BAB V

KESIMPULAN DAN SARAN

Pada bab ini akan dipaparkan beberapa kesimpulan dari hasil penelitian yang diperoleh serta saran yang dapat digunakan sebagai acuan untuk menyempurnakan sistem pada proses pengembangan berikutnya.

5.1 Kesimpulan

Berdasarkan pembahasan yang telah dilakukan pada bab sebelumnya, penulis dapat menarik kesimpulan dalam penelitian ini. Kesimpulan yang diperoleh dari pengujian yang dilakukan pada beberapa data set membuktikan bahwa metode kombinasi dapat mengatasi kelemahan pada metode KNN dengan proses waktu lebih cepat dan kelemahan pada Naïve Bayes dengan persentase akurasi sama dengan atau lebih tinggi.

5.2 Saran

Pengembangan yang dapat dilakukan pada penelitian di masa yang akan datang ialah menggunakan model komputasi paralel (*parallel computing*) dengan jumlah block dan thread sesuai dengan jumlah data pelatihan dan data uji untuk mempercepat tahap pemrosesan data.

DAFTAR PUSTAKA

Abraham, R., Simha, J.B. & Sitharama , S., 2009. Effective Discretization and Hybrid feature selection using Naïve Bayesian classifier for Medical datamining. *International Journal of Computational Intelligence Research*, 5(2), pp.116–29.

Adhatrao, K. dkk, 2013. Predicting Students' Performance using ID3 AND C4.5 Classification Algorithms. *International Journal of Data Mining & Knowledge Management Process*, 3(5), pp.39-52.

Baby, N. & T., P.L., 2012. Customer Classification And Prediction Based On Data Mining Technique. *International Journal of Emerging Technology and Advanced Engineering*, 2(12), pp.314-18.

Baradwaj, B.K. & Pal, S., 2011. Mining Educational Data to Analyze Students Perofmance. *International Journal of Advanced Computer Science and Applications*, 2(6), pp.63-69.

Beniwal, S. & Arora, J., 2012. Classification and Feature Selection Techniques in Data Mining. *International Journal of Engineering Research & Technology*, 1(5), pp.1-6.

Bergeron, B., 2003. Essential of Knowledge Management. John Wiley & Sons, Inc. New Jersey.

Bhargavi, P. & Jyothi, S., 2011. Soil Classification Using Data Mining Techniques: A Comparative Study. *International Journal of Engineering Trends and Technology*, 2(1), pp.55-59.

Bhuvaneswari, R. & Kalaiselvi, K., 2012. Naive Bayesian Classification Approach in Healthcare Applications. *International Journal of Computer Science and Telecommunications*, 3(1), pp.106-12.

Chou, K. & Shen, H., 2006. Predicting Eukaryotic Protein Subcellular Location by Fusing Optimized Evidence-Theoretic K-Nearest Neighbor Classifiers. *Journal of Proteome Research*, 5(8), pp.1888-1897.

Christobel, Y.A. & Sivaprakasam, P., 2013. A New Classwise k Nearest Neighbor (CKNN) Method for the Classification of Diabetes Dataset. *International Journal of Engineering and Advanced Technology*, 2(3), pp.396-200.

Danesh, A., dkk, 2007.Improve Text Classification Accuracy Based on Classifier Fusion Methods. *Information Fusion*, pp.1-6.

Farid, D.M., Harbi, N. & Rahman, M.Z., 2010. Combining Naive Bayes and Decision Tree. *International Journal of Network Security & Its Applications*, 2(2), pp.12-25.

Guo, G., dkk, 2006. Using KNN Model for Automatic Text Categorizaton. *Soft Computing*, 10(5), pp.423-430.

Hall, M., 2007. A Decision Tree-Based Attribute Weighting Filter for Naïve Bayes. *Knowledge-Based Systems*, 20(2), pp.120-126.

Jain, V., Narula, G.S. & Singh, M., 2013. Implementation of Data Mining in Online Shopping System using Tanagara Tool. *International Journal of Computer Scienceand Engineering*, 2(1), pp.47-58.

Jiang, L. dkk, 2005. Learning k-Nearest Neighbor Naïve Bayes for Ranking. *Advanced Data Mining and Applications*, 3584, pp.175-185.

