SBDIO I4.0
Big Data
for Industry

O4.3

Prototipo degli algoritmi per la rilevazione delle anomalie e l'identificazione del guasto dei sistemi di produzione (non online e online)

Prototipo degli algoritmi per la rilevazione delle anomalie e l'identificazione del guasto dei sistemi di produzione (non online e online)

Code	O4.3
Date	31/05/2021
Type	Confidential
Participants	CIRI-MAM
Authors	Alberto Regattieri (CIRI-MAM), Francesca Calabrese (CIRI-MAM), Raffaele Piscitelli (CIRI-MAM)
Corresponding Authors	Alberto Regattieri

POR FESR EMILIA-ROMAGNA 2014/2020
UNIONE EUROPEA
Fondo europeo di sviluppo regionale
RegioneEmilia-Romagna

Abstract

Il paradigma dell'Industria 4.0 prevede la connessione tra il mondo fisico e quello digitale e l'utilizzo di tecnologie quali IoT, edge computing, cloud computing e Machine Learning per la realizzazione delle cosiddette fabbriche intelligenti. La manutenzione predittiva rappresenta uno dei pilastri del paradigma, che può trarre il massimo beneficio dall'utilizzo di queste tecnologie. Date le difficoltà riscontrate da molte industrie, soprattutto relative alla scarsa possibilità di raccogliere dati strutturati per l'applicazione dei modelli supervisionati di Machine Learning, è necessario predisporre un sistema che sfrutti le tecnologie già disponibili al fine di facilitare la raccolta di dati. In particolare, la classificazione in real-time degli stati in cui lavora un impianto, sia nominali che anomali, può aiutare nella fase successiva di training dei modelli e al contempo fornire delle prime informazioni sul suo stato di salute in qualsiasi istante temporale. Gli algoritmi di apprendimento non supervisionato, che puntano a combinare ciò che è noto con ciò che non lo è, rappresentano un potenziale strumento per il raggiungimento di tale obiettivo. Il presente report si focalizza sullo sviluppo e soprattutto sull'applicazione di alcuni algoritmi semi-supervisionati e in streaming di *anomaly detection* e *novelty detection*, che hanno l'obiettivo di riconoscere in tempi brevi comportamenti anomali o diversi da quelli noti.

SBDIO I4.0
Big Data
for Industry

Indice

Il contesto di riferimento

Alla base del paradigma Industria 4.0 si trovano elementi chiave come l'interconnessione tra sistemi fisici e digitali, utilizzo di macchine intelligenti, analisi complesse sui Big Data e analisi real-time. Lo sviluppo di tecnologie come l'Internet of Things (IoT), il cloud computing e gli algoritmi di Machine Learning (ML) rendono possibile l'applicazione di questi concetti all'interno di realtà industriali con l'obiettivo di rendere le proprie fabbriche intelligenti [1]. All'interno dei numerosi campi di applicazione, la manutenzione degli impianti è una delle aree che più può trarre vantaggio da tali tecnologie. La possibilità di monitorare componenti e sistemi complessi, di raccogliere dati ed estrarre informazioni attraverso algoritmi più o meno complessi, di determinare la loro condizione di salute in qualsiasi istante di tempo rappresenta sicuramente un'opportunità per diverse industrie, che possono ridurre i costi dovuti ai fermi impianto non previsti e quelli dovuti ad una non appropriata gestione dei ricambi [2].

Nelle smart factory, la raccolta dati può infatti avvenire attraverso opportuni sensori installati sui macchinari che misurano grandezze quali vibrazioni, emissioni acustiche, correnti, temperature, pressioni, in grado di fornire informazioni sullo stato del componente. Nella forma più tradizionale, la manutenzione su condizione si basa sull'analisi di questa tipologia di dati, in particolare nel dominio della frequenza, per identificare soglie oltre cui il componente è da considerarsi guasto. Forme più evolute, invece iniziano ad utilizzare tecniche di Machine Learning, o apprendimento automatico, per la classificazione della tipologia di guasto [3]. Si passa dunque al concetto di diagnostica: i dati raccolti attraverso simulazioni in diversi comportamenti di guasto vengono opportunamente trasformati e dati in pasto ad algoritmi di classificazione, che imparano le relazioni che sussistono tra essi e la classe di guasto cui si riferiscono e, sulla base di tali relazioni, classificano le osservazioni future [4]. Si tratta, tuttavia, di un approccio che richiede molti dati di input, la conoscenza di tutte le possibili modalità di guasto e la possibilità di simulare tali modalità, seppur restituendo l'output, ovvero la classe di guasto, una volta che il guasto si è già verificato. In altre parole, anche con queste tecniche evolute di analisi dati, non si riesce a prevenire il guasto. Si arriva quindi a parlare di manutenzione predittiva e prognostica. Quello che cambia è il tipo di informazione in input e in output de i modelli di apprendimento. Infatti, l'input è rappresentato da una grandezza in grado di descrivere l'evoluzione del guasto, chiamato indicatore di salute, mentre l'output non è più una classe, ma una funzione continua, che prende il nome di vita utile residua. In questa ottica, la diagnostica ancora può essere utile per suddividere la vita del componente in diversi stadi, ma l'obiettivo non è più conoscere se e quale guasto si è verificato, quanto prevedere, durante il funzionamento della macchina, il momento in cui si guasterà, sulla base dell'attuale stato di salute [5].

Nonostante in letteratura si trovino centinaia di studi relativi a modelli ed approcci per la manutenzione predittiva, conoscere lo stato di salute di una macchina in un determinato momento non è banale, soprattutto considerando le diverse condizioni in cui può operare e l'influenza dell'ambiente esterno sui segnali prelevati [6]. Se da un lato la letteratura degli ultimi mira a trovare approcci indipendenti dalla condizione operativa, dall'altro le aziende possono trarre vantaggio dal raccogliere in modo più strutturato e conoscere come le condizioni al contorno influenzino il comportamento.

