Remaing Useful Lifetime Analysis - PHME 2020 Data Challenge 2020

1st Denis Bernovschi
s1102854@studenti.univpm.it

2st Emanuele Incicco
s1097841@studenti.univpm.it

***Abstract*—In questo progetto visioneremo alcune delle molteplici tecniche atte a valutare la RUL (Remaining Useful Life) tramite l'utilizzo dell'applicativo MATLAB.**

***Index Terms*—RUL, Degradation Model, Similarity Model**

I. Introduzione

L'obiettivo del progetto [3] è quello di stimare la RUL [1] di un sistema dedito al filtraggio. Nella figura Fig.1 è possibile osservare lo schema completo del sistema. Lo scopo è quello di effettuare una manutenzione predittiva, ovvero stimare il momento di rottura del filtro prima del danneggiamento reale, che può avvenire per occlusione. Nel dettaglio il sistema è così composto:

- L'elemento sensibile, il filtro;
- Un composto di acqua e particelle di Polietereterchetone (PEEK);
- Una pompa peristaltica, che mantiene stabile il flusso del fluido;
- Flussimetro, sensore che ci permette di misurare la portata;
- Due sensori di pressione posti a monte e a valle del filtro, permettendoci così di valutare la variazione di pressione dovuta anche all'occlusione del filtro.

Il filtro viene considerato occluso quando la differenza di pressione è maggiore di 20 psi.

II. Dataset

Il *dataset* contiene 32 esperimenti *Run-to-Failure* del filtro in questione. Ciascun esperimento riporta le seguente informazioni (acquisite a $10Hz$):

1) *Flow Rate*, valore della portata;
2) *Up Stream Pressure*, valore della pressione a monte del filtro;
3) *Down Stream Pressure*, valore della pressione a valle del filtro.

I dati relativi ai vari esperimenti sono così suddivisi:

- *Training Set*, contiene 24 esperimenti *Run-to-Failure*;
- *Validation Set*, contiene 8 esperimenti *Run-to-Failure*.

Successivamente alla divisione precedente, gli esperimenti sono stati ulteriormente suddivisi in:

- *Small Size*, 12 esperimenti con particelle di dimensioni *small*;

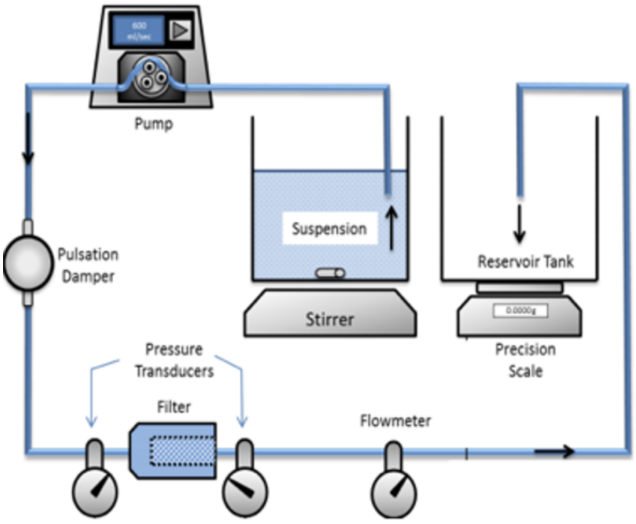


Fig. 1: Sistema di Filtraggio

- *Large Size*, 12 esperimenti con particelle di dimensioni *large*.

I dati sono generati in condizioni controllate, sono note ad esempio le dimensione e la concentrazione delle particelle di contaminazione. Gli esperimenti descritti precedentemente sono ulteriormente divisi in base alla concentrazione delle particelle presenti nel liquido, si hanno infatti differenti valori di *Solid Ratio*: 0.4, 0.425 e 0.45. Il *Training Set* contiene 4 esperimenti, per ciascuna di queste concentrazioni e per ognuna delle dimensioni delle particelle. Il *Validation Set* contiene esperimenti con una concentrazione non nota durante la fase di training.
Il dataset è composto da diversi file CSV[2] dove sono memorizzati il tempo e i valori per il *flow rate*, pressione a monte e a valle del filtro.

[1] RUL $\models$ Remaining Useful Lifetime

[2] CSV $\models$ Comma Separated Value

III. ETL(Extract, Transform, Load) Process

Al fine di utilizzare tale *dataset*, per l'allenamento dei diversi modelli che vedremo in seguito, si è dovuta svolgere la fase di ETL [3]. In questa fase, attraverso i *tool* forniti da **Mathworks**, come Matlab stesso e l'App *Diagnostic Feature Designer* disponibile al suo interno, abbiamo eseguito i passi seguenti:

1) Import dei vari file CSV, tenendo separati i file relativi al *Training Set* da quelli relativi al *Validation Set*;
2) Aggiunta dell'informazione relativa alla **differenza di pressione**, necessaria a valutare quando il filtro risulta essere occluso. Secondo le specifiche della *Challenge* in esame, tale condizione si verifica quando:

$$\Delta_{Press} = Up_{Press} - Down_{Press} \quad (1)$$

Δ_{Press} risulta superiore a 20 psi.
Nella figura Fig. 2 è possibile osservare gli andamenti temporali, delle misurazioni presenti nel *dataset* e del *Delta Pressure*, realizzati utilizzando le funzione messe a disposizione dal *Diagnostic Feature Designer*. Nello specifico sono riportati gli andamenti relativi alle misure del *Training Set* con dimensioni delle particelle pari a *small*.