Jnanamurthy, H. dkk, 2013. Discovery of Maximal Frequent Item Sets using Subset Creation. *International Journal of Data Mining & Knowledge Management Process*, 3(1), pp.27-38.

Kabir, M.F., Hossain, A. & Dahal, K., 2011. Enhanced Classification Accuracy on Naive Bayes Data Mining Models. *International Journal of Computer Applications* , 28(3), pp.9-16.

Kaur, G. & Aggarwal, S., 2013. Perfomance Analysis of Association Rule Mining Algorithms. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(8), pp.856-58.

Kaur, H. & Kaur, H., 2013. Proposed Work for Classification and Selection of Best Saving Service for Banking Using Decision tree Algorithms. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(9), pp.680-84.

Kumar, R. & Verma, R., 2012. Classification Algorithms for Data Mining:A Survey. *International Journal of Innovations in Engineering and Technology*, 1(2), pp.7-14.

Nithyasri, B., Nandhini, K. & Chandra, E., 2010. Classification Techniqes in Education Domain. *International Journal on Computer Science and Engineering*, 2(5), pp.1679-84.

Patil, T.R. & Sherekar, S.S., 2013. Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification. *International Journal Of Computer Science And Applications*, 6(2), pp.256-61.

Phyu, T.N., 2009. Survey of Classification Techniques in Data Mining. *Proceedings of International MultiConference of engineers and Computer Scientist*.

Raviya, K.H. &Gajjar, B., 2013. Perfomance Evaluation of Different Data Mining Classification Algorithm Using WEKA. *Indian Journal of Research* 2(1), pp.19-21.

Sahu, H., Shrma, S. & Gondhalakar, S., 2011. A Brief Overview on Data Mining Survey. *International Journal of Computer Technology and Electronics Engineering*, 1(3), pp.114-21.

Shazmeen, S.F., Baig, M.M.A. & Pawar, M.R., 2013. Performance Evaluation of Different Data Mining Classification Algorithm and Predictive Analysis. *IOSR Journal of Computer Engineering*, 10(6), pp.1-6.

Singh, K.S., Wayal, G. & Sharma, N., 2012. A Review: Data Mining with Fuzzy Association Rule Mining. *International Journal of Engineering Research & Technology*, 1(5), pp.1-4.

Suresh, J., Rushyanth, P. & Trinath, C., 2013. Generating Association Rule Mining using Apriori and FP-Growth Algorithms. *International Journal of Computer Trends and Technology*, 4(4), pp.887-92.

Ting, S.L., Ip, W.H. & Albert, H.C.T., 2011. Is Naïve Bayes a Good Classifier for Document Classification? *International Journal of Software Engineering and Its Applications*, 5(3), pp.37-46.

Xie, Z. dkk, 2002. SNNB: A Selective Neighborhood Based Naïve Bayes for Lazy Learning. Advances in Knowledge Discovery and Data Mining, 2336, pp.104-114.

Umamaheswari, K. & Niraimathi, S., 2013. A Study on Student Data Analysis Using Data Mining Techniques. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(8), pp.117-20.

Wu, X., dkk, 2007.Top 10 Algorithms in Data Mining. *Knowledge and Information System*, 14(1), pp.1-37.

