On the Role of Data Engineering Decisions in AI-Based Applications

Aurek Chattopadhyay[a], Matthew Van Doren[b], Reese Johnson[b] and Nan Niu[a]

[a]University of Cincinnati, USA
[b]Metropolitan Sewer District of Greater Cincinnati, USA

Abstract

Artificial Intelligence (AI) and Deep Learning (DL) solutions are becoming increasingly popular in our daily life. However, one of the main concerns that stakeholders face is the lack of accountability for the inevitable errors made by AI. In this paper, we build on the emerging research of developing DL solutions to predict combined sewer overflows, and present our vision of linking data engineering decisions to key requirements such as predictive accuracy and explainability.

Keywords

data and requirements engineering, deep learning, explainability

1. Introduction

Artificial Intelligence (AI) has become an increasingly incorporated part of many decisions that we take in our daily life. Recent advancements have made AI an appealing tool for improving the efficiency of manual work [1]. The unstoppable penetration of AI also reaches into the public sector. Although AI mistakes are inevitable, the lack of *explainability* raises significant concerns from the citizens and public organizations about AI-based decision making's accountability, fairness, responsibility, and transparency.

In our recent work [2], we reported a case study of using Deep Learning (DL) solutions to predict combined sewer overflow events for a municipal wastewater treatment organization. In particular, we built a Long Short Term Memory (LSTM) model by taking different time-series data, such as flow and velocity collected from the sensor networks, as well as the rainfall measures obtained from a weather service.

A crucial issue emerged from our case study is to what extent data engineering decisions influence DL solution's performances. Specifically, the data of our LSTM model are available in different timestamps, e.g., the sensor network data are collected in every five minutes

In: J. Fischbach, N. Condori-Fernández, J. Doerr, M. Ruiz, J.-P. Steghöfer, L. Pasquale, A. Zisman, R. Guizzardi, J. Horkoff, A. Perini, A. Susi, M. Daneva, A. Herrmann, K. Schneider, P. Mennig, F. Dalpiaz, D. Dell'Anna, S. Kopczyńska, L. Montgomery, A. G. Darby, and P. Sawyer (eds.): Joint Proceedings of REFSQ-2022 Workshops, Doctoral Symposium, and Poster & Tools Track, Birmingham, UK, 21-03-2022, published at http://ceur-ws.org/
Corresponding author: A. Chattopadhyay
chattoak@mail.uc.edu (A. Chattopadhyay); matthew.vandoren@cincinnati-oh.gov (M. V. Doren); reese.johnson@cincinnati-oh.gov (R. Johnson); nan.niu@uc.edu (N. Niu)
https://homepages.uc.edu/~niunn (N. Niu)
0000-0001-5566-2368 (N. Niu)

whereas the rainfall measures are stored in every minute. For a DL solution like LSTM to work, synchronizing the heterogeneous data is necessary. However, different data engineering decisions can be made to achieve synchronization. One can take the mean from a five-minute span of the rainfall measures to align with the sensor network data, yet others may regard the median, mode, minimum, or maximum rainfall values as a plausible data engineering decision to make. Clearly, different decisions can be made but whether or not they impact the LSTM's working is less understood.

In this paper, we concentrate on the influence of data engineering decisions on DL solution's performances by focusing on LSTM's predictive accuracy and explainability. We anticipate this detailed investigation to pave ways for integrating data as an integral component into requirements engineering of AI-based applications. In what follows, we introduce the background of the combined sewer overflows and our ongoing research of developing DL solutions in Section 2. We then present our study design and discuss the study results in Section 3. Finally, we draw some concluding remarks in Section 4.

2. Background and Related Work

Combined sewer overflows are significant problems affecting human and environmental health. For instance, nearly 860 cities and towns across the U.S. have combined sewer systems, which manage stormwater as well as wastewater, creating what the U.S. Environmental Protection Agency (EPA) considers to be the largest unaddressed risk to human health from the water infrastructure.

We recently began to collaborate with a regional wastewater treatment organization, Metropolitan Sewer District of Greater Cincinnati (MSDGC) [3]. MSDGC services an operating area of about 300 square miles, over 850,000 customers, and over 3,000 miles of combined sewers. Our stakeholder organization has set up a large scale sensor network to collect data and remotely operate their system. Their current practice is to reference weather forecast, then alert citizens if a combined sewer overflow event may occur within the next day. Since they are a public service, they need to be able to justify their reasoning for their decisions, especially when their decisions affect the safety of customers. This need for transparency is why their current system of mostly relying on weather forecasts is preferred.

Our collaboration with MSDGC experimented a DL solution of predicting the combined sewer overflows based on the LSTM implementation [2]. The data used for our LSTM algorithm was taken from various sensors at a MSDGC's outflow site, a manhole approximately 450 ft upstream of the outflow site, and a rainfall sensor for the area. The site is considered to be "overflowing" whenever the level of water exceeds the site's capacity. Notably, the data were collected at different rates. The slowest sampling rate is one sample for every five minutes while the fastest is every minute. As illustrations with fictitious data, Table 1 shows a sample of the rainfall data, and Table 2 shows samples from sensors in a manhole a few minutes upstream in a pipe upstream.

