O3.3

Prototipo finale delle funzionalità di advanced analytics

Code	O3.3
Date	7/5/2021
Type	Confidential
Participants	AIRI - UNIMORE
Authors	Andrea Baraldi (UNIMORE), Francesco Del Buono, (UNIMORE), Francesco Guerra (UNIMORE), Matteo Paganelli (UNIMORE), Maurizio Vincini (UNIMORE)
Corresponding Authors	Francesco Guerra

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

Indice

Contents

Abstract 3
1. Repository 4
2. Algoritmi di analisi dei dati 4
2.1 Algoritmi di anomaly/novelty detection 4
2.2 Algoritmi di forecasting 9
2.3 Trasformatori 10
3. Valutazione in ambito anomaly detection 11
3.1 Valutazione outlier detection 12
3.2 Valutazione in ambito novelty detection 13
3.2.1 Dataset 13
3.2.2 Offline 14
3.2.3 Online 15
3.3 Performance 17
4. Interfaccia grafica 17
5. Pubblicazioni 17

Abstract

Questo documento descrive gli algoritmi costituiscono la libreria di funzionalità di advanced analytics implementata e valutata all'interno del progetto. Riporta quindi per completezza al suo interno anche quanto sviluppato nel primo ciclo del progetto. La libreria è disponibile all'indirizzo https://github.com/softlab-unimore/SBDIOI40

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

1. Repository

Il repository del progetto si trova all'indirizzo https://github.com/softlab-unimore/SBDIOI40
Per il funzionamento è richiesto il compilatore Python 3.* nella distribuzione Anaconda. L'installazione del pacchetto richiede l'esecuzione delle seguenti operazioni:

```
$ cd source

$ virtualenv -p python3 venv

$ source venv/bin/activate

$ pip install -r requirements.txt
```

Figura 1: Installazione pacchetto python

Il repository è strutturato nel seguente modo:

- *models* contiene gli algoritmi implementati e testati in ambito anomaly detection e forecasting
- *transforms* contiene i trasformatori ed estrattori di feature che possono essere utilizzati su serie temporali
- *evaluation* contiene le funzioni, il codice e i parametri utilizzati per ottenere le valutazioni finali
- *demo* contiene una serie di esempi su come utilizzare i modelli sia in fase di training e sia in fase di testing

2. Algoritmi di analisi dei dati

La cartella "*models*" contiene gli algoritmi implementati e testati nei dataset del progetto. Sono state valutate implementazioni di algoritmi di anomaly detection/novelty detection e di forecasting. Inoltre, la cartella "*transforms*" contiene una serie di algoritmi in grado di estrarre feature significative e trasformare le serie temporali in ingresso consentendo un'esecuzione più efficace dei modelli in alcuni scenari.

2.1 Algoritmi di anomaly/novelty detection

L'obiettivo di questa tipologia di algoritmi è quello di individuare se ci sono delle anomalie di comportamento da parte dell'applicazione che si sta analizzando. Nel contesto industriale analizzato dal progetto l'obiettivo è quello di identificare in maniera automatica:

- Condizioni di setup/avviamento dei macchinari
- Anomalie e sbalzi nei valori osservati, in modo tale da poter avvertire l'operatore di un eventuale guasto e/o risultato sbagliato nella produzione rispetto a una condizione normale della macchina.

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

- Cambi di stato dovuti a un deterioramento dei componenti o a un diverso setting della macchina

Più in generale, molte applicazioni richiedono la capacità di decidere se una nuova osservazione appartiene alla stessa distribuzione che si sta osservando ("*inlier*"), oppure no ("*outlier*"), le tecniche in grado di farlo prendono il nome di *Anomaly Detection* e si dividono in due categorie:

- *Outlier Detection*: metodo unsupervised in cui si identificano gli outlier come deviazioni nei dati, andando a identificare le regioni più concentrate e segnalando eventuali scostamenti.
- *Novelty Detection*: metodo semi-supervised in cui si cerca di imparare un comportamento/storia nei dati con l'obiettivo di identificare nelle nuove osservazioni un eventuale scostamento dal comportamento normale imparato.

Nella nostra situazione ci siamo focalizzati sul secondo caso, *novelty detection*, proponendo una serie di algoritmi che possono adatti anche in un contesto di *outlier detection*.

