Rakenteiden Mekaniikka (Journal of Structural Mechanics)
Vol. 50, No 3, 2017, pp. 224 – 228
https://rakenteidenmekaniikka.journal.fi/index
https://doi.org/10.23998/rm.64942

VERTAISARVIOITU
KOLLEGIALT GRANSKAD
PEER-REVIEWED
www.tsv.fi/tunnus

Analyzing 3 TB field measurement data set

Jukka Aho[1] and Tero Frondelius

Summary. This article describes the use of intelligent algorithms for analyzing field measurement data. The main focus is on describing the general work-flow and practical issues when the amount of data is "big" and typical data analysis methods for small data cannot be used. When the amount of data is tens of terabytes, it is no longer fitting to computer memory. Data visualization is also challenging: visualization tools can only render a small fraction of data to computer screen and visual inspecting of the whole dataset is not meaningful at all. The data is simply too big. Thus, new approaches to study data are needed where the data is processed automatically in calculation clusters without manual human work. The basic idea of data mining is to gradually reduce the amount of data by various techniques, as long as the final data contains only information relevant to the research question and in such a compact form that its viewing from the human point of view is rational use of time. This article is a tutorial, ending with some opinion of the authors (technical aspect), based on their experience.

Key words: big data, data analysis, machine learning, artificial intelligence, deep learning

Received 16 June 2017. Accepted 11 August 2017. Published online 21 August 2017.

Introduction

The rapidly growing amount of data creates the need to develop new methods for processing large data volumes automatically. Data analysis should be seen as important feedback: data is acquired from the field, and it can be used in product development to make better products in future. Collecting field data makes it possible to study and learn from the product in real-life conditions which are usually totally different from the ones in a R&D laboratory. Thus, the data is essential for product development. By collecting and analyzing data, a company can achieve remarkable competitive advantages over other companies in same industry. If already delivered products start to malfunction, the only way to solve the problem and learn from it is to acquire measurement data from the field, do a data analysis and provide options for problem solving based on the analysis results. The importance of data analysis is well understood nowadays, and companies have a rapidly growing need to find data analysts and software engineers or partners to improve the company data analysis capabilities. [4, 5, 6, 9].

The data itself has been collected for a long time, but its systematic utilization in product design is often not done in companies. Many times, the problem is the amount of

[1]Corresponding author. jukka.aho@gbw.fi

data: it does not have to be very large so that its analysis with, e.g., spreadsheed program turns out to be totally impossible. It is also often the case that easy-to-use, ready-made programs for the spesific needs of a company's data do not exist. The company may need to find partners who are capable of delivering customized solutions for data analysis. Such services are provided by, for example, Global Boiler Works Oy. The company has years of experience in delivering a wide range of data analysis solutions, ranging from small desktop applications to the large browser-based analysis services capable of analysing hundreds of terabytes of data in computational clusters with thousands of calculation cores.

Basic steps of data analysis using machine learning

Next, the basic steps of data analysis are described. The focus is on cyclic measurement data, like cylinder pressure data of a running engine, but the techniques described here can be used as well in other applications. In this context, the term "data analysis" means the use of intelligent algorithms, many times referred as "machine learning algorithms", to analyze data. These methods include, for example, neural networks, deep learning, Bayesian networks, support vector machines, and many others. [7] The basic idea behind most of them is that a small part of data is first used to fit the model which then automatically processes the rest of the data.

In the first step, data is often subjected to preprocessing with the aim of converting data from a closed file format of a measurement instrument provider to an open standard. This ensures that data can be processed on a wide variety of platforms, programming languages and environments. Depending on the nature of the data, such file formats include HDF [8], XML, JSON [2], SQL [3] and so on. It is also worth keeping in mind the scalability: how data can be later transferred to a cloud computing platform if more calculation resources are needed.

In practice, a variety of events are identified from time series data. For each event, several features are calculated. A feature is usually a single numeric value describing the properties of the event. This procedure is called "feature extraction". For example, in the cylinder pressure measurement of a combustion engine, one event can be one engine cycle, and from the event it is possible to calculate features like the peak cylinder pressure, ignition time, energy, frequency components, mean cylinder pressure, and so on.

