# KLASIFIKASI TULISAN TANGAN BERUPA ANGKA MENGGUNAKAN *RANDOM FOREST* DAN *HISTOGRAM OF ORIENTED GRADIENT*

**Anugerah Ganda Putra**[1], **Tjokorda Agung Budi Wirayuda ST. MT.**[2]

Fakultas Informatika Universitas Telkom Bandung

[1]**agps_frick@yahoo.com**, [2]**cokagung2001@gmail.com**

**Abstrak**

**Membuat mesin atau komputer mngenali tulisan tangan masih merupakan masalah yang terus diteliti di bidang Computer Vision. Angka sendiri tidak bisa dipisahkan dari kehidupan manusia sehingga kemampuan mengenali tulisan tangan berupa angka akan sangat membantu pekerjaan manusia. Berbagai jenis algoritma klasifikasi yang tersedia seperti ANN dan SVM mampu mengenali tulisan tangan, namun kelemahan keduanya terletak pada komplesitas waktu training yang cukup lama**
**Random Forest menjadi salah satu algoritma alternatif dan diuji dalam kasus klasifikasi tulisan tangan berupa angka pada Tugas Akhir ini. Random Forest tidak membutuhkan preprocessing khusus dalam pengimpelementasiannya untuk mencapai akurasi yang bagus. Diuji juga performansi Random Forest menggunakan fitur ciri Histogram of Gradient yang sering dipakai untuk mengenali objek dalam citra. Hasil pengujian menunjukkan Random Forest dengan jumlah 10 tree dan vektor ciri HOG yang berukuran mampu mengklasifikasi tulisan tangan berupa angka dalam dataset MNIST dengan akurasi bagus yang mencapai 97% dan waktu training sekitar 4 menit.**

***Abstract***

***Making machines or computer be able to recognize people's handwritings is still an ongoing research in Computer Vision. Digits themselves are still important parts of the humanity, therefore the ability to recognize handwritten digits will certainly helpful. The general choices for the classification algorithm includes Artifical Neural Network and Support Vector Machines. These algorithms could achieve good accuracy but nevertheless their training complexity are still considered to be high.***
***Another choice for the algorithm is Random Forest. Random Forest could achieve good results with little or no special preprocessing. In this book, Random Forest was tested with the MNIST handwritten digit dataset. Also tested was the feature vector called Histogram of Oriented Gradient. The result shows that Random Forest coupled with HOG could achieve good accuracy, as high as 97% with training time of only four minutes long.***

## 1. Pendahuluan

Mengenali objek merupakan salah satu masalah utama dalam bidang Computer Vision. Salah satu cabang dalam masalah pengenalan objek adalah mengenai tulisan. Tulisan adalah hal yang tidak terpisahkan dari kehidupan manusia sehingga tidak heran mendapatkan perhatian khusus dan mendapatkan nama bidang tersendiri yaitu OCR atau Optical Character Recognition.

Kemampuan mengenali tulisan tangan memiliki aplikasi yang beragam. Komputer yang mampu mengenali tulisan tangan dapat membantu penyortiran surat pada kantor pos misalnya, melakukan penyortiran secara otomatis dengan hanya melihat isian boks kode pos yang ditulis pada masing-masing surat. Untuk contoh aplikasi yang lebih modren, saat ini kita bisa menemui smartphone yang mampu menerima input sentuhan (touch sensitive). Tidak jarang smartphone ini menawarkan aplikasi yang mampu mengenali tulisan tangan yang di-input-kan oleh pengguna. Aplikasi-aplikasi ini mengimplementasikan klasifikasi tulisan tangan untuk mengenali tulisan tangan manusia dan memasukkannya ke dalam smartphone [11]. Pengenalan tulisan tangan sendiri adalah masalah yang cukup menantang karena

begitu variatif dan kompleksnya tulisan tangan yang dihasilkan oleh manusia.