http://archive.ics.uci.edu/ml/, diakses pada 8 Agustus 2014

LAMPIRAN

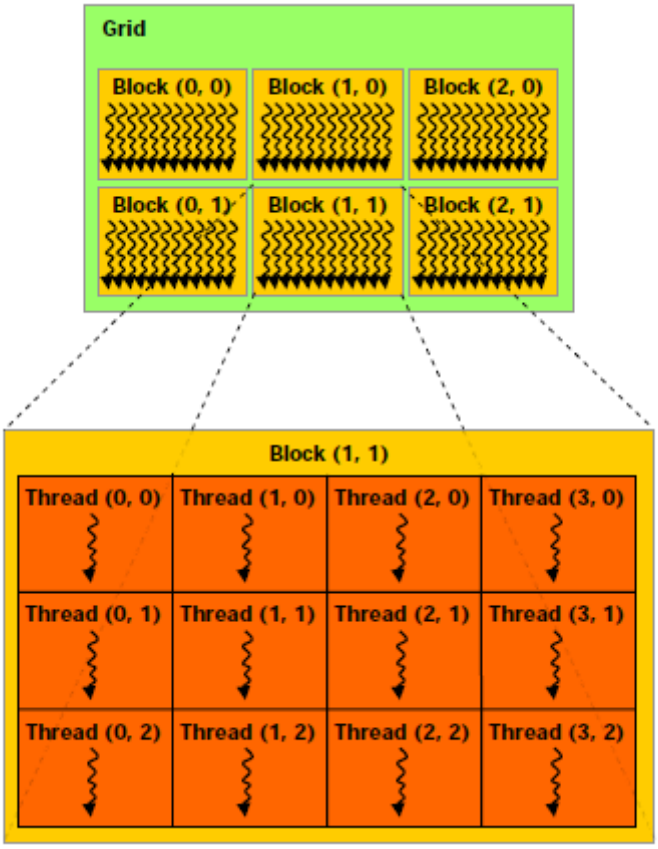
Implementasi Metode Kombinasi KNN-Naïve Bayes pada CUDA

1. Landasan Teori

GPU (Graphics Processing Unit) ialah prosesor parallel yang biasa digunakan untuk pengolahan grafis 3-dimensi berkualitas tinggi. Pada umumnya GPU digunakan pada PC (Personal Computer) untuk membantu CPU dalam menangani program atau perhitungan umum. GPU banyak digunakan untuk aplikasi grafis seperti video game atau perangkat lunak simulasi dan dioptimalkan untuk menangani tugas dimana CPU membutuhkan waktu yang cukup lama untuk menanganinya.

CUDA (Compute Unified Device Architecture) adalah model pemrograman dan perangkat lunak untuk mengelola perhitungan pada GPU sebagai perangkat komputasi data parallel dengan menggunakan bahasa pemrograman standar seperti C. Arsitektur CUDA dirancang untuk dapat mengembangkan aplikasi dengan parallel computing dengan bahasa pemrograman C dan C++. CUDA terdiri dari dua bagian. Bagian pertama dijalanan pada GPU yaitu kernel. Kernel diimplementasikan dalam bahasa pemrograman CUDA yang dasarnya adalah bahasa C yang ditambahkan sejumlah kata kunci. Bagian kedua, bagian yang dijalankan pada CPU (host) dan memberikan kontrol atas transfer data antara CPU-GPU dan pelaksanaan kernel.

Kernel adalah fungsi yang dieksekusi secara parallel oleh threads pada GPU. Kumpulan threads disebut dengan block. Kernel dipanggil dengan kode host berparameter yang menentukan dimensi hirarki pada thread dan jumlah memori yang harus dialokasikan untuk setiap blok. Thread diidentifikasi dengan index yang unik atau disebut dengan thread ID. Terdapat 3 level thread pada CUDA. Level pertama yaitu thread yang merupakan unit pemrosesan terkecil. Level kedua yaitu block, yang merupakan kumpulan 1 dimensi, 2 dimensi atau 3 dimensi dari threads. Setiap thread direferensikan dengan (x,y,z). Level ketiga yaitu grid, merupakan 1 dimensi atau 2 dimensi dari block. Seperti pada thread, blok direferensikan dengan (x,y). Struktur level thread seperti pada gambar 1.



Gambar 1. Struktur Level Thread Pada CUDA

Pada pemanggilan kernel diperlukan dua kunci utama yaitu untuk alokasi dimensi block dan dimensi thread. Sebagai contoh untuk menjalankan fungsi `kernel<<<2,1>>>()`, berarti membuat dua salinan dari kernel dan menjalankan fungsi tersebut secara paralel. Contoh lain yaitu `kernel<<<256,1>>>()`, berarti terdapat 256 block yang dijalankan pada GPU. Ketika kernel dijalankan, dengan jumlah block yang sudah didefinisikan thread akan memberikan nilai yang berbeda-beda untuk blockIdx.x. Fungsi blockIdx.x untuk memberikan index. blockIdx.x pertama bernilai 0 hingga nilai terakhir jumlah blocks-1. Kata kunci untuk menggunakan compiler pada GPU ialah dengan menambahkan `__global__` sebelum fungsi. Seperti pada bahasa pemrograman C, untuk alokasi (`malloc()`), salin data (`memcpy()`) dan menghapus alokasi data (`free()`), CUDA juga memiliki hal serupa dengan alokasi memori device menggunakan `cudaMalloc()`, salin data dari host ke device menggunakan `cudaMemcpy()` dan hapus alokasi data menggunakan `cudaFree()`.

