NOISE MINING USING MODIFIED SHARED NEAREST NEIGHBORS ALGORITHM

Rifki Fahrial Zainal

Lecturer, Department of Informatics Engineering, University of Bhayangkara Surabaya

Jl. Ahmad Yani 114 Surabaya

ABSTRACT

Removing objects that are noise is an important goal of data cleaning as noise hinders most types of data analysis. Most existing data cleaning methods focus on removing noise that is the result of low-level data errors that result from an imperfect data collection process, but data objects that are irrelevant or only weakly relevant can also significantly hinder data analysis. One of the way to enhance the data analysis as much as possible, is finding and removing the right noise data. Consequently, if the attributes for the noise can be found, a new and better way to remove the noise in a large data set can be applicated.

1. INTRODUCTION

Noise as described in Oxford Dictionary is “Random Fluctuations that obscure or do not contain meaningful data or other information”. The term has often been used as a synonym for corrupt data. For most existing data cleaning methods, the focus is on the detection and removal of noise (low-level data errors) that is the result of an imperfect data collection process. However, its meaning has expanded to include any data that cannot be understood and interpreted correctly by machines, such as unstructured text. Any data that has been received, stored, or changed in such a manner that it cannot be read or used by the program that originally created it can be described as noisy.

The need to address this type of noise is clear as it is detrimental to almost any kind of data analysis. However, ordinary data object that are irrelevant or only weakly relevant to a particular data analysis can also significantly hinder the data analysis, and thus these objects should also be considered as noise, at least in the context of a specific analysis. For instance, in document data sets that consist of news stories, there are many stories that are only weakly related to the other news stories. If the goal is to use clustering to find the strong topics in a set of documents, then the analysis will suffer unless irrelevant and weakly relevant documents can be eliminated. Consequently, there is a need for data cleaning techniques that remove both types of noise.

In some cases, the amount of noise in a data set is relatively small. For example, it has been claimed that field error rates for business are typically around 5% or less if an organization specifically takes measures to avoid data errors [24]. However, in other cases, the amount of noise can be large. For example, a significant number of false-positive protein interactions are present in current experimental data for protein complexes. Gavin *et al.* [9] estimates that more than 30% of the protein interactions they detect may be spurious, as inferred from duplicate analyses of 13 purified protein complexes. Although this is an example of a data set that has a large amount of noise due to data collection errors, the amount of noise due to irrelevant data objects can also be large. Examples include the document sets mentioned earlier [7] and Web data [25], [26]. Therefore, data cleaning techniques for the enhancement of data analysis also need to be able to discard a potentially large fraction of the data.

Noisy data can give a lot of problems, but the main problems with noisy data are:

1. Noisy data unnecessarily increases the amount of storage space required and can also adversely affect the results of any data mining analysis. Statistical analysis can use information gleaned from historical data to weed out noisy data and facilitate data mining.
2. Noisy data can be caused by hardware failures, programming errors and gibberish input from speech or optical character recognition programs.
3. Spelling errors, industry abbreviations and slang can also delay in machine reading.

Schema-related data quality problems occur because of the lack of appropriate model-specific or application specific integrity constraints. For examples, due to data model limitations of poor schema design, or because only a few integrity constraints were defined to limit the overheard for integrity control. Instance-specific or application specific problems relate to errors and inconsistencies that cannot be prevented at the schema level. Table 1 and Table 2 shows the single source problem at schema level and single source problems at instance level.

