C'est quoi votre problématique !

Notre problématique est la minimisation de la consommation des ressources dans une architecture conteneurisée.

c'est quoi votre objectif !

notre objectif est de proposer une solution en utilisant un système d'orchestration de conteneurs ainsi qu'une méthode de prédiction afin de minimiser
la consommation des ressources
on peut citer d'autre objectifs, tels que:
- apprentissage de l'orchestrateur Kubernetes
- apprentissage de LSTM
- apprentissage de Python et les différents outils tels que Grafana, InfluxDB

Quelles sont vos contributions !

Les principaux contributions de notre travail sont:
- Le choix de la méthode de prédiction LSTM Encodeur-Décodeur
- L'entraînement continu du système de prédiction LSTM
- La modularité de l'approche proposée (MEAP) afin de permettre l'extensibilité du système de mise à l'échelle avec de nouveaux modèles et méthodes
  de prédiction
- implémentation de l'approche proposée (MEAP) sur un cluster Kubernetes réel

Est ce que votre travail est unique !

dans notre travail nous sommes basés sur d'autres approches proposés dans la littérature,ensuite, nous avons proposé des nouvelles caractéristique
et modèle

C'est quoi la relation entre l'orchestration et votre travail !

l'orchestrateur est le responsable de mettre à l'échelle les microservices, à cause de ça nous avons utilisé un orchestrateur qui est kubernetes

C'est quoi la relation entre le cloud et votre travail?

l'orchestration est généralement utilisée dans les clouds à cause de l'existence de plusieurs microservices à orchestrer, donc notre a une relation
avec le cloud

c'est quoi un système d'orchestration !

c'est un système qui permet d'assurer:
- le déploiement et placement des microservices
- la mise à l'échelle des microservices
- l'équilibrage de la charge entre les microservices
- le bon fonctionnement des microservices

Existe-t- il un système d'orchestration pour les machines virtuelles !

il existe des systèmes d'orchestration pour les VM, tels que: Vagrant pour VirtualBox

c'est quoi la virtualisation !

La virtualisation est une technologie qui permet de créer plusieurs environnements simulés à partir d'un seul système physique.
en utilisant un hyperviseur on peut fragmenter ce système unique en plusieurs environnements sécurisés distincts appelés machines virtuelles,
ou chaque machine virtuelle a son propre système d'exploitation

Le principal avantage de la virtualisation !

Si on a une infrastructure capable de prendre en charge un grand nombre d'applications, on peut l'optimiser en utilisant la virtualisation.
la virtualisant.

c'est quoi la conteneurisation !

c'est une forme de virtualisation du système d'exploitation consiste à découper une application en services distincts et différents,
et à exécuter ces services dans des espaces utilisateurs isolés appelés conteneurs qui utilisent le même système d'exploitation partagé.

différence entre la virtualisation et la conteneurisation :

La virtualisation est une technologie qui permet de créer plusieurs environnements simulés à partir d'un seul système physique. alors que,
La conteneurisation vise à découper les systèmes d'exploitation en différents morceaux que nous pouvons l'utiliser plus efficacement.

c'est quoi un conteneur || pour quoi la conteneurisation c'est une forme de virtualisation de système d'exploitation !

Un conteneur c'est une machine virtuelle sans noyau qui partage le même noyau du système d'exploitation hôte, un conteneur ne contient
que l'application et les dépendances de l'application ou un moteur de conteneurisation comme docker n'a pas besoin d'émuler un système d'exploitation complet pour le conteneur,
plus précisément, un conteneur est un processus "init" fils du processus "init" du système d'exploitation hôte, et comme le processus "init" du système
d'exploitation hôte est le processus parent de tous les processus exécutés dans un système d'exploitation, un conteneur c'est un système virtuel
qui partage virtuellement la carte réseau, l'entré sortie, c'est à cause de ça que la conteneurisation c'est une forme de virtualisation au niveau
système.

c'est quoi la différence entre Docker et container !

