PENGENALAN FORMULA MATEMATIKA DARI POLA TULISAN TANGAN

Yuliana Melita Pranoto, Edwin Pramana, dan Renato Budiman
Teknik Informatika
Sekolah Tinggi Teknik Surabaya
ymp@stts.edu, epramana@gmail.com, dan renato.budiman07@gmail.com

ABSTRAK

Matematika bukan menjadi hal yang asing, bahkan telah menjadi kebutuhan utama. Formula matematika sederhana pun dipakai dalam kehidupan sehari-hari. Dalam perkembangannya semakin banyak perangkat *mobile* maupun *desktop* yang berbasis *touchscreen*. Hal inilah yang mendasari penelitian tentang Pengenalan Formula Matematika dari Pola Tulisan Tangan.

Seiring dengan pesatnya kemajuan teknologi dalam bidang Computer Vision, algoritma Hidden Markov Model (HMM) dikenal baik dalam pengenalan pola. Diawali dengan tahapan preprocessing, yang memanfaatkan fitur-fitur yaitu pen up/down, normalized y coordinate, normalized distance, vicinity slope, dan curvature. Fitur-fitur yang ada diubah ke bentuk codeword berdasarkan codebook yang dibuat dengan menggunakan data training dengan Vector Quantization. Kumpulan codeword tersebut kemudian dibandingkan dengan model-model HMM yang telah dibuat dalam data training.

Akurasi tertinggi yang dihasilkan dari pemanfaatan fitur dan codeword adalah 72.92%, yaitu dengan kombinasi fitur 1, 3, 4, dan 5 fitur, serta 60 buah codeword. Sedangkan waktu rata-rata yang dibutuhkan untuk mengenali formula matematika dari pola tulisan tangan dengan HMM adalah 283.69 ms.

Kata kunci: *Hidden Markov Model, HMM, Pola Tulisan Tangan, Formula Matematika*

ABSTRACT

Matemathic is extremely common, in fact it has become a major subject now. Even a simple mathematic equation is used in our daily life. Currently, a lot of mobile devices and desktop are implementing touchscreen. It is the very foundation of this research, which is about Mathematic Equation Recognition based on Hand-Writing Pattern.

Along with the advancement of technology in Computer Vision subject, Hidden Markov Model (HMM) algorithm is well known in recognizing pattern. Starting with preprocessing phase (utilizing features such as pen up/down, normalized Y coordinate, normalized distance, vicinity slope and curvature). Available features are then transformed into codewords based on predefined-codewords made by using Vector Quantization Data Training. These codewords collection are then compared with HMM models which has been prepared in Data Training.

Highest accuracy achived from utilizing features and codeword is 72.92%, combining features 1,3, 4 and 5, along with 60 codewords. Mean time required to recognize a mathematic equation from handwriting pattern with HMM is 283.69 ms.

Keywords: Hidden Markov Model, HMM, Hand-Writing Pattern, Mathematic Equation

I. PENDAHULUAN

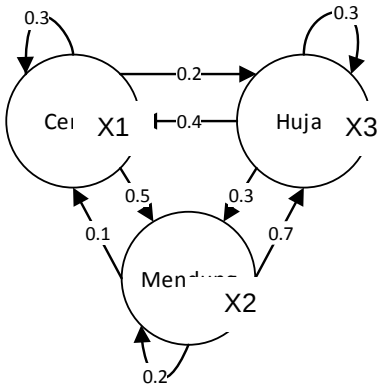
Matematika bukan menjadi hal yang asing, bahkan telah menjadi kebutuhan utama. Kegiatan sehari-hari pun tidak lepas dari formula matematika sederhana. Bukan rahasia lagi bila matematika kurang diminati oleh kebanyakan orang, padahal matematika telah dikenalkan dan dipelajari semenjak di bangku TK. Untuk alasan inilah beberapa metode terus dikembangkan guna mempermudah dalam mempelajari matematika dan menarik minat pelajar agar menyukai matematika.

Kalkulator sebagai alat bantu dalam menghitung formula matematika, telah menjadi kebutuhan utama dalam perangkat *mobile* maupun *desktop*. Kemampuan aplikasi kalkulator yang ada saat ini semakin kompleks, sehingga bukan sekedar perhitungan matematika dasar saja melainkan juga perhitungan yang lebih rumit seperti misalnya kalkulator scientific.

