Deep Leaning for predictive asset maintenance - project plan

June 28, 2019

Supervisors: Tolu Ogunseye (Shell), Martin Blunt (Imperial) and Lukas Mosser (Imperial)

Background reading

Improvements in predictive asset maintenance (PAM) are in great demand in every industry that relies on heavy and expensive machinery[**1**], especially compressors on offshore platforms. With frequent readings coming in from many sensors on many machines in each pipeline, we now have more than enough data for deep learning models to be trained and optimised for the task of predicting failures in these components. Both experienced engineers and data analysts agree that such machines display anomalous behaviour before failure[**2**], but of course, constant monitoring of all such components is infeasible. The automation of making reliable, real time inferences from this time series data could enable cost effective failure prediction and prevent the long down-times and large expenses incurred in repair.

Long short-term memory based recurrent neural networks (LSTM-RNNs) have shown great results in using time series data to make inferences[**3**], with groups at Microsoft[**4**] and NASA[**5**] having undertaken projects exploring this and have even created public data sets to encourage experimentation in this direction. With this, it's quite likely that a similar project, albeit on a different data set would also show good results.

Introduction

From a high-level view, the project objectives are as such:
- Develop a model that can use time series data to predict when/if an anomaly will occur in a given machine
- Use this model and a generative model to generate time series data corresponding to anomalous behaviour

With an abundance of time series data, this project will initially aim to train a LSTM-RNN model to predict upcoming component failure based on 'recent' data. This will involve large volumes of data from sensors on active Shell assets, with failures labelled.

The second part of this project will aim to use a model that has been optimised to distinguish anomalous and normal behaviour, in conjunction with a similar, generative adversarial model (GAN) to produce time series that both the previous model, and experienced engineers and analysts agree is anomalous. Once the generative model can be further optimised to produce anomalous behaviour that is also physically meaningful in the context of what could possibly happen to these machines, then as well as predicting failure, our models will also be capable of generating data that, if the machines were to display, would lead to failure.

A non-exhaustive list of milestones is:

- Gain access to the full data-set, as well as computing resources and internal repositories for this project. - Trivial, < 1 day
- Explore and assess the data, cleaning and doing any pre-processing that may be model independent - 2 weeks (03/06 to 14/06)

- Create a problem statement based on the intended approach, and then label/categorise the data to model the problem - 1 week (17/06 to 21/06)
- Develop a model and infrastructure to train and assess the it, and to feed the data and gather inferences from it - 1 week (24/06 to 28/06)
- Optimise the model until reliable inferences can be made for the solution of the problem defined earlier - 1 week (01/07 to 05/07)
- Define a new problem statement for the purpose of generating data that previous models would classify as anomalous - 1 week (08/07 to 12/07)
- Develop models and infrastructure to train and assess them - 2 weeks (15/07 to 26/07)
- Repeat the process of optimising/changing the model until the generated time series match the requirements of both the Shell supervisor and engineers who specialise in the machines the data comes from - 3 weeks (29/07 to 16/08)
- Project administration and finishing write up, finalising repository and reviews from Imperial supervisors before submission, and cleaning code-base - 2 weeks (19/08 to 30/08)

The dates provided are subject to change if there are any issues at any stage, but the milestones defined will still be met. Weekly progress updates will be carried out at Shell offices with the Shell supervisor, and less frequent updates and reviews of the project write ups will be carried out remotely with the Imperial supervisors.

Security and access

The project will be carried out at Shell offices, and hence the data, code base and any hardware used will be stored either locally at the office or on the private repository created for the project, accessible only by those involved with the project. The data will only be copied from the shared access folders on the cloud to the local machine, where it will not be distributed any further.

Upon the signing of confidentiality agreements by the relevant Imperial staff, the code base and any written reports by the student will be available for assessment and review.

0.0.1 Techincal details

The main tools used here will be from the PyTorch library**[6]**, making use of the optimised and highly customisable Recurrent neural network models available. PyTorch was chosen as the main software for the project as it is not only more familiar, but better suited to handling variable length time series inputs than similar deep learning libraries such as TensorFlow.

Data exploration and pre-processing will mainly consist of looking at what is available, and learning some context around what each of the features mean. This will also involve some reading on the relevant machines such as compressors. Missing data and bad features will have to be accounted for and algorithms to detect and remove/interpolate these will be developed.

The rest of the code such as supplementary functions and infrastructure around the model to enable it and assess it will be written in Python, and the development environment will be both Juypyter notebooks and an appropriate IDE.

References

[**1**]: Sean otto, How Predictive Maintenance is Improving Asset Effici
ency - 12/12/2018 - Available from: https://www.machinedesign.c om/industrial-automation/how-predictive-maintenance-improving-asset-efficiency - Accessed on 24/06/2019.
[**2**]: Machine learning for predictive maintenance, where to start? - 29/08/2017 - Available from: https://medi um.com/bigda tarepublic/machine-learning-for-predictive-maintenance-where-to-start-5f3b7586acfb - Accesse d on 24/06/2019.
[**3**]: Jakob Aungiers, Time series prediction using LSTM Neural networks - 15/09/2018 - Available from: https://www.altumin
telligence.com /articles/a/Time-Series-Prediction-Using-LSTM-Deep-Neural-Networks - Accessed on 24/06

/2019.
[**4**]: Fidan Boylu Uz, Deep learning for predictive maintenance with Long Short Term Memory Networks - 21/06/17 - Available from: https://azure.microsoft.com/en-gb/blog/deep-learning-for-predictive-maintenance/ - Accessed on 24/06/2019.
[**5**]: Nikunj Oza (publisher), Turbofan engine degradation simulation data set - 26/06/2018 - Available from: https://data.nasa.gov/widgets/vrks-gjie - Accessed on 24/06/2019.
[**6**]: Facebook Reasearch, PyTorch - https://pytorch.org/ - Accessed on 27/06/2019.