3) Come si può osservare dalla figura Fig.2 è presente una discreta quantità di rumore che affligge i diversi dati. Si è quindi andati a individuare possibili operazioni di *pre-processing* per elidere il rumore e le componenti non necessarie, così a rendere più accurata la stima della RUL.
   a) Per rimuovere la componente di rumore dei segnali si è applicato un filtro passa-basso. Per stabilire la frequenza di taglio del filtro si è analizzato il *Power Spectrum* (visibile in figura Fig.3, relativo al Training Set Small). Dopo una prima analisi dello spettro in frequenza si è scelta la frequenza di taglio pari a $f_t = 3Hz$.

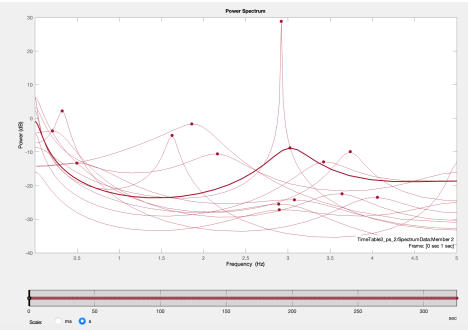
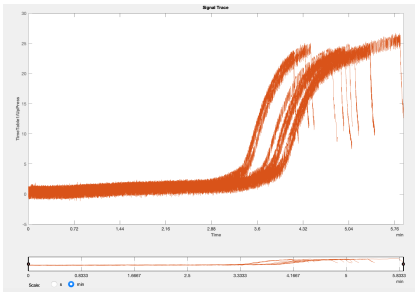


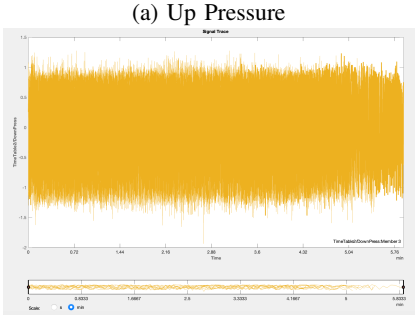
Fig. 3: Power Spectrum

   b) Per elidere la componente transitoria iniziale visibile nella figura Fig. 2c abbiamo effettuato un *clipping* dei segnali per i primi 16 secondi e il conseguente *shifting* a 0 sec dei segnali restanti. Tali operazioni sono state svolte per tutti i segnali considerati.

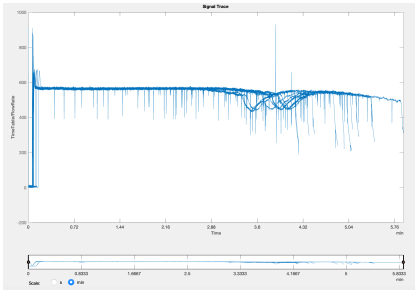
[3]ETL ⊨ Extraction Trasforming Loading



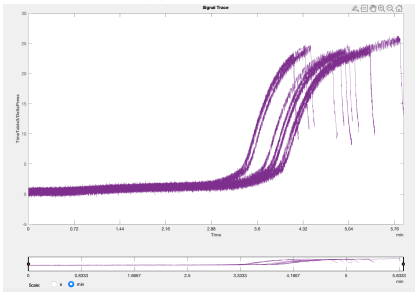
(a) Up Pressure



(b) Down Pressure



(c) Flow Rate



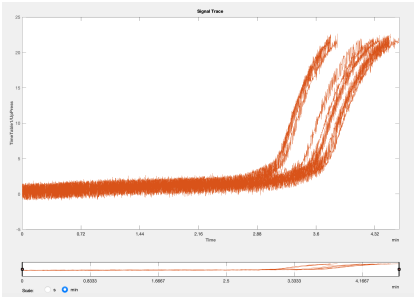
(d) Delta Pressure

Fig. 2: Misure Training Set Small senza alcun preprocessing

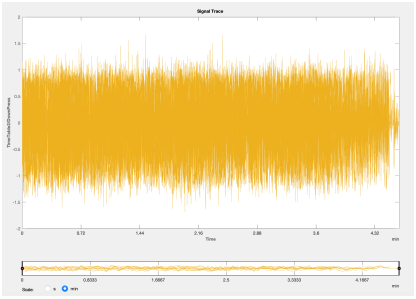
   c) Per ridurre il contenuto informativo non necessario è stato svolto un ulteriore *clipping* di tutti i segnali al superamento del valore di *Delta Pressure* di 22 psi.

I segnali risultanti dal processo di *pre-processing* sono graficati in figura Fig. 4.

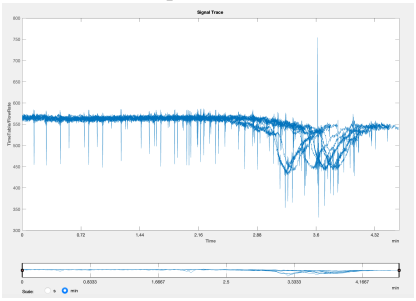
4) Per ultimare la fase di ETL, si è svolta l'analisi delle *feature* sia nel dominio del tempo che nel dominio della frequenza. In seguito a tale analisi si è effettuato il *ranking* delle *feature*. Nella sezione successiva si andrà a visionare i passi svolti per ottenere tale *ranking*.