In generale, i metodi di "novelty detection" imparano durante il training una rappresentazione delle operazioni normali del macchinario, le quali vengono usate per identificare deviazioni dal comportamento normale codificato. Successivamente durante la fase di test, i modelli allenati assegnano un novelty score a ogni dato di test per identificare se i record appartengono a una condizione normale o anomala.

Gli algoritmi riportati nella libreria si basano su tecniche di Machine Learning (ML) e di Deep Learning (DL). Quelli che si basano su tecniche ML sperimentati nell'ambito del progetto sono:

- *IsolationForest*: si tratta dell'implementazione del modello di Isolation Forest implementato dalla libreria *scikit-learn* così come introdotto in Liu, Fei Tony, Ting, Kai Ming and Zhou, Zhi-Hua. "*Isolation Forest*" Data Mining, 2008, ICDM'08, e "*Isolation-based anomaly detection*" ACM Transactions on Knowledge Discovery from Data (TKDD). Isolation Forest "isola" le osservazioni andando a selezionare casualmente le feature e un valore tra il minimo e il massimo della feature selezionata. Il numero di split richiesti per isolare i record è equivalente alla lunghezza dalla radice alla foglia, e la distanza del cammino mediato su una foresta di "random tree" è la misura di normalità della nostra funzione di decisione, dove i cammini più corti appartengono alle anomalie essendo più facile isolarle;

- *LOF* (*Local Outlier Factor*): implementazione fornita da *scikit-learn* dell'algoritmo Breunig, M. M., Kriegel, H. P., Ng, R. T., & J. Sander "*LOF: identifying density-based local outliers*" in ACM SIGMOD record. L'algoritmo LOF calcola la deviazione della densità locale di un dato punto rispetto al suo vicino, andando a considerare gli outlier i record che hanno una sostanziale bassa densità rispetto al vicinato;

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

- *OneClassSVM*: implementazione fornita da *scikit-learn* dell'algoritmo One Class SVM. L'algoritmo One-Class Support Vector Machine si basa sulla regressione lineare e deve essere allenato solamente sui record normali, andando a imparare i confini di questi punti e classificando come anomalie punti distanti dal confine trovato;

- *PCA*: è l'implementazione fornita dal *LogPAI Team* di algoritmi di anomaly detection basati su PCA. Il riferimento è Wei Xu, Ling Huang, Armando Fox, David Patterson, Michael I. Jordan, "*Large-Scale System Problems Detection by Mining Console Logs*", SOSP 2009. La Principal Component Analysis è un metodo unsupervised che cerca di identificare le correlazioni tra le variabili determinando la combinazione dei valori che meglio catturano le differenze e le variazioni. In ambito anomaly detection, viene calcolata la proiezione nello spazio degli autovettori di un nuovo record e il suo errore di ricostruzione, il quale viene utilizzato per definire se il record in esame è anomalo (alto errore di ricostruzione);

- *One Threshold*: tecnica in grado di identificare variazioni significative rispetto una condizione di equilibrio della serie temporale che si sta analizzando andando a definire una soglia;

- *Setup Clustering*: metodo basato sulla distanza e sulle tecniche di clustering, dove viene imparata una rappresentazione (template) della classa normale durante il training. Durante la fase di predizione, viene calcolata la distanza tra i "punti normali" e i nuovi record, e se supera una certa soglia, definita in precedenza, si determina se il nuovo record appartiene a un cluster già individuato oppure deve essere assegnato a un nuovo cluster che sarà classificato come anomalo.

Gli algoritmi che si che si basano su tecniche DL sperimentati nell'ambito del progetto sono basati su auto-encoder. Gli auto-encoder sono delle architetture basate su reti neurali che comprimono il file in una rappresentazione vettoriale compatta (fase di encoding) e ricostruiscono poi i dati originali a partire da questa rappresentazione intermedia (fase di decoding). Nel contesto della novelty detection, queste architetture permettono di identificare nuove condizioni operative, quelle in cui l'errore di ricostruzione ottenuto supera una certa soglia. In questo caso, l'input elaborato non può fare riferimento a nessuna condizione normale incontrata in la fase di training e si è in presenza di una "novelty". Gli algoritmi sperimentati sono:

- *Feed-Forward AutoEncoder*, si basa sul Multilayer Perceptrons (MLP) per codificare e decodificare i dati in input e le rappresentazioni intermedie rispettivamente;
- *CNN AutoEncoder* che applica un insieme di filtri convolutivi disposti in griglia per imparare rappresentazioni intermedie che codificano pattern spaziali nei dati originali (fase di codifica) che poi vengono espanse per ricostruire il dato originale (fase di decodifica);

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

- *RNN AutoEncoder* si basano una sequenza di celle ricorrenti che imparano una rappresentazione delle informazioni in ingresso, andando a comprimere e ricostruire i dati preservando la sequenzialità delle informazioni.