Dalam perkembangannya semakin banyak perangkat *mobile* maupun *desktop* yang berbasis *touchscreen* dan masih jarang aplikasi kalkulator yang dapat menghitung ekspresi matematika melalui tulisan tangan pengguna. Umumnya, aplikasi kalkulator berbasis *mobile* menggunakan tombol untuk menginputkan ekspresi matematika. Hal-hal inilah yang melatarbelakangi peneliti untuk mencoba mengembangkan sebuah aplikasi yang dapat mengenali formula matematika dari pola tulisan tangannya sendiri.

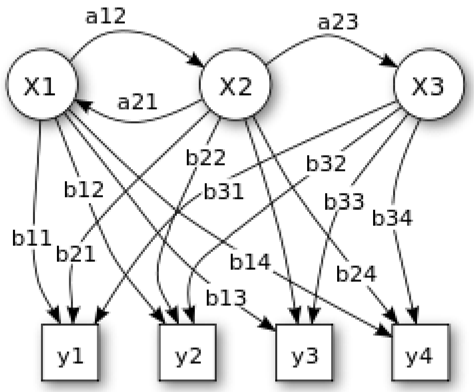
II. HIDDEN MARKOV MODEL

Hidden Markov Model (HMM) merupakan sistem pemodelan yang banyak digunakan untuk pengenalan pola, seperti pengenalan suara, tulisan tangan, POS Tagging, gesture. Dalam penelitian ini HMM digunakan untuk proses klasifikasi. HMM merupakan pengembangan dari Markov Model atau lebih dikenal dengan proses Markov dengan state tersembunyi.



Gambar 1. Markov Model

Pada Markov Model atau Markov Chain (gambar 1), setiap state terlihat jelas dan hanya memiliki transisi state untuk parameternya. Sedangkan pada HMM, state tidak secara langsung terlihat, namun output yang bergantung pada state terlihat. Pada HMM, setiap state memiliki distribusi probabilitas dari setiap output yang mungkin.



Gambar 2. Hidden Markov Model

Pada gambar 2, variabel y adalah observasi yang mungkin, a_{ij} adalah kemungkinan transisi dari state i state j, dan b_{ij} adalah probabilitas y_j di state i. Terdapat tiga buah parameter dalam HMM, yaitu a, b, dan π. Parameter a mewakili kemungkinan transisi dari state, parameter b mewakili distribusi kemungkinan dari observasi, dan π mewakili distribusi state awal. Model HMM biasa diwakili dengan simbol λ, karenanya λ(a, b, π).

Training/Estimasi ulang model HMM (λ) ditujukan untuk mendapatkan model HMM yang lebih baik. Estimasi ulang akan dilakukan dengan menggunakan algoritma *Baum-Welch* (persamaan 1, 2, 3, 4) dan *Forward-Backward* (persamaan 5, 6, 7, 8). Proses estimasi akan dilakukan terus hingga nilai dari $P(O|\lambda') \geq P(O|\lambda)$ atau sudah mencapai batas iterasi yang ditentukan.

$$a_{ij}^* = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)} \quad \dots\dots (1)$$

$$b_i^*(k) = \frac{\sum_{t=1}^{T} 1_{o_t=v_k} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)} \quad \dots\dots (2)$$

dengan nilai $1_{o_t=v_k} = \begin{cases} 1, jika\ o_t = v_k \\ 0, lainnya \end{cases}$

$$\gamma_i(t) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(T)} \quad \dots\dots (3)$$

$$\xi_{ij}(t) = \frac{\alpha_i(t) a_{ij} \beta_j(t+1) b_j(o_t+1)}{\sum_{k=1}^{N} \alpha_k(T)} \quad \dots\dots (4)$$

a_{ij}^* mewakili probabilitas transisi baru dari state i ke state j, sedangkan $b_i^*(k)$ mewakili probabilitas observasi baru state i, pada observasi ke k. T mewakili total

observasi yang ada dan v adalah data observasi yang mungkin. Pada persamaan untuk mencari ξ dan γ, variabel N mewakili jumlah state yang ada. α adalah nilai forward, sedangkan β adalah nilai backward. Nilai parameter baru yang didapatkan dengan algoritma Baum-Welch akan menggantikan nilai parameter yang lama.