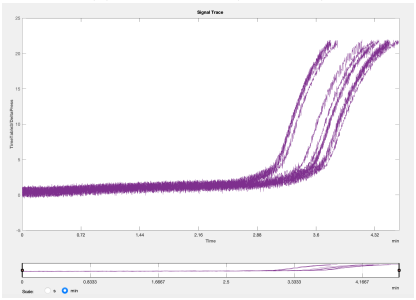
(a) Up pressure (SMALL)



(b) Down pressure (SMALL)



(c) Flow Rate (SMALL)



(d) Delta pressure (SMALL)

Fig. 4: Misure del Dataset Small dopo il pre-processing

IV. Scelta delle feature

Al termine della fase di *pre-processing* dei segnali, si è reso necessario procedere all'estrazione delle *feature* utili ai vari modelli che vedremo di seguito. Per una più ampia analisi, come già detto, si sono considerate le *feature* sia nel dominio del tempo che della frequenza.

La selezione delle *feature* è stata realizzata tramite l'App, interna a MATLAB, *Diagnostic Feature Designer (DFD)*, che, tramite una GUI, ci permette di svolgere tutte le analisi fin qui introdotte. Una precisione doverosa è relativa al settaggio dei frame temporali, tramite il DFD è possibile infatti settare che l'analisi venga realizzata in modalità *frame based*, ovvero si vanno a considerare quelli che sono dei frame temporali con la medesima durata.
Dopo aver effettuato diversi test si sono identificati i valori per il *Frame Size* (**FS**) e il *Frame Rate* (**FR**). Nel caso in esame i valori settati sono: $FR = 1s$ e $FS = 1s$, in quanto considerati un giusto compromesso fra il non avere valori *NaN* e l'avere *feature* poco accurate.

Successivamente, per le singole *feature* ottenute da tale processo sono stati calcolati i valori di *monotonicity*, *trendability* e *prognosability*.

Ulteriori considerazioni sono da svolgere sulla scelta delle *feature* che si è realizzata. Si sono selezionate le *feature* con i valori più elevati di *trendability* e *monotonicity* non considerando le *feature* dell'*Up Pressure*. Tale segnale, infatti, risulta avere un andamento simile al *Delta Pressure*, quindi la scelta di tali *feature* avrebbe portato ad aumentare della complessità computazionale aggiungendo poca informazione utile.

Una precisazione finale, è da fare sulle *feature* considerate per i vari modelli. Nel caso dell' *Exponential Degradation Model*, abbiamo utilizzato le *feature* migliori considerando i valori di *monotonicity*, nella fattispecie le prime 5 nel *ranking*; discorso analogo nel caso del *Residual Similarity Model* in cui si sono considerate le prime 5 *feature* ordinare rispetto ai valori di *trendability*.
Procedendo come descritto sopra si sono selezionate le seguenti 5 *feature*:

1) Band Power del *Delta Pressure*;
2) Peak Ampitude del *Delta Pressure*;
3) Peak Value del *Delta Pressure*;
4) RMS del *Delta Pressure*;
5) Mean del *Delta Pressure*;

Selezionando le prime 5 *feature*, ordinate sia per *monotonicity* che per *trendability*, siamo giunti alla selezione delle medesime 5 *feature* riportate nelle tabelle seguenti:

- Dataset Large: Tab.I
- Dataset Small: Tab.II
- Dataset Large + Small: Tab.II

Feature	Monotonicity	Trendability	Prognosability
DeltaPress Mean	0.7541	0.9983	0.9844
DeltaPress RMS	0.7419	0.9983	0.9844
DeltaPress Peak Value	0.6258	0.9959	0.9924
DeltaPress Band Power	0.5431	0.9980	0.9720
DeltaPress Peak Amp	0.5424	0.9980	0.9720

TABLE I: Feature selezionate relative al dataset Large

Feature	Monotonicity	Trendability	Prognosability
DeltaPress Mean	0.7737	0.9944	0.9897
DeltaPress RMS	0.7270	0.9942	0.9897
DeltaPress Peak Value	0.6464	0.9944	0.9956
DeltaPress Band Power	0.6451	0.9886	0.9827
DeltaPress Peak Amp	0.6514	0.9886	0.9826

TABLE II: Feature selezionate relative al dataset Small

Feature	Monotonicity	Trendability	Prognosability
DeltaPress Mean	0.7459	0.9015	0.9866
DeltaPress RMS	0.7344	0.9013	0.9866
DeltaPress Peak Value	0.6361	0.9035	0.9937
DeltaPress Band Power	0.5941	0.8298	0.9579
DeltaPress Peak Amp	5969	0.8298	0.9579

TABLE III: Feature selezionate relative al dataset Large+Small

V. Modelli

A. Degradation Model

Un Exponential Degradation Model [1] è usato per modellare un processo di degradazione esponenziale per stimare la vita utile residua (RUL) di un componente. I modelli di degradazione stimano la RUL predicendo quando un segnale monitorato supererà una soglia predefinita. I modelli di degradazione esponenziale sono utili quando il componente subisce una degradazione cumulativa.
L'Exponential Degradation Model implementa il seguente modello di degradazione esponenziale in tempo continuo

$$S(t) = \phi + \theta(t) \cdot e^{(\beta(t) \cdot t + \varepsilon(t) - \frac{\sigma^2}{2})} \tag{2}$$

dove:

- ϕ è l'intercetta del modello, che è costante. Si può inizializzare ϕ come limite inferiore o superiore sulla regione fattibile della variabile di degradazione usando Phi. Se il segno di θ è:
  - Positivo, allora ϕ è un limite inferiore.
  - Negativo, allora ϕ è un limite superiore.
- $\theta(t)$ è una variabile casuale modellata come una distribuzione log_{normal}.
- $\beta(t)$ è una variabile casuale modellata come una distribuzione gaussiana.
- $\varepsilon(t)$ è il rumore additivo del modello ed è modellato come una distribuzione normale con media zero.
- σ^2 è uguale a *NoiseVariance*.