The number of extracted features can be further reduced by suitable dimension reduction techniques, in simple cases, for example, by principal component analysis. In this way, the feature space can be reduced by making suitable linear combinations of features so that the data variance is still explained in the best possible way and "insignificant" features are suppressed from the data.

After feature extraction, the data is usually reduced so much that it is possible to visualize the features. Typically this is done using scatter plots, and the aim is to recognize the basic structure of data, find outliers and find dependencies between the features. It is essential that, if an interesting feature is found, the original data can be accessed easily for manual inspection. Therefore, each sample must be accurately identified by a running number or similar so that it can be uniquely extracted from raw data at a later stage in the analysis.

Features tend to cluster in the feature space. The clusters can be found automatically by several different "unsupervised learning" methods, like k-means clustering. As a result of clustering, each event is labeled by a number which then describes to what cluster that event belongs to. For example, in a combustion engine, such clusters could be 1) startup, 2) stop, 3) partial power, 4) maximum power, 5) heavy knocking, 6) pre-ignition, 7) misfiring, 8) measurement error, and so on.

Unsupervised learning is a good starting point for further data analysis. It doesn't need training data, so it's a cost-effective way of trying to find interesting features in the data without thinking too much about what can be expected. Yet another advantage of unsupervised learning is that these classes mentioned before are not defined in advance but are learned from data in a automatic way.

In this way, unsupervised learning can spot entirely new types of defects that were not even previously suspected or known. For example, if the engine control system has a bug that causes malfunctions, it is an unknown event that cannot be known in advance, and the only way to systematically capture this new class "9) bug in engine controls system" is to use unsupervised learning techniques.

After unsupervised learning, the model is capable to classify new events automatically. In this way, it is possible to build a fully automated system that monitors the condition of the machine and automatically sends an alert to the operator if any malfunctions occur in the device.

Typically, the extracted features form continuous distributions so there is an obvious problem in what is the limit or threshold that should be considered, e.g. "knocking cylinder" versus "normal-running cylinder". It may be that the data is not so well clustered that its automatic classification using unsupervised learning would be possible. Mathematically this means that events are not clearly separable to two or more classes. In this case it is possible to use supervised learning techniques. In order to use supervised learning, a training data set must be constructed. In practice, it goes so that some of the events are manually inspected and classified on the basis of the original data. In this way we proceed for as long as there is a representative number of events in each class. Once the learning data has been collected and the algorithm is fitted using the training data, it can be used to classify new unidentified features to classes like before.

However, it is not enough that the algorithm is performing well on learning data: the decision making must be generalizable to new, unseen data. Typically, manually classified data is split to learning and testing data, and testing data is used to validate a classifier. This way it's possible to find out how accurately the algorithm can classify new samples because the right class is already known. The use of test data protects the model against the over-fitting problem.

Yet another question arises: Let's suppose that the model is fitted, i.e. the decision surface has been found. The question now arising is that, if the sample is near the decision surface, what are the probabilities that the data is classified properly? For this question, discriminant-based classifiers, like support vector machines, are not giving answers but the full posterior distribution has to be estimated, usually numerically by MCMC sampling [1]. It is also possible to combine different fields of machine learning concepts, for example hierarchically modeled Bayesian neural networks aware of the uncertainty of results.

After model fitting, the last step is to squeeze all the information provided by the model to a human-readable report for company management. Typically the model can answer questions like how many events are in a class X compared to the total number of classes, "how many knocking cycles are in this time period", or what is the mean and variance of features in class X, "what is the mean peak pressure when engine is running on full power".

Technical aspects

From a technical point of view, a large amount of data brings its own challenges to the technologies, platforms and devices used in data analysis. With the current knowledge, it would be desirable for the platform to be scaled in a sensible way as the amount of data grows. Everything should be done in "cloud-ready" way, making it possible to migrate services to cloud computing resources when necessary.

First, the file format would be desirable to be one that is well-known and readable widely on various platforms and by various programming languages. Is should support partial reading as it's not practical to read the whole data at one time into the computer memory but just slice a part of it. The reading of data should be multi-threaded, and on-the-fly compress and extract of data should be supported. Essential is to make it possible to open data not only today but also after 10 years. That's the reason why the first thing to do is to get rid of manufacturers' closed file formats, since there are no quarantees that they will open later, at least without paying license costs maybe to companies that no longer exist. And for sure they are not accessible widely with different programming languages and platforms. It is important to make sure that the data is accessible in future, and to do that, the data must be converted to some open source standard format.