$$\alpha_i(1) = \pi_i b_i(o_1) \quad (5)$$

$$\alpha_j(t+1) = b_j(o_t + 1) \sum_{i=1}^{N} \alpha_i(t) a_{ij} \quad (6)$$

$$\beta_i(T) = 1 \quad (7)$$

$$\beta_i(t) = \sum_{j=1}^{N} \beta_j(t+1) a_{ij} b_j(o_{t+1}) \quad (8)$$

$$\sum_{i=1}^{N} \alpha_i(T) \quad (9)$$

Pada tahap klasifikasi, setiap kumpulan data observasi akan dibandingkan dengan semua model HMM yang ada menggunakan algoritma Forward. Model HMM yang memiliki probabilitas terbesar akan dipilih sebagai model yang mewakili data observasi tersebut (persamaan 10).

III. PREPROCESSING

Preprocessing berfungsi untuk mengurangi noise dan meningkatkan kualitas dari data yang akan diklasifikasi. Ada 4 jenis preprocessing yang digunakan, yaitu: Duplicate Point Filtering, Size Normalization, Smoothing, dan Speed Normalization.

1. Duplicate Point Filtering

Pada tahap ini, semua koordinat yang memiliki nilai x dan y yang sama dengan yang sebelum atau sesudahnya pada sebuah stroke akan dihapus. Koordinat kembar dianggap tidak dapat memberikan informasi yang berguna pada proses pengenalan tulisan tangan.

2. Size Normalization

Untuk mengurangi variasi dari ukuran stroke yang mungkin, Size Normalization digunakan untuk menormalisasi titik y menjadi nilai 0 sampai dengan 1.

3. Smoothing

Smoothing berfungsi untuk mengurangi noise yang ada pada stroke. Smoothing dapat dilakukan dengan mengganti nilai setiap koordinat, dengan rata-rata dari koordinat sebelum, sekarang, dan sesudah.

4. Speed Normalization

Speed Normalization (Resampling) digunakan untuk menghapus pengaruh dari kecepatan tulisan. Resampling dilakukan karena adanya perbedaan jarak antar koordinat yang ada di setiap stroke.

IV. EKSTRAKSI FITUR

Fitur-fitur yang ada dari sebuah stroke akan di ekstrak untuk kemudian digunakan dalam tahap Vector Quantization. Fitur-fitur yang diambil adalah:

1. Pen Up / Down

Fitur ini akan menyimpan keterangan apakah alat tulis yang digunakan menyentuh layar atau tidak pada waktu ke t. Fitur ini berbentuk *binary* (hanya terdiri dari nilai 0 atau 1).

2. Normalized Distance to Stroke Edge

Fitur ini memanfaatkan fitur Pen Up/Down untuk mendapatkan nilainya dengan mengambil jarak pada awal dan akhir dari stroke.

$$N(s,t) = \begin{cases} 1 - \frac{|de-db|}{ls}, actual \\ -\left(1 - \frac{|de-db|}{ls}\right), interpolated \end{cases} \quad (10)$$

Pada persamaan 10, $N(s,t)$ adalah normalized distance, ls adalah panjang dari stroke, db adalah jarak dari titik sekarang ke titik awal stroke, dan de adalah jarak dari titik sekarang ke titik akhir stroke.

3. Normalized Y-Coordinate

Normalized Y-Coordinate akan mengambil koordinat y yang telah melewati proses size normalization.

4. Vicinity Slope

Vicinity dari sebuah koordinat (x(t), y(t)) didapatkan dengan mendapatkan sudut diantara garis lurus yang menghubungkan koordinat (x(t-2), y(t-2)) dan (x(t+2), y(t+2)) dan garis horizontal di koordinat (x(t-2), y(t-2)).

5. Curvature

Curvature dari sebuah koordinat (x(t), y(t)) didapatkan dengan mendapatkan sudut diantara garis lurus yang menghubungkan koordinat (x(t-2), y(t-2)) dengan (x(t), y(t)) dan garis lurus yang menghubungkan koordinat (x(t), y(t)) dengan (x(t+2), y(t+2)).