Berbagai algoritma klasifikasi sudah digunakan untuk mengenali tulisan tangan angka. Dua yang populer adalah Support Vector Machine dan Artificial Neural Network. Keduanya mampu menghasilkan akurasi yang baik untuk mengenali tulisan tangan berupa angka dari MNIST. MNIST adalah salah satu dataset tulisan tangan angka yang populer. Rata-rata akurasi yang dicapai sudah cukup tinggi, yakni mencapai lebih dari 90% (lihat tabel 1.1) [12]. Hal ini tentu sudah cukup bagus. Namun yang tidak terlihat dalam tabel tersebut adalah bagaimana waktu training yang cukup lama masih diperlukan sebelum algoritma dapat dipakai untuk melakukan klasifikasi. Sebuah algoritma SVM yang dioptimasi agar dapat melakukan training dengan cepat, untuk melakukan klasifikasi menggunakan dataset MNIST waktu training-nya bisa mencapai empat jam [5].

Random Forest adalah salah satu alternatif algoritma klasifikasi yang dikenal mampu mencapai akurasi yang bagus tanpa melakukan pencarian yang banyak pada parameter training. Hal ini turut menjadi alasan kembali populernya Random Forest dalam masalah klasifikasi[2]. Random Forest merupakan sekumpulan decision tree yang hasil prediksinya diambil dari suara terbanyak dari prediksi sekumpulan decision tree yang ada di dalamnya. Selain masalah klasifikasi seperti prediksi, Random Forest juga banyak diteliti dan dikembangkan untuk permasalahan Computer Vision.

## 2. Dasar Teori

### 2.1 Pengenalan Tulisan Tangan

Manusia bisa dengan mudah mengenali sebuah tulisan tapi tidak demikian dengan mesin atau komputer. Pengenalan karakter atau Optical Character Recognition (OCR) adalah masalah yang sudah menjadi bahan penelitian sejak awal 1950-an namun hingga saat ini masih sedikit sistem OCR yang mampu mengenali dokumen atau tulisan tangan dengan baik [2].

Template matching adalah salah satu metode paling awal yang dikembangkan untuk mengenali simbol atau karakter. Pertama-tama, untuk setiap data karakter atau glyph yang sudah ada, sebuah pola disimpan ke dalam database. Sistem kemudian menyimpan seluruh pola yang harus dikenali ke dalam database. Selanjutnya simbol baru dikenali dengan membandingkan pixel per pixel simbol baru dengan pola yang sudah tersimpan.

Algoritma yang menggunakan template masih bisa digunakan untuk mengenali karakter dari tulisan yang dicetak, namun pencocokan template menjadi semakin rumit ketika digunakan untuk mengenali tulisan manusia. Alasan utama dari hal ini adalah karena tulisan manusai inkonsisten: tulisan manusia untuk satu karakter yang sama sekalipun biasanya tidak akan sama [2].

Gambar 2.1 Contoh tulisan angka '3'

Untuk tulisan tangan berupa angka, misalnya. Gambar 2.1 menunjukkan contoh tulisan tangan untuk angka 3. Dalam gambar terlihat variasi tulisan untuk angka 3 dan inkonsistensi bentuk angka dari satu gambar ke gambar lainnya. Dataset tulisan tangan angka yang terkenal antara lain adalah MNIST dan USPS.

Sebuah citra tulisan yang akan dikenali biasanya mengalami tiga tahapan [2]. Yang pertama adalah tahap preprocessing yang bertujuan untuk memperbaiki kualitas gambar tulisan. Gambar tulisan biasanya diperoleh menggunakan scanner sehingga tidak jarang kualitas gambar yang ada tidak begitu bagus. Yang ke-2 adalah tahap ekstraksi ciri yang bertujuan untuk mendapatkan karakteristik unik dari citra untuk kemudian digunakan untuk pengenalan. Tahapan terakhir adalah tahap klasifikasi yaitu penggunaan teknik klasifikasi untuk mengenali citra tulisan.

Sudah banyak algoritma klasifikasi yang sudah dipakai untuk membangun sistem OCR. Algoritma-algoritma ini dapat dikelompokkan menjadi kelompok klasifikasi statistikal, ANN, SVM, struktural, dan multiple classifier [2]. Random forest termasuk dalam kelompok yang terakhir. Random forest adalah metode klasifikasi

yang menggabungkan beberapa classifier yang berbentuk decision tree.

