

Pendekatan Long Short-Term Memory untuk Memprediksi Kondisi Motor 10 kV pada PLTU Batubara

Muhammad Kamal Wisyaldin¹; Gita Maya Luciana²; Henry Pariaman³

^{1,2} PT. Pembangkitan Jawa Bali, Surabaya, 60231 Indonesia

³ Institut Teknologi PLN, Jakarta, 11750 Indonesia

¹ m.kamalw@ptjpb.com

ABSTRACT

Condition Based Maintenance is one method to predict motor condition. However, this method requires high accuracy of Remaining Useful Life estimation. Estimating Remaining Useful Life can be very difficult especially when the amount of data sensor is too large. Machine Learning approach is needed because it can understand all correlations between measurement sensor data, which can lead to accurate Remaining Useful Life estimation. Long Short-Term Memory network is a best machine learning methods to predict the condition of the equipment because it has the ability to store memories for a long time simultaneously and can understand complex correlations between data thus providing information that is very useful in determining Remaining Useful Life. This paper presents a comparison between the Long-Short Term Memory model and two conventional algorithm models to predict the condition of a 10 kV motor. Modeling results show signal prediction using the Long-Short Term Memory approach is more accurate than conventional algorithm models because the model predictions are closest to the actual data, that indicated by the smallest mean absolute percentage error of 3.8%. The accuracy of the Long-Short Term Memory approach shows the reliability of the architecture so the plant gets a warning before damage occurs.

Keywords: motor, data, prediction, Long Short-Term Memory

ABSTRAK

Conditon Based Maintenance merupakan salah satu metode untuk memprediksi kondisi motor. Namun metode ini membutuhkan keakuratan estimasi Remaining Useful Life yang tinggi. Estimasi Remaining Useful Life bisa sangat sulit terutama ketika jumlah data yang berasal dari sensor terlalu besar. Pendekatan Machine Learning dibutuhkan karena dapat memahami semua korelasi antara pengukuran data sensor, sehingga dapat menghasilkan prediksi Remaining Useful Life yang akurat. Long-Short Term Memory network merupakan salah satu metode pendekatan machine learning terbaik untuk memprediksi kondisi peralatan karena memiliki kemampuan untuk menyimpan ingatan dalam waktu yang lama secara bersamaan serta dapat memahami korelasi yang kompleks antara data sehingga memberikan informasi yang sangat berguna dalam menentukan Remaining Useful Life. Tulisan ini menyajikan perbandingan pendekatan menggunakan model Long-Short Term Memory dan dua model algoritma konvensional untuk memprediksi kondisi motor 10 kV. Hasil pemodelan menunjukkan prediksi sinyal menggunakan pendekatan Long-Short Term Memory lebih akurat daripada model algoritma konvensional karena menghasilkan prediksi model yang paling dekat dengan data aktual, ditunjukkan oleh hasil mean absolute percentage error terkecil yaitu 3,8%. Keakuratan hasil prediksi Long-Short Term Memory menunjukkan keandalan arsitektur sehingga pembangkit memperoleh peringatan sebelum kerusakan terjadi.

Keywords: Data, Prediksi, Machine Learning, Long-Short Term Memory

1. PENDAHULUAN

Aktivitas monitoring dan prediksi kondisi peralatan motor telah berkembang[1,2,3]. Salah satu penyebab kegagalan motor dapat dideteksi secara dini dari temperaturnya. Temperatur motor listrik abnormal yang terjadi disebabkan karena adanya kerusakan diantaranya kerusakan pada bearing, ketidakseimbangan beban, *misalignment*, kegagalan isolasi pada lilitan motor dan lain-lain[1]. Pendeteksian dini terhadap kegagalan motor dengan melihat pola temperatur motor yang terjadi merupakan jenis tindakan prediktif yang secara umum dilakukan oleh kalangan industri. Beberapa aktivitas monitoring dan prediktif pada kondisi motor 10 kV seperti *electromagnetic field monitoring*, *temperature measurements*, *infrared recognition*, *radio-frequency (RF) emissions monitoring*, *noise and vibration monitoring*, *chemical analysis*, *acoustic noise measurements*, *motor-current signature analysis (MCSA)*[2,3].

