RdS
RICERCA DI SISTEMA

ENEA
AGENZIA NAZIONALE PER LE NUOVE TECNOLOGIE, L'ENERGIA E LO SVILUPPO ECONOMICO SOSTENIBILE

MINISTERO DELL'AMBIENTE E DELLA SICUREZZA ENERGETICA

Ricerca di Sistema elettrico

Piattaforma di stream analytics e logiche di training e inference dei modelli di Machine Learning e Artificial Intelligence su acceleratori di varia natura

Angelo Mariano, Serena D'Onofrio

ENEA

RdS_PTR 22-24_PR 2.1_LA3.7_140

Rapporto tecnico per Deliverable LA3.7: Piattaforma di stream analytics e logiche di training e inference dei modelli di Machine Learning e Artificial Intelligence su acceleratori di varia natura

A. Mariano (ENEA), S. D'Onofrio (ENEA)

Gennaio 2025

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dell'Ambiente e della Sicurezza Energetica -ENEA Piano Triennale di Realizzazione 2022-2024

Obiettivo 2: Digitalizzazione ed evoluzione delle reti

Progetto 2.1: Cybersecurity dei Sistemi Energetici

Linea di attività: 3.7

Responsabile del Progetto: M. Valenti (ENEA)

Responsabile Linea di Attività: A. Mariano (ENEA)

Mese inizio previsto: luglio 2023

Mese inizio effettivo: luglio 2023

Mese fine previsto: dicembre 2024

Mese fine effettivo: dicembre 2024

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione: Si ringrazia per la collaborazione alle attività svolte

Indice

Indice delle figure

1 Introduzione

La piattaforma ottenuta nel suo complesso può essere concepita come un sistema integrato di più componenti: la componente di stream processing; la componente di data analytics (applicata a due casi d'uso diversi); la dashboard di monitoraggio e gestione.

Il sistema integrato si appoggia su un cluster Kubernetes[1]. Kubernetes è un sistema open source di orchestrazione di container, ovvero il processo che permette di automatizzare il deployment, la gestione, la scalabilità e il networking dei container. Kubernetes funziona tramite pod e nodi. Il pod è la risorsa che descrive l'unità elementare eseguibile su un nodo del cluster, gestisce uno o più container ed è molto efficace dal punto di vista della scalabilità orizzontale soprattutto alle applicazioni orientate ai microservizi. I nodi eseguono i pod e sono in genere raggruppati in un cluster Kubernetes, astraendo le risorse hardware fisiche sottostanti.

L'infrastruttura Hw/Sw nel suo complesso, associata alle tecniche di machine learning elaborate, consente di fornire un valore aggiunto in termini di sicurezza rispetto a possibili vulnerabilità delle reti smartgrid.

2 Architettura del sistema

La progettazione della piattaforma di stream analytics è basata su un'architettura distribuita e scalabile, progettata su un cluster Kubernetes ed utilizza Apache Kafka come backbone per l'elaborazione dei dati in tempo reale.

Kubernetes è un sistema open source di orchestrazione di container, ovvero il processo che permette di automatizzare il deployment, la gestione, la scalabilità e il networking dei container. Kubernetes funziona tramite pod e nodi. Il pod è la risorsa che descrive l'unità elementare eseguibile su un nodo del cluster, gestisce uno o più container ed è molto efficace dal punto di vista della scalabilità orizzontale soprattutto alle applicazioni orientate ai microservizi. I nodi eseguono i pod e sono in genere raggruppati in un cluster Kubernetes, astraendo le risorse hardware fisiche sottostanti.

Apache Kafka è una piattaforma distribuita open source di stream processing. Le tre funzionalità principali sono: consentire alle applicazioni di pubblicare o sottoscrivere flussi di dati o eventi, archiviare i record accuratamente in modo tollerante agli errori e durevole, elaborare i record in tempo reale. Inoltre, Kafka ha una bassa latenza e un'alta velocità di gestione, e permette di analizzare in modo affidabile e persistente grandi volumi di dati che provengono da diverse sorgenti.

Una piattaforma a microservizi basata sull'utilizzo combinato di Kubernetes e Kafka permette di avere un sistema scalabile, flessibile e resiliente, perfetto per gestire carichi di lavoro dinamici e per elaborare dati in tempo reale.

[1] https://kubernetes.io/it/

3 Prestazioni del sistema

La piattaforma di stream analytics introdotta risulta essere ad elevate prestazioni benché il deployment utilizzato sia semplice.