Di conseguenza, risulta vantaggioso associare ad ogni dato/insieme di dati un'etichetta relativa anche alla condizione operativa a cui sono associati. Inoltre, le tecniche che puntano all'eliminazione delle variabili esterne, in particolare quelle note come tecniche di transfer learning, presentano, sia dal

punto divista teorico che pratico, ancora molti limiti. Pertanto, bisogna che si inizi a pensare ad un sistema in grado anzitutto di facilitare la raccolta dei dati nel modo più strutturato possibile, grazie all'ausilio delle tecnologie già esistenti e di modelli consolidati. L'applicazione di modelli complessi e il loro miglioramento dal punto di vista sia di accuratezza della previsione che di complessità computazionale sarà possibile una volta che si raccoglie una maggior quantità di dati [7].
In quest'ottica, diventa necessario implementare modelli in grado di riconoscere comportamenti già noti e comportamenti non noti, in modo da poter utilizzare questi ultimi per il re-training dei modelli. In letteratura, queste attività vengono definite anomaly detection e novelty detection, che si basano sull'apprendimento semi-supervisionato. Nel primo caso, l'obiettivo è identificare comportamenti anomali rispetto a quelli immediatamente precedenti. L'anomaly detection, pertanto, può essere visto come un problema di classificazione binaria, in cui un'osservazione può essere classificata come normale o anomala, e in cui il training avviene soltanto sulla condizione normale.

La novelty detection, invece, ha l'obiettivo di identificare un comportamento anomalo rispetto a quanto visto fino a quel momento. Si tratta quindi di un problema di classificazione multi-classe, in grado di classificare differenti tipi di anomalie, di tipologia nota o non nota. In altre parole, gli approcci di novelty detection rientrano nella categoria degli approcci incrementali, che una volta indentificato un comportamento non noto, riescono ad introdurre una nuova classe e far sì che da quel momento in poi, punti "simili" non saranno più considerati anomalie.
Di seguito si riporta una breve descrizione degli approcci maggiormente utilizzati per i due tipi di problemi e l'applicazione di tali approcci a segnali di vibrazione raccolti da un prototipo costruito nel laboratorio del Dipartimento di Ingegneria Industriale dell'Università di Bologna.
Verrà inoltre mostrato un caso relativo al calcolo della vita utile residua di uno dei componenti del prototipo.
Essendo questo l'ultimo dei tre report all'interno del progetto, molte nozioni e concetti alla base dei modelli saranno omessi, in quanto oggetto dei report precedenti.

Cenni sugli algoritmi sviluppati

Gli algoritmi per la rilevazione delle anomalie e l'identificazione di nuovi comportamenti/diverse condizioni operative, possono essere sostanzialmente suddivisi in due gruppi:

- Algoritmo basato sul calcolo ricorsivo della densità
- Algoritmi di Machine Learning e Deep Learning

Algoritmo basato sul calcolo ricorsivo della densità. Il concetto alla base di questo algoritmo è la densità, che viene calcolata in modo ricorsivo ogni volta che una nuova osservazione si rende disponibile, al fine di identificare in real-time se essa differisce da quelle precedenti. La densità in pratica misura la vicinanza nello spazio delle features di una osservazione a tutte le osservazioni precedenti in un certo istante temporale. Questo algoritmo prevede che il sistema si possa trovare in due stati: normale e anomalo. Per ogni osservazione quindi si calcolano la densità globale e una locale, che va a zero ogni qualvolta il sistema cambia stato. Il cambiamento di stato da normale e anomalo avviene quando la densità globale è minore di quella locale per un certo numero di osservazioni. Se invece per un certo numero di osservazioni si verifica la situazione opposta, lo stato del sistema torna ad essere normale. Questo algoritmo, noto come Autonomous Anomaly Detection (AAD) [8], consente

soltanto di indentificare comportamenti anomali rispetto ai comportamenti immediatamente precedenti. Se invece l'obiettivo è identificare cambiamenti di stato, anche riferiti a stati noti (condizioni operative/guasti), è possibile aggiungere un algoritmo di clustering ogni volta che avviene il cambiamento di stato. Utilizzando lo stesso principio di calcolo ricorsivo della densità e il concetto di distanza, l'algoritmo di clustering online, noto come Autonomous Data Partitioning (ADP) [9], calcola, ad ogni cambio di stato, la distanza tra il punto corrente e i centri dei cluster esistenti. Se la distanza tra il punto e il centro del cluster più vicino è minore della metà della distanza media tra tutti i punti considerati fino a quel momento, allora il punto viene assegnato a tale cluster. I parametri di questo cluster, ovvero il numero di punti ad esso assegnati e il suo centro, vengono aggiornati, in modo da considerare il punto appena assegnato. Altrimenti, il punto corrispondente al cambiamento di stato genera un nuovo cluster.

In questo modo non solo è possibile riconoscere un cambiamento di stato da normale ad anomalo, ma è anche possibile sapere se l'anomalia si è già verificata in passato.

Algoritmi di Machine Learning e Deep Learning. In questa categoria, si possono distinguere modelli più tradizionali di Machine Learning, come l'algoritmo a vettori di supporto (SVM) e le reti neurali non profonde , e modelli di Deep Learning, come la rete Long-Short Time Memory (LSTM) e le reti convoluzionali (CNN), che rispetto alle reti non profonde presentano un numero elevato di livelli nascosti.

L'SVM si basa sulla teoria dell'apprendimento statistico e sulla minimizzazione di una funzione di rischio. L'obiettivo dell'SVM è costruire un iperpiano che separa lo spazio delle features in una o più classi con l'obiettivo di massimizzare la distanza tra le classi. In genere l'SVM è un algoritmo di apprendimento supervisionato, usato quindi per la classificazione di due o più comportamenti. Tuttavia, una variante dell'SVM, chiamata One-class SVM, viene spesso utilizzata sia per l'*anomaly detection* che per la *novelty detection* [10].