Docker c'est un système qui permet de gérer des conteneurs (création, modification, suppression, ...)

c'est quoi la différence entre un hyperviseur et Docker !

Un hyperviseur permet de gérer des machines virtuelles, tels que VirtualBox, Vmware. Tandis que, Docker est un système pour gérer les conteneurs.

quels sont les types de virtualisations !

il y'a deux types de virtualisant:
Le type 1, consiste à installer l'hyperviseur directement sur le matériel, tel que ESXI Vmware
Le type 2, consiste à installer l'hyperviseur au-dessus d'un système d'exploitation comme VirtualBox.

La conteneurisation est un type de virtualisation?

la conteneurisation c'est une méthode de virtualisation, on peut l'utiliser sur un système installé sur une machine physique, ou sur un système
installé sur une machine virtuelle.

Quels sont les avantages de la conteneurisation !

En comparant avec la virtualisation basée sur des machines virtuelles, la conteneurisation présente l'avantage principalement en raison d'une

meilleure portabilité, d'une infrastructure légère, rapide et isolée pour exécuter des applications ainsi qu'une mise à d'échelle facile.
on peut citer d'autres avantages, telles que:
- facile à créer, redémarrer, déplacer, réparer, ...
- consommation de consommation des ressources, cela est expliqué par le fait que dans les conteneurs on a pas besoin d'émuler les périphériques d'entrée sortie

c'est quoi Kubernetes !

c'est un orchestrateur de conteneurs qui permet d'assurer :
- le déploiement et placement des microservices
- la mise à l'échelle des microservices
- l'équilibrage de la charge entre les microservices
- le bon fonctionnement des microservices

Décrivez l'architecture de Kubernetes !

walah masme3tak :
l'Architecture de Kubernetes se compose d'un ou plusieurs masters et un ou plusieurs workers
- Le Nœud master: exécute le plan de contrôle qui est le responsable de la gestion des workers
- Le Nœud worker: chargés de gérer les les microservices des applications via des pods.

Le pod est l'unité la plus petite dans Kubernetes. Il représente une instance unique d'un microservice. Chaque pod est constitué d'un conteneur ou
d'une série de conteneurs étroitement couplés
Le master se compose de 4 composants principaux:
1- Equipement de Terminaison de Circuit de Données (ETCD): C'est une BDD qui représente le magasin de données principal de Kubernetes, Elle permet
de stocker et répliquer tous les états de clusters et les informations nécessaires à son fonctionnement, comme (les services déployés, les clés, ...)
2- L'API Server est la partie front-end du plan de contrôle Kubernetes, il est utilisée pour créer, configurer et gérer les clusters Kubernetes, il est
le moyen d'interaction entre les utilisateurs et le cluster Kubernetes
3- Le scheduler est le planificateur de Kubernetes, il gère la planification de tous les pods, (ç-à-dire: il décider sur le nombre de réplicas,
leurs emplacement sur les workers, ...,)
4- Un contrôleur se réfère au planificateur pour assurer qu'un nombre suffisant de pods est exécuté, si un pod est défaillant, le contrôleur
le remarque et réagit, c'est l'entité qui va interagir avec le système de virtualisation comme Docker pour gérer l'ensemble des pods (création, suppression,
redémarrage, ...)

- Un service : Indique comment accéder à un microservice (c'est-à-dire à un ensemble de pods). Ils sont utilisés pour contrôler l'accès interne et
externe à un cluster.
- Kubelet est un agent qui s'exécute sur chaque nœud worker du cluster. Il communique avec le plan de contrôle, et exécute ses actions, Il assure le
bon fonctionnement des conteneurs
- kube-proxy: pour l'accessibilité des microservices depuis l'extérieur.

C'est quoi la différence entre Docker et Kubernetes?

