Deep Ordinal Regression for Remaining Useful Life Estimation from Censored Data

Conference Paper · July 2018

4 authors:

Vishnu Tv
Tata Consultancy Services Limited
20 PUBLICATIONS **652** CITATIONS
SEE PROFILE

Pankaj Malhotra
Tata Consultancy Services Limited
47 PUBLICATIONS **2,918** CITATIONS
SEE PROFILE

Lovekesh Vig
Tata Consultancy Services Limited
171 PUBLICATIONS **4,746** CITATIONS
SEE PROFILE

Gautam Shroff
Tata Consultancy Services Limited
156 PUBLICATIONS **4,005** CITATIONS
SEE PROFILE

Deep Ordinal Regression for Remaining Useful Life Estimation from Censored Data

Vishnu TV, Pankaj Malhotra, Lovekesh Vig, Gautam Shroff
TCS Research
New Delhi, India
{vishnu.tv,malhotra.pankaj,lovekesh.vig,gautam.shroff}@tcs.com

ABSTRACT

Estimating remaining useful life (RUL) for equipment using sensor data streams is useful to enable condition-based maintenance, and avoid catastrophic shutdowns due to impending failures. Recently, supervised deep learning approaches have been proposed for RUL estimation that leverage historical sensor data of failed instances for training the models. However, access to a large number of failed instances required to train deep networks is often rare in real-world scenarios. On the other hand, data from a large number of currently operational instances is easily accessible but deemed unusable for training due to obvious lack of target (RUL) labels for these instances. In this work, we propose a novel approach to leverage these operational instances while training deep Recurrent Neural Networks (RNNs) for RUL estimation from time series of sensor data. We formulate RUL estimation as an ordinal regression (OR) problem, and employ deep RNNs to learn the OR function. We show that OR-based formulation of the RUL estimation problem naturally allows incorporating the censored operational instances into training data, leading to more robust learning. Through experiments on C-MAPSS turbofan engine benchmark datasets, we demonstrate that our approach is significantly better than the commonly used deep metric regression based approaches, especially when failed training instances are scarce.

1 INTRODUCTION

In the current Digital Era, streaming data is ubiquitous, and various original equipment manufacturers are taking a keen interest in Industrial Internet of Things-enabled remote health monitoring services to provide operational support, ensure high reliability and availability, and reduce operational cost of equipment [5]. A large number of sensors are being installed to capture the operational behavior of equipment. Data-driven remaining useful life (RUL) estimation module is a key component of such health monitoring applications.

Recently, deep learning approaches have been proposed for various sensor data-driven health monitoring tasks including anomaly detection [19, 21] and prognostics [8, 20, 35], yielding state-of-the-art results for RUL estimation [8] using Recurrent Neural Networks (RNNs). However, deep learning approaches for health monitoring have certain limitations as highlighted in [8, 13]. We consider one such limitation in this work: Deep neural networks require a large number of labeled training instances to avoid overfitting. However, such instances are often not available as failures are rare. If failure time for an instance is known, a target RUL can be obtained at any time before the failure time. Any currently operational instance (or any instance for which failure time is not known) is considered to be *censored* as target RUL cannot be determined for such an instance. We consider the often encountered real-world scenario of using supervised deep neural networks for RUL estimation using sufficient censored instances and few failed instances.

Deep RNNs [8, 10, 20, 34, 35] and Convolutional Neural Networks (CNNs) [1] have been proposed for RUL estimation. Most of these approaches consider RUL estimation to be a metric regression (MR) problem where a normalized estimate of RUL is obtained given time series of sensor data via a non-linear regression metric function learned from the data.

We note that MR formulation of RUL estimation cannot directly leverage censored data typically encountered in RUL estimation scenarios. On the other hand, Ordinal Regression (OR) [9] for neural networks has been proposed in [4], which can be extended to incorporate censored data during training [18]. In contrast to MR, the response variables in OR are discrete and finite. OR has been extensively used for various survival analysis and prognostics applications in medical domain [28, 29], age estimation from facial images [17, 22], etc. These approaches have been shown to perform better than MR.

In this work, we propose a novel approach for RUL estimation using Deep Long Short Term Memory (LSTM) [11] RNNs for Ordinal Regression (LSTM-OR) for scenarios dealing with censored data. We show that RUL estimation can also be formulated as an OR problem

Presented at Joint Workshop on Deep Learning for Safety-Critical Applications in Engineering at ICML-IJCAI-AAMAS, 2018
2018.

with different RUL ranges being the discrete variables of interest (refer Section 4 for details). LSTM-OR leverages the advantages of deep learning to learn non-linear mapping from time series of raw sensor data to RUL, and uses censored data to overcome small labeled data issues arising due to limited failure instances.