Table 1. Examples for Single-Source Problems at Schema Level

Scope/Problems		**Noise**	**Reasons/Remarks**
Attribute	Illegal Values	Bdate = 01.09.78	Values outside the domain range
Record	Violated Attribute Dependencies	Age=22, bdate=01.09.78	Age = current year-birth year should hold
Record Type	Uniqueness Violation	Emp1=(name="Rifki F", ID="123456") Emp2=(name="Hartatik S", ID="123456"	Uniqueness for ID violated
Source	Referential Integrity Violation	Emp=(name="Rifki F", ID_Dept=127)	Referenced Departement (127) not defined

Table 2. Examples for Single-Source Problems at Instance Level

Scope/Problems		**Noise**	**Reasons/Remarks**
Atribute	Missing Values	Phone=9999-9999	Unavailable values during data entry(dummy values or null)
	Misspelling	City="Sursbaya"	Usually typos, phonetic errors
	Cryptic values, Abbrevations	Experiences="B" Occupation="DB Prog"	
	Embedded Values	Name="Rifki F 01.09.78 Surabaya"	Multiple values entered in one attribute
	Misfielded Values	City="Surabaya"	
Record	Violated Attribute Dependencies	City="Surabaya", zip=101200	City and zip code should correspond
Record Type	Word Transpositions	Name1="Rifki F", name2="Hartatik S"	Usually in free form field
	Duplicated	Emp1=(name="Rifki F") Emp2=(name="Rifki F")	Same employee represented twice due to some data entry errors

When multiple sources are integrated, the problems present in single source are aggregated. Each source may contain dirty and inconsistent data and the data in the sources may be represented differently, overlap or contradict. This is because the sources are typically developed, deployed and maintained independently to serve specific needs. This results in a large degree of heterogeneity with data management systems, data models, schema designs and the actual data. Tabel 3 shows the multisource problems.

Table 3. Examples of Multisource Problems at Schema & Instance Level
Data Source 1

CID	Name	Street	City	Sex
11	Rifki Fahrial	Jemursari VI/26	Surabaya	1
24	Hartatik Sri	Kutisari Selatan 2/14	Surabaya	0

Data Source 2

CNO	LastName	FirstName	Gender	Address	Phone/Fax
24	Rahayu	Rejeki	F	Rungkut Sier 16 Surabaya	333-222-6542
493	Fahrial	Rifki	M	Jemursari VI/26 Surabaya	444-555-6666

Integrated Target with Cleaned Data

No	LName	FName	Gender	Street	City	Phone	CID	CNO
1	Fahrial	Rifki	M	Jemursari VI/26	Surabaya	444-555-666	11	493
2	Sri	Hartatik	F	Kutisari Selatan 2/14	Surabaya		24	
3	Rahayu	Rejeki	F	Rungkut Sier 16	Surabaya	333-222-6542		24

Data cleaning routines is to clean the data by filling in missing values, smoothing noisy data, identifying or removing outliers, and resolving inconsistencies. Dirty data can cause confusion for the mining procedure, resulting in unreliable and poor output. There is necessity for useful preprocessing step to be used some data-cleaning routines, specially the missing values. The missing values are to be corrected by following measures

a) Ignore the missing values
b) Fill in the missing values manually
c) Use a global constant to fill in the missing values
d) Use the attribute mean to fill in the missing values
e) Use the most probable value to fill in the missing values.

The missing values can be ignored if the class label is missing. This method is not very effective, unless the missing values contains several attributes with missing values. It is especially poor when the percentage of missing values per attribute varies considerably.

The second approach is time consuming and may not be feasible to the large data set with many missing values. To shorten the time of the process, the missing value in the data sets are filled by using a global constant in the missing value and replace all missing attribute values by the same constant. The other way is to fill the missing values with the attribute mean. The last approach, which is using the most probable value to fill in the missing value may be determined with regression, inference-based tools using a Bayesian formalism, or decision tree induction.

For example, using the other attributes in the soil data set, we may construct a decision tree to predict the missing values for income. This is a popular strategy. In comparison to the other methods, it uses the most information from the present data to predict missing values. This procedure is bias the data. The filled-in value may not be correct. It is important to have a good design of databases or data entry procedures which would minimize the number of missing values or errors.

