Expose

Janek Prange
February 15, 2022

1 Motivation

Big collections of data and therefore databases are becoming increasingly important as more data is produced with accelerating speed. In the best case, this data is directly saved in a format that contains a lot of metadata such as data types, primary keys and information about dependencies between columns.

In many scenarios however, it is not possible to design a proper schema suited for specific data beforehand. Instead, the data is already present in an arbitrary format or is being taken from other sources such as the internet, where the data format can rarely be influenced. These use cases require tools to – as much as possible automatically – detect the needed metadata from the raw data.

The thesis introduced in this expose tries to propose a new solution to this problem making use of machine learning.

2 Naive Procedures

One of the main characteristics of a primary key is the uniqueness of its values. Based on that constraint, a naive way of finding primary key candidates would be to search for columns or column combinations which don't have any duplicate values. This could be achieved either by sorting or hashing.

Especially when using a hashing algorithm, the primary key candidates could be found relatively efficiently. It is however necessary to read all values in all columns, which can become a very large amount of data very fast, especially if no single column candidate can be found and column combinations have to be examined too. It could be possible though to increase the efficiency by ignoring some columns, for example the ones containing images or long texts.

3 Proposed Solution

The thesis introduced in this expose will present a solution to find primary keys respectively primary key candidates. The solution will be primarily based on a neural network, which will be trained on a large set of tables. The hope is that the network will discover primary key candidates with better efficiency than the naive solution described in Section 2.

In addition to the usage of neural networks, there may be other ideas to increase the efficiency of the algorithm. One possibility would be to sample the columns of the table instead of reading all values. This way, the data that has to be retrieved from the database could be reduced which may result in a significant speedup when the algorithm is used with large datasets.

Another opportunity for increased efficiency could be the implementation of a two phase algorithm. In the first phase, columns which can't practicably be primary keys will be eliminated, such as columns containing images, videos or relatively long texts. In the second phase, the neural network would try to detect the best primary key candidates from the remaining columns.

To reduce the scope of the thesis, the focus of the algorithm will be primarily on detecting single column primary key candidates. If the results are promising and there is time left, multicolumn candidates will be included.

4 Evaluation Method and Metric

To train the neural network that is at the center of the solution proposed in Section 3, methods have to be defined to evaluate the success or failure of the network. In this case, there are two favorable and relatively easy to measure metrics, which are the correctness and the efficiency of the algorithm.

A primary key candidate is correct when it has no duplicate elements and no subset of it is a primary key. These requirements are easy to check, especially if the algorithm is limited to single column candidates.

For the purposes of this thesis, the algorithm is efficient if it is faster and uses less main memory than the naive solution presented in Section 2, which will be tested on the same data as the neural network.

5 Possible Titles

- Enhancing Primary Key Detection using Machine Learning
- Improved Primary Key Detection with Neural Networks