The key contributions of this work are as follows:
1) We propose LSTM-OR for deep ordinal regression from time series data with application to RUL estimation.
2) To address the aforementioned issue of scarce labeled training data, we propose an approach to generate *partially labeled training instances* from the readily available operational (non-failed) instances to augment the labeled training data, yielding robust RUL estimation models.

Through empirical evaluation on C-MAPSS aircraft turbofan engine degradation benchmark datasets [26], we demonstrate that: i) LSTM-OR provides a robust end-to-end learning framework for RUL estimation that inherently extracts relevant features from raw time series of multiple sensors, and ii) LSTM-OR performs significantly better than LSTM-based metric regression (LSTM-MR) with decreasing number of failure instances by effectively leveraging censored data, while maintaining performance comparable to LSTM-MR when large number of failure instances are available.

2 RELATED WORK

An important class of approaches for RUL estimation is based on trajectory similarity, e.g. [8, 14, 15, 20, 30]. These approaches compare the health index trajectory or trend of a test instance with the trajectories of failed train instances to estimate RUL using a distance metric such as Euclidean distance. Such approaches work well when trajectories are smooth and monotonic in nature but are likely to fail in scenarios when there is noise or intermittent disturbances (e.g. spikes, operating mode change, etc.) as the distance metric may not be robust to such scenarios [8].

Another class of approaches is based on metric regression. Unlike trajectory similarity based methods which rely on comparison of trends, metric regression methods attempt to learn a function to directly map sensor data to RUL, e.g. [1, 2, 6, 8, 10, 35]. Such methods can better deal with non-monotonic and noisy scenarios by learning to focus on the relevant underlying trends irrespective of noise. Within metric regression methods, few methods consider non-temporal models such as Support Vector Regression for learning the mapping from values of sensors at a given time instance to RUL, e.g. [2, 6].

Deep temporal models such as those based on RNNs [8, 10, 20, 35] or Convolutional Neural Networks (CNNs) [1] can capture the degradation trends better compared to non-temporal models, and are proven to perform better. Moreover, these models can be trained in an end-to-end learning manner without requiring feature engineering. Despite all these advantages of deep models, they are prone to overfitting in often-encountered practical scenarios where the number of failed instances is small, and most of the data is censored. Our approach based on ordinal regression provisions for dealing with such scenarios, by using censored instances in addition to failed instances to obtain more robust models.

Ordinal Regression has been extensively used for applications such as age estimation from facial images [3, 17, 22, 31], however for non-temporal image data using Convolutional Neural Networks. [4, 18] use feed-forward neural networks based ordinal regression for survival analysis. To the best of our knowledge, the proposed LSTM-OR approach is the first attempt to leverage ordinal regression based training using temporal LSTM networks for RUL estimation.

A set of techniques for deep survival analysis have been proposed in medical domain, e.g. [12, 18]. On similar lines, an approach to combine deep learning and survival analysis for asset health management has been proposed in [16]. However, it is not clear as to how such approaches can be adapted for RUL estimation applications, as they focus on estimating the survival probability at a given point in time, and cannot provide RUL estimates. On the other hand, LSTM-OR is capable of providing RUL estimates using time series of sensor data.

3 BACKGROUND: DEEP LSTM NETWORKS

We use a variant of LSTMs as described in [33] in the hidden layers of the neural network. Hereafter, we denote column vectors by bold small letters and matrices by bold capital letters. For a hidden layer with h LSTM units, the values for the input gate $\mathbf{i}_t$, forget gate $\mathbf{f}_t$, output gate $\mathbf{o}_t$, hidden state $\mathbf{z}_t$, and cell state $\mathbf{c}_t$ at time t are computed using the current input $\mathbf{x}_t$, the previous hidden state $\mathbf{z}_{t-1}$, and the cell state $\mathbf{c}_{t-1}$, where $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{o}_t$, $\mathbf{z}_t$, and $\mathbf{c}_t$ are real-valued h-dimensional vectors.