Docker c'est un système qui permet de gérer des conteneurs (création, modification, suppression, ...), Tandis que Kubernetes est un système
d'orchestration des conteneurs, ou le contrôleur de Kubernetes peut communiquer avec Docker pour gérer des conteneurs.

Comment peut-on créer plusieurs containers dans le même système?
créer plusieurs processus "init" dans le même système d'exploitation explique la possibilité de créer plusieurs conteneurs sur le même système,
car un conteneur c'est un processus "init" fils du processus "init" du système d'exploitation.

C'est quoi le Container-runtime pour Kubernetes?
Le container-runtime c'est le système de gestion des conteneurs que Kubernetes doit communiquer avec pour la gestion de conteneurs et poids.

C'est quoi la différence entre kubectl et kubelet?
"kubectl" est un programme pour la gestion des déploiements dans kubernetes, il utilise REST API pour communiquer avec l'api-server, tandi que
"kubelet" est un agent qui s'exécute sur chaque nœud worker du cluster et communique avec le plan de contrôle et exécute ses actions

C'est quoi la différence entre persistent volum (pv) et persistent volum claim (pvc)?
dans Kubernetes, persistent volume (pv) c'est un espace disque pour un ensemble d'applications (app1, app2, app3, …, appk) créé par
l'administrateur de Kubernetes, tandis que le persistent volume claim (pvc) correspond à l'attribution d'un (pv) par exemple (app1) à un utilisateur,
Le pvc est créé par l'utilisateur.

Les conteneurs correspondant au déploiement de votre microservice (Pas-ss-0) sont dans le Master ou Worker?
les containers sont dans le worker

C'est quoi le système qui permet de faire le monitoring sur Kubernetes !
le système qui permet de faire le monitoring sur Kubernetes est Metrics-Server, il correspond à un déploiement, qu'on peut installer via:
"kubectl apply -f ..."

C'est quoi la mise à l'échelle !
consistent à ajouter des ressources chaque fois que la moyenne CPU, mémoire franchit un certain seuil

Quels sont les types de mise à l'échelle?
Il y a 3 types de mise à l'échelle des conteneurs.
- La mise à l'échelle verticale: consiste à ajouter des ressources (CPU, Mémoire) au même ensemble de conteneurs.
- La mise à l'échelle horizontale consiste à ajouter un ensemble de conteneurs à un déploiement existant.
- La mise à l'échelle hybride: consiste à utiliser les 2 premiers (Horizontal et Vertical) en même temps.

Quel type de mise à l'échelle avez-vous choisi dans votre travail? Pourquoi?
dans notre travail, nous avons choisi le type de mise à l'échelle Horizontal, car il est facile à implémenter,
tandis que le Type Vertical très difficile à implémenter

Est-ce que Kubernetes a l'option de mise à l'échelle?
Oui, Kubernetes a l'option de mise à l'échelle mais réactive et horizontale.

On fait mise à l'échelle à un déploiement ou à un pod ou à un container ?
On fait mise à l'échelle à un déploiement en ajoutant un replicas (pod) ou en augmentant les ressources pour ce pod

Dans le cloud, qui est le responsable de faire la mise à l'échelle?
Dans une architecture cloud, c'est l'administrateur de Kubernetes qui doit le configurer et le paramétrer afin qu'il puisse faire la mise à l'échelle.

C'est quoi un réseaux de neurone RN
Un réseau de neurones est une séquence de fonctions simples, prenant en entrée les données du problème et les poids "W".
L'entraînement se déroule comme suit :
1- le Feed-Forward : consiste à calculer les valeurs prédite à partir des entrés X (point 1) , [tels que: en premier temps, les poids W sont initialisés aléatoirement]
2- calculer la valeur de la fonction de coût qui représente la distance entre la valeur prédite et la valeur réel (point 2)
3- estimation des quantités avec lesquels chaque poids doit être mis à jour, pour cela, on calcule le gradient de cette fonction de coût par rapport aux paramètres W, en faisant la rétro-propagation à travers le réseau de neurones,
4- minimiser la fonction de coût en mettant à jour les paramètres W
5- la répétition des point (1,,,,4) pour tous les séquence de tous les batch d'entraînement corresponds à une époque d'entraînement,
6- on peut faire feed-forward pour plusieurs vecteur avant de faire backpropagation, le nombre de fois correspond au batch size