La rete neurale utilizzata qui, è una rete *feedforward* che utilizza l'algoritmo di back propagation per l'apprendimento. Una rete neurale è un insieme di nodi e connessioni, a cui sono associati a dei pesi. Una rete è *feedforward* quando le connessioni sono concesse solo tra nodi appartenenti a due livelli consecutivi. Inizialmente, i pesi associati alle connessioni tra i livelli sono generati in modo random nell'intervallo tra 0 e 1. L'algoritmo di back propagation consiste nella determinazione dei pesi al fine di minimizzare l'errore tra il valore attuale della classe e quello predetto dalla rete. Come l'SVM, anche l'apprendimento di questo tipo di reti neurali è di tipo supervisionato. Tuttavia alcune varianti vengono utilizzate anche per l'*anomaly detection* [11].

La LSTM appartiene alla categoria delle reti neurali ricorrenti. A differenza delle reti *feedfarward,* le reti ricorrenti abilitano connessioni anche con nodi appartenenti ai livelli precedenti. La LSTM, in particolare, è in grado di processare non solo osservazioni singole, ma anche insieme di punti e pertanto è molto appropriata per l'analisi delle *time series*. In genere, una LSTM include un cella in grado di memorizzare dei dati per un certo intervallo di tempo, e tre gates, uno di input, uno di output e uno di *forget*, che regolano il flusso di dati verso o da la cella di memoria. Più nel dettaglio, la cella di memoria determina se c'è stato un cambio rispetto all'informazione contenuta precedentemente, mentre la rete calcola il valore da memorizzare nella memoria in un preciso istante, basandosi sull'informazione a disposizione in quel momento e il valore di input della cella di memoria [12]. Grazie quindi alla possibilità di ricordare ciò che è successo in precedenza, la LSTM è molto utilizzata per la *novelty detection*.

Le reti convoluzionali (CNNs), infine, sono nate principalmente per analizzare e classificare immagini e foto, in quanto capaci di operare soltanto con dati in 2D. Recentemente, sono state sviluppate reti convoluzionali per dati ad una dimensione, come i segnali, e si è dimostrato che in molti casi, data la

riduzione della complessità computazionale, sono piuttosto adatte per applicazioni di *anomaly detection* in real-time e su dispositivi mobili. Inoltre, le CCN sono particolarmente efficaci in mancanza di molti dati di training [13].

Sulla base di queste due diverse tipologie di algoritmi, si sono sviluppati due approcci differenti, entrambi volti ad identificare, in real-time, comportamenti non noti/diversi da quelli rilevati in un certo istante. Mentre il primo approccio non richiede alcuna fase di training e può essere applicato in qualsiasi momento e condizione del sistema, il secondo prevede una fase di training volta

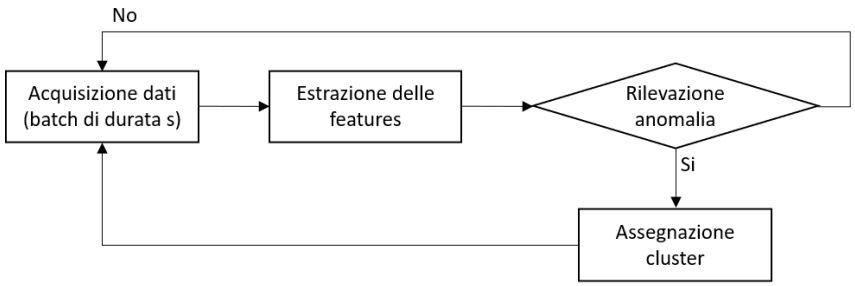

Figura 2 Approccio basato sul calcolo ricorsivo della densità

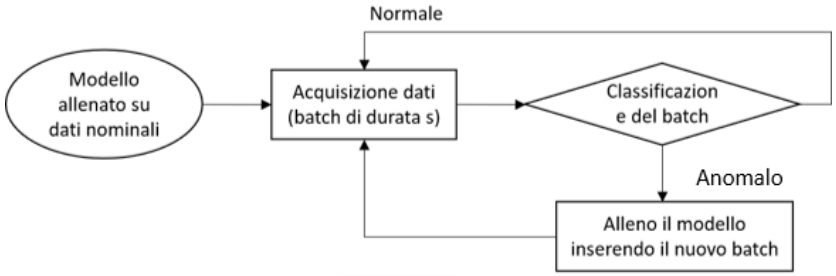

Figura 1 Approccio basato su algoritmi di Machine Learning e Deep Learning

all'apprendimento da parte dei modelli di Machine Learning e Deep Learning almeno della condizione nominale di funzionamento. Inoltre, mentre il primo approccio richiede necessariamente l'estrazione delle features come fase preliminare, il secondo approccio può essere applicato direttamente ai segnali grezzi. Gli approcci sviluppati sono mostrati in Figura 1 e Figura 2, rispettivamente. L'utilizzo di uno o dell'approccio dipende sostanzialmente dalla tipologia di dati disponibili, dalla capacità computazionale del dispositivo di implementazione e dall'accuratezza della predizione che si vuole ottenere.

Applicazione su dati reali

In questa sezione, si riporta l'applicazione degli approcci descritti per l'*anomaly detection* e la *novelty detection* ad un dataset raccolto da un prototipo costruito nel laboratorio di Ingegneria Industriale dell'Università di Bologna. L'obiettivo della presente analisi è quello di fornire un confronto tra le diverse tipologie di tecniche e modelli sviluppati, in termini di capacità di riconoscere comportamenti nuovi, nominali e di guasto, in tempi accettabili per un sistema continuo di monitoraggio e di diagnostica online.

