Using Autoencoder to Generate Data Quality Tests

Hajar Homayouni, Sudipto Ghosh, Indrakshi Ray
Department of Computer Science
Colorado State University
{hhajar, ghosh, iray}@colostate.edu

Abstract

Data quality tests check the properties of data stored in databases and data warehouses to detect violations of syntactic and semantic constraints. Domain experts define the constraints based on the needs of the stakeholders and knowledge of the application domain. Approaches that can automatically generate the constraints on data without requiring domain knowledge are lacking. We propose an approach that uses unsupervised autoencoder to (1) discover the constraints in data records that must be satisfied, and (2) classify the records as valid or invalid based on these constraints. We evaluate our approach using records from a health data warehouse.

Keywords: data quality tests, data warehouse, database, software testing, unsupervised learning, autoencoder

1 Introduction

Enterprises use data stores such as databases and data warehouses to store, manage, access, and query their data. While a database focuses on data transactions, a data warehouse accumulates data from multiple sources for analysis and research. Data stores use a data model to describe the data attributes and their relationships. The quality and correctness of the data are critical to every enterprise.

Data records stored in data stores may become invalid because of how the data is collected, transformed, and managed. Invalid data can violate constraints present in the data model and in the application domain. For example, a health data store may contain records with a non-numeric value for the *Weight* attribute and a *Pregnancy* value for a *Male* patient. Data quality tests are required to validate the syntactic and semantic properties of the data to ensure that data model and domain-specific constraints are not violated.

Existing data quality tools typically check for violations of syntactic and semantic constraints in a specific application domain. Tools that can be used across domains are generally intended only for checking syntactic constraints, such as not-null and uniqueness checks. Domain-specific constraints are defined and periodically updated by domain experts, who can miss important constraints.

The key idea of our approach is to automatically discover domain-specific patterns from the data to be used to define new constraints for an application domain. We use an unsupervised [1] machine learning approach called autoencoder [2] to learn hidden patterns in the attributes of the unlabeled data records (i.e., records whose validity is not known in advance). This technique can model both linear and complex non-linear relationships between attributes of the data without having any prior knowledge about the data. The new patterns are used to generate data quality tests that validate the rest of the data in the data store.

We evaluate the fault detection effectiveness of our approach by comparing the faults detected by our approach with those detected by the existing data quality test tools that are created by the domain experts. We demonstrate that our approach can detect (1) the faults that were already detected by the existing tools and also (2) new faults that were not previously detected by the existing tools.

2 Related Work

Existing data quality test approaches usually address a domain specific database or data warehouse project and are not generic enough to be applicable to other projects [3].

Measurement to Understand the Reclassification of Cabarrus/Kannapolis (MURDOCK) [4] and Achilles [5] are two approaches that generate domain specific data quality tests for health data. These tools define a set of tests as queries that are written to check the properties specified by domain experts using mathematical formulas or natural language. The process of generating tests to check the inconsistently specified properties requires manual effort, which makes it difficult to update the tests when requirements evolve.

Informatica [6] is a data quality validation tool that is applicable to all domains. The tool provides a set of data quality checks as queries to validate the syntactic properties of the target data, such as data type and not-null constraint checks. The tool allows users to specify semantic properties to be verified by the tests. Although this tool is general enough for use in any project, it still requires domain knowledge to define and update the domain specific properties. There is a lack of tools that automatically generate the properties to be checked by the data quality tests.

Unsupervised learning techniques have been used in the literature to investigate hidden properties in the attributes of the data without having any prior knowledge about the data [7]. Clustering is an unsupervised technique that has been widely used to investigate properties of the data through grouping similar data into several categories [8]. The similarity of the records are measured using distance functions, such as Euclidean [9] and Manhattan [10] distances. Distance-based clustering algorithms cannot derive the complex non-linear relationships that exist among attributes of the data in their clusters [11], which is a problem in real-world applications, where non-linear associations are prevalent among attributes of the data [12].

Representation learning is an unsupervised learning technique that investigates hidden associations in data attributes by capturing a representation of the attributes present in the data. Principal Component Analysis (PCA) [13] is a representation learning approach that investigates the relationships among the data attributes by converting a set of correlated attributes into a set of linearly uncorrelated attributes called principal components. The PCA representation learning can only investigate the linear relationships among the attributes.

Autoencoder is another representation learning approach that investigates an efficient encoding from the data in an unsupervised manner. Autoencoder is a type of neural network [14] that efficiently models complex associations among attributes of the data through the composition of several layers of nonlinearity [15]. This network (1) compresses the data from the input layer into a short representation and (2) decompresses that representation into a new representation that closely matches the original data. This technique can detect both linear and complex non-linear associations among the data attributes.