C'est quoi un réseau de neurones récurrent RNN?
Un réseau de neurones récurrent est un type de réseau de neurones utilisé pour la prédiction des données séquentielles , L'idée derrière Les RNN est de permettre au modèle de Deep Learning d'avoir de la mémoire, où chaque cellule prend comme entrée une valeur «x» de séquence et la sortie «h» de la cellule précédente (ç à dire: les neurones dans la même couche sont reliés pour trouver des relations entre les valeurs l'entré X).

C'est quoi la différence entre RN et RNN?
dans un RN simple les neurones d'une couche ne sont pas reliés, sont généralement utilisés pour classification, tandis que dans RNN les neurones d'une couche sont reliés pour trouver des relations entre les valeurs l'entré, ils sont généralement utilisés pour la prédiction des données séquentielles comme la traduction.

C'est quoi la différence entre entraînement et test?
dans l'entraînement d'un RN, nous connaissons l'entré X et la sortie réelle Y et nous cherchons à déterminer les poids "w" associés aux neurones de façon que la sortie prédite (Y-bar) est plus proche de la valeur réelle Y, tandis que dans le test, nous connaissons l'entré X, nous avons la liste des poids et on calcule les valeur prédites en utilisant "x" et "w"

Quels sont les poids d'un neurone?
dans un RN, chaque neurone est associé à un "W" et une erreur "b", sont utilisés pour déterminer une certaine caractéristique dans l'entrée X, par exemple : pour prédire si un chat ou chien, on peut détecter séparément (la tête, le pied, les yeux, ...).

Quels sont les objets qui changent en faisant l'entraînement d'un RNN?
Les objets qui changent en faisant l'entraînement sont les poids ("W", "b").

C'est quoi Feed-Forward dans l'entraînement?
le Feed-Forward : consiste à calculer les valeurs prédite à partir des entrés X, en passant par les poids et les fonctions d'activation.

C'est quoi Back-Propagation dans l'entraînement?
c'est une technique utilisée pour mettre à jour les poids pendant l'entraînement, plus précisément, elle est utilisée pour calculer le gradient de la fonction de perte par rapport aux poids w d'une cellule ou d'une couche.

C'est quoi une fonction Loss d'un RNN?
La fonction loss ou perte permet de calculer la distance entre la valeur prédite et la valeur réelle pendant le processus d'entraînement.

Que signifie une fonction Loss convergente après entraînement?
le processus d'entraînement se compose de plusieurs epoches, dans chaque époque on calcule la valeur de la fonction perte puis en fait la mise à jour des poids, si les valeurs de la fonction perte tendent vers 0 après plusieurs époques on dit qu'elle converge, sinon on dit qu'elle ne converge pas.

Citez un exemple d'une fonction Loss pour RNN?
On trouve généralement les bibliothèques Keras et TensorFlow utilisent l'optimizer "Adam" qui correspond à une fonction de perte.

Pourquoi on fait la dérivation en faisant le Back-Propagation?
Pour estimer les quantités avec lesquelles chaque poids doit être mis à jour.

C'est quoi la différence entre Epoch et Batch?
Le batch size correspond au nombre de séquences à traiter avant la mise à jour des poids, alors que le nombre d'époques correspond au nombre de passages complets pour chaque cellule.

C'est quoi une fonction de mise à jour des poids d'entrainement ?
La fonction de mise à jour de poids permet d'attribuer de nouvelles valeurs aux poids après le calcul de la valeur du gradient de la fonction perte par rapport à les poids w.