Consider $W_{n_1,n_2} : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$ to be an affine transform of the form $\mathbf{z} \mapsto \mathbf{W}\mathbf{z}+\mathbf{b}$ for matrix $\mathbf{W}$ and vector $\mathbf{b}$ of appropriate dimensions. In the case of a multi-layered LSTM network with L layers and h units in each layer, the hidden state $\mathbf{z}_t^l$ at time t for the l-th hidden layer is obtained from the hidden state at $t-1$ for that layer

$\mathbf{z}^l_{t-1}$ and the hidden state at t for the previous $(l-1)$-th hidden layer $\mathbf{z}^{l-1}_t$. The time series goes through the following transformations iteratively at l-th hidden layer for $t = 1$ through T, where T is length of the time series:

$$\begin{pmatrix} \mathbf{i}^l_t \\ \mathbf{f}^l_t \\ \mathbf{o}^l_t \\ \mathbf{g}^l_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} W_{2h,4h} \begin{pmatrix} \mathbf{D}(\mathbf{z}^{l-1}_t) \\ \mathbf{z}^l_{t-1} \end{pmatrix} \tag{1}$$

where the cell state $\mathbf{c}^l_t$ is given by $\mathbf{c}^l_t = \mathbf{f}^l_t\mathbf{c}^l_{t-1} + \mathbf{i}^l_t\mathbf{g}^l_t$, and the hidden state $\mathbf{z}^l_t$ is given by $\mathbf{z}^l_t = \mathbf{o}^l_t tanh(\mathbf{c}^l_t)$. We use dropout for regularization [23], which is applied only to the non-recurrent connections, ensuring information flow across time-steps for any LSTM unit. The dropout operator $\mathbf{D}(\cdot)$ randomly sets the dimensions of its argument to zero with probability equal to a dropout rate. The sigmoid (σ) and $tanh$ activation functions are applied element-wise.

In a nutshell, this series of transformations for $t = 1 \dots T$, converts the input time series $\mathbf{x} = \mathbf{x}_1 \dots \mathbf{x}_T$ of length T to a fixed-dimensional vector $\mathbf{z}^L_T \in \mathbb{R}^h$. We, therefore, represent the LSTM network by a function f_{LSTM} such that $\mathbf{z}^L_T = f_{LSTM}(\mathbf{x}; \mathbf{W})$, where $\mathbf{W}$ represents all the parameters of the LSTM network.

4 DEEP ORDINAL REGRESSION FOR RUL ESTIMATION

4.1 Terminology

Consider a learning set $\mathcal{D} = \{\mathbf{x}^i, r^i\}^n_{i=1}$ of n failed instances, where r^i is the target RUL, $\mathbf{x}^i = \mathbf{x}^i_1 \dots \mathbf{x}^i_{T^i} \in \mathcal{X}$ is a multivariate time series of length T^i, $\mathbf{x}^i_t \in \mathbb{R}^p$, p is the number of input features (sensors). The total operational life of an instance i till the failure point is F^i, s.t. $T^i \leq F^i$. Therefore, $r^i = F^i - T^i$ is the RUL in given unit of measurement, e.g., number of cycles or operational hours. Hereafter, we omit the superscript i in this section for better readability, and provide all the formulation considering an instance (unless stated otherwise).

We consider an upper bound r_u on the possible values of RUL as, in practice, it is not possible to predict too far ahead in future. So if $r > r_u$, we clip the value of r to r_u. The usually defined goal of RUL estimation via Metric Regression (MR) is to learn a mapping $f_{MR} : \mathcal{X} \to [0, r_u]$. With these definitions, we next describe LSTM-based Ordinal Regression (LSTM-OR) approach as summarized in Figure 3, and then describe how we incorporate censored data into the LSTM-OR formulation.

4.2 LSTM-based Ordinal Regression

Instead of mapping an input time series to a real-valued number as in MR, we break the range $[0, r_u]$ of RUL values into K intervals of length c each, where each interval is then considered as a discrete variable. The j-th interval corresponds to $((j-1)\frac{r_u}{c}, j\frac{r_u}{c}]$, and r is mapped to the k-th interval with $k = \lceil \frac{r}{c} \rceil$, where $\lceil . \rceil$ denotes the ceiling function.

We consider K binary classification sub-problems for the K discrete variables (intervals): a classifier C_j solves the binary classification problem of determining whether $r \leq j\frac{r_u}{c}$.

We train an LSTM network for the K binary classification tasks simultaneously by modeling them together as a multi-label classification problem: We obtain the multi-label target vector $\mathbf{y} = [y_1, \dots, y_K] \in \{0, 1\}^K$ from r such that

$$y_j = \begin{cases} 0 & j < k \\ 1 & j \geq k \end{cases} \tag{2}$$

where $j = 1, 2, \dots, K$.