3 Proposed Approach

Our approach uses an unsupervised autoencoder that investigates complex patterns in the attributes of the data without having any prior knowledge about the data. We generate a test script that detects invalid records based on these patterns. Figure 1 shows the architecture of our data quality test approach with two components; these are (1) data quality pattern discovery module and (2) test script generator. The first component uses the data stored in a database or data warehouse to investigate the underlying patterns, such as linear and non-linear associations among the attributes of the data. This component trains an unsupervised autoencoder model to best fit the input data. The second component produces a test script that uses the trained model to determine whether or not the data records conform to the discovered patterns. The test script reports as output the IDs of the invalid records. We describe each component below.

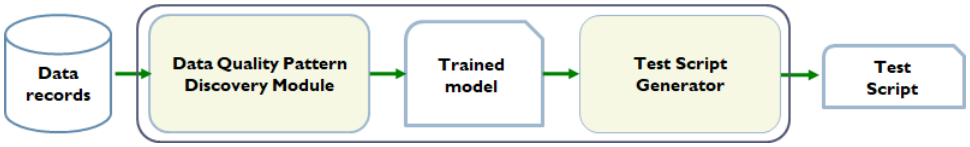

Figure 1: Data Quality Test Approach Architecture

Data quality pattern discovery module. This component retrieves the data records from the data store, preprocesses the data to convert it into a format that is feasible for analysis, and trains an autoencoder to best fit the data. An autoencoder is composed of two parts, namely *encoder* and *decoder* as shown in Figure 2. The encoder maps the input record X with d attributes into a hidden representation Z with d' attributes,

which are non-linear combinations of the input attributes. The encoder discovers non-linear associations among attributes of data. The decoder maps back the hidden representation Z to an output record Y with d attributes, where Y is a reconstruction of X [16]. The objective of the decoder is to determine whether or not the new representation Z has captured useful patterns in the attributes of the input data record. For this purpose, the autoencoder is trained to minimize the reconstruction error, which is the squared distance between the original data and its reconstruction [17].

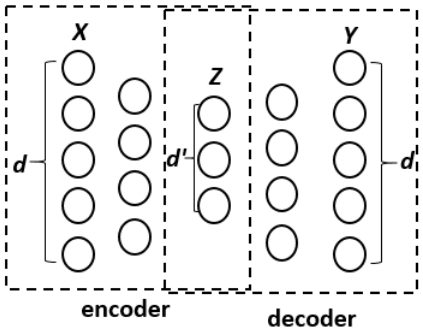

Figure 2: Structure of an Autoencoder [16]

The output of the data quality pattern discovery module is a trained autoencoder model with its parameters learned to best describe the patterns in the input data records.

Test script generator. This component automatically generates a test script that detects faulty records using the following steps:

1. *Reconstruct the data records*: This step reconstructs each data record using the trained autoencoder model to determine whether or not the records can be correctly reconstructed using the patterns discovered by the autoencoder. For this purpose, the step calculates for each record an invalidity score that is equal to the reconstruction error of the record.

2. *Detect invalid records*: The objective is to detect invalid records that do not conform to the patterns discovered by the autoencoder. This step flags as faulty those records whose invalidity score is greater than the threshold.

4 Demonstration and Evaluation

We evaluated the fault detection effectiveness of our approach by comparing the faults detected by our approach with those detected by the Achilles and MURDOCK tools to demonstrate that our approach can detect (1) the faults that were already detected by the existing tools and also (2) new faults that were not previously detected by the existing tools.

We evaluated our approach on four data sets from a health data warehouse. This data warehouse integrates patient clinical data from hospitals into a single target data store to support medical research on diseases, drugs, and treatments. The data sets involved 94,165, 600,000, 600,000, and 1,000,000 patient records from joins of multiple tables in the data warehouse.