C'est quoi le paramètre Learning Rate?
Le taux d'apprentissage est un hyperparamètre qui représente la rapidité de la descente de gradient, s'il prend des valeurs plus proches de 0 le processus d'entraînement devient très long, sinon la fonction "loss" ne converge pas.

Comment définir le nombre de paramètres d'un RNN?
le nombre de paramètres d'un RNN correspond au nombre de poids (W,b) pour tous les neurones, si un RNN se compose de 10 neurones, alors le nombre de paramètres est 20.

C'est quoi les hyperparamètres dans un réseau RNN?
un hyperparamètre correspond au nombre de paramètres, nombre de couches, la valeur de learning rate, batch-size, epochs …

Pourquoi on utilise les fonctions d'activation Sigmoid et Tanh?
Les fonctions d'activation comme Sigmoid et Tanh sont utilisées pour normaliser les valeurs prédites pour avoir entre (0, 1) pour Sigmoid ou (-1 et 1) pour tanh, ceci va aider à converger la fonction de perte.

C'est quoi la différence entre LSTM et RNN?

Un RNN cause un problème pendant l'entraînement, ou la dérivation pour les cellule les plus loins tend vers «0 , ce problème est connu sous le nom «Vanishing gradient», Pour résoudre ce problème, un vecteur nommé: «état de la cellule» a été introduit aux RNN, le nouveau réseau de neurones est connu sous le nom LSTM,

C'est quoi le rôle du Cell State dans LSTM?
permet de ne sauvegarder que les informations pertinentes parmi la mémoire du passé et les entrées de la céllule Xt et Ht-1, plus précisément, elle permet de faire la dérivation que pour les cellules importantes dans la séquence.

C'est quoi le role du vecteur 'h' dans LSTM?
c'est un vecteur qui représente la valeur prédite à l'instant t par une cellule LSTM, à l'instant t+1 une cellule LSTM prend ce vecteur comme entrée (Ht-1)

Quelles sont les motivations qui ont posé pour utiliser LSTM au lieu du RNN?
nous avons trouvé que LSTM est très recommandé, cela est expliqué par le fait que les RNN cause un problème pendant l'entraînement pour les dépendance à long termes

C'est quoi Vanishing Gradient problem?
problème pendant l'entraînement, ou la dérivation pour les cellules les plus loins tend vers 0

C'est quoi Long Term dans LSTM?
Long Terme correspond aux cellules les plus loins.

C'est quoi Short Term dans LSTM?
Short Terme correspond au cellules les plus proches

C'est quoi une couche LSTM?
une couche LSTM est une succession de cellules, chaque cellule corresponde à 4 réseaux de neurones en parallèles, un pour le forget gate, 2 pour le input gate et un pour le output gate, ces 4 réseaux de neurones donnent comme sortie deux vecteurs "c" et "h".

Comment définir le nombre de paramètres d'une cellule LSTM?
le nombre de paramètres pour une cellule LSTM, correspond au nombre de poids (W,b) pour les 4 réseaux de neurones.

Comment entraîner une couche LSTM?
l'entraînement d'une couche LSTM consiste à faire le Feed-Forward pour toutes les cellules de la couche, ensuite le calcule de la valeur de la fonction perte, ensuite la mise à jour des poids pour chaque cellule.

Le nombre d'unités d'une couche LSTM correspond à quoi?
le nombre d'unité d'une couche LSTM correspond à la taille du vecteur "h",
Il représente le nombre de neurones à concaténer avec ceux de l'entrée "X" dans chaque cellule LSTM, et l'augmentation du nombre de neurones implique l'augmentation du nombre de poids "W", ce qui augmente la possibilité de détecter plus de caractéristiques dans chaque séquence.

les modes d'utilisation de LSTM !
One-to-One, Many-to-One, Many-to-Many

Pourquoi avez- vous choisi Many-to-Many ! Parce que dans notre travail, on vise à prédire une séquence de consommation à partir d'une autre séquence.