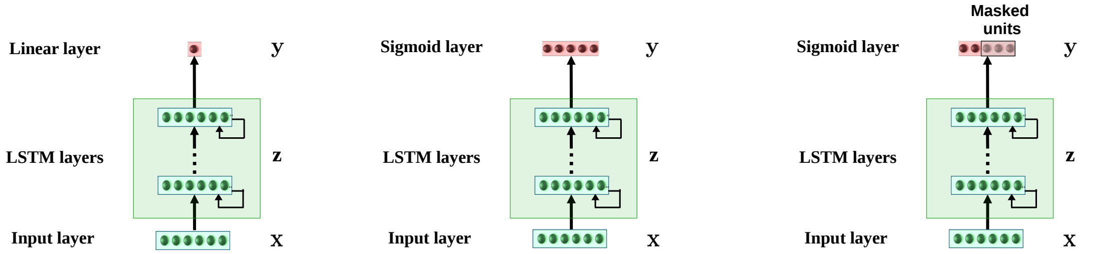
For example, consider a scenario where $K = 5$, and r maps to the third interval such that $k = 3$. The target is then given by $\mathbf{y} = [0, 0, 1, 1, 1]$, as illustrated in Figure 2(a). Effectively, the goal of LSTM-OR is to learn a mapping $f_{OR} : \mathcal{X} \to \{0, 1\}^K$ by minimizing the loss function $\mathcal{L}$ given by:

$$\begin{aligned} \mathbf{z}^L_T &= f_{LSTM}(\mathbf{x}; \mathbf{W}) \\ \hat{\mathbf{y}} &= \sigma(\mathbf{W}_C\, \mathbf{z}^L_T + \mathbf{b}_C) \\ \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) &= -\frac{1}{K}\sum_{j=1}^{K} y_j \cdot log(\hat{y}_j) + (1 - y_j) \cdot log(1 - \hat{y}_j) \end{aligned} \tag{3}$$

where, $\hat{\mathbf{y}}$ is the estimate for target $\mathbf{y}$, $\mathbf{W}$ represents the parameters of the LSTM network, and $\mathbf{W}_C$ and $\mathbf{b}_C$ are the parameters of the layer that maps $\mathbf{z}^L_T$ to the output sigmoid layer.

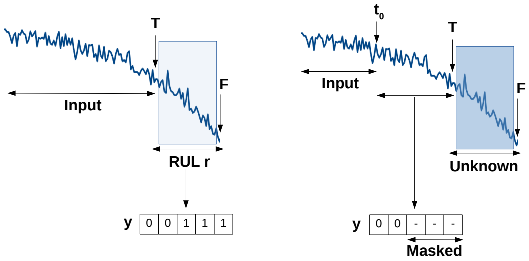
4.3 Using Censored Data for Training

For any censored instance, the data is available only till a time T prior to failure and the failure time F is unknown (illustrated in Figure 2(b)). Therefore, the target RUL r is also unknown. However, at any time t_0 s.t. $1 \leq t_0 < T$, it is known that the RUL $r > T - t_0$ since the instance is operational at least till T. Considering $\mathbf{x} = \mathbf{x}_1 \dots \mathbf{x}_{t_0}$ as the input time series, we next show how we assign labels to few of the dimensions y_j of the target vector $\mathbf{y}$: Assuming $T - t_0$ maps to the interval $k' = \lceil \frac{T-t_0}{c} \rceil$, since $T - t_0 < r$, we have $\lceil \frac{T-t_0}{c} \rceil \leq \lceil \frac{r}{c} \rceil \implies k' \leq k$. Since k



(a) Metric Regression (b) Ordinal Regression (c) Ordinal Regression For Censored Data

Figure 1: Deep Ordinal Regression versus Deep Metric Regression.



(a) Failed Instance (b) Censored Instance

Figure 2: Target vector creation for failed versus censored instance.

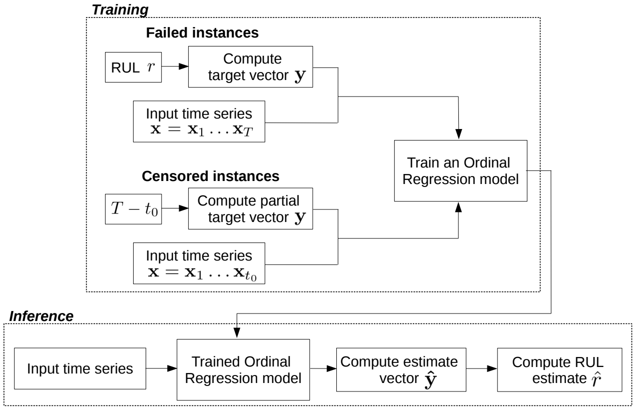

Training
Failed instances
RUL r
Compute target vector $\mathbf{y}$
Input time series $\mathbf{x} = \mathbf{x}_1 \dots \mathbf{x}_T$
Train an Ordinal Regression model
Censored instances
$T - t_0$
Compute partial target vector $\mathbf{y}$
Input time series $\mathbf{x} = \mathbf{x}_1 \dots \mathbf{x}_{t_0}$
Inference
Input time series
Trained Ordinal Regression model
Compute estimate vector $\hat{\mathbf{y}}$
Compute RUL estimate $\hat{r}$