Di seguito è possibile osservare ulteriori parametri necessari all'implementazione:

- LifeTime Unit: indica l'unità temporale da considerare per la stima della RUL
- Slope Detection: parametro necessario alla eliminazione della parte transitoria del segnale, ovvero è necessario per la determinazione dell'inizio del processo di degradazione.

Nel listato Listing.1 è possibile osservare una prima possibile implementazione del Degradation Model.

```
mdl = exponentialDegradationModel(...
    'Theta', 1, ...
    'ThetaVariance', 1e6, ...
    'Beta', 1, ...
    'BetaVariance', 1e6, ...
    'Phi', -1, ...
    'NoiseVariance', (0.1*threshold/(threshold + 1))
    ^2, ...
    'LifeTimeUnit', 'seconds',...
    'SlopeDetectionLevel', 0.05);
```

Listing 1: Implementazione in MATLAB del Degradation Model

Dati i risultati poco soddisfacenti uniti ad una maggior complessità computazionale, si è preferito introdurre un nuovo modello, più semplice da comprendere. L'implementazione relativa al nuovo modello è visibile nel listato Listing.2.

```
mdl = exponentialDegradationModel(...
'LifeTimeUnit', 'seconds',...
'NoiseVariance', (0.1*threshold/(threshold + 1))^2,
    ...
'SlopeDetectionLevel', 0.01);
```

Listing 2: Implementazione in MATLAB del Degradation Model

Nella fattispecie il *Degradation Model* opera con una singola *feature* per volta, ecco quindi che si hanno due possibili strategie, la prima prevede di considerare singolarmente le *feature* scelte precedentemente e verificare il comportamento del modello per ciascuna di esse e infine effettuare quella che è una "media" dei possibili risultati. La seconda strategia, è quella di ridurre lo spazio dimensionale delle *feature*, attraverso un algoritmo di PCA (Principal Component Analisys), la quale permette di "fondere" le varie *feature* in un'unica *feature* che possiamo identificare come *Health Indicator*, ovvero lo stato di salute del filtro. Nel corso del progetto da noi preso in esame sono state implementante entrambe le strategie.

B. Similarity Model

Un *Residual Similarity Model* [2] è utilizzato per stimare la vita utile residua (RUL) di un componente utilizzando un modello di somiglianza basato sul confronto di residui. Questo modello è utile quando si hanno profili di degradazione per un insieme di componenti simili, come macchine multiple prodotte con le stesse specifiche, e si conosce la dinamica del processo di degradazione. I dati storici per ogni membro del gruppo di dati sono dotati di un modello di struttura identica. I dati di degradazione del componente di prova sono utilizzati per calcolare errori di previsione in 1-step, o residui, per ogni modello di insieme. L'ampiezza di questi errori indica quanto il componente di prova sia simile ai membri corrispondenti dell'*ensemble*.

```
mdl = residualSimilarityModel(...
    'Method', 'poly2',...
    'Distance', 'absolute',...
    'NumNearestNeighbors', height(trainData),...
    'Standardize', 1);
```

Listing 3: Implementazione in MATLAB del Similarity Model

VI. Metriche per la Valutazione

L'errore viene considerato come la differenza tra RUL predetta e RUL reale, la valutazione dell'errore del modello è stata realizzata durante il processo di *testing*.
Si sono considerati tre indici principali relativi all'errore, nella fattispecie:

- Media Errore

$$error_{Mean} = \frac{\sum_{i=0}^{N}(RUL_{PREDICT} - RUL_{TRUE})}{N} \tag{3}$$

- Mediana del errore
- Deviazione Standard Errore

$$\sigma = \sqrt{\frac{\sum_{i=0}^{N}(x - \bar{x}_i)^2}{N}} \tag{4}$$

I valori degli errori e le metriche introdotte precedentemente verranno poi graficate all'utente tramite l'utilizzo di *Box Plot* ed un grafico che riporta i valori medi degli errori di predizione e i valori della deviazione standard. Tali valori saranno suddivisi per diverse percentuali di utilizzo del *test set* su cui si sta valutando il modello (i grafici verranno mostrati nella sezione Sez. VIII).
Un'ultima doverosa precisazione è inerente alla questione relativa alla percentuale di utilizzo dei dati del *test set*, durante la fase di *testing*. Infatti, si sono utilizzate percentuali diverse di dati di test per vedere la risposta dei modelli a queste diverse percentuali. In particolare, si è inizialmente utilizzati i dati di test fino al 50% della durata complessiva, successivamente al 70% e infine al 90%; così da osservare come ciò possa influire sul risultato finale.