One format satisfying all the above mentioned requirements is the Hierarchical Data Format (HDF). NASA is using it to store hundreds of terabytes of measurement data. Thus, quite a good proof of applicability for big data can be seen. Another option is to choose something built on top of SQL engines if the ultimate goal is to make a framework naturally living on cloud services. Some other alternatives also exist, e.g. Hadoop and its different variants. When choosing an appropriate data format, it's important not only to take into account the storage costs EUR/TB but also how the data can be accessed in future. For example, it may be practical to use a desktop application at a prototype stage of analysis, but later on it may be realized that it would be more appropriate to build a browser-based data visualization and analysis solution so that more people in the company can have access to the data and results easily.

One practical restriction for analysis performance is giving the IO bandwidth of the devices used for analysis. Expecially in the feature extraction stage, the entire data must be read in order to calculate the necessary features. If it is needed to process the whole data, e.g. 3 terabytes, in 5 minutes, the data processing speed should be 10 GB/s to achieve that. Such a requirement poses significant challenges even for the most recent calculation clusters. It is therefore meaningless whether the core count available is 100 or 10000, if the bottleneck is outside of the number of processors.

Although data is extracted from dozens of different features, the amount of data in the feature extraction is reduced to a fraction of the original. At this point there are no more challenges related to IO bandwidth of devices, but for computing power as well. Fitting the machine learning algorithm is very computationally intensive. Typically, from the mathematical point of view, the aim is to optimize the parameters of the algorithm so that it classifies new samples as effectively as possible. This is often a global optimization problem for its mathematical nature. For example, teaching the neural network is a non-convex global optimization problem. In many cases algorithms cannot be parallelized well. For example in the Bayesian methods, the new state of Markov chain depends on the previous state, so a single trace cannot be parallelized. Classifiers based on support vector machines are in the field of the convex optimization problem containing inequality constraints, and interactive methods are needed to solve the optimization problem. In the principal component analysis, an eigenvalue problem has to be solved. And so on.

Conclusions

This article gave a brief review of what practical problems are encountered during the analysis of large amounts of data and what are the basic steps of data analysis. The use of machine learning and artificial intelligence in data analysis opens entirely new dimensions to how data can be used to improve a company's competitiveness.

References

[1] Siddhartha Chib and Bradley P Carlin. On mcmc sampling in hierarchical longitudinal models. *Statistics and Computing*, 9(1):17–26, 1999.

[2] Douglas Crockford. The application/json media type for javascript object notation (json). 2006.

[3] Chris J Date and Hugh Darwen. *A Guide to the SQL Standard*, volume 3. Addison-Wesley New York, 1987.

[4] Tero Frondelius, Pasi Halla-aho, and Antti Mäntylä. Crankshaft development with virtual engine modelling. In *CIMAC Congress Helsinki*, 2016.

[5] Pasi Halla-aho, Antti Mäntylä, Tero Frondelius, Tommi Helander, and Juha Hautala. Counterweight measurements device development. *Rakenteiden Mekaniikka*, 50(3):318–322, 2017. URL https://doi.org/10.23998/rm.65050.

[6] Juho Könnö, Hannu Tienhaara, and Tero Frondelius. Wärtsilä digital design platform. *Rakenteiden Mekaniikka*, 50(3):234–238, 2017. URL https://doi.org/10.23998/rm.64621.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[8] The HDF Group. Hierarchical data format version 5, 2000-2010. URL http://www.hdfgroup.org/HDF5.

[9] Ilkka Väisänen, Antti Mäntylä, Antti Korpela, Teemu Kuivaniemi, and Tero Frondelius. Medium speed engine crankshaft analysis. *Rakenteiden Mekaniikka*, 50(3):341–344, 2017. URL https://doi.org/10.23998/rm.64916.

Jukka Aho
Global Boiler Works Oy
Lumijoentie 8
90400 Oulu
jukka.aho@gbw.fi

Tero Frondelius
Wärtsilä
Järvikatu 2-4
65100 Vaasa
tero.frondelius@wartsila.com