Pemeliharaan berbasis kondisi (CBM) merupakan salah satu metode untuk memprediksi kondisi saat ini dari peralatan atau sistem peralatan. Menurut Jardine et al. CBM merekomendasikan tindakan berdasarkan informasi yang dikumpulkan dari sistem. Tujuan utama CBM adalah menghindari tindakan pemeliharaan yang tidak perlu dan merekomendasikan tindakan pemeliharaan jika anomali terdeteksi [4]. Deteksi anomali dini sebelum masalah yang muncul sangat penting, dan dapat memicu strategi pemeliharaan yang diperlukan untuk menghindari situasi berbahaya sehingga membantu memastikan keandalan peralatan dan waktu operasi, serta mengurangi tingkat kegagalan dan downtime [5]. CBM yang efektif dapat dikembangkan dengan keakuratan estimasi sisa waktu manfaat (RUL) yang tinggi. RUL dapat diprediksi melalui pengumpulan sinyal dengan sensor yang terletak pada unit sistem terkait. Estimasi RUL bisa sangat sulit terutama ketika jumlah data yang berasal dari sensor terlalu besar. Penggunaan pendekatan *Machine Learning* dibutuhkan untuk memahami semua korelasi tersembunyi antara pengukuran data sensor, dan memungkinkan untuk menghasilkan prediksi RUL yang akurat [6].

Long-Short Term Memory (LSTM) merupakan salah satu jenis *Machine Learning* berbasis pendekatan *Recurrent Neural Network* yang dapat memprediksi kondisi mesin saat ini dengan menggunakan spark mesin pengolah data skala besar [7]. Dong et.al menunjukkan bahwa LSTM dapat dianggap sebagai solusi yang valid dan lebih baik dari teknik lain seperti regresi Naif Bayesien [8]. LSTM network menjadi pilihan terbaik berkat kemampuannya untuk menyimpan ingatan dalam waktu yang lama pada saat yang sama, korelasi yang kompleks antara data memberikan informasi yang sangat berguna dalam menentukan RUL [6]. Pada studi ini dilakukan pemodelan untuk memprediksi kondisi motor 10 kV menggunakan pendekatan LSTM.