Vector Quantization (VQ) digunakan untuk mengkompresi data. Cara kerja dari VQ adalah dengan membagi sejumlah besar vektor yang ada ke dalam grup yang memiliki jarak terdekat dengan vektor tersebut. Untuk membagi vektor ke dalam grup dapat menggunakan teknik k-means ataupun teknik clustering lainnya. Grup-grup tersebut di dalam VQ disebut sebagai codeword, sedangkan kumpulan dari codeword disebut sebagai codebook. Dalam penelitian ini, VQ digunakan untuk mengkonversi fitur-fitur yang didapatkan dari sebuah stroke ke bentuk kumpulan codeword.

V. PENUTUP

Uji coba yang dilakukan mencakup kombinasi fitur, eksperimen jumlah codeword, dan jumlah data training.

Tabel 1. Simbol yang Dikenali

Simbol					
0	1	2	3	4	5
6	7	8	9	x	+
/	-	(	)	∑	π
sin	cos	tan	√	!	log
∏	%	=			

$$acc = \frac{N_{berhaasil}}{N_{simbol}} \; x \; 100\% \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (12)$$

Pada persamaan 12, acc mewakili tingkat akurasi yang didapatkan, $N_{berhasil}$ mewakili jumlah simbol yang berhasil dikenali, dan N_{simbol} adalah jumlah simbol yang ada. Hasil testing dilakukan dengan mengambil contoh dari 6 orang yang berbeda ditambah beberapa sumber dari internet, dengan total 91 ekspresi matematika. Simbol matematika yang ada mencakup simbol yang ada pada tabel 1. Kecepatan rata-rata yang diperlukan untuk melakukan klasifikasi adalah 283.69 ms.

Untuk percobaan modifikasi fitur (tabel 2), fitur 1 mewakili Pen Up/Down, fitur 2 mewakili Normalized Distance to Stroke Edge, fitur 3 mewakili Normalized Y-Coordinate, fitur 4 mewakili Vicinity Slope, dan fitur 5 mewakili Curvature.

Tabel 2. Hasil Percobaan Modifikasi Fitur

Kombinasi Fitur	Tingkat akurasi
1, 2, 3, 4, 5	70.03%
1, 2, 3, 4	62.09%
1, 2, 3, 5	71.48%
1, 2, 4, 5	58.85%
1, 3, 4, 5	71.84%
2, 3, 4, 5	69.68%

VI. KESIMPULAN

Dari penelitian yang dilakukan, didapatkan beberapa kesimpulan:

1. HMM cukup baik digunakan untuk pengenalan tulisan tangan, dengan akurasi tertinggi adalah 72.92%.
2. Dari hasil percobaan dalam variasi data training, tingkat akurasi yang dihasilkan oleh 91 ekspresi matematika hasil tulisan tangan seseorang menghasilkan tingkat akurasi yang lebih tinggi daripada 20 data training untuk masing-masing simbol, dan jika keduanya digabungkan dengan beberapa tambahan simbol maka dapat mencapai tingkat akurasi yaitu 72.92%.
3. Untuk beberapa simbol masih ada yang sering mengalami ambiguitas dalam pengenalannya karena kemiripan yang cukup tinggi dengan simbol lainnya, sehingga perlu dilakukan penanganan khusus untuk simbol-simbol tersebut.

VII. DAFTAR PUSTAKA

[1] Chaninthorn Amornsawaddirak, Cholwich Natte, dan Nirattaya Khamsemanan. *Mathematical Handwritten Formula Recognition*. http://saki.siit.tu.ac.th/icictes2014/app/webroot/uploads_final/18__5facaeb540351fe572bbc96711fc1ed5/ChaninthornAmornsawaddirak.pdf. Februari 2014.

[2] Lei Hu dan Richard Zanibbi. 2011. *HMM-Based Recognition of Online Handwritten Mathematical Symbols Using Segmental K-Means Initialization and a Modified Pen-up/downs Feature*. 6 Februari 2014. http://www.cs.rit.edu/~rlaz/files/HuZanibbiICDAR2011.pdf.

[3] Han Shu. 1996. *On-Line Handwriting Recognition Using Hidden Markov Models*. http://dspace.mit.edu/bitstream /handle/1721.1/42603/37145316.pdf. Februari 2014.
[4] Gernot A. Fink. 2010. *Markov Models for Handwriting Recognition.* http://www.isical.ac.in/~icfhr2010/images/Fink-ICFHR2010-MMF.pdf. Februari 2014.