Infatti, l'elaborazione in tempo reale è molto efficace: per quanto riguarda gli eventi/processi al secondo, la piattaforma è in grado di elaborare più di 50 pacchetti trasmessi al secondo, garantendo una risposta rapida e in tempo reale alle variazioni dei dati in ingresso. La latenza media di elaborazione di un singolo evento è di circa 150 millisecondi, assicurando una bassa latenza e una reattività ottimale, come si evince dal test di carico effettuato durante stress su dieci messaggi consecutivi (vedi Figura 1).

```
Message consumed with latency: 0.1685 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114300.146349}'}
Message consumed with latency: 0.1709 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114302.6280499}'}
Message consumed with latency: 0.1660 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114304.848991}'}
Message consumed with latency: 0.1680 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114307.117176}'}
Message consumed with latency: 0.0644 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114309.991528}'}
Message consumed with latency: 0.1696 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114313.317048}'}
Message consumed with latency: 0.1671 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114316.819328}'}
Message consumed with latency: 0.1677 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114319.237053}'}
Message consumed with latency: 0.1665 seconds
Received: {'value': '{"key": "value", "timestamp": 1737114323.167892}'}
Message consumed with latency: 0.0641 seconds
```

Figura 1: Test di carico su dieci messaggi consecutivi

Inoltre, anche la scalabilità del sistema ha ottime performance, poiché sottoponendo la piattaforma a test di carico crescenti, si è dimostrata la capacità di scalare linearmente con l'aumento del volume di dati e del numero di dispositivi connessi. La piattaforma è stata interessata dal processamento di quantità crescenti di elementi di traffico di rete mantenendo costante il rapporto tra numero di eventi processati e tempo di processamento.

Il sistema è stato progettato per essere altamente scalabile, grazie all'utilizzo di tecnologie basate su kubernetes, e quindi sulla realizzazione di microservizi containerizzati gestiti su un sistema cloud. In particolare, i singoli pod che compongono il sistema di stream analytics possono essere scalati sia orizzontalmente, aggiungendo nuovi pod per l'elaborazione dei dati, sia verticalmente andando a modificare a runtime le configurazioni di storage e risorse di calcolo da utilizzare.

Per quanto riguarda le prestazioni della dashboard in termini di tempi di risposta, essi sono variabili in base a diversi fattori, quali la dimensione del cluster Kubernetes, il tipo di operazione che si vuole svolgere sul cluster, il carico del cluster, la latenza di connessione con il cluster e non ultimo la potenza di calcolo del computer su cui la dashboard è in esecuzione. Tenute presenti queste considerazioni, in una configurazione come quella di progetto con un cluster leggero, anche geograficamente distante dalla dashboard e con l'uso di hardware di base senza particolari esigenze i tempi di esecuzione dei comandi della dashboard si mantengono sempre inferiori al secondo nelle fasi di monitoraggio.

Per quanto riguarda le prestazioni della dashboard in termini di scalabilità, per la sua natura, si tratta di un software che può essere installato su qualsiasi piattaforma e non necessita di

repliche per essere efficiente e monitorare e orchestrare i container sul cluster Kubernetes, ma si affida alle API di Kubernetes per operare.

Per quanto riguarda le prestazioni della dashboard in termini di usabilità, emerge chiaramente, sulla base delle figure presentate e dell'uso quotidiano, che lo strumento utilizza una interfaccia grafica intuitiva, presenta una notevole facilità d'uso, consente l'accesso alla gamma completa di funzionalità del cluster Kubernetes, fornisce la possibilità di integrarsi con altri strumenti dell'ecosistema Kubernetes. Di conseguenza lo strumento si presta in maniera eccellente alle funzionalità di monitoraggio e di gestione dei componenti software descritti in precedenza sia per utenti non esperti che in condizione in cui la complessità del cluster sottostante può aumentare.

4 Utilizzo delle risorse

Analizziamo l'utilizzo delle risorse di memoria, CPU, GPU/FPGA del sistema.

Durante l'esecuzione delle logiche di stream analytics e dei modelli di machine learning, la CPU è utilizzata in media al 30% della sua capacità massima, con alcuni picchi vicini al 40% della capacità come si evince dalla Figura 2, in cui i picchi di utilizzo della CPU si verificano principalmente nelle fasi in cui c'è un traffico più intenso sulla rete dati che si sta processando, ma in ogni caso rimangono nei limiti di tollerabilità del sistema.