On the other hand, noisy data or data that contains errors can be corrected with Binning, Regression or Clustering. Binning methods have the tendency to smooth a sorted data by consulting its “neighborhood”, that is, the data around it. The sorted data are distributed into a number of “buckets” or bins. Because binning methods consult the neighborhood of values, they perform local smoothing. In smoothing by bin means each value in a bin is replace by the mean value of the bin.

Smoothing by bin medians can be employed, in which each bin value is replaced by the bin median. In smoothing by bin boundaries, the minimum and maximum values in a given bin are identified as the bin boundaries. Each bin value is then replaced by the closest boundary value. In general, the larger the width, the greater the effect of the smoothing. Alternatively, bins may be equal-width, where the interval range of values in each bin is constant. Binning is also used as a discretization technique.

The Regression methods smoothed the data by fitting the data to a function. Linear regression involves finding the “best” line to fit two attributes (or variables), so that one attribute can be used to predict the other. Multiple linear regression is an extension of linear regression where more than two attributes are involved and the data are fit to a multidimensional surface.

The last methods is by using clustering methods. Outliers or noise may be detected by clustering where similar values are organized into groups or clusters. Intuitively, values that fail outside of the set of clusters may be considered outliers or noise. This method can be used not only to find class for each of the data in the data sets, but also able to clean the noise in its process. But the process of finding the noise among one big data set, is a long and time-consuming process.

In this paper, we propose a way to use the clustering methods to find the noise and examining it, thus finding the attributes for the noise. Then by using the noise attribute, we can do several things such as avoiding filling in the missing values with the noise values, or cleaning the data set with the right attribute for the noise.

2. METHODOLOGY

There are several clustering methods that can be used to find the noise. This paper use the Shared Nearest Neighbors (SNN). SNN is a density based clustering algorithm which is capable of finding clusters of arbitrary shapes, sizes and densities and we need not mention the number of clusters as parameter. This algorithm makes use of a similarity measure which is obtained from the number of neighbors two points share. This can be computed from the k-nearest neighbors of each point. In order to identify the k-nearest neighbors, we need a distance function like Euclidean distance.

2.1 SNN Algorithm

This algorithm requires the following input:

a. K: number of nearest neighbors to be identified for each point.
b. Eps: Density threshold-minimum number of points shared by two points in order to be considered close to each other.
c. MinPts: minimum density a point should have to be considered a core point.

Algorithm includes the following steps:

1) Create the distance matrix using a given distance function and identify for each point, the k nearest neighbors.
2) For each two points, calculate the similarity, which is given by the number of shared neighbors.
3) Establish the SNN density of each point. The SNN density is given by the number of nearest neighbors that share Eps or more neighbors.
4) Identify the core points of the data set. Each point that has a SNN density greater or equal to MinPts is considered a core point.
5) Build clusters from core points. Two core points are allocated to the same cluster if the share Eps or more neighbors with each other.
6) Handle noise points. Points not classified as core points and that are not within Eps of a core point are considered noise.
7) Assign the remaining point to cluster. All non-core and non-noise points are assigned to the nearest cluster.

The inmportant step in the SNN algorithm is identification of k-nearest neighbors. In order to identify the nearest neighbors, we make use of a distance function. Various distance functions that could be applicable to numerical data are:

1) Euclidean Distance:
   It is the straight-line distance between two points. It computes the root of squares difference between coordinates of pair of objects.

$$Dist_{xy} = \sqrt{\sum_{i=1}^{n}(X_i - Y_i)^2} \quad (1)$$

2) Manhattan Distance:
   Manhattan Distance computes the absolute differences between coordinates of pair of objects.

$$Dist_{xy} = \sum_{i=1}^{n}((|X_i - Y_i|)) \quad (2)$$

3) Minkowski Distance:
   Minkowski Distance can be defined as the generalized metric distance. It is formulated as

$$Dist_{xy} = \sum_{i=1}^{n}((|X_i - Y_i|)^p)^{\frac{1}{P}} \quad (3)$$

When p=2, it becomes Euclidean Distance.