Pourquoi vous avez choisi Encoder-Decoder? Quels sont ses avantages!
parce que on a trouvé que type Encoder-Decoder de LSTM est très recommandé pour la prédiction des données séquentielles, et cela est expliqué par le fait que ce type réseau LSTM est constitué de 2 couches LSTM la couche Encoder et la couche Decoder, ce qui augmente la capacité de prédiction.

Est qu' il y a déjà des travaux dans la littérature sur la mise à l'échelle prédictive!
Il existe des travaux dans la littérature sur la mise à l'échelle prédictive.

S'il existe déjà des travaux, pourquoi vous avez répété l'approche!
Notre sujet est un sujet de recherche, et dans la recherche, on a deux directions:
- la première c'est de proposer une solution originale pour un certain problème,
- la deuxième c'est de baser sur une solution déjà proposée, et proposer des nouvelles caractéristiques et méthodes pour l'améliorer.

Quels sont les avantages de votre approche par rapport aux autres approches dans la littérature!
les avantages de notre approche par rapport aux autres approches proposés dans la littérature sont:
- la modularité de notre architecture *afin de permettre l'extensibilité du système de mise à l'échelle avec de nouvelle méthode et modèle prédictif (ç à dire notre architecture peut être amélioré facilement)*
- le choix de la méthode de prédiction LSTM Encoder-Decoder
- la continuité d'entraînement

Que représentent les tableaux comparatifs entre votre approche et les autres? Comment vous l'avez rempli!
Dans le tableau comparatif nous avons montré la valeur ajoutée de notre travail par rapport aux autres approches proposées dans la littérature.

Est-ce que vous avez réalisé l'état de l'art avant ou après l'implémentation!
Nous avons réalisé l'état de l'art avant l'implémentation, car nous avons basé sur elle pour choisir les systèmes et les technologies à utiliser pour arriver à une bonne contribution.

Pourquoi avez-vous proposé un modèle mathématique!
nous avons proposé le modèle mathématique, pour généraliser notre approche MEAP pour qu'elle soit indépendante de la technologie à implémenter

L'algorithme que vous avez proposé représente quoi !
L'algorithme que nous avons proposé représente le fonctionnement et l'idée globales derrière la mise à l'échelle automatique et prédictive.

Dans l'architecture de LSTM Encoder-Decoder, que représente RepeatVector?
RepeatVector est utilisé pour répéter les vecteurs "h" et "c" (ç à dire la sortie de la couche encodeur) pour avoir la dimension 3 au lieu de 2, car la couche décodeur de LSTM nécessite une dimension 3 comme entrée (c'est une question de programmation).

Dans l'architecture de LSTM Encoder-Decoder, c'est quoi la différence entre couche LSTM 1 et 2 !
La couche encodeur permet de créer une représentation d'une séquence et de fournir deux vecteurs à la fin "h" et "c" d'où vient le nom "encodeur", tandis que le décodeur permet de prédire la séquence de sortie, d'où vient le nom "décodeur" ou chaque cellule donne comme sortie un vecteur h_bar utilisé pour prédire une valeur dans la séquence prédite.

Dans l'architecture de LSTM Encoder-Decoder, comment fonctionne la couche dense !

Pourquoi avez- vous choisi Python pour implémenter vos scripts !
parce que la plupart des bibliothèques qu'on peut utiliser pour implémenter LSTM sont en Python comme Keras et TensorFlow.

C'est quoi la modularité dans votre plateforme !
La modularité signifie que chaque composant dans notre plateforme MEAP est implémenté et fonctionne séparément des autres

Comment les différents modules communiquent entre eux!
-Le module du monitoring communique avec Kubernetes pour stresser le CPU du pod Pas-ss-0 en utilisant "kubectl" qui utilise la commande "bash" pour lancer le stresse.
-Nous avons aussi utilisé "kubectl" pour collecter l'utilisation de CPU avec la commande :
"kubectl top pods",
-et pour lancer la mise à l'échelle automatique, nous avons utilisé la commande "kubectl scale ..."