Figure 3: Process overview for LSTM-OR.

is unknown (as r is unknown) and we have $k' \leq k$, the target vector $\mathbf{y}$ can only be partially obtained:

$$y_j = \begin{cases} 0 & j < k' \\ unknown & j \geq k' \end{cases} \quad (4)$$

For all $j \geq k'$, the corresponding binary classifier targets are masked, as shown in Figure 2(b), and the outputs from these classifiers are not included in the loss function for the instance. The loss function $\mathcal{L}$ given by Equation 3 can thus be modified for including the censored instances in training as:

$$\mathcal{L}_m(\mathbf{y},\hat{\mathbf{y}}) = -\frac{1}{K'}\sum_{j=1}^{K'} y_j \cdot log(\hat{y}_j)+(1-y_j)\cdot log(1-\hat{y}_j) \quad (5)$$

where $K' = k' - 1$ for a censored instance and $K' = K$ for a failed instance.

4.4 Map OR estimates to RUL

Once trained, each of the K classifiers provides a probability $\hat{y}_j$ for RUL being greater than the upper limit of the interval corresponding to the j-th classifier. We obtain the point-estimate $\hat{r}$ for r from $\hat{\mathbf{y}}$ for a test instance as follows (similar to [3]):

$$\hat{r} = r_u(1 - \frac{1}{K}\sum_{j=1}^{K}\hat{y}_j) \quad (6)$$

It is worth noting that once learned, the LSTM-OR model can be used in an online manner for operational instances: at current time instance t, the sensor data from the latest T time instances can be input to the model to obtain the RUL estimate r at t.

5 EXPERIMENTAL EVALUATION

We consider three approaches for evaluation: i) **MR**: LSTM-MR using failed instances only (as in [8, 10, 35]), ii) **OR**: LSTM-OR using failed instances only and using loss as in Equation 3, iii) **ORC**: LSTM-OR leveraging censored data along with failed instances using loss as in Equation 5. We use the publicly available C-MAPSS aircraft turbofan engine benchmark datasets [26] for our experiments.

5.1 Dataset Description

We consider datasets FD001 and FD004 from the simulated turbofan engine datasets[1] [26]. The training sets (*train_FD001.txt* and *train_FD004.txt*) of the two datasets contain time series of readings for 24 sensors (21 sensors and 3 operating condition variables) of several instances (100 in FD001 and 249 in FD004) of a turbofan engine from the beginning of usage till end of life. The time series for the instances in the test sets (*test_FD001.txt* and *test_FD004.txt*) are pruned some time prior to failure, such that the instances are operational and their RUL needs to be estimated. The actual RUL values for the test instances are available in *RUL_FD001.txt* and *RUL_FD004.txt*. We randomly sample 20% of the available training set instances, as given in Table 1, to create a validation set for hyperparameter selection.

For simulating the scenario for censored instances, a percentage $m \in \{0, 50, 70, 90\}$ of the training and validation instances are randomly chosen, and time series for each instance is randomly truncated at one point prior to failure. We then consider these truncated instances as censored (currently operational) and their actual RUL values as unknown. The remaining $(100 - m)\%$ of the instances are considered as failed. Further, the time series of each instance thus obtained (censored and failed) is truncated at 20 random points in the life prior to failure, and the exact RUL r for failed instances and the minimum possible RUL $T - t_0$ for the censored instances (as in Section 4 and Figure 2) at the truncated points are used for obtaining the models. The number of instances thus obtained for training and validation for $m = 0$ is given in Table 2. The test set remains the same as the benchmark dataset across all scenarios (with no censored instances). The MR and OR approches cannot utilize the censored instances as the exact RUL targets are unknown, while ORC can utilize the lower bound on RUL targets to obtain partial labels as per Equation 4.

An engine may operate in different operating conditions and also have different failure modes at the end of its life. The number of operating conditions and failure modes for both the datasets are given in Table 1. FD001 has only one operating condition, so we ignore the corresponding three sensors such that $p = 21$, whereas FD004 has six operating conditions determined by the three operating condition variables. We map these six operating conditions to a 6-dimensional one hot vector as in [35], such that $p = 27$.