The algorithm has been implemented by using the above measures in order to obtain the efficient distance measure. It is found that using Manhattan as distance measures, the algorithm doesn't perform well. Using Euclidean it has produced the effective results. Since Minkowski also behaves like Euclidean, it also has produced the same results. Therefore, we have implemented the algorithm by using Euclidean distance in this paper.

In the first step, we create a distance matrix and identify the k-nearest neighbors of each point. Then for each two points, we calculate similarity i.e. number of nearest neighbors two points share. Then SNN density is computed. The points whose SNN density is greater than MinPts are considered as core points. All core points that share Eps or more neighbors are allocated to the same cluster and continue to be core points. Then we start clustering remaining points with the help of core points.

2.2 Modified SNN Algorithm

For the noise mining, we need data considered as the noise in the data set. In this paper, we propose a modified SNN algorithm by using only 1 to 4 steps in the original SNN algorithm. After we form and compute the SNN density, we can find the noise data by using the MinPts. For each points that have SNN density greater than MinPts are considered core points. While the points not classified as core points and that are not within Eps of a core point are considered noise.

This noise data than can be used as the core points for our noise mining or to avoid confusion now called the noise core points. We modified the next step whereas the original is build cluster from core points, we use the noise core points as the core points for the clusters. Two noise core points are allocated to the same cluster if they share Eps or more neighbors with each other. This modified SNN algorithm gives us a fast solution to perform analysis for the noise data.

The modified Algorithm includes the following steps:

1) Create the distance matrix using a given distance function and identify for each point, the k nearest neighbors.
2) For each two points, calculate the similarity, which is given by the number of shared neighbors.
3) Establish the SNN density of each point. The SNN density is given by the number of nearest neighbors that share Eps or more neighbors.
4) Identify the noise core points of the data set. Each point that has a SNN density lower to MinPts is considered a noise core point.
5) Build noise clusters from noise core points. Two noise core points are allocated to the same cluster if the share Eps or more neighbors with each other.

For each noise cluster can gives us a new point of view to see the bigger pictures. It can reveal a new perspective. Thus, opening the way of noise mining.

3. CONCLUSION

The modified SNN algorithm can gives a new way to perform noise mining. By using the result of noise mining, we can find the noise attributes to help us cleaning the data. Another way of using noise mining is to help providing the values of the missing values. In this case, is the value that should not be given to the missing values. Still need a further research to find the full ability of the noise mining.

REFERENCES

[1] Aggrawal, R, Imielinski, T and Swami, A. (1993), *Mining Association Rules between Sets of Items in Large Databases*, In Proc. Of the ACM SIGMOD.

[2] Angiulli, F and Pizzuti, C, (2002), *Fast Outlier Detection in High Dimensional Spaces*, In Proceedings of the Sixth European Conference on the Principles of Data Mining and Knowledge Discovery.

[3] Bay, Stephen D, and Schwabacher, M (2003), *Mining Distance-based Outliers in Near Linear Time with Randomization and A Simple Pruning Rule*, In KDD '03: Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 29-38, New York, NY, USA, ACM Press

[4] Breunig, Markus M, Kriegel, Hans-Peter, Ng, Raymond T and Sander, Jorg, (2002) *Lof: Identifying Density Based Local Outliers*, In Proc. of the 2000 ACM SIGMOD International Conference on Management of Data.

[5] Brodley, Carla E and Friedl, Mark A., (1999), *Identifying Mislabeled Training Data*, Journal of Artificial Intelligence Research, 11:131-167.

[6] Eisen, Michael B, Spellman, Paul T, Browndagger, Patrick O, and Botstein, David, (1998), *Cluster Analysis and Display of Genome-wide Expression Patterns*, Proceeding of the National Academy of Sciences of the United States of America (PNAS).

[7] Ertoz, L, Steinbach, M and Kumar, V, (2003), *Finding Clusters of Different Sizes, Shapes and Densities in Noisy, High Dimensional Data*, In Proceedings of Third SIAM International Conference on Data Mining, San Francisco, CA, USA.