Il prototipo di laboratorio

Per la costruzione del prototipo, si è scelto di impiegare diversi elementi, quali cuscinetti, ruote dentate, pulegge e cinghie, che sono presenti in qualsiasi macchinario industriale moderno. Scelte le componenti meccaniche, si è passato alla fase di scelta di quelle che sono le componenti che garantiscono una variabilità oltre che una ripetibilità nei test da effettuare. Sono stati inseriti quindi un motore elettrico con la possibilità di due diverse configurazioni, un freno elettromagnetico regolabile, un sistema di messa in tiro della cinghia ed un disco metallico removibile simulante un carico sul secondo albero del prototipo. Il motore elettrico utilizzato è un motore trifase a 6 poli con potenza pari a 0,55 kW, coppia di 5,7 Nm e velocità di rotazione pari a 920 rpm. Il sistema frenante utilizzato è un freno a polveri elettromagnetiche con coppia frenante regolabile nell'intervallo 0-7,5 Nm tramite un potenziometro posto sull'interfaccia di regolazione dello stesso. Per la raccolta dati, sono stati scelti 3 accelerometri triassiali Dytran 3093D3 con tecnologia IEPE, frequenza di campionamento impostata a 12,8 kHz per asse e range di accelerazione di 500 Gpeak. I sensori di accelerazione sono stati inoltre collegati ad un computer per l'acquisizione dati attraverso schede di acquisizione I/O NI9230 a tre canali dotati di una capacità di campionamento max di 12,8 kS/s per ogni canale.

Per il test degli approcci descritti nella sezione precedente, sono stati eseguiti dei test sul prototipo in 4 diverse condizioni operative, i cui parametri e la cui durata sono riassunti in Tabella 1. La durata dell'ultimo test, chiamato setting 4, è molto inferiore rispetto alle altre. Questo è dovuto al verificarsi

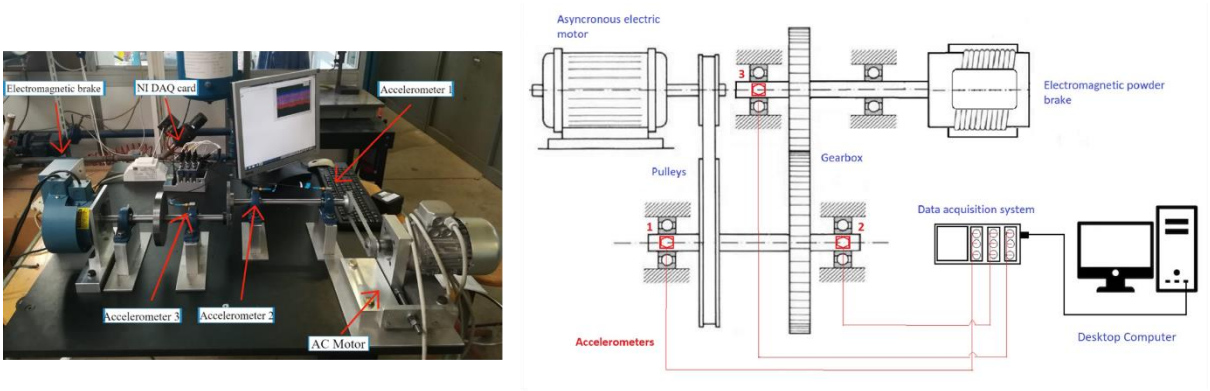

Figura 3 Il prototipo e lo schema meccanico corrispondente

di un guasto al motore che non era previsto.

Gli obiettivi, dunque, del sistema di diagnostica online che si vuole implementare sono due:

1. Si vuole riconoscere il momento in cui avviene il cambio di stato del sistema, ovvero quando viene modificato il setting della macchina, e distinguere ciò che è noto da ciò che non è noto
2. Si vuole riconoscere l'anomalia dovuta al guasto improvviso verificatosi alla fine dell'ultimo test

Tabella 1 Parametri durante i test

Condizione operativa	Distanza tra pulegge (mm)	leCoppia frenante (Nm)	Durata (min)	Velocità di rotazione (rpm)
Setting 1	27.33	0.1	70	660

Setting 2	27.33	0.5	150	660
Setting 3	27.54	0.1	70	660
Setting 4	27.54	0.328	30	660

Test degli approcci

Approccio basato sul calcolo ricorsivo della densità. Per questa tipologia di approccio, si sono considerati noti i primi due setting. Pertanto, questi sono stati utilizzati sia per la selezione delle features più rilevanti e sia per il training del modello. Per quanto riguarda la prima fase, data la diversa natura dei due obiettivi, da uno stesso insieme di features estratte nel dominio del tempo su segmenti di segnale di durata uguale a 1 secondo, si sono selezionati due sottoinsiemi diversi. Il primo, che include la media di tre dei nove segnali raccolti, è stato utilizzato per il riconoscimento del cambio di setting; il secondo, costituito soltanto dal fattore di impulso di un segnale, è stato invece utilizzato per il riconoscimento delle anomalie dovute al guasto. Nella fase di training, invece, si sono calcolati i parametri locali di due cluster, uno corrispondente al setting S1 e l'altro corrispondente al setting S2. Dopo queste due fasi preliminari, l'algoritmo viene lanciato su tutto il dataset, i cui risultati sono riassunti in Tabella 2. Per quanto riguarda il primo obiettivo dell'analisi, ovvero il riconoscimento della condizione operativa, i passaggi dal setting S1 al setting S2 e dal setting S2 al setting 3 sono stati rilevati con una latenza di 15 e 10 secondi, rispettivamente. Pertanto, si hanno 15 e 10 osservazioni assegnate a cluster sbagliati. Tuttavia, per il riconoscimento della condizione operativa, la latenza e l'errore ottenuti sono accettabili, in quanto non si tratta di un'attività che non richiede interventi immediati e il numero di punti erroneamente assegnati ad un cluster è molto minore rispetto al totale delle osservazioni che si possono raccogliere durante l'implementazione di un determinato setting. Il passaggio dal setting S3 al setting S4, invece, non viene riconosciuto, in quanto il guasto si verifica subito dopo la modifica del setting e l'algoritmo non riesce a identificare nessuna anomalia. Per quanto riguarda il secondo obiettivo, ovvero la rilevazione dell'anomalia relativa al guasto, l'algoritmo riesce a riconoscere un comportamento anomalo già dopo 170 secondi l'implementazione del setting 4. Inoltre, crea un nuovo cluster, che può implicare quindi non solo un'anomalia ma un vero e proprio cambiamento del comportamento, 90 secondi prima della fermata del sistema dovuta al guasto.