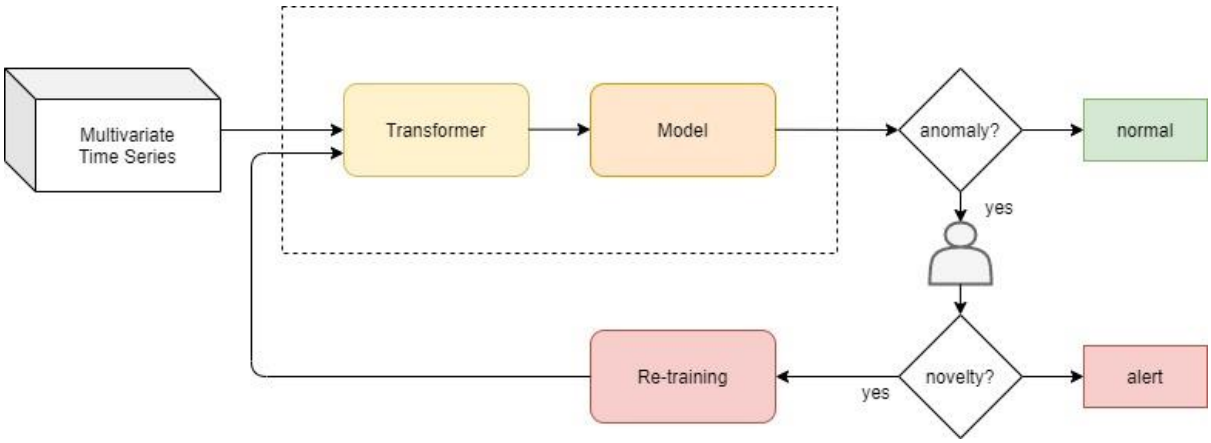

Figura 2: Algoritmi di Novelty Detection in produzione

In aggiunta, gli algoritmi una volta allenati su un comportamento normale possono essere distribuiti in un sistema online fornendo la possibilità agli operatori di allenarli nuovamente su nuovi comportamenti o stati del macchinario, senza segnalare eventuali cambi di stato corretti e andando a filtrare eventuali anomalie, in questo modo ci saranno 3 possibili risultati per i segnali in ingresso: normale, anomalia, e nuovo stato.

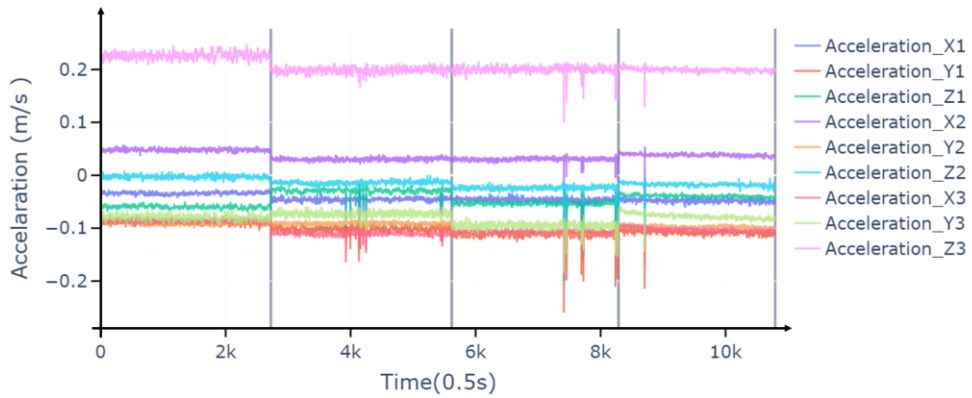

Figura 3: Segnale grezzo corrispondente a ogni condizione operativa del macchinario

Di conseguenza, come viene mostrato in Figura 3, non verranno segnalate eventuali anomalie che corrispondono a cambiamenti di stato, infatti l'algoritmo sarà in grado di identificare correttamente ogni stato normale.

