A data-driven approach for performance evaluation for edge-computing

None

Radarweg 29, Amsterdam

Elsevier Inc[a,b], Global Customer Service[b,]*

[a] *1600 John F Kennedy Boulevard, Philadelphia*
[b] *360 Park Avenue South, New York*

Abstract

Service providers are increasingly using data-driven mechanisms to build their performance model of their service-providing systems. To build a model to accurately describe the performance of the existing infrastructure is very crucial to make resource management decisions. Conventional approaches that use hand-tuned parameters has its drawback. Recently, data-driven paradigm have been shown to greatly outperform traditional methods in many applications, in both accuracy and their quick reactions to the changing environment. We design a framework that using these techniques to add. LSTM has emerged as a powerful tool in sequence modeling of machine learning problems. High dimensionality; anomaly. Our approach shows an average 6.98% improvement in terms of weighted mean absolute percent error (WMAPE) compared to the baseline models.

Keywords: *edge computing, deep learning, content delivery networks, sequence learning, predictive analysis, high dimensional data*

*Corresponding author
Email address: support@elsevier.com (Global Customer Service)
URL: www.elsevier.com (Elsevier Inc)

1. Introduction

There is a trend [1] [2] [**?**]that using data-driven methods to model complex networked systems. Traditional approach typically simple huristics. These methods have several drawbacks drawback [2]. They cannot quickly respond to the change of the environment. They cannot accurately reflect and oversimplified the complex systems due to the lack of knowledge of real-word environment. VM Scheduling. Internet Telphony. CDN selection. Models that accurately describe the complex networked system

In this paper, we provide a framwork for performance evaluation of edge-computing.

Previous approaches have two drawbacks: instant models, sequence model; huristics, data-driven Resource management: Resouoce management involves discovering and identifying all available resources, partitioning them to maximize a utility function — which can be in terms of cost, energy performance, etc., and finally scheduling the tasks on available physical resources.

the relationship between CDN and edge computing (the earliest form of edge computing). A content delivery network (CDN) is a globally distributed network system deployed across the Internet. Composed with geographically distributed cache servers, CDNs deliver cached content to customers worldwide based on their geographic locations. Extensively using cache servers, content delivery over CDN has low latency and high reliability, and supports better quality of experience.

In recent year, with the development of AI and data analytics system, its a trend using data data-driven techniques to optimize networked systems. Driven by the opportunity to collect and analyze data (e.g., application quality measurement from end users), many recent proposals have demonstrated the promise of using data-driven optimization of networked systems. Drawing pfrom the sucsess of Instead of finding out the complex interaction of different trying to We modeling treat networked system as a black-box

Deep Learning (DL) is a rapidly growing discipline that, during the last few

years, has revolutionalised machine learning and artificial intelligence research due to the availability of big data , new algorithms for neural networks training, and extremely fast dedicated hardware. Companies like Google, Microsoft, Amazon, Facebook and Apple use deep learning to solve difficult problems in areas such as speech and image recognition, machine translation, natural language processing, resource planning or even to reduce power consumption by manipulating computer servers and related equipment like cooling systems.

Our prediction consists of stages: (1) feature selection (2) feature embedding (3) fully connected network/ svm/ other black-box machine learning algorithm to output the predictions. lstm,lstm auto encoder and decoder

our contributions are listed follow:

- data-driven approach
- performance modeling as sequence modeling problem
- anomaly detection(Collective Anomalies a) and prediction

first build a prediction model, and then use the prediction model to do the anomaly detection.

the organization of this paper is as follows. In Section II, we first describe the performance evaluation problem and then introduce our LSTM based structure. In Section III, we introduce anomaly detection algorithms based on the auto-encoder and decoder. In Section IV, we demonstrate performance improvements over baseline models. Finally, we provide concluding remarks in Section V.

2. Background

Drawing on prior work, we discuss several applications. Then, we highlight key limitations of huristics approaches.

2.1. Limitations of prior approaches

2.1.1. inaccuracy

The underlying systems are complex and often impossible to model accurately. For instance, in cluster scheduling, the running time of a task varies

with data locality.

2.1.2. Problem and description

2.2. CDN and edge computing

2.3. Model: performance evaluation as sequence learning

[3]

we argue that performance modeling as a sequence problem. Traditionally, deep learning (Relatively recently,sequence modeling based on LSTM technique gained popularity due to its end-to-end modeling, ease of incorporating exogenous variables, and automatic feature extraction abilities.[4])

2.4. deep learning in sequence prediction

deep learning Deep learning can be thought of asfeature engineering done automatically by algorithms Sequence prediction often involves forecasting the next value in a real valued sequence or outputting a class label for an input sequence.

2.4.1. RNN

[5] [6]

what is RNN and RNN applied in sequence forecast.