## 2.2 Random forest

Decision Tree adalah metode learning yang menggunakan struktur tree di mana di setiap node leaf-nya terdapat informasi mengenai prediksi yang akan dikeluarkan. Decision tree dibangun dengan menggunakan proses training di mana untuk setiap node dan sampel data training, algoritma learning dalam decision tree memilih satu atribut dalam

sampel yang memenuhi kriteria tertentu. Sampel kemudian dipecah sesuai dengan nilai atribut yang telah dipilih tersebut dan untuk setiap sampel pecahan tersebut, node turunan dibuat. Hal ini berlangsung terus menerus hingga kondisi tertentu terpenuhi, misalnya jumlah maksimum kedalaman tree atau jumlah minimum sampel yang tersedia. Gambar 2.3 memberikan algoritma umum pembangunan decision tree [6].

Algoritma learning untuk Decision Tree yang populer antara lain adalah ID3, C45, dan CART

[6]. Dalam membangun decision tree, untuk setiap

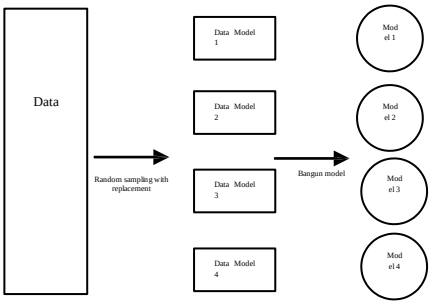
masuk. Dalam ID3 dan C45, pemilihan atribut yang akan dijadikan kondisi pemisahan node dilakukan dengan menghitung information gain. Berbeda dengan ID3, CART menggunakan gini index untuk mengukur kemurnian suatu kumpulan data (2.1)

[6].

( 2.1)

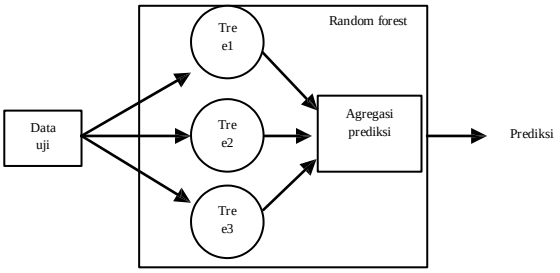
— — ( 2.2)

Untuk menghitung gini index binary split, misal untuk data D dipartisi menjadi dua data D1 dan D2, maka nilai gini D dihitung dengan menjumlahkan nilai gini tiap partisi yang sudah diberi bobot (2.2). Decision tree melakukan klasifikasi dengan melakukan penelusuran node mulai dari root hingga ke node leaf, sesuai dengan kondisi atribut tiap node. Ketika mencapai sebuah node leaf, hasil prediksi didapatkan dengan mengambil informasi prediksi yang ada dalam node leaf tersebut. Untuk mendapatkan skor atau probabilitas prediksi, rasio jumlah data training kelas hasil prediksi dihitung untuk setiap leaf node.




Gambar 2.2 Proses pembangunan ensemble learner

Random forest adalah algoritma klasifikasi dan regresi yang merupakan bagian dari kelompok ensemble learning. Model base classifier yang dipakai adalah decision tree sehingga membentuk 'forest' atau hutan. Random forest dipopulerkan oleh Leo Breiman [1].




Gambar 2.3 Proses prediksi random forest

Dalam random forest, pemilihan atribut dalam setiap kali sebuah node akan dipecah diambil secara acak. Pertama-tama setiap tree diberi sampel data training dengan menggunakan metode bagging, dan tiap tree dibangun menggunakan metode yang sama untuk membangun CART. Perbedaan yang mencolok terdapat pada proses pemilihan splitting criterion. Alih-alih mempertimbangkan seluruh atribut yang ada pada data, random forest hanya mempertimbangkan subset dari keseluruhan atribut (biasanya jumlah atribut yang diseleksi ditentukan oleh user). Atribut yang akan diseleksi ini didapatkan secara acak. Seperti CART, pembangunan tree akan berhenti ketika data sudah homogen atau batas jumlah data minimum terlewati. Namun terdapat variasi Random Forest yang menentukan besar kedalaman tree maksimum [3].