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

Gli algoritmi utilizzati nel corso del progetto possono essere testati eseguendo il codice all'interno della cartella *demo* in cui viene mostrato un caso d'uso di *anomaly detection,* in cui l'output viene presentato in forma di triplette dove ogni record rappresenta una condizione anomala con timestamp di inizio e di fine.

```
$ python -m demo.anomaly -h
usage: anomaly.py [-h] --params PARAMS

Anomaly Detection Algorithms

optional arguments:
  -h, --help       show this help message and exit
  --params PARAMS  params input file
```

Figura 4: Parametri richiesti per la demo

```
$ python -m demo.anomaly --params ./params/example.json

Read input data
train: ./data/industrial/ts_normal1.CSV
test: ./data/industrial/ts_anomaly_setup1.CSV
from 2018-01-11 21:19:12 to 2018-01-12 01:22:53
Select features
Model initialization: cnn
Create training set
Training...
CNN AutoEncoder Fit
Create test set
Testing...
CNN AutoEncoder Predict

Results:
+--------+---------------+----------------------+----------------------+
|        | feature       | start                | end                  |
|--------+---------------+----------------------+----------------------|
| 0      | all features  | 2017-12-12 10:47:48  | 2017-12-12 11:07:47  |
| 1      | all features  | 2017-12-12 13:47:48  | 2017-12-12 16:04:27  |
+--------+---------------+----------------------+----------------------+
```

Figure 5: Esecuzione codice demo

```
{
  "train": "./data/industrial/ts_normal1.CSV",
  "test": "./data/industrial/ts_anomaly_setup1.CSV",

  "features_list": [
    "%E1", "TEMP_E1",
    "%E2", "TEMP_E2",
    "%E3", "TEMP_E3",
    "%E4", "TEMP_E4",
    "%E5", "TEMP_E5"
  ],

  "kernel": 180,
  "stride": 1,
  "resample_rate": 1,
  "custom_resample":  false

  "model_type":  "cnn",
  "model_params": "./params/params_{}.json",
}
```

Figure 6: Esempio del file di configurazione per eseguire la demo

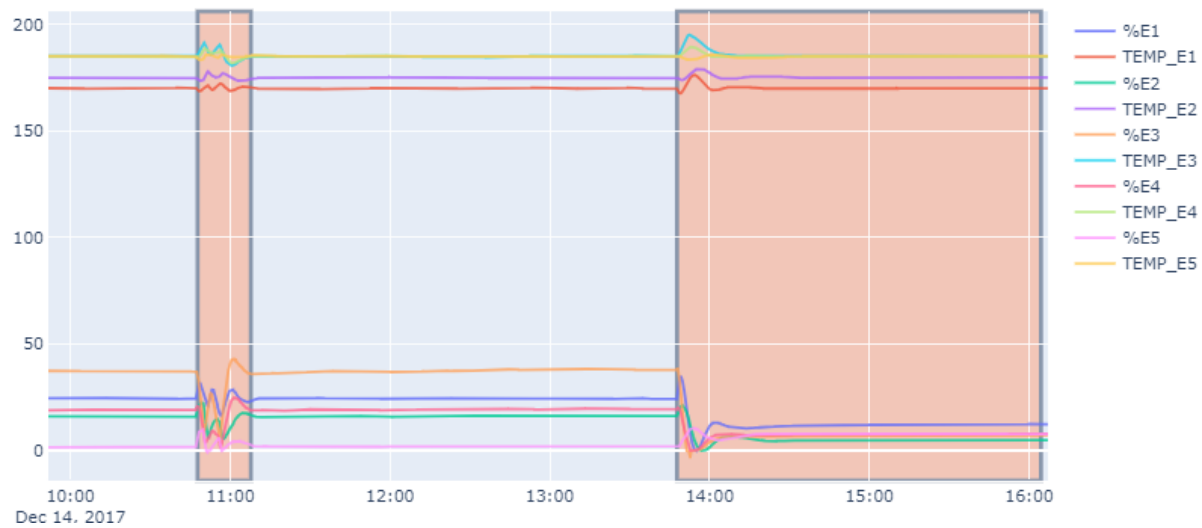

Figura 7: Applicazione delle tecniche di anomaly detection e visualizzazione delle anomalie

2.2 Algoritmi di forecasting

L'obiettivo di questa tipologia di algoritmi è quello di individuare le condizioni operative della macchina, prevedendo l'andamento futuro e se il processo è soggetto a usura. Gli algoritmi di forecasting si basano sia sull'analisi di serie temporali, sia sull'applicazione di algoritmi di

predizione. Gli algoritmi che si sono (preliminarmente) sperimentati e che sono riportati nella libreria sono:

- *ARIMA*, *VAR*: tecniche di analisi di dati basate su time series
- *LinearRegression*, *GradientBoostingRegressor*, *RandomForestRegressor*: regressore lineare e altri regressori combinati con tecniche di ensemble.