5.2 Performance metrics

There are several metrics proposed to evaluate the performance of prognostics models [25]. We measure the performance of our models in terms of Timeliness Score (S) and Root Mean Squared Error (RMSE): For a test instance i, the error in estimation is given by $e_i = \hat{r}_i - r_i$. The timeliness score for N test instances is given by $S = \sum_{i=1}^{N}(exp(\gamma \cdot |e_i|) - 1)$, where $\gamma = 1/\tau_1$ if $e_i < 0$, else $\gamma = 1/\tau_2$. Usually, $\tau_1 > \tau_2$ such that late predictions are penalized more compared to early predictions. We use $\tau_1 = 13$ and $\tau_2 = 10$ as proposed in [27]. The lower the value of S, the better is the performance. The root mean squared error (RMSE) is given by: $RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} e_i^2}$.

5.3 Experimental Setup

Table 1: Number of train, validation and test instances. Here, OC: number of operating conditions, FM: number of fault modes.

Dataset	Train	Validation	Test	OC	FM
FD001	80	20	100	1	1
FD004	199	50	248	6	2

Table 2: Number of truncated instances.

Dataset	Train	Validation	Test
FD001	1600	400	100
FD004	3980	1000	248

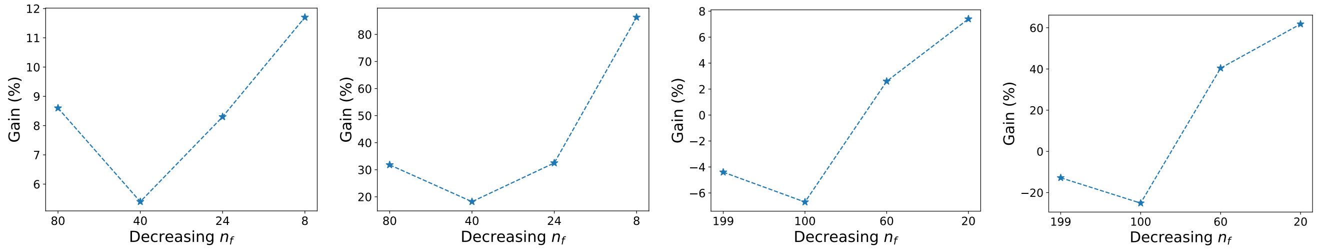
We consider $r_u = 130$ cycles for all models, as used in [1, 35]. For OR and ORC, we consider $K = 10$ such that interval length $c = 13$. For training the MR models, a normalized RUL in the range 0 to 1 (where 1 corresponds to a target RUL of 130 or more) is given as the target for each input. We use a maximum time series length of $T = 360$; for any instance with more than 360 cycles, we take the most recent 360 cycles. Also, we use the standard z-normalization to normalize the input time series sensor wise using mean and standard deviation of each sensor from the train set.

The hyperparameters h (number of hidden units per layer), L (number of hidden layers) and the learning rate are chosen from the sets $\{50, 60, 70, 80, 90, 100\}$, $\{2, 3\}$, and $\{0.001, 0.005\}$, respectively. We use a dropout rate of 0.2 for regularization, and a batch size of 32 during training. The models are trained for a maximum of 2000 iterations with early stopping. The best hyperparameters are obtained using grid search by minimizing the respective loss function on the validation set.

[1] https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository/#turbofan

Table 3: Comparison of various LSTM-based approaches considered in terms of RMSE and Timeliness Score (S) for FD001 and FD004 datasets. n_f and n_c denote number of failed and censored instances in training set, respectively.

	FD001								**FD004**							
	Instances		RMSE			Timeliness Score (S)			Instances		RMSE			Timeliness Score (S) $\times 10^{-3}$		
m(%)	n_f	n_c	MR	OR	ORC	MR	OR	ORC	n_f	n_c	MR	OR	ORC	MR	OR	ORC
0	80	0	15.62	**14.28**	**14.28**	507.2	**346.77**	**346.77**	199	0	**26.88**	28.07	28.07	**4.92**	5.55	5.55
50	40	40	17.56	19.06	**16.61**	444.1	564.14	**363.19**	100	99	**29.71**	32.85	31.7	**7.97**	17.9	9.97
70	24	56	19.92	**16.48**	18.26	713.31	**362.21**	481.73	60	139	33.17	33.65	**32.3**	18.8	17.4	**11.2**
90	8	72	25.32	24.83	**22.37**	1.26×10^4	3.07×10^4	$\mathbf{1.73 \times 10^3}$	20	179	41.23	43.88	**38.17**	102.0	111.0	**39.0**



(a) RMSE FD001 (b) Timeliness Score (S) FD001 (c) RMSE FD004 (d) Timeliness Score (S) FD004

Figure 4: %age gain of ORC over MR with decreasing number of failed instances (n_f) in training.

5.4 Results and Observations

Table 4: Performance comparison of the proposed approach with existing approaches in terms of RMSE and Timeliness Score (S).