2.5. big data application stack

spark[7];spark streaming[8];kafka;

2.6. comparison of exsisting approach

2.6.1. Fundamental Limitations of Prior Approaches

hard and tedious tuning of parameters

unable to adapt to the change of the environment

2.6.2. our approach

3. Methods

3.1. Spark Streaming

data preprocessing

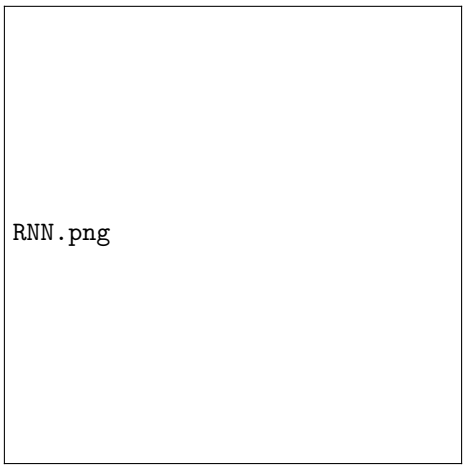


Figure 1: RNN Architecture

feature	meaning
cpu	cpu ratio
memory	
disk	

Table 1: list of candidate input features

3.2. Feature Engineering

need more data to do the experiments

Feature selection: The benefit of this feature selection process is two-fold: (a) a reduced feature set will control the model complexity during model learning, and (b) the processes gain more insights on the complex interaction of different matrics. [9] Dimension Reduction: Some Deep Learning algorithms can become prohibitively computationally-expensive when dealing with high-dimensional data

3.3. Model Design

3.4.

Samples are constructed using a sliding window with step size one, where each sliding window contains the previous 28 minutes as input, and aims to

forecast the upcoming reach rate. the reach rateis stationary ....

3.4.1. RNN

RNNs maintain a hidden vector $\mathbf{h}$, which is updated at time step t as follows:

$$\mathbf{h}_t = \tanh(\mathbf{W} * \mathbf{h}_{t-1} + \mathbf{I} *_t) \tag{1}$$

where tanh is the hyperbolic tangent function, $\mathbf{W}$ is the recurrent weight matrix and I is a projection matrix. The hidden state $\mathbf{h}$ is then used to make a prediction

$$\mathbf{y}_t = \text{softmax}(\mathbf{W} * \mathbf{h}_{t-1}) \tag{2}$$

where *softmax* provides a normalized probability distribution over the possible classes and $\mathbf{W}$ is a weight matrix. By using $\mathbf{h}$ as the input to another RNN, we can stack RNNs, creating deeper architectures [**?**]

$$\mathbf{h}_t^l = \sigma(\mathbf{W} * \mathbf{h}_{t-1}^l + \mathbf{I} * \mathbf{h}_t^{l-1}). \tag{3}$$

Training vanilla RNNs is known to be particularly difficult, with vanishing and exploding gradients being one possible explanation [**?**].

3.4.2. RNN encoder-decoder

[6] applications: machine translation, learning to excute, image captioning, conversational modeling

RNN Encoder-Decoder, consists of two recurrent neural networks (RNN) that act as an encoder and a decoder pair. The encoder maps a variable-length source sequence to a fixed-length vector, and the decoder maps the vector representation back to a variable-length target sequence. [6] also known as sequence embedding.

Why RNN encoder-decoder

The point of training an autoencoder is to make an RNN learn how to compress a relatively long sequence into a limited, dense vector.

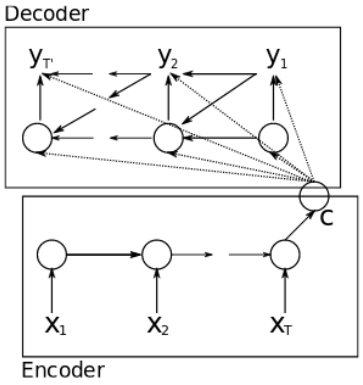


Figure 2: neural network architecture

3.4.3. LSTM

LSTM, introduced in [10], addresses the problem of vanishing gradients by introducing a memory cell which ensures constant error flow and gating units. The inner working of LSTM are listed follows:

$$\begin{aligned}
\mathbf{g}^u &= \sigma(\mathbf{W}^u * \mathbf{h}_{t-1} + \mathbf{I}^u *_t) \\
\mathbf{g}^f &= \sigma(\mathbf{W}^f * \mathbf{h}_{t-1} + \mathbf{I}^f *_t) \\
\mathbf{g}^o &= \sigma(\mathbf{W}^o * \mathbf{h}_{t-1} + \mathbf{I}^o *_t) \\
\mathbf{g}^c &= \tanh(\mathbf{W}^c * \mathbf{h}_{t-1} + \mathbf{I}^c *_t) \\
\mathbf{m}_t &= \mathbf{g}^f \odot + \mathbf{g}^u \odot \mathbf{g}^c \\
\mathbf{h}_t &= \tanh(\mathbf{g}^o \odot \mathbf{m}_{t-1})
\end{aligned} \tag{4}$$