Tabella 2 Risultati dell'algoritmo basato sul calcolo ricorsivo della densità per il riconoscimento del cambio di setting in termini di latenza

Algoritmo	Condizione operativa	Istante di tempo del cambio di setting (reale)	Istante di tempo del cambio di setting (rilevato)	Latenza (sec)
AAD + ADP	S1	-	-	
	S2	4.200	4.215	15
	S3	13.200	13.210	10
	S4	17.400	-	-

Tabella 3 Risultati dell'algoritmo basato sul calcolo ricorsivo della densità per il riconoscimento del guasto in termini di latenza

Algoritmo	Condizione operativa	Istante del guasto (reale)	Istante del guasto (rilevato)	Latenza (sec)
AAD + ADP	Nominale	-	-	-
	Guasto	-	17.570	-
	Fermata del sistema	18.950	18.860	- 90

Approcci basati sugli algoritmi di Machine Learning e Deep Learning**.** Per questa tipologia di approcci, sono stati considerati due scenari: offline e online.

1. Nel primo scenario, i modelli vengono allenati su pochi dati di una singola condizione operativa, per valutare quanto bene riescono a riconoscere come normali gli altri dati della stessa condizione e come anomale le altre tre condizioni. Si tratta di uno scenario tradizionale, in cui il modello deve essere riallenato ogni volta che si presenta una nuova condizione
2. Nel secondo scenario, i modelli vengono valutati sulla base della capacità di incorporare nuova conoscenza, quando disponibile. Questo scenario utilizza un approccio incrementale, in cui viene valutata la capacità di imparare dei comportamenti della macchina che non erano noti al momento del training.

Per ogni scenario, sono state condotte due analisi. Nel primo caso, ogni osservazione viene considerata separatamente. Pertanto, l'accuratezza di un modello viene calcolata come il rapporto tra il numero di osservazioni correttamente classificate e il numero totale di osservazioni. Un secondo livello di analisi considera un batch di dati invece di una singola osservazione. Pertanto, viene introdotta una nuova metrica, definita accuratezza del batch, che considera una classificazione come corretta se tutti i dati del batch sono correttamente classificati. I batch hanno una durata di 10 minuti. I risultati di entrambe le analisi relative allo scenario offline sono riassunti in Tabella 4, dove per ogni condizione con cui il modello è stato allenato, vengono riportate l'accuratezza e l'accuratezza dello batch. Quello che emerge è che con l'SVM si ottengono buoni risultati se si considera l'accuratezza per singola osservazione, ad eccezione del setting S3. Tuttavia, per quanto l'analisi sui batch, l'SVM presenta un'accuratezza molto bassa. Inoltre, è possibile vedere come i modelli l'LSTM e la CNN raggiungano accuratezze superiori al 90% anche quando vengono allenati sul setting S3.

Tabella 4 Risultati degli approcci di Machine Learning e Deep Learning per il riconoscimento delle anomalie (scenario offline)

Algoritmo	Tutte		S1		S2		S3	
	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.
SVM	0.658	0.351	0.982	0.715	0.703	0.290	0.189	0.067
BP-ANN	0.957	0.911	1.000	1.000	0.945	0.906	0.933	0.821
LSTM	0.989	0.944	0.998	0.977	0.984	0.927	0.990	0.942
CNN	0.989	0.939	0.998	0.980	0.984	0.919	0.989	0.933

Per la valutazione dello scenario online, sono state considerate tre diverse configurazioni, riassunte in Tabella 5. Nella configurazione C1, l'allenamento dei modelli è stato effettuato soltanto su un batch del setting S1. Pertanto, l'insieme noto corrisponde ai rimanenti batch del setting S1, mentre tutti i batch di S2, S3 e S4 corrispondono alle nuove condizioni da identificare. Nella configurazione C2, l'allenamento è stato fatto sul primo batch dei setting S1 e S2. Pertanto le condizioni nuove da indentificare sono la S3 e la S4. Infine, nell'ultima configurazione, l'allenamento viene eseguito sul primo batch dei primi 3 setting e l'obiettivo è riconoscere che il setting S4 è anomalo/diverso rispetto a tutti gli altri. I risultati sono mostrati in Tabella 6, dove è stata riportata soltanto l'accuratezza dello batch. I risultati migliori sono stati ottenuti nella configurazione C1, in cui l'allenamento è avvenuto solo sul setting S1 che, essendo molto diversa da tutte le altre, facilita il riconoscimento di comportamenti diversi. Nell'ultima configurazione invece, solo l'LSTM e la CNN, seppur con bassa accuratezza, riescono a riconoscere il setting S4, quello in cui si è verificato il guasto, come anomalo.

Inoltre, mentre i batch dei setting appartenenti alle condizioni note vengono classificati con maggiore accuratezza dalla rete neurale più tradizionale, i setting "nuovi" vengono riconosciuto con un'accuratezza migliori dai modelli di Deep Learning.

