti bernilai salah.
3. Untuk setiap pasangan elemen yang akan dilakukan pembelajaran, kerjakan:

Feedforward

a. Setiap unit *input* (X_i, i=1,2,3,...,n) menerima sinyal x_i dan meneruskan ke semua unit pada lapisan yang ada di atasnya (lapisan tersembunyi).

b. Setiap unit pada suatu lapisan tersembunyi (Z_j, j=1,2,3,...,p) menjumlahkan sinyal*input* terbobot :

$$z\_in_j = b1_j + \sum_{i=1}^{n} x_i v_{ij} \quad \text{.....(1)}$$

menggunakan fungsi aktivasi untuk menghitung sinyal *output* :

$z_j = f(z\_in_j)$ .....(2)

mengirim sinyal ke semua unit di lapisan atasnya (unit-unit *output*).

c. Setiap unit *output* (Y_k, k=1,2,3,...,m) menjumlahkan sinyal-sinyal *input* terbobot.

$$y\_in_k = b2_k + \sum_{i=1}^{n} z_i w_{jk} \quad \text{.....(3)}$$

menggunakan fungsi aktivasi untuk menghitung sinyal *output* :

$y_k = f(y\_in_k)$ .....(4)

mengirim sinyal ke semua unit di lapisan atasnya (unit-unit *output*).

Catatan :Langkah (b) dilakukan sebanyak jumlah lapisan tersembunyi.

Backpropagation

d. Setiap unit *output* (Y_k, k=1,2,3,...,m) menerima target pola yang berhubungan dengan pola *input* pembelajaran, hitung informasi errornya :

$\delta 2_k = (t_k - y_k)\, f'(y\_in_k)$ .....(5)

$\varphi 2_{jk} = \delta_k z_j$ .....(6)

$\beta 2_k = \delta_k$ ..... (7)

hitung koreksi bobot (digunakan untuk memperbaiki nilai w_{jk}) :

$\Delta w_{jk} = \alpha \varphi 2_{jk}$ ..... (8)

hitung juga koreksi bias (digunakan untuk memperbaiki nilai $b2_k$):

$\Delta b2_k = \alpha \beta 2_k$ ..... (9)

Catatan : Langkah (d) dilakukan sebanyak jumlah lapisan tersembunyi, menghitung informasi error dari suatu lapisan tersembunyi ke lapisan tersembunyi sebelumnya.

e. Setiap unit tersembunyi (Z_j, j=1,2,3,...,p) menjumlahkan delta *input*:

$$\delta\_in_j = \sum_{k=1}^{m} \delta 2_k w_{jk} \quad \text{.....(10)}$$

nilai δ_in_j dengan turunan dari fungsi aktivasinya untuk menghitung informasi *error* :

$\delta 1_j = \delta\_in_j \, f'(z\_in_j)$ .....(11)

$\varphi 1_{ij} = \delta 1_j x_j$ ..... (12)

$\beta 1_j = \delta 1_j$ .....(13)

hitung koreksi bobot (digunakan untuk memperbaiki nilai v_{ij}) :

$\Delta v_{ij} = \alpha \varphi 1_{ij}$ ..... (14)

hitung koreksi bias (digunakan untuk memperbaiki nilai $b1_j$) :

$\Delta b1_j = \alpha \beta 1_j$ ..... (15)

f. Setiap unit *output* (Y_k, k=1,2,3,...,m) memperbaiki bias dan bobotnya (j=0,1,2,...,p) :

$w_{jk}(\text{baru}) = w_{jk}(\text{lama}) + \Delta w_{jk}$ ..... (16)

$b2_k(baru) = b2_k(lama) + \Delta b2_k$ ..... (17)

Setiap unit tersembunyi (Z_j, j=1,2,3,...,p) memperbaiki bias dan bobotnya (i=0,1,2,...,n) :

$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij}$ ..... (18)

$b1_j(baru) = b1_j(lama) + \Delta b1_j$ ..... (19)