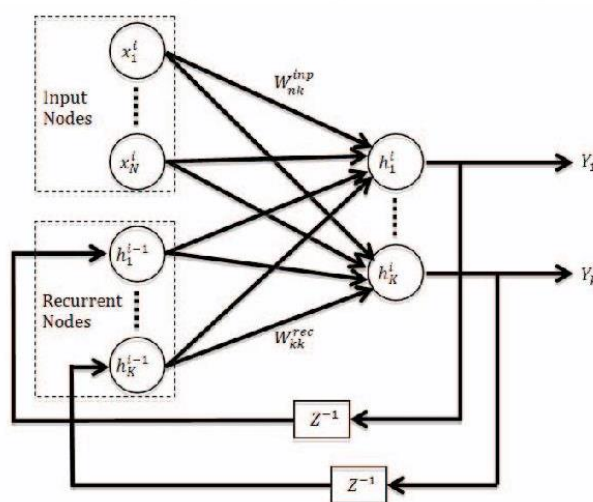
1.1. MACHINE LEARNING

Machine Learning (ML), sebuah teknik komputasi yang didasarkan pada *Theory of Statistical Learning* [9], yang berupaya untuk mengeksplorasi pola variabilitas yang melekat pada data untuk menggeneralisasi hubungan antara *variabel of interest* (sering disebut “target”) dan satu set variabel lain (disebut “fitur” atau “prediktor”) yang mana dianggap sebagai deskripsi yang bagus dari target. Meskipun formulasi ini terdengar sangat mirip dengan model statistik, perbedaan utama antara *machine learning* dan statistik terletak pada penggunaan asumsi matematika yang kaku. Model statistik bergantung pada apa yang disebut distribusi standar asumsi (atau “distribusi asimtotik”) untuk berbagai jenis data; misalnya distribusi normal untuk data dengan nilai kontinu, distribusi binomial untuk data bernilai biner, distribusi poisson untuk menghitung peristiwa langka, dll. Asumsi distribusi tipikal seperti itu tidak akan praktis ketika melakukan pemodelan dengan data yang sangat banyak karena akan menghasilkan estimasi yang bias dan prediksi yang salah. Model berbasis *machine learning* dibangun dengan mempelajari kumpulan pragmatis antara perubahan pada variabel

target dan perubahan pada set fitur, yang mengarah pada error yang lebih kecil. Gabungan tersebut dipelajari secara terprogram (teralgoritmik) secara efisien dan otomatis menggunakan berbagai macam perangkat lunak *open source* yang banyak tersedia. Pemodelan *machine learning* diperlukan untuk mencapai tingkatan otomatisasi yang lebih tinggi.

1.2. LONG-SHORT TERM MEMORY (LSTM)

Long-Short Term Memory (LSTM) merupakan tipe spesial dari *neural network*, dimana termasuk bagian dari *Recurrent Neural Network* (RNN). Tidak seperti *feed forward neural network* konvensional, RNN menggunakan n umpan balik dari *output layers* kembali ke *input layer*, dimana setiap koneksi umpan balik dapat digunakan sebagai *time-delay gate* (Gambar 1). Arsitektur RNN mampu mewakili secara eksplisit pengaruh nilai output masa lalu pada perhitungan output saat ini, menjadikannya ideal untuk memodelkan struktur autokorelasi dari data deret waktu atau *time series*.



Gambar 1. Recurrent Neural Network (RNN) konvensional sederhana

Dibandingkan dengan model ARMA-class klasik untuk prediksi deret waktu, mis. ARMA (p, q):

$$X_t = \{\sum_{i=1}^p a_i X_{t-i} + \sum_{j=1}^q \beta_j \varepsilon_{t-j}\} + \varepsilon_t \quad (1)$$

yang mengasumsikan ketergantungan linear dari nilai sekarang (pada waktu t) pada nilai masa lalu (pada waktu $t-1, t-2, \dots$), model RNN berbasis *machine learning* untuk prediksi deret waktu:

$$X_t = f + (\sum_{i=0}^N w_i X_{t-i}) \varepsilon_t \quad (2)$$

menggabungkan transformasi non-linear, $f(\cdot)$, untuk menyatakan ketergantungan dari nilai saat ini pada jeda waktu. Model RNN mampu mengekspresikan struktur dependensi *lag serial* yang jauh lebih kompleks dengan menangkap pengaruh non-linear dari nilai masa lalu pada nilai saat ini (diprediksi). Bengio et.al (1994) menunjukkan prediksi yang dibuat oleh RNN biasanya jauh lebih akurat karena dapat memodelkan kurvatur dalam hubungan antara variabel target dan prediktor atau fitur, yang hanya dapat direpresentasikan dengan buruk oleh konsep *rigid linear* dari model ARMA-class [10].

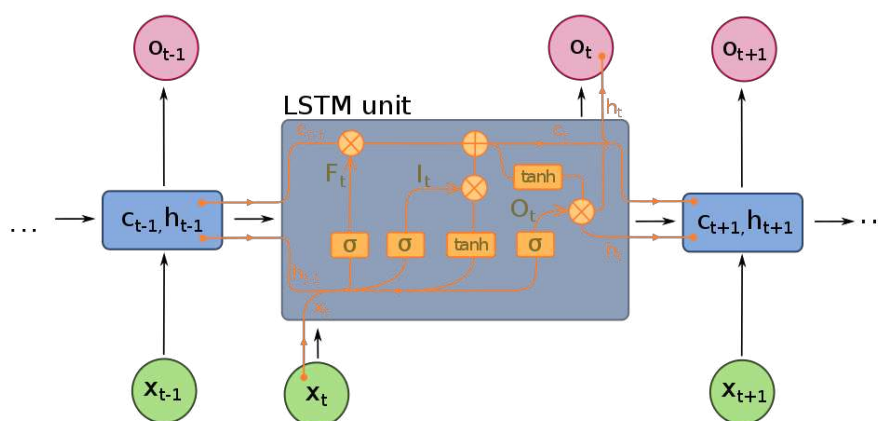
Tipe klasik *plain-vanilla* RNN sulit untuk dilatih dan sering diganggu dengan masalah “vanishing gradient” [10], terutama ketika memprediksi untuk jangka waktu yang panjang di masa depan dimana prediksi pada dasarnya menjadi garis datar - mirip dengan prediksi *random walk*.