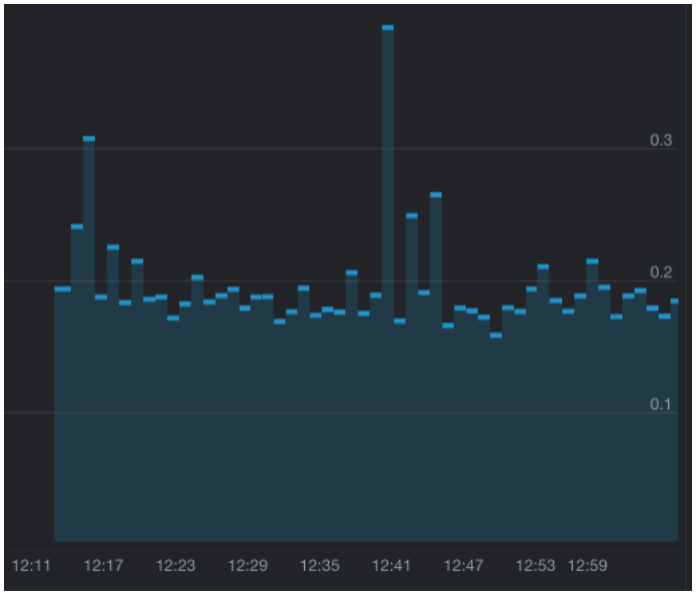

Figura 2: Schermata relativa all'uso della CPU da parte dei processi di stream analytics

Per quanto riguarda la quantità di memoria utilizzata per l'elaborazione dei dati e l'esecuzione dei modelli, questa è di poco superiore ai 9 GB di utilizzo della RAM (vedi Figura 3), in cui la maggior parte della memoria è utilizzata per il caching dei dati durante la fase di stream processing e si mantiene tendenzialmente stabile lungo tutta la durante dei test di esecuzione.

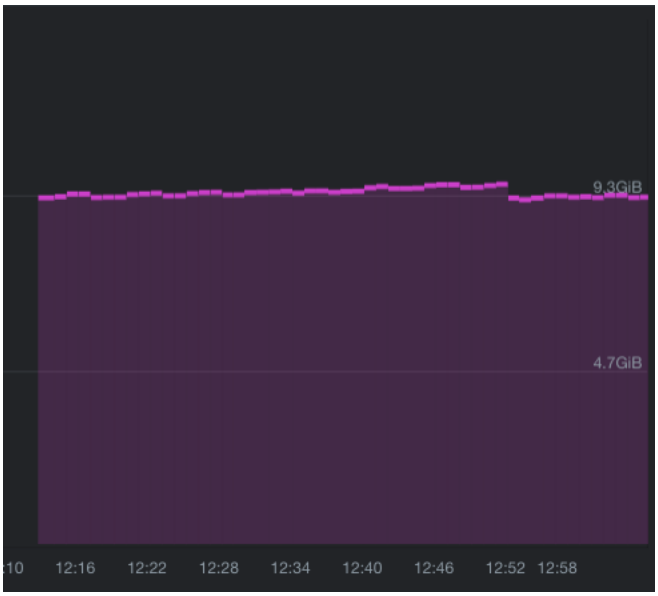

Figura 3: Schermata relativo all'uso della RAM da parte dei processi di stream analytics

In questa specifica applicazione non si è rivelato di marcato beneficio l'utilizzo di architetture speciali per il calcolo quali quelle basate su GPU e FPGA. Infatti, tutti i modelli utilizzati lavorano con la massima precisione e affidabilità già sulla CPU standard della piattaforma. Di conseguenza i modelli elaborati hanno ampia affidabilità e basso costo rispetto alle richieste hardware/software e quindi possono essere messi in produzione senza particolari attenzioni.

5 Precisione e accuratezza dei modelli

I modelli di machine learning implementati sulla piattaforma sono tre: un Autoencoder impilato sviluppato dall'Università RomaTre (per i dettagli rimandiamo al report della Linea di Attività 3.9), ed una Random Forest e un Extreme Gradient Boosting sviluppati dall'Università di Bari (per ulteriori dettagli vedere il report della Linea di Attività 3.10).

5.1 Autoencoder

Il primo modello implementato sulla piattaforma è un Autoencoder impilato (stacked) non supervisionato, che identifica attività dannose del traffico di rete apprendendo i modelli di comportamento normale, e segnalando variazioni significative come potenziali attacchi.

Il modello ha ottenuto, su un dataset di 1.288.025 record i seguenti risultati in termini di:

- Precision: 98%
- Recall: 97%
- F1-score: 97%
- Accuratezza: 100%

5.2 Random forest

Il secondo modello testato sulla piattaforma è una Random Forest addestrata per identificare minacce di tipo DoS.

Il modello ha prodotto su un dataset di 640.144 osservabili e 43 variabili, i seguenti risultati in termini di:

- Accuratezza: 99.987%
- Recall: 99.965%
- F1: 99.974%

5.3 Extreme Gradient Boosting

Il terzo modello è un Extreme Gradient Boosting (XGBoost), ovvero una versione altamente ottimizzata dell'algoritmo di gradient boosting, trainato per riconoscere attacchi di tipo DoS.

Il modello ha prodotto su un dataset di 640.144 osservabili e 43 variabili, i seguenti risultati in termini di:

- Accuratezza: 99.986%
- Recall: 99.979%
- F1: 99.977%