VII. Codice

Come già accennato precedentemente il codice è stato interamente sviluppato tramite il tool MATLAB fornito da Mathworks. Per una maggior comprensione, replicabilità dei test fin qui svolti e adattabilità del codice, abbiamo codificato tale progetto secondo una struttura modulare. Ciascun "modulo", sarà dedito ad una specifica funzione.
Nello specifico, troviamo un file che si occupa della fase di ETL: cioè dell'importazione dei file CSV iniziali, dei *clipping*, dello *shift* dei segnali e dell'aggregazione dei diversi file CSV (tali funzioni sono state descritte nella sezione III).
Troviamo poi un file che si occupa del *preprocessing* delle *feature* estratte dal DFD. Tale modulo modifica il formato delle *feature* estratte così da rendere possibile il successivo utilizzo nell'allenamento dei diversi modelli.
Abbiamo poi tre diverse cartelle, una prima cartella è relativa al *Similarity Model*, contiene tre programmi principali (uno per ogni tipologia di dataset: Small, Large e Small+Large) e le funzioni di servizio utilizzate dai programmi principali. Le altre due cartelle sono relative al *Degradation Model*, nello specifico il primo fa utilizzo della PCA (Principal Component Analysis) mentre il secondo utilizzata una singola *feature* per volta.
Nella figura Fig. 5 è visibile la struttura del codice del progetto sviluppato.

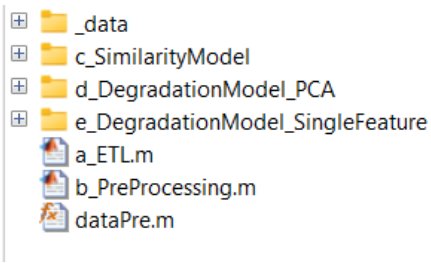


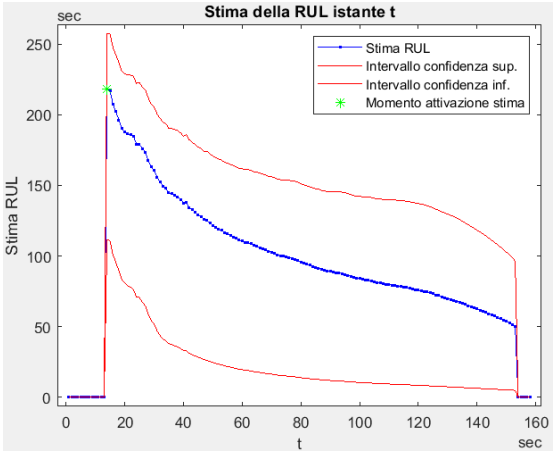
Fig. 5: Struttura Codice Progetto

VIII. Results

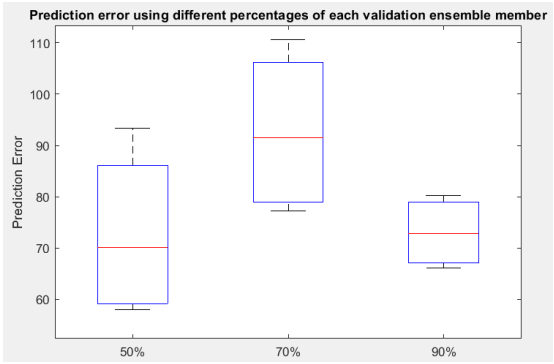
A. Degradation Model

In questa sezione verranno riportati i risultati ottenuti utilizzando il *degradation model* nelle due varianti, con PCA e utilizzando le singole *feature*.

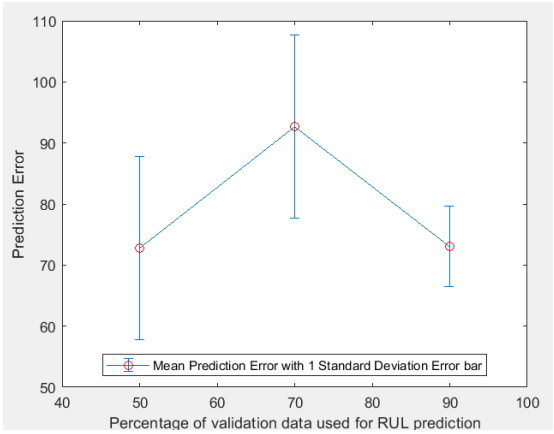
1) PCA: Nella figura Fig.6 vengono riportati i risultati di test ottenuti dal *Degradation Model* che utilizza la PCA.
Nello specifico, la figura Fig.6a raffigura la RUL predetta dal modello, sono inoltre visibili gli intervalli di confidenza e il momento di attivazione della predizione.
Nella figura Fig.6b tramite il *box plot* si rappresentano i valori ottenuti per gli errori della stima della RUL, considerando le diverse percentuali di utilizzo dei dati di test (50%, 70% e 90%).