[8] Ester, M, Kriegel, H. P, Sander, J and Xu, X, (1996), *A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise*, In Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining.

[9] Gavin, A et al, (2002), *Functional Organization of the Yeast Proteome by Systematic Analysis of Protein Complexes*, Nature, 415:141-147.

[10] Gaede, V and Gunther, O, (1998), *Multidimensional Access Methods*, ACM Computing Surveys, 30(2): 170-231.

[11] Galhardas, H, Florescu, D, S, and Simon, E, (2000), *Ajax: An Extensible Data Cleaning Tool*, In Proceedings of the ACM SIGMOD International Conference on Management of Data.

[12] Galhardas, H, Florescu, D, S, Simon, E and Saita, C (2001), *Declarative Data Cleaning: Language, Model and Algorithms*, In Proceedings of the 2001 Very Large Data Bases (VLDB) Conference.

[13] Guha, S, Rastogi, R and Shim, K, (1998), *Cure: An Efficient Clustering Algorithm for Large Databases*, In Laurea M. Haas and Ashutosh Tiwary, editors, SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA, pages 73-84, ACM Press.

[14] Han, E, Boley, D, Gini, M, Gross, R, Hastings, K, Karypis, G, Kumar, V, Mobasher, B, and Moore, J, (1998), *Webace: A Web Agent for Document Categorization and Exploration*, In Proc. Of the 2nd International Conference on Autonomous Agents.

[15] Hernandez, M, and Stolfo, S, (1995), *The Merge/Purge Problem for Large Databases*, In Proceedings of the ACM SIGMOD International Conference on Management of Data, pages 127-138.

[16] Hernandez, M and Stolfo, S (1998), *Real Word Data is Dirty: Data Cleansing and the Merge/Purge Problem*, Data Mining and Knowledge Discovery, 2:9-37.

[17] Hodge, V.J, and Austin, J, (2004), *A Survey of Outlier Detection Methodologies*, Artificial Intelligent Review, 22:85-126.

[18] Jain, A.K, and Dubes, R.C, (1988), *Algorithms for Clustering Data*, Prentice Hall Advanced Reference Series, Prentice Hall, Englewood Cliffs, New Jersey.

[19] Karypis, G, (2006), *Cluto: Software for Clustering High Dimensional Dataset*, Data Mining and Knowledge Discovery, 3:12-16.

[20] Knorr, E.M, Ng, R.T, and Tucakov, V, (2000), *Distance-based Outliers: Algorithms and Applications*, VLDB Journal: Very Large Databases, 8:237-253.

[21] Kohavi, R, and John, G.H, (1997), *Wrappers for Feature Subset Selection*, Artificial Intelligence, 97(1-2):273-324.

[22] Larsen, B, and Aone, C., (1999), *Fast and Effective Text Mining Using Linear-Time Document Clustering*, In Proceedings of the fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[23] Lee, M.L, Ling, T.W, and Low, W.L, (2000), *Intellclean: A Knowledge-based Intelligent Data Cleaner*, In Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

[24] Monge, A.E, and Elkan, C.P, (1997), *An Effiecient Domain-Independent Algorithm for Detecting Approximately Duplicate Database Record*, In Proc. of the ACM-SIGMOD Workshop on Research Issues on Knowledge Discovery and Data Mining

[25] Yang, Y, (1995), *Noise Reduction in a Statistical Approach to Text Categorization*, In Edward A, Fox, Peter Ingwersen, and Raya Fidel, editors, SIGIR, pages 256-263, ACM Press.

[26] Yi, L, Liu, B, and Li, X, (2003), *Eliminating Noisy Information in Web Pages for Data Mining*, in Lise Getoor, Ted E, Senator, Pedro Domingos, and Christos Faloutsos, editors, KDD, pages 296-305.