2.3 Trasformatori

La cartella "*transformers*" contiene tecniche per modificare il dataset di partenza in modo da prepararlo per le analisi successive. Le tecniche utilizzate sono:

- *Moving Average*, *EWMA*: tecniche utilizzate nell'analisi di time series che servono a normalizzare i dati per evitare fluttuazioni transitorie nei dati. La prima è una media mobile, la seconda è una media pesata in cui i valori più recenti assumono una importanza maggiore.
- *Normalizer*, *RobustScaler*, *StandardScaler*, *PCA*. Il normalizer e gli scaler sono operatori che portano i valori assunti da un attributo in un intervallo ristretto normalmente tra 0 e 1. Questo evita che un diverso range di valori abbia effetto sulla analisi dei dati. La tecnica del PCA serve a definire gli attributi del dataset che portano un maggiore peso informativo.
- *Ricampionamento con "Time Feature"*. In caso di serie temporali campionate con una frequenza molto elevata vengono applicate tecniche di ricampionamento che vanno calcolare un insieme di time feature su una finestra mobile per comprimere la complessità del segnale grezzo andando a evidenziare eventuali pattern comuni all'interno di uno stato. Le feature che vengono calcolate sono: "mean", "peak", "peak to peak", "rms", "crest factor", "skewness", "kurtosis", "shape factor"