	FD001		**FD004**	
	RMSE	S	RMSE	S
CNN-MR [1]	18.45	1.29×10^3	29.16	7.89×10^3
LSTM-MR [35]	16.14	$\mathbf{3.38 \times 10^2}$	28.17	5.55×10^3
MR (ours)	15.62	5.07×10^2	**26.88**	$\mathbf{4.92 \times 10^3}$
ORC (proposed)	**14.28**	3.47×10^2	28.07	5.55×10^3

As summarized in Table 3, we observe that: As the number of failed training instances (n_f) decreases, the performance for all models degrades (as expected). However, importantly, for scenarios with small n_f, ORC is significantly better than MR and OR. For example, with $m = 90\%$ (i.e. with $n_f = 8$ and 20 for FD001 and FD004, respectively), ORC performs significantly better than MR, and shows 11.6% and 7.4% improvement over MR in terms of RMSE, for FD001 and FD004, respectively. The gains in terms of timeliness score S are further higher because of the exponential nature of S (refer Section 5.2). This is further evident with increasing %age gain of ORC over MR with decreasing number of failed instances in training, as shown in Figure 4. While MR and OR have access to only a small number failed instances n_f for training, ORC has access to n_f instances as well as partial labels from n_c censored instances for training. Therefore, MR and OR models tend to overfit while ORC models are more robust.

We also provide a comparison with existing deep CNN-based [1] and LSTM-based [35] MR approaches in Table 4. ORC (same as OR for m =0%) performs comparable to existing MR methods. More importantly, as noted above, ORC may be advantageous and more suitable for practical scenarios with few failed training instances.

6 DISCUSSION

In this work, we have proposed a novel approach for RUL estimation using deep ordinal regression based on multilayered LSTM neural networks. We have argued that ordinal regression formulation is more robust compared to metric regression, as the former allows to incorporate more labeled data from censored instances. We found that leveraging censored instances significantly improves the performance when the number of failed instances is small. In future, it would be interesting to see if a semi-supervised approach (e.g. as in [7, 32]) with initial unsupervised pre-training of LSTMs using failed as well as censored instances can further improve the robustness of the models. Further, an extension to the proposed approach to address the usually encountered non-stationarity scenario using approaches similar to [24] can be considered.

REFERENCES

[1] Giduthuri Sateesh Babu, Peilin Zhao, and Xiao-Li Li. 2016. Deep convolutional neural network based regression approach for estimation of remaining useful life. In *International Conference on Database Systems for Advanced Applications*. Springer, 214–228.

[2] T Benkedjouh, Kamal Medjaher, Noureddine Zerhouni, and Saïd Rechak. 2013. Remaining useful life estimation based on nonlinear feature reduction and support vector regression. *Engineering Applications of Artificial Intelligence* 26, 7 (2013), 1751–1760.

[3] Kuang-Yu Chang, Chu-Song Chen, and Yi-Ping Hung. 2011. Ordinal hyperplanes ranker with cost sensitivities for age estimation. In *Computer vision and pattern recognition (cvpr), 2011 ieee conference on*. IEEE, 585–592.

[4] Jianlin Cheng, Zheng Wang, and Gianluca Pollastri. 2008. A neural network approach to ordinal regression. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 1279–1284.

[5] Li Da Xu, Wu He, and Shancang Li. 2014. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics* 10, 4 (2014), 2233–2243.

[6] Hancheng Dong, Xiaoning Jin, Yangbing Lou, and Changhong Wang. 2014. Lithium-ion battery state of health monitoring and remaining useful life prediction based on support vector regression-particle filter. *Journal of power sources* 271 (2014), 114–123.

[7] Narendhar Gugulothu, Vishnu TV, Priyanka Gupta, Pankaj Malhotra, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2018. On Practical Aspects of Using RNNs for Fault Detection in Sparsely-Labeled Multi-sensor Time Series. *Annual Conference of the Prognostics and Health Management Society*.

[8] Narendhar Gugulothu, Vishnu TV, Pankaj Malhotra, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2017. Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks. *International Journal on Prognostics and Health Management, IJPHM, arXiv preprint arXiv:1709.01073* (2017).

[9] Frank E Harrell. 2001. Ordinal logistic regression. In *Regression modeling strategies*. Springer, 331–343.

[10] Felix O Heimes. 2008. Recurrent neural networks for remaining useful life estimation. In *Prognostics and Health Management, 2008. PHM 2008*. IEEE, 1–6.

[11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[12] Jared L Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. 2018. DeepSurv: Personalized treatment recommender system using A Cox proportional hazards deep neural network. *BMC medical research methodology* 18, 1 (2018), 24.