3.5. lstm auto-encoder

Autoencoders are data-specific. Autoencoders are lossy. Autoencoders are learned automatically from data examples, which is a useful property: it means that it is easy to train specialized instances of the algorithm that will perform well on a specific type of input. It doesn't require any new engineering, just appropriate training data. [11]

An autoencoder contains: an encoding function, a decoding function, and a distance function between the amount of information loss between the com-

pressed representation of your data and the decompressed representation (i.e. a "loss" function). The encoder and decoder will be chosen to be parametric functions (typically neural networks), and to be differentiable with respect to the distance function, so the parameters of the encoding/decoding functions can be optimize to minimize the reconstruction loss, using Stochastic Gradient Descent. It's simple! And you don't even need to understand any of these words to start using autoencoders in practice. [11]

Today two interesting practical applications of autoencoders are data denoising (which we feature later in this post), and dimensionality reduction for data visualization. With appropriate dimensionality and sparsity constraints, autoencoders can learn data projections that are more interesting than PCA or other basic techniques. [11]

[12] applied in time series. Long Short-Term Memory (LSTM) is able to solve many time series tasks unsolvable. by feedforward networks using fixed size time windows[13].

LSTM attention[14];applied in time series [15] [16]

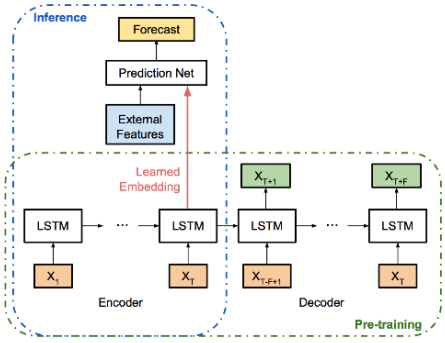


Figure 3: neural network architecture

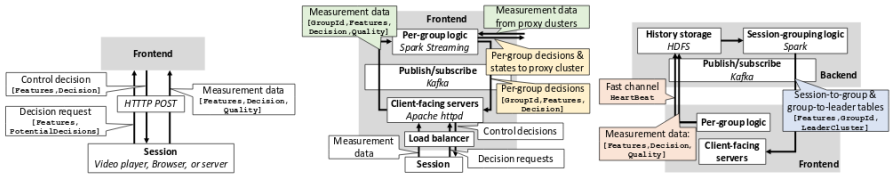
As you can see in the figure 3, the function grows near 0. Also, in the page 8 is the same example.

4. implementation

kafka, spark streaming

tensorflow:[17]

online/offline training



(a) API to application sessions (b) Frontend (c) Backend

Figure 4: implementation

5. Evaluation

5.1. Experimental Settings

data description

how samples are constructed

5.2. Baseline

1. Persistent Model: a naive model that takes last ouput as the prediction
2. Single-LSTM
3. LSTM encoder-decoder with multiple-layers perceptions
4. LSTM encoder-decoder with multiple-layers perceptions and attention

5.3. Performance

compare four different models in terms of training time and accuracy

Table 2: My caption

location	Persistent	LSTM	LSTM encoder-decoder	Our model
Shanghai	10.0	9.9	8.8	7.7
Shenzhen	10.0	9.9	8.8	7.7
Zhejiang	10.0	9.9	8.8	7.7

5.4. Deployed at ChinaNetCenter

6. Discussion and Future Work

a unified models for different cache groups.

further improve accuracy

uncertainty

qualified rate is an indirect measurement;collecting client data.

change detection

7. Related Work

edge-computing performance.[18]

data-driven networking. there are data-driven mechanisms

CDN: two types: long-term, short term;CDN selection [1]

deep-learning;RNN;RNN encoder-decoder;LSTM;LSTM time-series application;LSTM with attention

Geo-distributed data analytics:

deep learning and streaming data [19] incremental feature learning and ex traction, denoising autoencoders, and deep belief networks

Our work: [20]

8. Conclusion

We have present a system that provides esimation for . Using the LSTM encoder-decoder with multiple-layers perceptions and attention techinques we improve

References

[1] J. Jiang, S. Sun, V. Sekar, H. Zhang, Pytheas: Enabling Data-Driven Quality of Experience Optimization Using Group-Based Exploration-Exploitation, 14th USENAIX Symposium on NSDI.