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

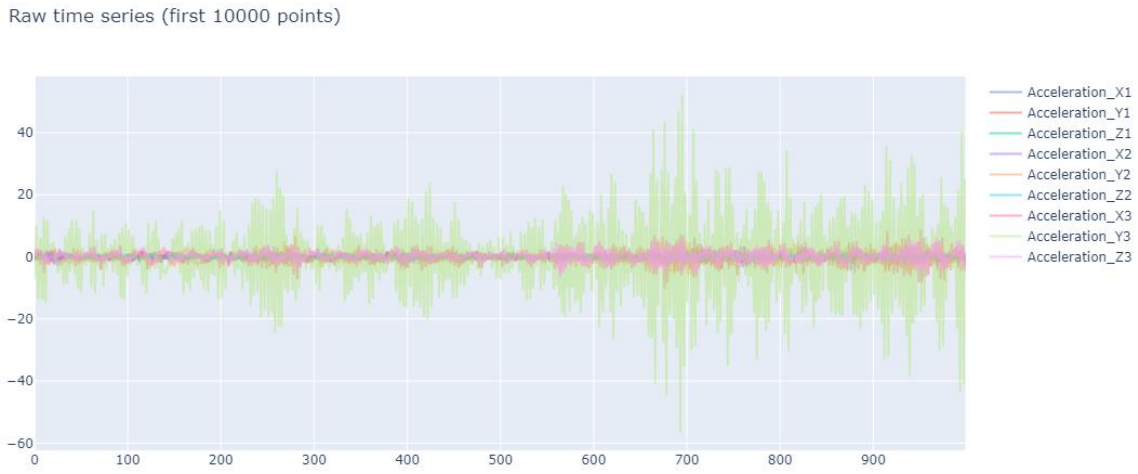

Figure 8: Primi 1000 record del segnale grezzo

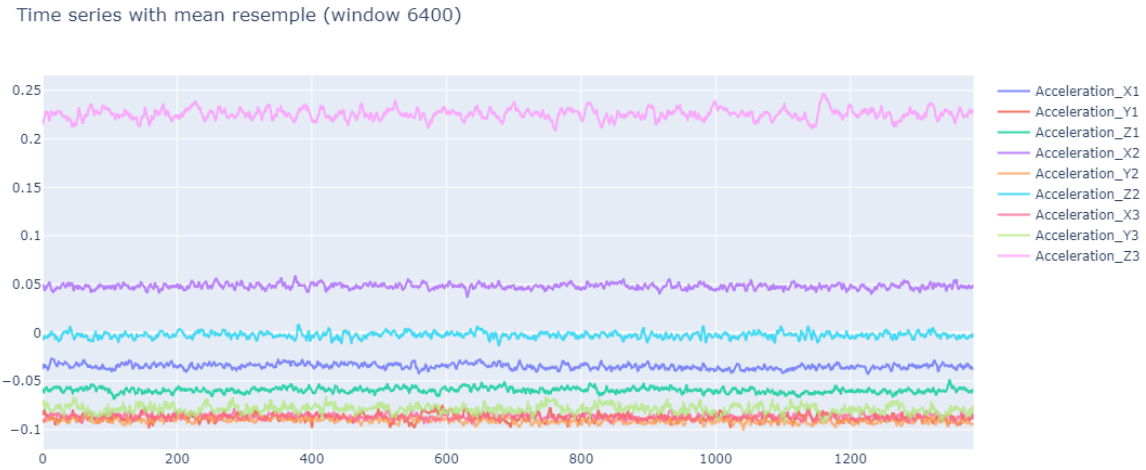

Figure 9: Ricampionamento del segnale utilizzando la media su una finestra di 6400 campioni

Le figure mostrano un applicazione della compressione del segnale applicando una media dei valori su una finestra di 6400 campioni, andando così a rimuovere parte del rumore, semplificando le analisi successive e l'applicazione dei modelli di anomaly detection e forecasting.

3. Valutazione in ambito anomaly detection

La valutazione degli algoritmi è stata divisa in due fasi:

1. Valutazione dell'efficacia degli algoritmi di ML per identificare anomalie, in particolare outlier, dovute a condizioni di setup o guasti del macchinario;
2. Valutazione dell'efficacia dell'efficienza degli algoritmi di ML e DL nel identificare e imparare i diversi stati di un macchinario.

3.1 Valutazione outlier detection

In questa prima fase è valutato il processo di anomaly detection utilizzando gli algoritmi di ML e un dataset fornito e labellato contente serie temporali multi-variate con 25 feature. Le label andavano a indicare solamente la presenza e l'assenza delle anomalie in un chunk di dati, quindi non fornivano informazioni riguardo alla locazione temporale.

La valutazione è stata fatta sia un contesto di serie temporali uni-variate, quindi andando a considerare le feature singolarmente, e sia nel contesto multi-variato.

Model	Accuracy	Precision	Recall	F-score
PCA	90.53	99.76	84.10	91.26
Clustering	72.06	83.10	65.90	73.51
SVM	58.71	59.11	96.70	73.37
IF	66.24	68.52	78.80	73.30
LOF	58.58	59.06	96.50	73.27

Tabella 1: Valutazione anomaly detection su serie temporali uni-variate

Model	Accuracy	Precision	Recall	F-score
PCA	100.0	100.0	100.0	100.0
Clustering	91.17	100-0	85.00	91.89
SVM	61.76	60.60	100.0	75.47
IF	83.82	91.42	80.00	85.33

LOF	60.29	59.70	100.0	74.77

Tabella 2: Valutazione anomaly detection su serie temporali multi-variate

In entrambe le situazioni, gli algoritmi ML, in particolare PCA e Clustering, sono in grado di identificare correttamente sbalzi del segnale, andando a segnalare eventuali errori all'operatore.

3.2 Valutazione in ambito novelty detection

L'obiettivo di questa analisi è di fornire una valutazione comparativa in termini di efficacia ed efficienza degli algoritmi di ML e DL nel contesto della novelty detection. In particolare abbiamo considerato due scenari:

1. Il primo scenario, definito offline, i modelli sono prima allenati su una singola condizione operativa, e viene valutata la loro abilità di identificare la stessa condizione nota ("known set") rispetto alle altre ("novel set");
2. Nel secondo scenario, definito online, i modelli sono valutati in termini della loro abilità di imaprare nuova conoscenza (nuovi stati). In particolare viene adottato un approccio di apprendimento incrementale dove nuove condizioni operative diverse da quella iniziale sono aggiunte nel training del modello.