Related work on DL-based smart sewer systems includes the use of a recurrent neural network to forecast the stormwater runoff in terms of the precipitation and the previous runoff discharge at a combined sewer overflow site near the District of Columbia [4]. The experiment on the

Table 1
Sample of rainfall data which is sampled every minute and reports the depth of rainfall measured in an area upstream of the combined sewer outfall site.

Timestamp	Rainfall (inches)
Oct 12, 2018 14:30	0.0007
Oct 12, 2018 14:31	0.0006
Oct 12, 2018 14:32	0.0006
Oct 12, 2018 14:33	0.0009
Oct 12, 2018 14:34	0.0013
Oct 12, 2018 14:35	0.0014

Table 2
Sample of flow, level, and velocity data from the manhole site upstream of the outflow sensor. The data were sampled at a rate of once every five minutes.

Timestamp	Level	Velocity	Flow
Aug 20, 2019 2:15:00	1.706	0.920	0.066
Aug 20, 2019 2:20:00	1.673	0.861	0.060
Aug 20, 2019 2:25:00	1.648	0.789	0.054
Aug 20, 2019 2:30:00	1.634	0.753	0.051
Aug 20, 2019 2:35:00	1.618	0.779	0.052

34,721-timestep data showed that runoff prediction accuracy was high when the hidden layer reached the maximum capacity of hardware constraints. Our ongoing work compared three variants of the recurrent neural network architecture, namely LSTM, GRU, and IndRNN [3], and further showed that LSTM exhibited a high robustness and stationarity [5]. In this paper, our interests focus specifically on the decision of synchronizing data sampled at different rates (cf. Tables 1 and 2), and the decision's impacts on the key requirements of predictive accuracy and explainability.

3. Study Design and Results

Using the data shown in Table 1 as an example, aggregating the rainfall measures from a five-minute span to a single value represents an important yet subtle data engineering decision. Suppose a synchronization decision at 14:35 (Oct 12, 2018) needs to be made, a baseline choice may be taking the instantaneous value of 0.0014. However, these exist several competing decisions: from 14:31 to 14:35, mode would take 0.0006, median would use 0.0009, mean would equal 0.00096, minimum would point to 0.0006, and maximum would match 0.0014.

Our study thus tests these different data engineering decisions' impacts on a DL solution's performances. Specifically, we reuse the LSTM implementation of Maltbie *et al.* [2] where the related artifacts (including source code, hyper parameter search scripts, etc.) are available at https://doi.org/10.5281/zenodo.4818970 . This LSTM model is trained with 12 hours of continuous data and predicts whether a combined sewer overflow event would occur in the next hour (i.e., between the 12th and 13th hour). In addition, we exploit the LIME tool [6] for generating the explanations of why LSTM makes certain predictions in the same way as [2]. LIME creates

Table 3
Confidence scores for different data engineering decisions on a true positive case.

Data Engineering Decision	True Positive Case
Baseline	0.62
Mean	0.64
Median	0.50
Mode	0.57
Maximum	0.55
Minimum	0.60

a local approximation of the deep learning model's output space by sampling various inputs from our dataset, and then uses this approximation of the output space to determine which features in the input space are the most significant to determine the model's prediction.

In our study, six different datasets are prepared, each corresponding to a data engineering decision: baseline, mean, median, mode, maximum, and minimum. To generate a data point for a particular timestamp (at a five-minute interval), five timestamps from the original collected rainfall data between two five-minute interval timestamps are taken according to the statistical measures. To assess predictive accuracy, we compute the confidence scores of the LSTM's predictions made for elevated flow for a true positive case where the combined sewer overflow event actually occurs. Table 3 lists the results, from which we can conclude that, everything else being equal, the different data engineering decisions do lead to different DL performances. Furthermore, using the mean and the median gives rise to the best (0.64) and worse (0.50) confidence score respectively.

While the confidence scores reported in Table 3 appear quantitatively similar, we run LIME to generate qualitative insights into why the LSTM has made its prediction on our chosen case. Figures 1 and 2 show the LIME explanation results for the true positive case under mean and median respectively. It can be seen from these figures that the factors contributing to the LSTM predictions are noticeably different. When the mean is used, for instance, a top-ranked factor turns out to contribute to a negative prediction (i.e., contributing to normal flow). The LIME results of Figure 2, on the other hand, show the top factors indeed contribute to a positive, elevated flow prediction; however, the confidence score of such a decision is as strong as a coin flip (0.50 shown in Table 3).