LSTM dipahami sebagai arsitektur alternatif untuk menangani masalah dalam RNN konvensional. LSTM pertama kali diperkenalkan pada tahun 1997, karya awalnya adalah [11,12], tetapi telah disempurnakan untuk penggunaan skala praktis besar pada tahun 2014 [13]. LSTM telah mencapai performa terbaik dalam tugas *artificial intelligence* (A.I.) beragam seperti dalam *natural language processing* (NLP) (untuk *handwriting recognition* pada tahun 2009 dan *natural speech recognition* pada tahun 2013) [14,15], untuk *robotic control* pada tahun 2018 [16], dan untuk *automated game control* pada tahun 2019 [17]. Mulai tahun 2016, perusahaan teknologi besar termasuk Google, Apple, Facebook, Amazon, dan Microsoft telah menggunakan LSTM sebagai komponen inti dalam *speech recognition* dan sistem terjemahan bahasa otomatis.

2. METODE/PERANCANGAN PENELITIAN

2.1. LSTM NEURAL NET FOR MODELLING SEQUENCE DATA

Arsitektur generik dari unit LSTM ditunjukkan pada Gambar 2 [18]. Struktur ini terdiri dari neuron yang berfungsi sebagai sel, $c(t)$, dan tiga jenis *gates*, yang disebut *input gates*, $i(t)$, *forget gates*, $f(t)$, dan *output gates*, $o(t)$. Tiga tipe *gates* yang berbeda ini bekerja untuk mengatur input dari titik data yang masuk (baru), $x(t)$, ke neuron sel, yang pada gilirannya secara langsung mempengaruhi perhitungan output, $h(t)$, dari seluruh unit LSTM. Dalam arsitektur RNN keseluruhan, setiap unit LSTM berfungsi sebagai sebuah blok memori yang akan 'mengingat' struktur autokorelasi yang terkandung dalam segmen data deret waktu yang diumpankan ke dalamnya. Mengacu pada arsitektur RNN asli pada gambar 1, sebuah unit LSTM akan diposisikan setelah *time-lag gate*, z , dan sebelum diumpankan kembali ke neuron lapisan pertama.



Gambar 2. Unit LSTM generic [18]

Menggunakan istilah atau gagasan dari bidang identifikasi sistem dinamis, sel-sel berhubungan dengan seperangkat tambahan filter dinamis yang memodifikasi *internal system state* $h(t)$ pada setiap kali iterasi. Pada RNN yang umum, *internal system state* ini berfungsi sebagai satu-satunya mekanisme yang menguasai informasi ketergantungan waktu dengan kata lain, sebagai memori satu arah yang sangat sederhana. Pada LSTM, perilaku memori yang lebih kompleks dapat diwujudkan dengan bantuan *dynamic cell states* dan *gates*. Secara khusus, *forget gates* berfungsi untuk menghapus ("forget") beberapa isi sel yang ada dari iterasi waktu sebelumnya, *input gates* menambahkan ("input") informasi baru ke isi sel, dan *output gates* mengontrol berapa banyak

penyesuaian ini akan mengubah keadaan sel pada iterasi waktu saat ini. Oleh karena itu, informasi lama dapat dihapus, dan informasi baru dapat ditambahkan secara dinamis ke urutan memori.