Per ogni scenario vengono calcolate due misure che cercano di mappare due situazioni diverse, che possono verificarsi durante le normali condizioni operative. La prima considera separatamente ogni sample del dataset, e viene calcolata l'accuratezza in termini del numero di record correttamente predetti rispetto al numero totale. Nella seconda analisi, vengono considerati dei batch di campioni invece dei singoli record, e viene definita "l'accuratezza batch", in questo modo una predizione viene considerata corretta quando tutti i campioni di un batch vengono predetti correttamente, in modo da valutare il numero e la qualità degli avvisi inviati all'operatore.

3.2.1 Dataset

Il dataset è stato fornito dal laboratorio CIRI-MAM di Bologna, e rappresenta delle serie multi variate (con 9 feature) che rappresentano 4 diverse condizioni operative. Nella seguente tabella sono mostrate le caratteristiche meccaniche del segnale per ogni stato.

Condizione operative	**Distanza tra le pulegge (mm)**	**Frenata Coppia (Nm)**	**Durata (min)**	**Numero di batch**
C1	27.33	0.1	70	7

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

C2	27.33	0.5	150	15
C3	27.54	0.1	70	7
C4	27.54	0.1	30	3

Tabella 3: Descrizione dataset per la novelty detection

3.2.2 Offline

La valutazione offline è stata eseguita andando ad allenare un diverso modello per ogni batch di dati, e successivamente valutato sui rimanenti. Questo valutazione è stata eseguita per ogni batch di ogni stato disponibile. Durante la fase di predizione, ci aspettiamo che i dati associati alla stessa condizione operativa utilizzata durante il training non produca segnalazioni di anomalie, mentre per dati appartamenti a un altro stato della macchina venga segnalato il cambiamento.

I risultati sono mostrati in tabella, dove per ogni modello e per ogni condizione operativa usata come training è stata calcolata l'accuratezza normale e batch su tutti i dati disponibili.

Algorithm	All		C1		C2		C3	
	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.	Acc.	B. Acc.
Clustering	0.988	0.758	0.998	0.941	0.978	0.627	0.995	0.830
LOF	0.817	0.572	0.990	0.840	0.915	0.544	0.411	0.326
PCA	0.965	0.808	0.999	0.945	0.939	0.752	0.981	0.772
SVM	0.658	0.351	0.982	0.715	0.703	0.290	0.189	0.067
IF	0.880	0.619	0.994	0.867	0.967	0.598	0.561	0.379
MLP	0.957	0.911	1.000	1.000	0.945	0.906	0.933	0.821

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

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

Tabella 4: Accuratezza dei modelli nella valutazione offline

3.2.3 Online

Nella valutazione online abbiamo simulato l'adozione del modello in un contesto dinamico dove un continuo monitoraggio della macchina è eseguito, e un conoscenza incrementale della condizione operativa è imparata dal sistema di diagnostica.

Per simulare questo scenario abbiamo creato tre diverse strategie di apprendimento e valutazione, mostrate nella tabella seguente.

Conf	Training Set	Test set	
		Known Set	Novel Set
S1	C1 (10 min.)	C1 (70 min)	C2, C3, C4 (150 + 70 min + 10 min)
S2	C1, C2 (10 + 10 min.)	C1, C2 (70 + 150 min)	C3, C4 (70 min + 10 min)
S3	C1, C2, C3	C1, C2, C3	C4 (10 min)

Tabella 5: Strategie di apprendimento online

Nella prima configurazione, ogni modello è allenato esclusivamente su 10 minuti del batch dello stato C1. Nella seconda configurazione il modello assume che abbia imparato le condizioni operative C1 e C2. Nella terza configurazione, il modello viene allenato su 3 condizioni operative, e dovrà segnalare le anomalie solo per campioni appartenenti alla condizione 4.

Algorithm	S1	S2	S3

	Test	Known	Novel	Test	Known	Novel	Test	Known	Novel
Clustering	0.941	0.732	1.000	0.567	0.503	0.700	0.410	0.455	0.000
LOF	0.840	0.304	0.990	0.456	0.395	0.586	0.690	0.767	0.000
PCA	0.945	0.750	1.000	0.235	0.000	0.729	0.000	0.000	0.000
SVM	0.715	0.321	0.825	0.618	0.565	0.729	0.790	0.878	0.000
IF	0.867	0.411	0.995	0.618	0.626	0.600	0.757	0.841	0.000
MLP	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

Tabella 6: Accuratezza dei modelli nella valutazione online

La tabella mostra per ogni scenario l'accuratezza batch del modello calcolata sui dati appartenenti:

- a ogni possibile condizione del macchinario conosciuta e/o anomala (all, know e novel);
- a una condizione nota (know);
- a una condizione sconosciuta quindi anomala (novel).