Dengan proses pemilihan atribut yang acak pada pembangunan tree, Random Forest dapat

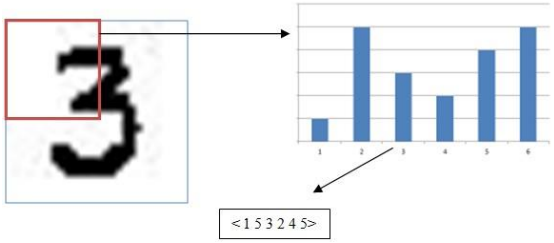
menangani data dengan dimensi tinggi dengan baik dibanding model classifier lainnya. Berbeda dengan Decision Tree biasa, overfitting diatasi dengan menjaga variansi model tree dalam forest. Dalam Random Forest umumnya tidak terdapat leaf pruning (penghilangan node leaf), tetapi variasi Random Forest seperti Hough Forest dalam implementasinya menerapkan leaf pruning untuk menghilangkan leaf node dengan probabilitas rendah [3].

Random forest yang juga memiliki nama Decision Forest dalam pengaplikasiannya memiliki beberapa varian, antara lain Extremely Randomized Tree, dan Hough Forest. Random forest dapat melakukan klasifikasi, regresi, dan bahkan clustering. Hal ini menjadi daya tarik random forest dan turut mendorong aplikasi dan penelitian random forest dalam bidang computer vision [3].

### 2.3 Histogram of Oriented Gradient

Vektor ciri yang dibangun dengan membentuk histogram terhadap orientasi gradien pada gambar merupakan salah satu deskriptor yang populer untuk mengenali objek. Histogram of Oriented Gradient atau HOG telah dipakai untuk mengenali objek pejalan kaki dalam gambar [4]. Jenis deskriptor serupa juga dipakai untuk melakukan klasifikasi objek pada dataset yang menantang seperti CALTECH.

Gradien suatu pixel adalah sebuah vektor yang memiliki sifat selalu menunjuk ke arah perubahan pixel terbesar di sekitarnya[5]. Besaran vektor ini disebut magnitude dan arah vektor disebut orientation (orientasi). Gambar gradien dari suatu gambar adalah gambar yang setiap nilai pixelnya berisi magnitude pixel [9][5].




Gambar 2.4 HOG diambil dari histogram tiap cell

Untuk membangun vektor ciri berbasis histogram dari orientasi gradient, pertama-tama gradient tiap pixel dihitung dan didapatkan gambar gradient. Gambar gradient kemudian dibagi ke dalam sejumlah cell dengan ukuran tertentu. Untuk setiap cell, nilai orientasi dari tiap pixel kemudian dihitung dan dimasukkan ke dalam sebuah histogram. Vektor dibentuk dengan menggabungkan histogram dari seluruh cell.

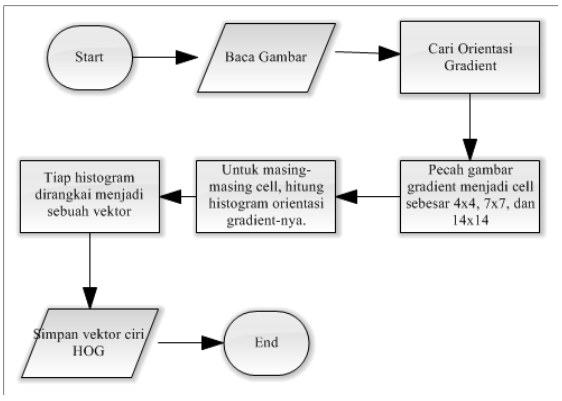
Vektor ciri berbasis histogram dari orientasi gradient cukup tahan terhadap perubahan gambar seperti rotasi, skala, dan translasi. Perubahan rotasi yang tidak signifikan tidak akan membuat perubahan signifikan pula pada nilai histogram apabila nilai rotasi tidak lebih besar dari besar interval bin. Sedangkan perubahan skala dan rotasi ditangani dengan membuat histogram untuk berbagai macam ukuran cell yang saling overlap [7].

## 3. Perancangan Sistem

Sistem dibangun dengan membuat modul sesuai jenis pengujian. Setiap pengujian melalui tiga tahapan: preprocessing, training dan testing.

### 3.1 Preprocessing

Dalam pengujian akan dipakai dua jenis vektor ciri, yaitu vektor ciri dari MNIST (nilai pixel mentah) dan vektor ciri HOG. Vektor ciri HOG merupakan vektor dengan panjang 784 yang diambil dari gambar tulisan tangan berupa angka yang berukuran 28x28. Tiap elemen dalam vektor memiliki range nilai 0-1. Nilai ini menunjukkan intensitas suatu pixel.