Given A, the set of faulty records detected by an existing data quality test approach, and E, the set of faulty records detected by our approach, we define the *Previously Detected* (PD), *Newly Detected* (ND), and *UnDetected* (UD) metrics to evaluate our approach:

PD: Percentage of faulty records detected by the existing data quality test approach that could also be detected by our approach.

$$PD = \frac{|A \cap E|}{|A|} \tag{1}$$

ND: Percentage of detected faulty records by our approach that were not previously detected.

$$ND = \frac{|E - A|}{|E|} \tag{2}$$

UD: Percentage of faulty records detected by the existing approach that could not be detected by our approach.

$$UD = \frac{|A - E|}{|A|} \tag{3}$$

Table 1 shows the previously detected, newly detected, and undetected faults by our data quality test approach with respect to the faults detected by Achilles and MURDOCK in the four data sets under test. It took two minutes to generate and execute our test script against each dataset. Our approach could detect between 96.14% and 100% of faults that were previously detected by the existing data quality test approaches for the four data sets. A number of faults detected by our approach (between 0% to 16.75%) were faults not previously detected; these are suspicious records that were missed by the domain experts. Our approach could not detect a maximum of 3.86% of faults that were previously detected by the existing approaches. This indicates that autoencoder could not discover all of the associations among the data attributes.

Table 1: Known and New Faults Detected by Our Approach

Data Set ID	**PD**	**ND**	**UD**
1	100	0	0
2	99.99	16.75	0.01
3	97.21	4.24	2.79
4	96.14	10.63	3.86

5 Conclusions and Future Work

We developed a data quality test approach that automatically detects the patterns in the data and generates a test script to detect invalid data based on these patterns. Evaluating the new faulty records detected by our approach to demonstrate whether or not they are actually faulty requires domain specific knowledge and is the subject of our future research. We will extend our data quality test approach to detect undetected faults. We will evaluate our approach using data sets from data warehouses in different domains.

Acknowledgment

This research is supported by grants from the Anschutz Medical Campus at the University of Colorado, Denver. We would like to thank Prof. Michael Kahn for his support and feedback.

References

[1] U. R. Hodeghatta and U. Nayak, *Unsupervised Machine Learning.* Berkeley, CA: Apress, 2017, pp. 161–186.

[2] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, "Autoencoder for Words," *Neurocomputing*, vol. 139, pp. 84 – 96, 2014.

[3] N. ElGamal, A. E. Bastawissy, and G. Galal-Edeen, "Towards a Data Warehouse Testing Framework," in *9th International Conference on ICT and Knowledge Engineering*, Bangkok, Thailand, 2012, pp. 65–71.

[4] S. Bhattacharya, A. A. Dunham, M. A. Cornish, V. A. Christian, G. S. Ginsburg, J. D. Tenenbaum, M. L. Nahm, M. L. Miranda, R. M. Califf, R. J. Dolor, and L. K. Newby, "The Measurement to Understand Reclassification of Disease of Cabarrus/Kannapolis (MURDOCK) Study Community Registry and Biorepository," *American Journal of Translational Research*, vol. 4, no. 4, pp. 458–470, 2012.

[5] "OHDSI/Achilles," https://github.com/OHDSI/Achilles (Accessed 2018-05-11).

[6] "Informatica," https://www.informatica.com/ (Accessed 2018-04-14).

[7] V. Hodge and J. Austin, "A Survey of Outlier Detection Methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[8] A. Barai and L. Dey, "Outlier Detection and Removal Algorithm in K-Means and Hierarchical Clustering," *World Journal of Computer Application and Technology*, vol. 5, no. 2, pp. 24–29, 2017.

[9] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean Distance Matrices: Essential Theory, Algorithms, and Applications," *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 12–30, 2015.

[10] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the Surprising Behavior of Distance Metrics in High Dimensional Space," in *Database Theory*, J. Van den Bussche and V. Vianu, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 420–434.

[11] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications*. SIAM, Society for Industrial and Applied Mathematics, 2007.

[12] K. Wang, Q. Zhao, J. Lu, and T. Yu, "K-profiles: A Nonlinear Clustering Method for Pattern Detection in High Dimensional Data," *BioMed Research International*, vol. 2015, pp. 1–10, 2015.

[13] S. Sehgal, H. Singh, M. Agarwal, V. Bhasker, and Shantanu, "Data Analysis Using Principal Component Analysis," in *International Conference on Medical Imaging, m-Health and Emerging Communication Systems*, 2014, pp. 45–48.

[14] M. Mishra and M. Srivastava, "A view of Artificial Neural Network," in *International Conference on Advances in Engineering Technology Research*, 2014, pp. 1–3.

[15] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[16] D. Chicco, P. Sadowski, and P. Baldi, "Deep Autoencoder Neural Networks for Gene Ontology Annotation Predictions," in *5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*. New York, NY, USA: ACM, 2014, pp. 533–540.

[17] S. Malek, F. Melgani, Y. Bazi, and N. Alajlan, "Reconstructing Cloud-Contaminated Multispectral Images with Contextualized Autoencoder Neural Networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 56, no. 4, pp. 2270–2282, April 2018.