Secara matematis (di sini, notasi vektor-matriks digunakan dan * menunjukkan perkalian poin / elemen), antarlain:

$x(t) \in \mathbb{R}^p$	menjadi nilai input (sinyal) p pada sistem saat ini, (t)
$h(t-1) \in \mathbb{R}^m$	menjadi <i>internal system states</i> m pada iterasi waktu sebelumnya, $(t-1)$
$h(t) \in \mathbb{R}^m$	menjadi <i>internal system states</i> m pada iterasi waktu saat ini, (t)
$c(t-1) \in \mathbb{R}^m$	menjadi cell states m pada iterasi waktu sebelumnya, $(t-1)$
$c(t) \in \mathbb{R}^m$	menjadi cell states m pada iterasi waktu saat ini, (t)
$i(t) \in \mathbb{R}^m$	menjadi nilai <i>input gates</i> m pada iterasi waktu saat ini, (t)
$f(t) \in \mathbb{R}^m$	menjadi nilai <i>forget gates</i> m pada iterasi waktu saat ini, (t)
$o(t) \in \mathbb{R}^m$	menjadi nilai <i>output gates</i> m pada iterasi waktu saat ini, (t)
$\sigma(\cdot) \in \mathbb{R}^m$	menjadi fungsi <i>generic nonlinearity</i> generik (sigmoid) m -dim dari persamaan $= \sigma(x) = \frac{1}{1+\exp(x)}$
$W_c, W_i, W_f, W_o \in \mathbb{R}^{m \times p}$	menjadi matrix $m \times p$ dari fungsi <i>feed-forward transfer</i> untuk <i>cells, input gate, forget gate, dan output gate</i>
$V_c, V_i, V_f, V_o \in \mathbb{R}^{m \times m}$	menjadi matrix (persegi) $m \times m$ dari <i>feed-back transfer</i> untuk <i>cells, input gate, forget gate, dan output gate</i>
$b_c, b_i, b_f, b_o \in \mathbb{R}^m$	menjadi vektor m -dim dari <i>learning bias</i> untuk <i>cells, input gate, forget gate, dan output gate</i>

Dan kemudian,

- 1) Dari *feed-forward input* saat ini, $x(t)$, dan koneksi *feed-back* dari *hidden system state* sebelumnya, $h(t-1)$, *current states* yang terdiri dari *input, forget, dan output gates* dihitung sebagai berikut:

$$i(t) = \sigma(W_i \cdot x(t) + V_i \cdot h(t-1) + b_i) \quad (3)$$

$$f(t) = \sigma(W_f \cdot x(t) + V_f \cdot h(t-1) + b_f) \quad (4)$$

$$o(t) = \sigma(W_o \cdot x(t) + V_o \cdot h(t-1) + b_o) \quad (5)$$

- 2) Lalu, *cell states* saat ini, $c(t)$, dihitung dengan memodifikasi status sel dari iterasi waktu sebelumnya dengan isi yang dihitung dari *input* dan *forget gates*:

$$c(t) = f(t) * c(t-1) + i(t) \tanh(W_c \cdot x(t) + V_c \cdot h(t-1) + b_c) \quad (6)$$

- 3) Akhirnya, *cell states* yang diperbarui, bersama dengan isi dari *output gates*, digunakan untuk menghitung *hidden states* baru dari sistem (yang pada gilirannya, dalam perhitungan *feedforward*, akan menghasilkan output akhir dari *neural network*):