Gambar 3.1 Proses ekstraksi HOG

Vektor ciri HOG didapatkan dengan mencari orientasi gradient pada gambar tulisan angka. Gambar ini kemudian dibagi menjadi beberapa area atau cell yang saling overlap. Terdapat tiga jenis pembagian cell (gambar 3.2), yaitu pembagian

dengan cell ukuran 14x14, 7x7, dan 4x4. Tiap cell kemudian dihitung orientasi gradient-nya dan dimasukkan ke dalam histogram. Seluruh histogram dari seluruh cell yang ada kemudian digabung menjadi sebuah vektor ciri HOG. Karena setiap histogram terdiri dari 12 bin histogram, maka panjang vektor HOG adalah 12*(4+16+49) = 828.

### 3.2 Training dan testing

Training bertujuan untuk membangun decison tree dalam sebuah random forest. Training random forest memiliki beberapa parameter yang akan menjadi bahan pengujian. Training menggunakan data training MNIST yang berjumlah 60.000 data gambar tulisan tangan berupa angka.

Testing dilakukan dengan menggunakan data testing MNIST. Data testing MNIST seluruhnya berjumlah 10.000.

## 4. Hasil Pengujian

### 4.1 Pengujian Dua Jenis Vektor Ciri

Vektor ciri pertama adalah vektor ciri berupa nilai mentah pixel dalam dataset MNIST. Vektor ciri ke-2 adalah vektor ciri yang didapatkan dari ekstraksi ciri Histogram of Oriented Gradient. Kedua pengujian dilakukan dengan parameter training yang sama, yaitu parameter training bawaan Matlab. Hasil pengujian terdapat pada Tabel 4.1.

Tabel 4.1 Hasil percobaan dengan vektor ciri berbeda

| Jenis vektor ciri | Waktu training (sekon) | Akurasi |
|---|---|---|
| Pixel mentah | 110,0102474 | 92,23 % |
| HOG | 104,1081668 | 94,19 % |

Ternyata akurasi menggunakan vektor ciri HOG lebih baik walaupun menggunakan parameter training yang sama. Ini tidak berarti akurasi random forest menggunakan vektor ciri nilai pixel yang mentah tidak tinggi; akurasi 92,23 % merupakan hasil yang bagus. Akan tetapi hasil pengujian skenario pertama ini menunjukkan HOG mencapai akurasi yang lebih baik. Hal ini sesuai dengan pengujian dua vektor ciri serupa tapi menggunakan SVM. Dengan demikian dapat disimpulkan vektor ciri HOG menghasilkan akurasi lebih baik, sehingga vektor ciri ini kemudian dipakai untuk skenario-skenario pengujian selanjutnya.

### 4.2 Pengujian Terhadap Parameter Jumlah Atribut Acak

Pengujian ke-2 dilakukan terhadap parameter jumlah atribut yang akan diseleksi setiap akan melakukan node splitting. Hasil pengujian disajikan di dalam tabel 4.2.

Tabel 4.2 Hasil pengujian dengan jumlah atribut acak berbeda

| Jumlah atribut seleksi | Waktu training (detik) | Akurasi (%) |
|---|---|---|
| 20 | 105,5206756 | 93,75 |
| 22 | 96,18154188 | 94,15 |
| 24 | 97,11942673 | 94,31 |
| 26 | 100,8524108 | 94,5 |
| 28 | 101,5894494 | 94,42 |
| 30 | 104,6529679 | 94,32 |
| 32 | 108,0514215 | 94,5 |
| 34 | 111,8917574 | 94,72 |
| 36 | 120,4990886 | 94,15 |
| 38 | 139,8353312 | 94,49 |
| 40 | 133,848706 | 94,58 |

Hasil pengujian menunjukkan tidak adanya hubungan yang jelas antara parameter jumlah atribut yang akan diseleksi. Hal ini menunjukkan jumlah atribut yang diambil secara acak untuk kemudian diseleksi dalam setiap node splitting tidak mempengaruhi akurasi secara signifikan. Bila mengamati proses pembangunan random forest, hasil pengujian ini sejalan dengan algoritma random forest itu sendiri. Karena pengambilan atribut diambil secara acak, maka sulit melihat adanya hubungan antara perubahan akurasi terhadap jumlah atribut yang akan diseleksi sebagai kondisi split. Hal yang serupa terlihat pada pengukuran waktu training.