(a) RUL Predetta

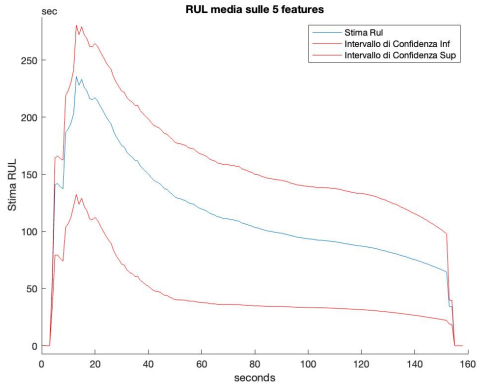


(b) Box Plot Errori

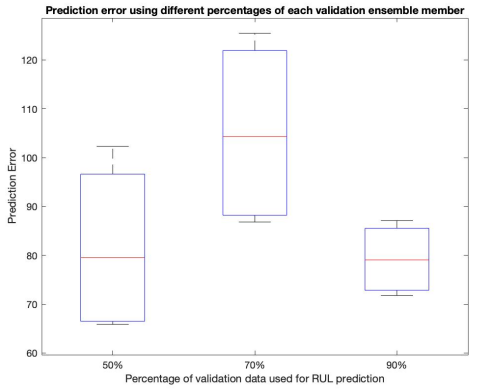


(c) Rappresentazione indici errori

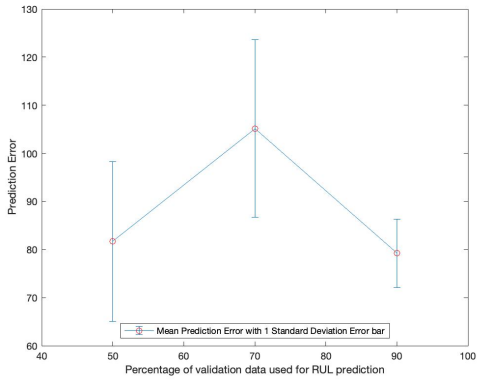
Fig. 6: Risultati Degradation Model (PCA) Small+Large Test Set



(a) RUL Predetta



(b) Box Plot Errori



(c) Rappresentazione indici errori

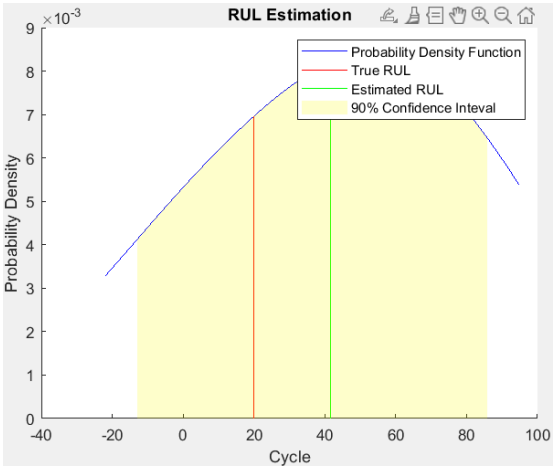
Fig. 7: Risultati Degradation Model (Singole Features) Small+Large Test Set

2) Con Singole Feature: In questa sezione si vanno ad analizzare i risultati del modello considerando le diverse *feature* in modo separato. Il modello *exponential degradation* è stato implementato considerando le diverse feature, successivamente al fine di fornire un visione olistica del problema, si è andati ad effettuare quella che è una media dei vari risultati. Nella figura Fig.7 è possibile visionare i vari risultati.

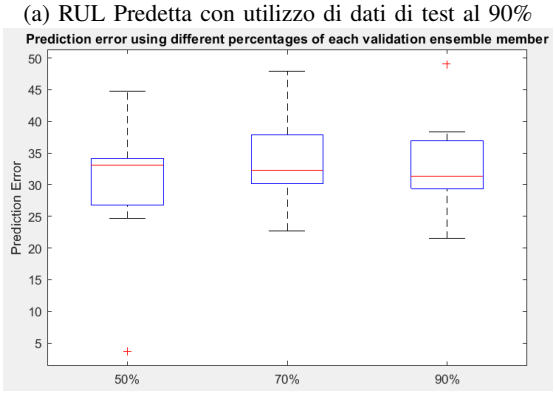
B. Similarity Model

Nella figura Fig. 8 si possono vedere i risultati ottenuto dal *Similarity Model* considerando il *dataset Small+Large*.
In particolare, nella figura Fig. 8a si possono vedere i valori di RUL predetta (in verde) e della RUL reale (in rosso), utilizzando il 90% dei dati di un esperimento del test set.
Nella figura Fig. 8b si può osservare, tramite i box plot, i valori degli errori calcolati considerando rispettivamente il 50%, 70%

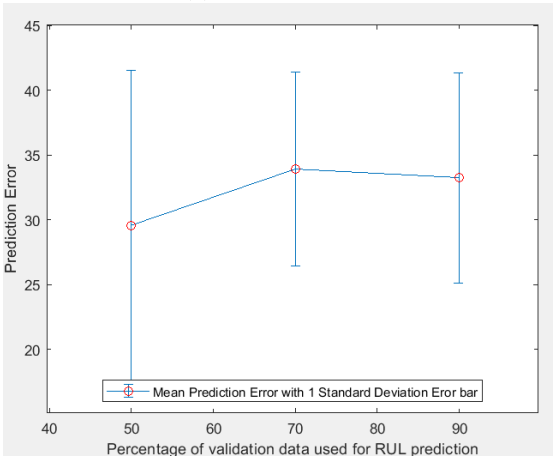
e 90% dei dati di test.
Nella figura Fig. 8c si può osservare i valori medi degli errori e la loro deviazione standard, anche in questo caso considerando percentuali differenti di dati di test.