Tabella 5 Configurazioni del test dello scenario online

Configurazione	Set per il training	Set per il test	
		Insieme noto	Insieme "nuovo"
C1	S1 (10 min.)	S1 (70 min)	S2, S3, S4 (150 + 70 min + 10 min)
C2	S1, S2 (10 + 10 min.)	S1, S2 (70 + 150 min)	S3, S4 (70 min + 10 min)
C3	S1, S2, S3	S1, S2, S3	S4

Tabella 6 Risultati degli approcci "intelligenti" per la novelty detection (scenario online) nelle diverse configurazioni

Algoritmo	S1			S2			S3		
	Test	Noto	Nuovo	Test	Noto	Nuovo	Test	Noto	Nuovo
SVM	0.715	0.321	0.825	0.618	0.565	0.729	0.790	0.878	0.000
BP-ANN	1.000	1.000	1.000	0.871	1.000	0.600	0.900	1.000	0.000
LSTM	0.997	0.893	1.000	0.935	0.952	0.900	0.743	0.772	0.476
CNN	0.980	0.911	1.000	0.922	0.912	0.943	0.790	0.847	0.286

Sviluppo di algoritmi per la stima della RUL

Il calcolo della vita utile residua di un componente/sistema può essere ricondotto ad un problema di regressione. Una volta individuato l'indicatore di salute ottimo e avendo a disposizione diverse traiettorie per uno stesso guasto, un primo approccio può essere quello di individuare la miglior curva di fitting sulla base dei dati storici e calcolare la vita utile residua, in real-time, sulla base sia dell'andamento dell'indicatore di salute calcolato per il componente in analisi e la curva di fitting. Si noti che l'indicatore è sostanzialmente una feature e, come tale, può essere estratta attraverso le tecniche di analisi dei segnali e/o costruita attraverso tecniche di riduzione della dimensionalità. La caratteristica di tale feature è che deve essere in grado di descrivere il progredire del failure e deve presentare lo stesso andamento per una stessa tipologia di guasto. Pertanto, l'indicatore di salute deve essere monotono ed avere, al punto di rottura, sempre lo stesso valore (o u valore molto vicino). Queste due caratteristiche vengono chiamate in letteratura *monotonicity* e *prognosability* [14].
Di seguito si riporta un esempio di tale approccio. I dati sono stati raccolti dal prototipo e sono relativi alla rottura della cinghia di trasmissione. A seguito di due diversi failure test in due diverse condizioni operative, si sono ricavati gli andamenti dell'indicatore di salute, mostrati in Figura 4. La curva utilizzata per il fitting è un'esponenziale di primo grado. La RUL è stata invece calcolata come la differenza tra il momento in cui è previsto che l'indicatore di salute superi la soglia di guasto (identificata nell'ultimo punto della curva) e il momento in cui si effettua il calcolo. Seppur più tradizionale rispetto ai modelli di Machine Learning, questo metodo consente di ottenere un errore medio sul calcolo della RUL, del 14,2% e del 10%, rispettivamente. L'errore si abbassa ulteriormente se si prende in considerazione solo l'ultima parte della vita. Infatti, mentre all'inizio l'errore è piuttosto

alto, dopo un certo punto inizia a tendere sempre di più a zero. Questo significa che la predizione diventa via via più accurata. Stabilire il momento della prima predizione è un aspetto critico. Gli approcci generalmente utilizzati sono due. La divisione della vita di un componente in diversi stati di salute, grazie all'ausilio degli algoritmi di classificazione, può essere una strada. Tuttavia, è necessario avere la traiettoria completa dell'indicatore di salute, dal momento in cui il componente entra in funzione, al momento del guasto. Seppur efficace e accurato, questo metodo presenta dei limiti soprattutto nei casi di componenti che si usurano in diversi anni. L'altra strada, probabilmente meno accurata, ma utile in mancanza di diversi dati di failure storici, è quello di identificare il punto di inizio di usura attraverso algoritmi di rilevazione delle anomalie.

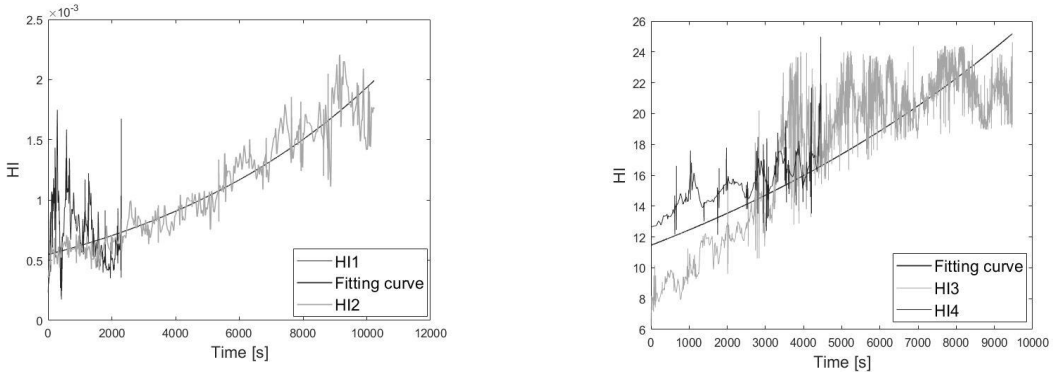

Figura 4 Curva di fitting nelle due diverse condizioni operative

Collaborazioni

Il lavoro sugli algoritmi di Machine Learning e Deep Learning per la rilevazione di anomalie e comportamenti non noti è stato svolto in collaborazione con il gruppo Softech-ICT (UNIMORE)

Un'altra collaborazione con i gruppi CIRI-ICT (UNIBO) e MechLav (UNIFE) si è invece focalizzata sulla raccolta dati in streaming. L'obiettivo della collaborazione è stato quello di integrare il prototipo realizzato in laboratorio con un'architettura cloud in modo da poter realizzare in live streaming non solo l'acquisizione dei dati dai sensori, ma anche la loro condivisione online e il calcolo della vita utile residua di un componente del prototipo. In particolare, si è deciso di focalizzarsi di nuovo sulle cinghie di trasmissione, vista la quantità di dati già raccolti in passato e la maggiore velocità di deterioramento rispetto ad altri componenti quali, cuscinetti ed ingranaggi.