### 4.3 Pengujian Terhadap Jumlah Data untuk Splitting

Pengujian ke-3 adalah pengujian terhadap parameter jumlah data minimum ketika mengecek apakah sebuah node harus di-split atau tidak. Hasil pengujian disajikan dalam tabel 4.3.

Sekali lagi, tidak terlihat ada hubungan yang jelas terhadap akurasi dengan jumlah data minimum untuk melakukan splitting (lihat gambar 4.2). Namun hal berbeda terlihat pada waktu

training. Waktu training semakin berkurang seiring pertambahan jumlah data. Hal ini terjadi karena semakin banyaknya jumlah data dalam tiap leaf, maka banyaknya node splitting semakin sedikit sehingga seleksi atribut yang memakan resource waktu paling banyak dalam training decision tree semakin sedikit. Dengan mengamati ini, maka diambil nilai parameter 10 sebagai nilai parameter optimal. Nilai ini dipakai untuk pengujian skenario selanjutnya.

Tabel 4.3 Hasil pengujian dengan jumlah data splitting berbeda

| Jumlah Data | Waktu training (detik) | Akurasi (%) |
|---|---|---|
| 1 | 126,3159038 | 94,02 |
| 2 | 100,7127874 | 94,55 |
| 3 | 94,67578279 | 95,24 |
| 4 | 90,50042335 | 95,07 |
| 5 | 83,68210277 | 95,34 |
| 6 | 81,30676448 | 95,07 |
| 7 | 80,85966249 | 94,99 |
| 8 | 79,45314535 | 95,25 |
| 9 | 75,65608459 | 95,2 |
| 10 | 73,27137896 | 94,83 |

### 4.4 Hasil Pengujian Terhadap Jumlah Tree Berbeda

Pengujian terakhir menguji performansi Random Forest apabila menggunakan jumlah tree berbeda. Dalam literatur disebutkan bahwa kenaikan jumlah tree akan menambah akurasi, sehingga selain membuktikan pernyataan ini, pengujian skenario 4 juga mencari akurasi terbaik yang mampu dicapai. Hasil pengujian terdapat pada tabel 4.4. Sampel salah satu tree yang diambil dari random forest berada pada Lampiran A.

Tabel 4.4 Hasil pengujian dengan jumlah tree berbeda

| Jumlah tree | Waktu training (detik) | Akurasi (%) |
|---|---|---|
| 5 | 106,9361278 | 94,93 |
| 10 | 147,2516865 | 96,81 |
| 15 | 222,4389108 | 97,29 |
| 20 | 295,8799572 | 97,38 |
| 25 | 377,0972041 | 97,47 |
| 30 | 446,6888901 | 97,63 |
| 50 | 739,7212715 | 97,83 |
| 100 | 1483,994183 | 97,85 |

Dari tabel hasil pengujian terlihat bahwa setiap kenaikan jumlah tree menambah akurasi klasifikasi. Namun ketika jumlah tree mencapai sekitar 25, kenaikan akurasi semakin kecil sementara kenaikan waktu training naik secara linear. Hal ini tidak menguntungkan dan terlihat ketika membandingkan hasil pengujian jumlah tree 50 dan 100: untuk mencapai kenaikan 0,02% pada akurasi, dibutuhkan waktu training dua kali lipat lebih besar. Karena hal ini, disimpulkan bahwa jumlah tree yang paling akurat dan efesien dalam hal waktu training adalah random forest dengan jumlah tree 10 dan 15 yaitu dengan akurasi 96.81 % dan 97,29%. Kenaikan akurasi random forest yang lebih besar kurang dari 1%.

### 4.5 Misklasifikasi Angka

Lantas bagaimana dengan kesalahan klasifikasi atau misklasifikasi yang dilakukan random forest? Tabel 4.5 menampilkan data misklafikasi yang dilakukan random forest dengan jumlah tree 30.