(a) RUL Predetta con utilizzo di dati di test al 90%



(b) Box Plot Errori



(c) Rappresentazione indici errori

Fig. 8: Risultati Similarity Model Small+Large Test Set

IX. Conclusioni

In conclusione possiamo affermare che l'obiettivo del progetto, ovvero ottenere un *Similarity Model* e un *Degradation Model* per la stima della RUL del filtro, è stato raggiunto.
Come riportato nella sezione Sez. VIII i risultati di tali modelli sono di bontà limitata. Come si può osservare l'errore medio di predizione del *Degradation Model* si aggira intorno gli 80 secondi per il *dataset Small+Large*. Risultati migliori si ottengono con il *Similarity Model*, questo infatti presenta un errore medio di predizione di 30 secondi.
Si sono ottenuti dei risultati mediocri in quanto i due modelli non riescono a generalizzare a sufficienza dai dati che essi apprendono. Si ha infatti che nei dati di test utilizzati sono presenti valori di *solid-ratio* non presenti nel *training set* e questo può portare a questi risultati di bontà limitata.
Per giustificare questa nostra affermazione si sono eseguiti dei test ulteriori: utilizzando la *K-Fold Cross Validation* si è diviso il *training set* in modo da ottenere da questo il *test set*, così da utilizzare dati di test con stessi valori di *solid-ratio* e vedere i risultati dei modelli precedenti sui dati così ottenuti.
Questi test sono stati realizzati utilizzando il *dataset Small + Large* e il parametro K della *K-Fold Cross Validation* settato a 7. Nel caso del *Degradation Model* si è effettuato il test con la versione che utilizza la PCA.
Come si può osservare dalla figura Fig. 9 si ha un netto miglioramento del *Similarity Model*, con un errore medio che si avvicina allo zero all'aumentare della percentuale dei dati di test utilizzati dal modello.
Dalla figura Fig. 10 si può osservare che si ha un miglioramento anche nel caso del *Degradation Model*, anche se di entità minore rispetto al caso del *Similarity Model*.
Possiamo quindi affermare che nel caso preso in esame in questo progetto il modello da preferire è quello di similarità, anche se nel caso di valori di *solid-ratio* non presenti nel *training set* entrambi i modelli mostrano una scarsa generalizzazione e quindi risultati di bontà limitata.
Un possibile sviluppo futuro di tale lavoro potrebbe essere quello di realizzare il task di stima della RUL del filtro applicando strumenti differenti rispetto ai modelli di similarità e di degradazione, ad esempio utilizzando una *Convolutional Neural Network*[4] così da confrontare i risultati e vedere se la *Convolutional Neural Network* riesce a generalizzare i dati di train e quindi stimare la RUL in maniera migliore rispetto a quanto fatto dal *Similarity Model* e dal *Degradation Model*.

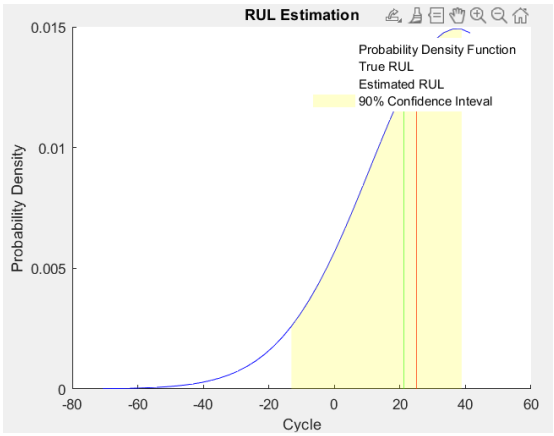
X. Tabelle Risultati

Si riportano di seguito le tabelle con i valori di media, mediana e deviazione standard degli errori di stima ottenuti per i diversi *dataset* di test e dai diversi modelli utilizzati nel progetto.

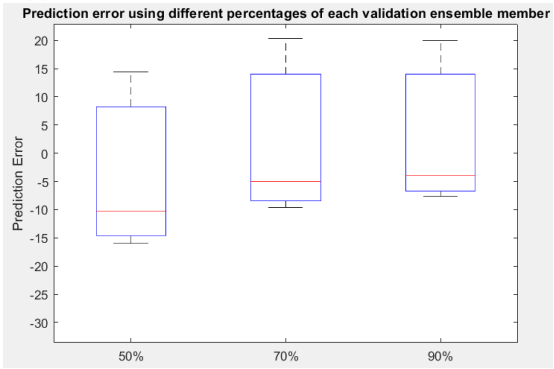
SIMILARITY MODEL (LARGE + SMALL)			
	50%	70%	90%
Error Mean	29,5705	33,9068	33,2466
Error Median	33,1114	32,2628	31,3433
Error SD	11,9256	7,4941	8,1118

TABLE IV: Similarity Model Large+Small

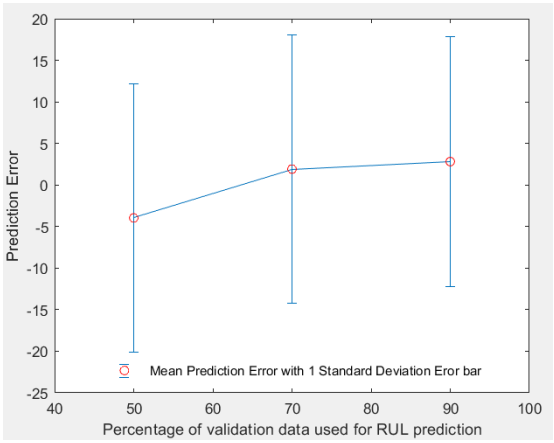
[4]https://it.mathworks.com/help/predmaint/ug/remaining-useful-life-estimation-using-convolutional-neural-network.html