4. Tes kondisi berhenti.

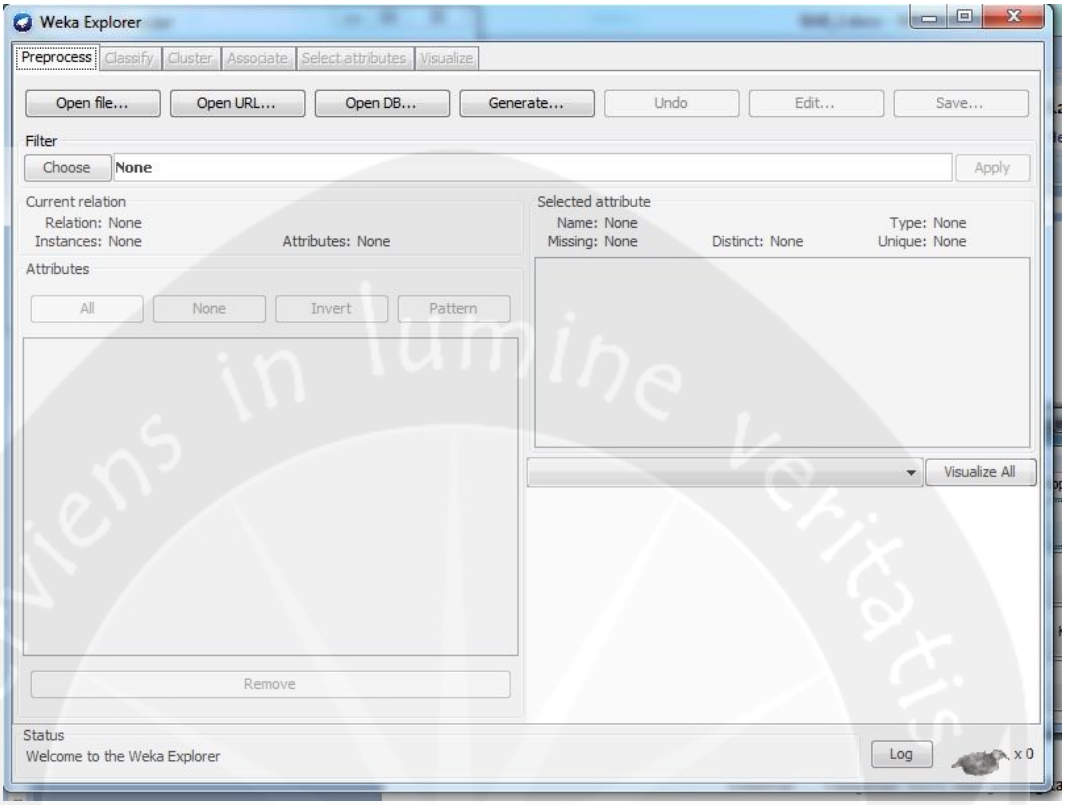
Masalah utama yang dihadapi dalam backpropagation adalah lamanya iterasi yang harus dilakukan. Backpropagation tidak dapat memberikan kepastian tentang berapa *epoch* yang harus dilalui untuk mencapai kondisi yang diinginkan, sehingga harus dilakukan eksperimen terhadap parameter jaringan untuk mendapatkan jumlah iterasi yang tepat. Beberapa cara untuk mengoptimalkan backpropagation adalah inisialisasi nilai bobot awal secara*random* dengan nilai antara -0.5 sampai 0.5 (atau -1 sampai 1, atau interval yang lainnya), penambahan jumlah layer tersembunyi, menentukan lama iterasi.

3.5 WEKA

WEKA adalah singkatan dari Waikato Environment for Knowledge Analysis. WEKA adalah salah satu perangkat lunak *machine learning* yang populer yang dibangun dalam bahasa JAVA, dikembangkan di University of Waikato. WEKA adalah perangkat lunak tidak berbayar yang tersedia dibawah GNU General Public License. Gambar 3.2 adalah tampilan halaman depan dari WEKA.

Gambar 3.2 Tampilan GUI dari perangkat lunak WEKA

WEKA *workbench* ditunjukkan pada Gambar 3.3 memiliki kumpulan dari *visualization tools* dan algoritma untuk data analisis dan permodelan prediktif, disertai juga dengan antarmuka grafik untuk memudahkan penggunaan secara fungsionalitas. Versi asli non-Java dari WEKA adalah sebuah TCL/TK *front-end* (kebanyakan adalah sebagai *third-party*) untuk algoritma permodelan yang diimplementasikan dalam bahasa pemrograman lain, ditambah *preprocessing data utilities* dalam bahasa C dan sebuah sistem *Makefile-based* untuk menjalankan eksperimen *machine learning*. Versi asli dari WEKA sebenarnya dirancang sebagai perangkat untuk menganalisis data dari area *agricultural*, tetapi lebih lanjut sepenuhnya dalam versi berbasis Java (WEKA 3), pengembangan dimulai pada tahun 1997. Sekarang digunakan pada banyak aplikasi yang berbeda, khususnya untuk tujuan pendidikan dan penelitian.



Gambar 3.3 Tampilan GUI dari WEKA *workbench*

Kekuatan utama dari WEKA adalah tersedia secara bebas dibawah GNU General Public License, sangat *portable* karena menggunakan bahasa pemrograman Java yang dapat berjalan pada hampir semua platform moderen, memiliki banyak *data preprocessing* dan teknik permodelan, mudah digunakan oleh pemula dengan antarmuka grafik. WEKA mendukung beberapa *taskdata mining* standar, khususnya seperti *data preprocessing, clustering, classification, regression, visualization* dan *feature selection*. Semua teknik pada WEKA didasarkan pada asumsi bahwa data yang tersedia sebagai *single flat file* atau relasi, yang mana setiap *point* data dideskripsikan oleh sejumlah *attribute* tetap (*numeric* atau nominal, tetapi juga mendukung beberapa jenis *attribute* lain juga). WEKA menyediakan akses ke basis data SQL

menggunakan Java Database Connectivity dan dapat memproses hasil dari *query*basis data. WEKA tidak mendukung *multi-relational data mining*. Area penting yang tidak tercakup oleh algoritma yang termasuk di dalam proses distribusi WEKA adalah urutan permodelan (Witten & Frank, 2005)(Bouckaer, et al., 2015).