La pipeline di acquisizione e data analytics, mostrata in Figura 5, parte dall'acquisizione effettuata con il software Labview. Di qui si è deciso di procedere creando un gateway in python, il quale monitora la cartella di destinazione dei file in formato lvm in uscita da Labview, ed ogni qual volta trova un nuovo file contenente i dati in arrivo dai 3 accelerometri, lo apre ed estrae le informazioni inviandole

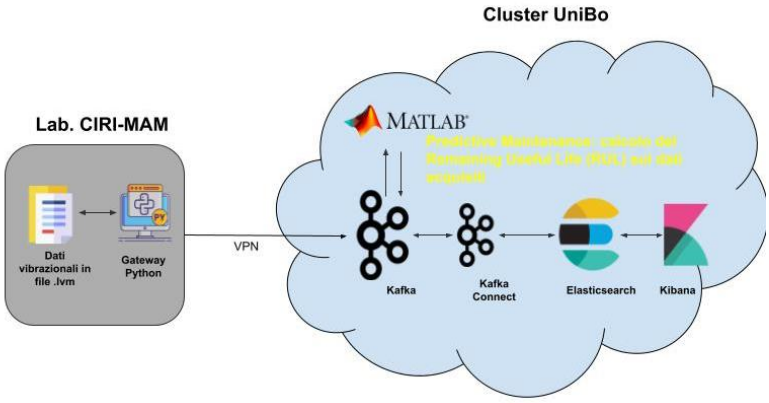

Figura 5 Pipeline acquisizione e data analytics

poi al Cluster remoto dell'UniBO.

I test che sono stati effettuati mostrano che il gateway risulta impiegare, per queste azioni di lettura ed invio informazioni, quasi il doppio del tempo che Labview impiega per generare il file in oggetto. Al momento si sta testando un altro tipo di architettura per eseguire la lettura dei file e l'invio sul cluster che prevede l'utilizzo di un toolkit per python interno a Labview che dovrebbe velocizzare notevolmente queste operazioni. I messaggi letti dal gateway, il quale è attivo grazie all'utilizzo di una VPN, vengono inviati su Kafka dove sono organizzati in 3 diversi topic. Questi topic sono gestiti da Kafka Connect che inoltra i nuovi messaggi ad Elasticsearch dove vengono creati quindi degli index per ogni topic di Kafka. Infine grazie alla creazione di index patterns su Kibana è possibile creare dashboards per la visualizzazione dei dati raw in arrivo dagli accelerometri. Due esempi di dashboard visualizzabile su Kibana sono mostrati in Figura 6 e Figura 7.

Successivamente questi dati raw vengono inviati su una macchina del datacenter UniBO grazie al collegamento con VPN ancora una volta. Qui viene sfruttata la potenza di calcolo della macchina per analizzare i segnali raccolti su Matlab in batch da 1 secondo per volta. Con Matlab viene quindi eseguito un algoritmo il quale estrae un indicatore di salute per ogni secondo di acquisizione dati e poi effettua un'inferenza sulla vita utile rimanente della cinghia di trasmissione basandosi su un modello costruito a partire da altri failures raccolti e presenti nella nostra banca di acquisizione dati. In particolate l'HI utilizzato si basa sul concetto di indice di correlazione, calcolato attraverso l'Eq. (1).

$$HI = \frac{\sum_{i=1}^{N} (x_i - \bar{x}) \,.\, (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N} (x_i - \bar{x})^2} . \sqrt{\sum_{i=1}^{N} (y_i - \bar{y})^2}} \qquad (1)$$

Figura 6 Visualizzazione segnali dell'accelerometro 1 su Kibana

La scelta dell'HI è stata condizionata soprattutto dalla necessità di dover lavorare con pacchetti di dati per volta, quindi risulta utile utilizzare l'indice di correlazione come indicatore di salute con il quale di fatto si pongono a confronto un batch di dati acquisito nelle condizioni nominali della macchina con i pacchetti successivi che rappresentano il deterioramento della componente cinghia, l'andamento quindi di questo HI sarà decrescente nel tempo.