Tabel 4.5 Error rate untuk setiap digit

| Digit | Error rate (%) | Paling Banyak Dimisklasifikasi Sebagai |
|---|---|---|
| 0 | 0.7143 | 6 |
| 1 | 1,2335 | 2 |
| 2 | 2,1318 | 0,7 |
| 3 | 1,5842 | 7 |
| 4 | 2.2403 | 9 |
| 5 | 2.4664 | 3 |
| 6 | 2.9228 | 5 |
| 7 | 2.6265 | 7 |
| 8 | 3.1828 | 2,4,7 |
| 9 | 4.7572 | 4 |

Dari tabel di atas terlihat bahwa digit 9 merupakan digit yang paling banyak dimisklasifikasi oleh model random forest dalam skenario pengujian. Angka 9 paling banyak salah dikenali sebagai angka 4, hal ini wajar mengingat bentuk 9 yang ditulis oleh tangan manusia terkadang terlihat mirip dengan angka 4 (lihat gambar 4.1).

Gambar 4.1 Angka 9 yang salah dikenali sebagai 4

Demikian halnya dengan angka 0 sebagai digit yang paling sedikit dimisklasifikasi. 0 merupakan angka dengan bentuk yang paling unik dibanding angka lain. Angka 0 paling banyak salah dikenali sebagai angka 6 pun merupakan hal yang wajar karena bentuk tulisan tangan angka 0 dan 6 hampir serupa yakni sama-sama memiliki bagian yang berbentuk simpul atau lingkaran.

## 5. Kesimpulan dan Saran

Kesimpulan yang dapat diambil dari tugas akhir ini antara lain :

1. Klasifikasi tulisan tangan berupa angka dari MNIST menggunakan random forest dapat mencapai akurasi tinggi, lebih dari 90% dengan waktu training kurang dari 10 menit.
2. Kenaikan jumlah tree pada random forest akan menambah akurasi random forest. Pada kasus MNIST, dengan menggunakan vektor ciri HOG, random forest cenderung naik drastis hingga jumlah tree mencapai 15. Untuk parameter pembangunan lain seperti jumlah atribut yang diseleksi secara acak dan jumlah data minimum splitting, tidak terlihat hubungan yang jelas antara nilai keduanya dengan akurasi klasifikasi.
3. Vektor ciri HOG menghasilkan akurasi lebih baik ketika dipakai melakukan klasifikasi tulisan tangan berupa angka dari dataset MNIST menggunakan random forest

Saran yang dapat diberikan terkait dengan tugas akhir ini antara lain :

1. Dalam pengujian dilakukan perbandingan dengan algoritma klasifikasi lain secara langsung.
2. Pengujian menggunakan vektor ciri lain selain nilai mentah pixel dan HOG.

## Daftar Pustaka


[1] Breiman, Leo. 2001. *Random Forests.* Kluwer-Academic Publishers

[2] Cheriet, M. et al. 2007. *Character Recognition Systems: A Guide for Students and Practitioners*. John Wiley & Sons, Inc.

[3] Criminisi, A., Shotton, J. 2013. *Decision Forest for Computer Vision and Medical Image Analysis*. Springer-Verlag London.

[4] Dalal, Navneet dan Triggs, Bill. 2005. *Histograms of Oriented Gradients for Human Detection*

[5] Gonzalez, R.C dan Woods,R.E. 2007. *Digital Image Processing*. Prentice Hall.

[6] Han, Jiawei., Kamber, Micheline., dan Pei, Jian. 2012. *Data Mining Concepts and Techniques*. Elsevier Inc.

[7] Maji, Subhransu. Malik, Jitendra. 2009. *Fast and Accurate Digit Classification*. Elsevier Inc.

[8] Mitchell, Tom. 1997, *Machine Learning*, McGraw-Hill

[9] Nixon, M.S., Aguado, A.S. 2012. *Feature Extraction & Image Processing for Computer Vision*. United Kingdom. Elsevier Ltd.

[10] Parker, J.R. 2011. *Algorithms for Image Proecssing and Computer Vision*. Wiley Publishing, Inc.

[11] Google unveils handwriting recognition system for smartphones. http://www.v3.co.uk/v3-uk/the-frontline-blog/2194860/google-unveils-handwriting-recognition-system-for-smartphones diakses pada 12 Agustus 2014

[12] THE MNIST DATABASE of handwritten digits. http://yann.lecun.com/exdb/mnist/ diakses pada 12 Agustus 2014