2. Implementasi CUDA

Metode kombinasi KNN-Naïve Bayes yang sudah diuji dengan program yang dibangun dengan bahasa pemrograman C, akan diimplementasikan menggunakan CUDA v5.0. Pada pemrograman menggunakan CUDA C, digunakan array 1 dimensi saat perhitungan jarak pada masing-masing block.

Block yang digunakan untuk perhitungan metode kombinasi sejumlah data training, dan masing-masing blok digunakan 1 thread.

Berikut beberapa barisan code c++ yang dijalankan secara serial pada perhitungan jarak untuk 1 data uji:

```
for(int i=0;i<rowsTraining;i++)
{
   cek=false;
   for(int j=0;j<columns-1;j++)
   {
     if(dataTest[0][j]==dataTraining[i][j])
     {
       cek=true;
       break;
     }
   }
   if(cek=true)
   {
     sum=0;
     for(int l=0;l<columns-1;l++)
     {
       sum+=pow((dataTest[0][l]-dataTraining[i][l]),2);
     }
     jarak[i]=sqrt(sum);
     jmlData++;
   }
   else
   jarak[i]=100;
}
```

Barisan code diatas ialah barisan code serial untuk pencarian jarak seperti yang sudah dipaparkan pada sub-bab sebelumnya. Serial mengakses data dengan menggunakan indeks dan diolah satu persatu pada CPU.

Barisan code c++ tersebut diubah kedalam bentuk CUDA C sehingga dapat dijalankan secara paralel pada perhitungan jarak untuk 1 data uji. Berikut merupakan barisan code dengan CUDA C:

```
int row=blockIdx.x;
int sum=0;
bool cek;
```

```
if(row<rowsTraining)
{
  cek=false;
  for(int j=0;j<columns-1;j++)
  {
     if(dataTraining[row*columns+j]== dataTest[j])
     {
       cek=true;
       break;
     }
  }
  if(cek==true)
  {
    for(int l=0;l<columns-1;l++)
    {
      sum+=pow((dataTraining[row*columns+l]-
dataTest[l]),2);
    }
    jarak[row]=sqrt(sum);
  }
  else
     jarak[row]=100;
}
```

Barisan code tersebut dijalankan secara paralel dengan menggunakan 1 block dan 1 thead untuk menghitung jarak antara 1 dataTraining ke dataTest. DataTraining dibagi ke sejumlah block. Setiap block akan melakukan pengecekan apakah dataTraining tersebut memiliki nilai atribut yang sama dengan dataTest. Jika ya, maka melakukan perhitungan jarak antara dataTraining dengan dataTest. Hal ini dilakukan secara bersamaan pada device. Kemudian hasil jarak akan dikirimkan dari device ke host untuk dilanjutkan dengan pencarian jarak terdekat.

3. Hasil dan Analisis

Hasil implementasi CUDA pada metode kombinasi KNN-Naïve Bayes akan dikaji pada tabel 4.9.

Tabel 1 Hasil Perbandingan Waktu Proses CPU dan GPU

Dataset	Waktu pengolahan (millisecond)	
	CPU	GPU
Nursery	33.868	36.441
Car Evaluation	1.076	1.139
Balance Scale	31	31

Pada tabel 1 dapat disimpulkan bahwa GPU belum dapat memproses data lebih cepat dibanding dengan CPU. Hal ini disebabkan karena hanya menggunakan 1 thread dari setiap block. Sehingga harus menyalin data uji dari host ke device secara berulang sebanyak jumlah data uji. Menyalin data dari host ke device dibutuhkan waktu yang cukup lama, sehingga pada penelitian selanjutnya, akan lebih baik untuk menggunakan thread 2 dimensi. Dengan menggunakan thread 2 dimensi, maka proses penyalinan data dari host ke device akan lebih cepat, dan data yang diolah secara parallel juga semakin mempersingkat waktu yang dibutuhkan untuk pengolahan data.