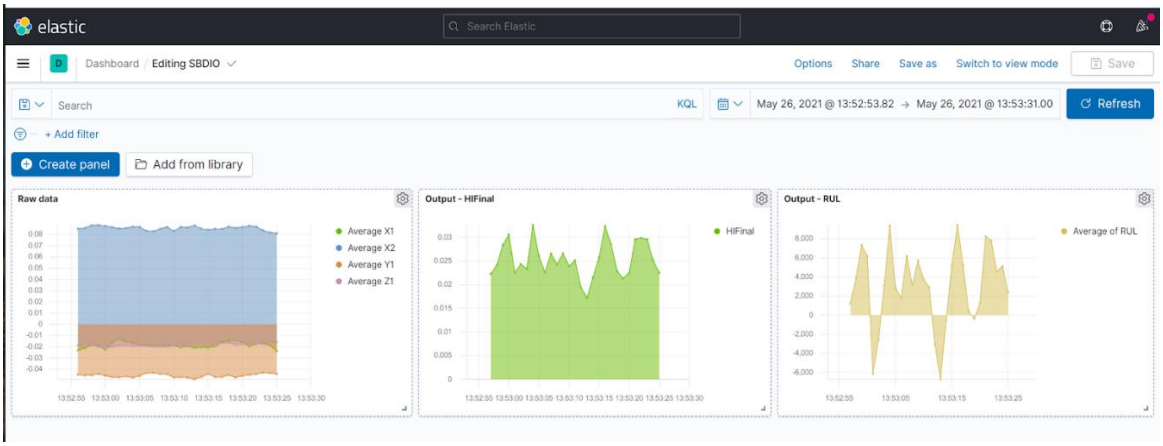

Figure 7 Visualizzazione analytics su dati grezzi e dati elaborati

Conclusioni

La rilevazione delle anomalie e di comportamenti non noti di un componente/macchina è fondamentale per l'implementazione di una strategia manutentiva predittiva. In questo deliverable, abbiamo descritto due diversi approcci, di complessità diversa e con input richiesti diversi. L'applicazione a dati raccolti in ambiente controllato mostra risultati buoni dal punto di vista del riconoscimento dei diversi setting e di guasti improvvisi. L'utilizzo di approcci basati sul calcolo ricorsivo della densità fornisce come output principale un *labelling* automatico, che consente di raggruppare dati simili, facilitando una successiva analisi di tipo supervisionato. Tuttavia, la difficoltà maggiore risiede nell'individuazione delle features che meglio distinguano dati relativi a condizioni diverse. Al contrario, le tecniche di Machine Learning e soprattutto di Deep Learning hanno la capacità di imparare autonomamente rappresentazioni sintetiche ed informative dai segnali, limitando l'attività manuale di estrazione e selezione delle features. Tuttavia, richiedono una maggiore quantità di dati, soprattutto se si vogliono raggiungere accuratezze di classificazione/rilevazione del cambio di stato elevate.

Le aziende ad oggi incontrano diverse difficoltà nel raccogliere dati di guasto, soprattutto nel caso di componenti che si usurano lentamente. È per questa ragione che la condizione necessaria per una possibile applicazione industriale è la predisposizione di una architettura anzitutto che consenta la raccolta di dati più "strutturati" e con un maggiore contenuto informativo. In seguito, è possibile implementare tecniche e modelli via via più complessi e accurati. In particolare, si può partire con applicazioni più semplici, basate su analisi statistiche, come nel caso degli approcci basati sul calcolo ricorsivo della densità e dell'approccio utilizzato per la predizione della vita utile residua. Solo successivamente, una volta disponibili i dati, è possibile applicare tecniche di apprendimento automatico volte a rendere le proprie macchine "intelligenti".

SBDIO I4.0
Big Data
for Industry

Bibliografia

[1] H. Nordal and I. El-Thalji, “Modeling a predictive maintenance management architecture to meet industry 4.0 requirements: A case study,” *Syst. Eng.*, vol. 24, no. 1, pp. 34–50, 2021, doi: 10.1002/sys.21565.

[2] M. Pech, J. Vrchota, and J. Bednář, “Predictive maintenance and intelligent sensors in smart factory: Review,” *Sensors*, vol. 21, no. 4, pp. 1–39, 2021, doi: 10.3390/s21041470.

[3] P. Poór, D. Ženíšek, and J. Basl, “Historical overview of maintenance management strategies: Development from breakdown maintenance to predictive maintenance in accordance with four industrial revolutions,” *Proc. Int. Conf. Ind. Eng. Oper. Manag.*, no. July, pp. 495–504, 2019.

[4] A. K. S. Jardine, D. Lin, and D. Banjevic, “A review on machinery diagnostics and prognostics implementing condition-based maintenance,” *Mechanical Systems and Signal Processing*, vol. 20, no. 7. pp. 1483–1510, Oct-2006, doi: 10.1016/j.ymssp.2005.09.012.

[5] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, “Machinery health prognostics: A systematic review from data acquisition to RUL prediction,” *Mech. Syst. Signal Process.*, vol. 104, pp. 799–834, 2018, doi: 10.1016/j.ymssp.2017.11.016.

[6] F. Calabrese, A. Regattieri, L. Botti, C. Mora, and F. G. Galizia, “Unsupervised Fault Detection and Prediction of Remaining Useful Life for Online Prognostic Health Management of Mechanical Systems,” *Appl. Sci.*, vol. 10, no. 12, p. 4120, 2020, doi: 10.3390/app10124120.

[7] F. Calabrese, A. Regattieri, M. Bortolini, M. Gamberi, and F. Pilati, “Predictive maintenance: a novel framework for a data-driven, semi-supervised, and partially online Prognostic Health Management application in industries,” *Appl. Sci.*, 2021.

[8] B. S. J. Costa, P. P. Angelov, and L. A. Guedes, “Fully unsupervised fault detection and identification based on recursive density estimation and self-evolving cloud-based classifier,” *Neurocomputing*, vol. 150, no. Part A, pp. 289–303, Feb. 2015, doi: 10.1016/j.neucom.2014.05.086.

[9] X. Gu, P. P. Angelov, and J. C. Príncipe, “A method for autonomous data partitioning,” *Inf. Sci. (Ny).*, vol. 460–461, pp. 65–82, 2018, doi: 10.1016/j.ins.2018.05.030.

[10] J. A. Cariño *et al.*, “Fault Detection and Identification Methodology Under an Incremental Learning Framework Applied to Industrial Machinery,” *IEEE Access*, no. i, pp. 1–1, 2018, doi: 10.1109/ACCESS.2018.2868430.

[11] H. S. Dhiman, D. Deb, S. M. Muyeen, and I. Kamwa, “Wind Turbine Gearbox Anomaly Detection based on Adaptive Threshold and Twin Support Vector Machines,” *IEEE Trans. Energy Convers.*, vol. 8969, no. c, pp. 1–8, 2021, doi: 10.1109/TEC.2021.3075897.

[12] D. Miki and K. Demachi, “Bearing fault diagnosis using weakly supervised long short-term memory,” *J. Nucl. Sci. Technol.*, vol. 57, no. 9, pp. 1091–1100, 2020, doi: 10.1080/00223131.2020.1761473.

[13] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, “1D convolutional neural networks and applications: A survey,” *Mech. Syst. Signal Process.*, vol. 151, p. 107398, 2021, doi: 10.1016/j.ymssp.2020.107398.

[14] P. Baraldi, G. Bonfanti, and E. Zio, “Differential evolution-based multi-objective optimization for the definition of a health indicator for fault diagnostics and prognostics,” *Mech. Syst. Signal Process.*, vol. 102, pp. 382–400, 2018, doi: 10.1016/j.ymssp.2017.09.013.


SBDIO I4.0
Big Data
for Industry