4. Concluding Remarks

Our study results indicate that data engineering decisions impact DL's results quantitatively and qualitatively. One may argue that which decision to take is a matter of domain expertise. Yet our experience shows that such expertise is subjective, e.g., maximum may be a logical choice for a wastewater treatment engineer who is willing to tolerate the false-positive overflow predictions, whereas mean or median might be reasonable choices for a statistical analyst.

A main contribution of our paper is to recognize the data engineering decisions as a first-class citizen in the software engineering process for AI-based systems as shown in Figure 3.

Figure 1: LIME explanations for the true positive case of mean.

Figure 2: LIME explanations for the true positive case of median.

Not only do the data engineering decisions influence DL solution's performances, but they also shape the requirements engineering activities in important ways much like aspects cut across the conventional object-oriented modularity [7, 8]. For instance, a wastewater treatment engineer's requirement of willing to tolerate false positives would be better understood and more quantifiable if the results of using maximum could be compared with other decisions like minimum.

In a similar fashion, the DL performance differences resulted from using the mean and the median may provoke a more rigorous discussion among the statistical analyst and related stakeholders to better operationalize data synchronization [9, 10], or even create new ways to handle asynchronous data (e.g., increasing the sampling frequency from the sensor networks dynamically informed by the rainfall measures and/or LSTM's predictions). Due to the dynamic nature of data engineering decisions, aligning requirements closely with testing may be valuable [11, 12]. Finally, it is interesting to realize that the data engineering decisions themselves may help define the emerging requirements of flexibility and customizability that allow the end users to bind such decisions on-the-fly; however, balancing the tradeoffs between being highly customizable and safely operable is of significant importance to AI-based systems.

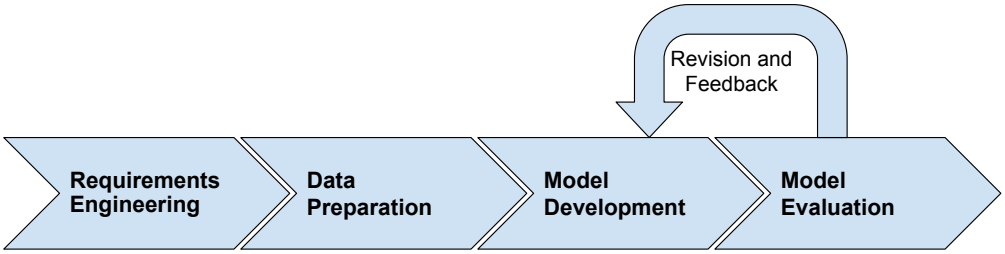

Figure 3: Software engineering process for AI-based systems (*adapted from Maltbie et al. [2]*).

Acknowledgments

We thank Nicholas Maltbie for his seminal and replicable work on which this paper builds.

References

[1] F. Dalpiaz, N. Niu, Requirements engineering in the days of artificial intelligence, IEEE Software 37 (2020) 7–10.

[2] N. Maltbie, N. Niu, M. V. Doren, R. Johnson, XAI tools in the public sector: A case study on predicting combined sewer overflows, in: ESEC/FSE, 2021, pp. 1032–1044.

[3] H. Gudaparthi, R. Johnson, H. Challa, N. Niu, Deep learning for smart sewer systems: Assessing nonfunctional requirements, in: ICSE-SEIS, 2020, pp. 35–38.

[4] N. Zhang, Urban stormwater runoff prediction using recurrent neural networks, in: ISNN, 2011, pp. 610–619.

[5] H. Challa, N. Niu, R. Johnson, Faulty requirements made valuable: On the role of data quality in deep learning, in: AIRE, 2020, pp. 61–69.

[6] M. T. Ribeiro, S. Singh, C. Guestrin, "Why Should I Trust You?" Explaining the predictions of any classifier, in: KDD, 2016, pp. 1135–1144.

[7] N. Niu, S. Easterbrook, Analysis of early aspects in requirements goal models: A concept-driven approach, Transactions on Aspect-Oriented Software Development 3 (2007) 40–72.

[8] N. Niu, B. G.-B. Yijun Yu, N. Ernst, J. Leite, J. Mylopoulos, Aspects across software life cycle: A goal-driven approach, Transactions on Aspect-Oriented Software Development 6 (2009) 83–110.

[9] N. Niu, S. Easterbrook, So, you think you know others' goals? A repertory grid study, IEEE Software 24 (2007) 53–61.

[10] N. Niu, A. Y. Lopez, J.-R. C. Cheng, Using soft systems methodology to improve requirements practices: An exploratory case study, IET Software 5 (2011) 487–495.

[11] T. Bhowmik, S. R. Chekuri, A. Q. Do, W. Wang, N. Niu, The role of environment assertions in requirements-based testing, in: RE, 2019, pp. 75–85.

[12] Z. Peng, P. Rathod, N. Niu, T. Bhowmik, H. Liu, L. Shi, Z. Jin, Environment-driven abstraction identification for requirements-based testing, in: RE, 2021, pp. 245–256.