Come si può notare dalla tabella, gli algoritmi di ML hanno difficoltà nell'imparare più comportamenti del macchinario che rappresentano diverse condizioni operative, andando diminuire le performance in termini di accuratezza, in particolare nell'identificazione di nuovi stati anomali. Invece, gli algoritmi di DL riescono a impare correttamente diverse rappresentazioni di normalità , mostrando la loro utilità in situazioni online dove c'è un continuo cambiamento del comportamento normale.

Bisogna specificare che questi risultati rappresentano l'accuratezza batch, mentre in un contesto in cui si valuta il singolo record, gli algoritmi DL ottengono nello scenario S3 (il più critico) valori di accuratezza intorno ai 0.81-0.83 nell'identificare i singoli record anomali, a differenza del 0.47-0.28 nella situazione mostrata in tabella.

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

3.3 Performance

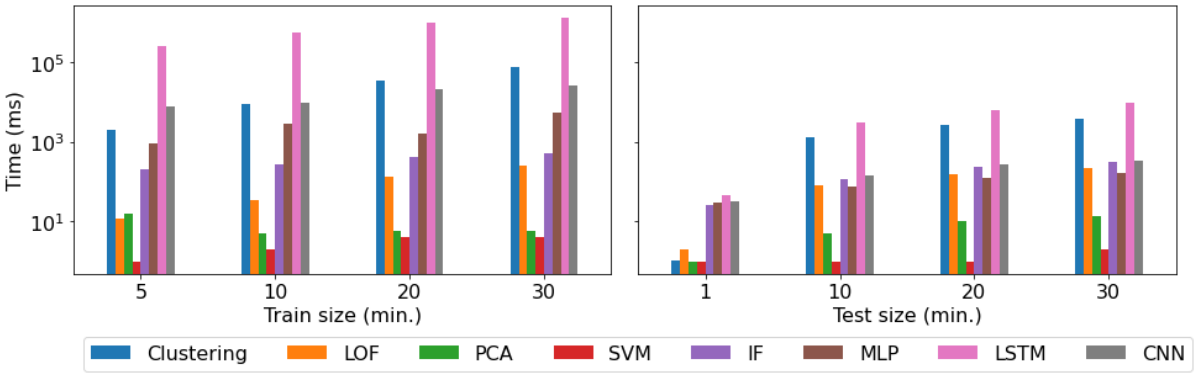

Figura 10: performance dei modelli in training e test

Nella figura vengono mostrati i tempi di training e predizione dei vari modelli all'aumentare della dimensione dei dati in input in termini di minutaggio delle serie temporali analizzate, dove 1 minuto viene rappresentano da 600 record del dataset.

Come si può notare nei tempi di training gli algoritmi SVM, LOF, e PCA hanno tempi davvero bassi per terminare l'allenamento, al contrario dei modelli DL che richiedono più tempo, in particolare il modello LSTM, dovuto alla sequenzialità delle operazioni che non prevedono una parallelizzazione. Al contrario i tempi di predizione sui dataset analizzati sono davvero bassi, consentendo un'applicazione real-time degli algoritmi.

4. Interfaccia grafica

Un'interfaccia grafica per l'attivazione della pipeline è stata realizzata in collaborazione con il team di UNIBO.

5. Pubblicazioni

Nel corso del progetto sono state prodotte tre pubblicazioni sfruttando le tecnologie sviluppate:

Matteo Paganelli, Francesco Del Buono, Marco Pevarello, Francesco Guerra, Maurizio Vincini: Automated Machine Learning for Entity Matching Tasks. EDBT 2021: 325-330

Andrea Baraldi, Francesco Del Buono, Matteo Paganelli, Francesco Guerra: Using Landmarks for Explaining Entity Matching Models. EDBT 2021: 451-456

Francesco Del Buono, Francesca Calabrese, Andrea Baraldi, Matteo Paganelli, Alberto Regattieri: Data-driven predictive maintenance in evolving environments: a comparison between machine learning and deep learning for novelty detection. KES 2021 (in corso di pubblicazione)

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