$$h(t) = o(t) * \tanh(c(t)) \quad (7)$$

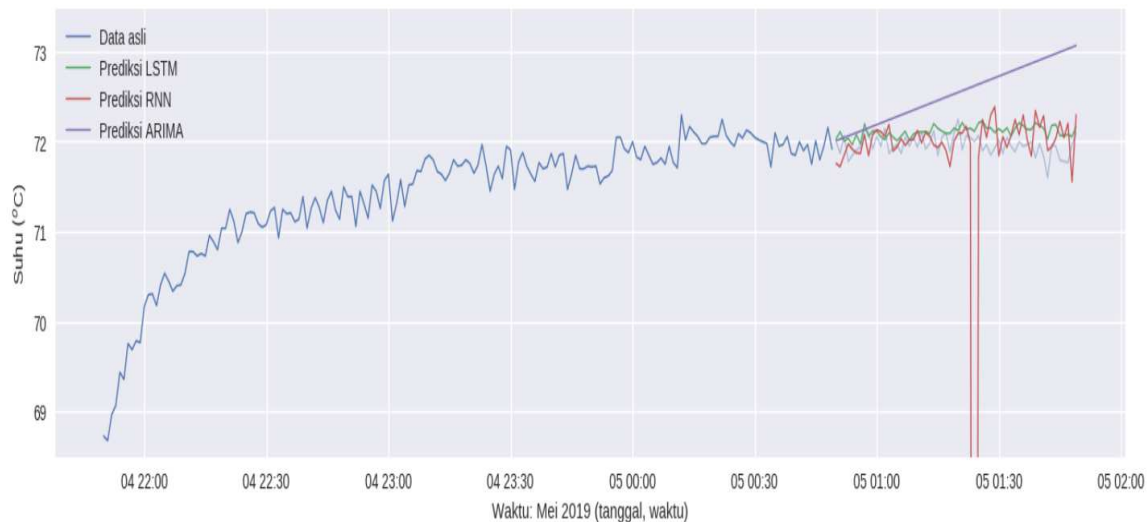
3. HASIL DAN PEMBAHASAN

Perbandingan model hasil dari LSTM versus dua alternatif (konvensional) algoritma: i) model ARMA, dan ii) RNN biasa, disajikan di bawah ini. Data input yang digunakan adalah segmen pendek (180 menit) dari urutan pengukuran sensor "ITE-FW459" motor 10 kV pada PLTU Paiton, untuk menghasilkan prediksi nilai masa depan 60 menit.

Metode pencarian jaringan ("brute force") digunakan untuk menentukan parameter model optimal untuk setiap algoritma. Untuk setiap model, berarti persentase kesalahan absolut (MAPE) dan waktu pelatihan (dalam menit) masing-masing dilaporkan sebagai ukuran kinerja model dan beban komputasi.

Tabel 1. MAPE & Training Time untuk Setiap Model pada motor 10 kV

Algorithm	MAPE (%)	Training Time (second)
SARIMA (1,1,0) [i.e. ARIMA with seasonality tweak]	13,279.97	1.27
RNN with 2 layers: 120 & 60 neurons	5.59	3,403
LSTM with 2 layers: 120 & 60 neurons	3.80	13,579



Gambar 3. Hasil Prediksi LSTM, RNN and ARMA terhadap Data Aktual Motor 10 kV

Plot pada Gambar 3, yang menunjukkan hasil prediksi dari 3 model (LSTM: hijau, RNN: merah, ARMA: violet) versus data aktual (biru muda). Seperti yang dapat dilihat dari plot pada Gambar 3, model ARMA gagal untuk belajar dari *training data* dan menghasilkan prediksi garis lurus dengan kemiringan positif, yang mana prediksi tren data yang naik ini tidak sesuai dengan data aktual, sedangkan model RNN menjadi tidak stabil di beberapa titik dan memprediksi adanya lonjakan (spike) ke bawah (negatif) yang tidak sesuai dengan data aktual. Sementara model LSTM menghasilkan prediksi terdekat dengan data aktual secara keseluruhan. Namun demikian, butuh sekitar 4 kali lebih lama untuk melatih model LSTM dibandingkan dengan model RNN, dan lebih dari 10.000 kali lebih lama dibandingkan dengan model ARMA.