(a) RUL Predetta con utilizzo di dati di test al 90%

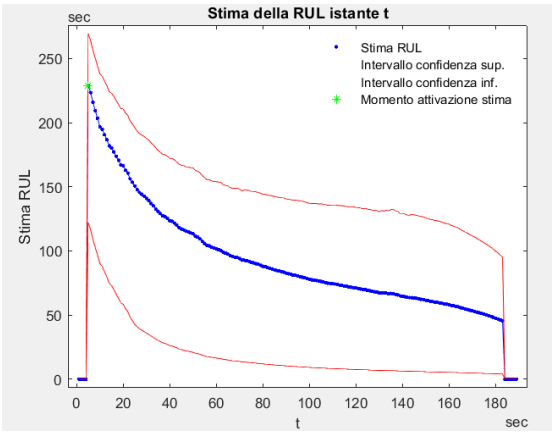


(b) Box Plot Errori

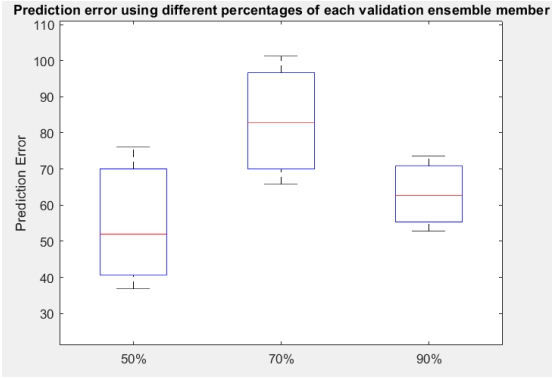


(c) Rappresentazione indici errori

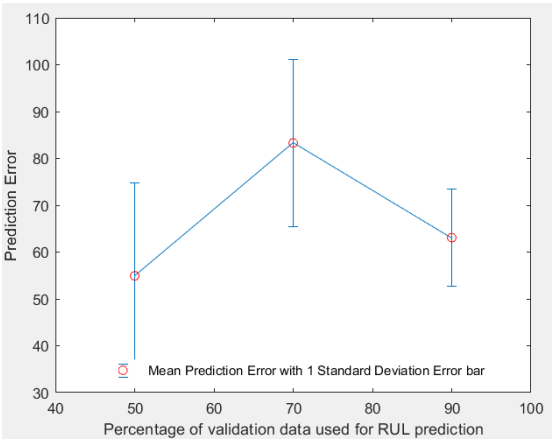
Fig. 9: Risultati Similarity Model Small+Large K-Fold



(a) RUL Predetta con utilizzo di dati di test al 90%



(b) Box Plot Errori



(c) Rappresentazione indici errori

Fig. 10: Risultati Degradation Model PCA Small+Large K-Fold

SIMILARITY MODEL (SMALL)

	50%	70%	90%
Error Mean	52,7698	54,1440	52,7444
Error Median	53,4844	54,3527	53,6900
Error SD	3,9325	2,7936	4,1390

TABLE V: Similarity Model Small

SIMILARITY MODEL (LARGE)

	50%	70%	90%
Error Mean	17,7630	18,1226	18,2295
Error Median	18,0578	18,2063	18,2163
Error SD	70%179	0,4990	0,3313

TABLE VI: Similarity Model Large

DEGRADATION PCA MODEL (LARGE + SMALL)			
	50%	70%	90%
Error Mean	72,7621	92,6608	73,0042
Error Median	70,0865	91,4674	72,8303
Error SD	14,9809	14,9852	6,5842

TABLE VII: Degradation Model PCA Large+Small

DEGRADATION PCA MODEL (SMALL)			
	50%	70%	90%
Error Mean	95,8694	118,0861	71,3898
Error Median	95,8063	117,7902	71,5798
Error SD	6,6436	3,3695	0,6113

TABLE VIII: Degradation Model PCA Small

DEGRADATION PCA MODEL (LARGE)			
	50%	70%	90%
Error Mean	87,5425	102,0876	94,4844
Error Median	87,5395	102,5310	94,2424
Error SD	2,2412	1,7347	1,0351

TABLE IX: Degradation Model PCA Large

DEGRADATION 5F MODEL (LARGE + SMALL)			
	50%	70%	90%
Error Mean	81,6739	105,1485	79,2206
Error Median	79,8269	104,3894	79,1434
Error SD	16,5794	18,4145	7,0809

TABLE X: Degradation Model Large+Small

DEGRADATION 5F MODEL (SMALL)			
	50%	70%	90%
Error Mean	81,6739	105,1485	79,2206
Error Median	79,8269	104,3894	79,1434
Error SD	16,5794	18,4145	7,0809

TABLE XI: Degradation Model Small

DEGRADATION 5F MODEL (LARGE)			
	50%	70%	90%
Error Mean	81,6739	105,1485	79,2206
Error Median	79,8269	104,3894	79,1434
Error SD	16,5794	18,4145	7,0809

TABLE XII: Degradation Model Large

REFERENCES

[1] Mathworks. *Exponential degradation model for estimating remaining useful life*. Last accessed 28/01/2022. 2022. URL: https://it.mathworks.com/help/predmaint/ref/exponentialdegradationmodel.html.

[2] Mathworks. *Residual comparison-based similarity model for estimating remaining useful life*. Last accessed 28/01/2022. 2022. URL: https://it.mathworks.com/help/predmaint/ref/residualsimilaritymodel.html.

[3] PHME20. "FIFTH EUROPEAN CONFERENCE OF THE PROGNOSTICS AND HEALTH MANAGEMENT SOCIETY 2020. PHME20 Data Challenge. An experimental filtration system is the subject of the competition and the challenge is to create high-performance prognostic models to estimate Remaining Useful Life (RUL)." In: *phmsociety* (). DOI: http://www.phmsociety.org/journal/.