[2] H. Mao, R. Netravali, M. Alizadeh, Neural Adaptive Video Streaming with Pensieve, Mohammad Alizadeh MIT Computer Science and Artificial Intelligence Laboratory (2017) 197–210doi:10.1145/3098822.3098843.
URL http://dx.doi.org/10.1145/3098822.3098843

[3] M. Längkvist, L. Karlsson, A. Loutfi, A review of unsupervised feature learning and deep learning for time-series modelingdoi:10.1016/j.patrec.2014.01.008.
URL https://ac.els-cdn.com/S0167865514000221/1-s2.0-S0167865514000221-main.pdf?_tid=73b58696-de42-11e7-99f2-00000aab0f27&acdnat=1512976445_b3a701a33285bf903a86108fc3a2b956

[4] L. Zhu, N. Laptev, Deep and Confident Prediction for Time Series at Uberdoi:10.1109/ICDMW.2017.19.

[5] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE Transactions on Neural Networks 5 (2) (1994) 157–166. doi:10.1109/72.279181.
URL http://www.ncbi.nlm.nih.gov/pubmed/18267787http://ieeexplore.ieee.org/document/279181/

[6] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation.
URL http://www.statnlp.org/wp-content/uploads/2016/02/rnn.pdf

[7] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, I. Stoica, Spark: Cluster Computing with Working Sets.

URL http://www1.ece.neu.edu/~ningfang/SimPaper/hotcloud_spark.pdf

[8] M. Zaharia, T. Das, H. Li, S. Shenker, I. Stoica, Discretized Streams: An Efficient and Fault-Tolerant Model for Stream Processing on Large Clusters.
URL https://www.sics.se/%7eamir/cloud14/papers/2012%20-%20Discretized%20Streams:%20An%20Efficient%20and%20Fault-Tolerant%20Model%20for%20Stream%20Processing%20on%20Large%20Clusters%20%28HotCloud%29.pdf

[9] J.-S. Yeom, J. J. Thiagarajan, A. Bhatele, G. Bronevetsky, T. Kolev, Data-Driven Performance Modeling of Linear Solvers for Sparse Matrices, in: 2016 7th International Workshop on Performance Modeling, Benchmarking and Simulation of High Performance Computer Systems (PMBS), IEEE, 2016, pp. 32–42. doi:10.1109/PMBS.2016.009.
URL http://ieeexplore.ieee.org/document/7836412/

[10] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Computation 9 (8) (1997) 1735–1780. doi:10.1162/neco.1997.9.8.1735.

[11] Building Autoencoders in Keras.
URL https://blog.keras.io/building-autoencoders-in-keras.html

[12] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, Long Short Term Memory Networks for Anomaly Detection in Time Series.
URL https://www.researchgate.net/profile/Pankaj_Malhotra3/publication/304782562_Long_Short_Term_Memory_Networks_for_Anomaly_Detection_in_Time_Series/links/5880506308ae71eb5dbfbd10/Long-Short-Term-Memory-Networks-for-Anomaly-Detection-in-Time-Series.pdf

[13] F. A. Gers, D. Eck, J. Schmidhuber, Applying LSTM to Time Series Predictable through Time-Window Approaches, 2001, pp. 669–676. doi:

10.1007/3-540-44668-0{_}93.
URL http://link.springer.com/10.1007/3-540-44668-0_93

[14] D. Bahdanau, K. Cho, Y. Bengio, NEURAL MACHINE TRANSLATION BY JOINTLY LEARNING TO ALIGN AND TRANSLATE.
URL https://arxiv.org/pdf/1409.0473.pdf

[15] Y. Cinar, H. Mirisaee, P. Goswami, E. Gaussier, A. At-Bachir, V. Strijov, Position-based Content Attention for Time Series Forecasting with Sequence-to-sequence RNNs.
URL https://arxiv.org/pdf/1703.10089.pdf

[16] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, G. W. Cottrell, A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction.
URL http://cseweb.ucsd.edu/~yaq007/DA-RNN.pdf

[17] TensorFlow.
URL https://www.tensorflow.org/

[18] M. Satyanarayanan, The Emergence of Edge Computing.
URL http://elijah.cs.cmu.edu/DOCS/satya-edge2016.pdf

[19] M. M. Najafabadi, F. Villanustre, T. M. Khoshgoftaar, N. Seliya, R. Wald, E. Muharemagic, Deep learning applications and challenges in big data analytics, Journal of Big Data 2. doi:10.1186/s40537-014-0007-7.
URL https://journalofbigdata.springeropen.com/track/pdf/10.1186/s40537-014-0007-7?site=journalofbigdata.springeropen.com

[20] C. Wang, Z. Lu, Z. Wu, J. Wu, S. Huang, Optimizing Multi-Cloud CDN Deployment and Scheduling Strategies Using Big Data Analysis, in: 2017 IEEE International Conference on Services Computing (SCC), 2017, pp. 273–280. doi:10.1109/SCC.2017.42.
URL http://ieeexplore.ieee.org/document/8034995/