4. KESIMPULAN DAN SARAN

Kondisi motor dapat diprediksi salah satunya dengan melihat pola temperatur motor *winding* yang terjadi. Metode pendekatan *Machine Learning* yang paling baik untuk memprediksi kondisi peralatan adalah Long-Short Term Memory Network karena memiliki kemampuan untuk menyimpan ingatan dalam waktu yang lama secara bersamaan. Hasil pemodelan menggunakan metode LSTM menunjukkan prediksi sinyal temperatur *winding* motor 10 kV yang lebih akurat dan mendekati kondisi aktual dibandingkan dengan prediksi sinyal menggunakan metode algoritma konvensional (ARMA dan RNN biasa). Hal ini terlihat dari nilai *mean absolute percentage error* (MAPE) model LSTM yang diperoleh yaitu 3,8%. Keakuratan model LSTM menunjukkan keandalan arsitektur sehingga pembangkit memperoleh peringatan sebelum kerusakan terjadi.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada rekan-rekan PT PJB UP Paiton yang telah memberikan dukungan yang membantu pelaksanaan penelitian dan atau penulisan artikel.

DAFTAR PUSTAKA

- [1] Tugino, "Rekayasa sistem monitoring dan analisis temperature motor listrik berbasis komputer, JFN, Vol. 3 No. 2, 2009.
- [2] P. Vas, "Parameter Estimation, Condition Monitoring, and Diagnosis of Electrical Machines", Oxford, U.K.: Clarendon, 1993.
- [3] G. B. Kliman and J. Stein, "Induction motor fault detection via passive current monitoring," in Proc. Int. Conf. Electrical Machines, Cambridge, MA, Aug. 1990, pp. 13–17.
- [4] A. Jardine, D. Lin, D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance", Mechanical Systems and Signal Processing, vol. 20, Oct 2006, pp. 1483-1510, doi: 10.1016/j.ymssp.2005.09.012.
- [5] Wang, K., Wang, Y., "How AI affects the future predictive maintenance: a primer of deep learning", In: IWAMA 2017, Lecture Notes in Electrical Engineering, vol. 451, 2018.
- [6] Vita, D.F., Bruneo, D., "On the use of LSTM networks for Predictive Maintenance in Smart Industries, IEEE International Conference on Smart Computing, 2019.
- [7] A. Saxena, K. Goebel, "Turbofan Engine Degradation Simulation Data Set," NASA Ames Prognostics Data Repository (<http://ti.arc.nasa.gov/project/prognostic-data-repository>), NASA Ames Research Center, Moffett Field, CA, 2008.
- [8] D. Dong, X. Li, and F. Sun, "Life prediction of jet engines based on LSTM-Recurrent Neural Networks," in 2017 Prognostics and System Health Management Conference (PHM-Harbin), July 2017, pp. 1–6.
- [9] Hastie, T., Tibshirani, R., and Friedman, J., "The Elements of Statistical Learning," Springer-Verlag, 2009.
- [10] Bengio, Y., Simard, P., and Frasconi, P., "Learning long-term dependencies with gradient descent is difficult," IEEE Transactions on Neural Networks Vol.5(2), 1994.
- [11] Hochreiter, S. and Schmidhuber, J., "Long short-term memory," Neural Computation Vol.9(8), 1997.
- [12] Gers, F.A., Schmidhuber, J., and Cummins, F.A., "Learning to forget: Continual prediction with LSTM," Neural Computation Vol.12(10), 2000.
- [13] Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y., "Learning phrase representations using RNN encoder-decoder for statistical

- machine translation,” Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.
- [14] Graves, A., Liwicki, M., Fernandez, S., Bertolami, R., Bunke, H., and Schmidhuber, J., “A novel connectionist system for unconstrained handwriting recognition,” IEEE Transactions on Pattern Analysis and Machine Intelligence Vol.31(5), 2009.
- [15] Graves, A., Mohamed, A.R., and Hinton, G., “Speech recognition with deep recurrent neural networks,” In 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2013.
- [16] “Learning dexterity,” OpenAI Blog, July 30, 2018.
- [17] Stanford, S., “DeepMind’s AI, AlphaStar, showcases significant progress towards AGI.” Medium ML Memoirs, Jan. 25, 2019.
- [18] Deloche, F., Own work, CC BY-SA 4.0.