[13] Samir Khan and Takehisa Yairi. 2018. A review on the application of deep learning in system health management. *Mechanical Systems and Signal Processing* 107 (2018), 241–265.

[14] Racha Khelif, Simon Malinowski, Brigitte Chebel-Morello, and Noureddine Zerhouni. 2014. RUL prediction based on a new similarity-instance based approach. In *IEEE International Symposium on Industrial Electronics*.

[15] Jack Lam, Shankar Sankararaman, and Bryan Stewart. 2014. Enhanced Trajectory Based Similarity Prediction with Uncertainty Quantification. *PHM 2014* (2014).

[16] Linxia Liao and Hyung-il Ahn. 2016. Combining Deep Learning and Survival Analysis for Asset Health Management. *International Journal of Prognostics and Health Management* (2016).

[17] Hao Liu, Jiwen Lu, Jianjiang Feng, and Jie Zhou. 2017. Ordinal Deep Feature Learning for Facial Age Estimation. In *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*. IEEE, 157–164.

[18] Margaux Luck, Tristan Sylvain, Héloïse Cardinal, Andrea Lodi, and Yoshua Bengio. 2017. Deep Learning for Patient-Specific Kidney Graft Survival Analysis. *arXiv preprint arXiv:1705.10245* (2017).

[19] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection. In *Anomaly Detection Workshop at 33rd International Conference on Machine Learning (ICML 2016)*. arXiv preprint arXiv:1607.00148.

[20] Pankaj Malhotra, Vishnu TV, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. Multi-Sensor Prognostics using an Unsupervised Health Index based on LSTM Encoder-Decoder. *1st ACM SIGKDD Workshop on ML for PHM. arXiv preprint arXiv:1608.06154* (2016).

[21] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2015. Long Short Term Memory Networks for Anomaly Detection in Time Series. In *ESANN, 23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*. 89–94.

[22] Zhenxing Niu, Mo Zhou, Le Wang, Xinbo Gao, and Gang Hua. 2016. Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4920–4928.

[23] Vu Pham, Théodore Bluche, Christopher Kermorvant, and Jérôme Louradour. 2014. Dropout improves recurrent neural networks for handwriting recognition. In *Frontiers in Handwriting Recognition (ICFHR)*. IEEE, 285–290.

[24] Sakti Saurav, Pankaj Malhotra, Vishnu TV, Narendhar Gugulothu, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2018. Online anomaly detection with concept drift adaptation using recurrent neural networks. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*. ACM, 78–87.

[25] Abhinav Saxena, Jose Celaya, Edward Balaban, Kai Goebel, Bhaskar Saha, Sankalita Saha, and Mark Schwabacher. 2008. Metrics for evaluating performance of prognostic techniques. In *IEEE International Conference on Prognostics and health management. PHM 2008.* . IEEE, 1–17.

[26] A Saxena and K Goebel. 2008. Turbofan Engine Degradation Simulation Data Set. *NASA Ames Prognostics Data Repository* (2008).

[27] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. 2008. Damage propagation modeling for aircraft engine run-to-failure simulation. In *IEEE International Conference on Prognostics and Health Management, 2008. PHM 2008*. IEEE, 1–9.

[28] Taysseer Sharaf and Chris P Tsokos. 2015. Two artificial neural networks for modeling discrete survival time of censored data. *Advances in Artificial Intelligence* 2015 (2015), 1.

[29] W Nick Street. 1998. A Neural Network Model for Prognostic Prediction.. In *ICML*. 540–546.

[30] Tianyi Wang, Jianbo Yu, David Siegel, and Jay Lee. 2008. A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *IEEE International Conference on Prognostics and Health Management, 2008. PHM 2008.* IEEE, 1–6.

[31] Huei-Fang Yang, Bo-Yao Lin, Kuang-Yu Chang, and Chu-Song Chen. 2013. Automatic age estimation from face images via deep ranking. *networks* 35, 8 (2013), 1872–1886.

[32] Andre S Yoon, Taehoon Lee, Yongsub Lim, Deokwoo Jung, Philgyun Kang, Dongwon Kim, Keuntae Park, and Yongjin Choi. 2017. Semi-supervised Learning with Deep Generative Models for Asset Failure Prediction. *arXiv preprint arXiv:1709.00845* (2017).

[33] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* (2014).

[34] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael Pecht. 2018. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology* (2018).

[35] Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. 2017. Long Short-Term Memory Network for Remaining Useful Life estimation. In *IEEE International Conference on Prognostics and Health Management (ICPHM), 2017.* IEEE, 88–95.