Pourquoi avez- vous créé un modèle de données simulé !
actuellement c'est très difficile de trouver une base de données sur la consommation de CPU des pods sur internet, car cette architecture est nouvelle,
de ce fait nous avons 2 choix:
1) soit essayer de prédire l'évolution aléatoire du CPU d'un microservice,
2) soit essayer de prédire un certain modèle de données simulés.
Dans la réalité, la consommation des ressources par un microservice n'est pas aléatoire,
ç à dire: elle suit une certaine évolution, par exemple (dans la journée il y beaucoup de trafic contrairement à la nuit), de ce fait, nous avons opté pour le choix 2.

C'est quoi la continuité d'entraînement dans votre plateforme !
La continuité d'entraînement c'est d'entraîner le modèle LSTM périodiquement sans affecter les ressources, ç à dire l'entraînement en premier temps rah ykoun b les données d'entraînement initiales et après chaque période de temps l'entraînement yedi comme entrée les données d'entraînement initiales concaténées avec les 10 données actuellement monitorées, le module autoscaler utilise le modèle sauvegardé qui est amélioré continuellement (chaque période).

Pourquoi avez-vous concaténé les données du modèle avec celles actuellement monitorées!
c'est pour adapter notre modèle entraîné aux données réelles de CPU.

Comment le module autoscaler communique avec Kubernetes?
le module autoscaler communique avec Kubernetes via "kubectl" pour lancer la mise à l'échelle automatique: "kubectl scale ..."

Comment le module Monitor_Cpu stresse et collecte l'utilisation de CPU?
nous avons installé l'outil stress-ng au niveau du pod "pas-ss-0", aussi nous avons activé le système de monitoring "Metrics-Server" sur Kubernetes afin de monitorer la consommation des ressource dans notre cluster, donc pour stresser "pas-ss-0" nous envoyons une requête au pod "pas-ss-0" via kubectl qui va lancer l'outil stress-ng, et pour monitorer nous utilisons la commande "kubectl top pods" qui affiche la consommation de CPU par les différents pods déployés dans le cluster.
****

Pourquoi vous avez fixé le nombre maximal de pods pour "Pas-ss-0" à 5?
c'est juste pour vous faire la démonstration, mais il existe un nombre infini de valeurs qu'on doit tester, ce qui motive à laisser cette question ouverte pour qu'elle soit traité comme perspective.

Dans votre plateforme, comment expliquez- vous que le seuil de confiance "Tetha" est fixé à 1?
Dans notre implémentation,si une valeur dépasse le seuil, on déclenche la mise à l'échelle,donc on essaye pas plusieurs fois avant de déclencher la mise à l'échelle, de ce fait, Theta = 1.

Pourquoi avez- vous choisi seulement le CPU !
Parce que l'utilisation de CPU est plus décisive dans une architecture d'orchestration, c'est pour ça que nous avons opté pour le CPU*, cependant, on a considéré l'ajout de la mémoire comme perspective.*

C'est quoi RMSE !
Le RMSE permet de calculer la précision de prédiction, il représente la distance entre la valeur prédite et la valeur réelle.

C'est quoi la différence entre RMSE Globale et par Période !
Le RMSE Globale permet de mesurer la distance entre toutes les valeurs dans plusieurs séquence réelles et prédites à la fois, tandis que le RMSE par période permet de mesurer la distance entre les valeurs dans plusieurs séquence réelles et prédites mais période par période. Considérons 10 Séquences, chaque séquence contient 10 valeurs (périodes ou timesteps), nous avons une seule valeur de RMSE Globale pour 10*10 et nous avons 10 RMSE, chaque RMSE par période.

C'est quoi la relation entre RMSE par période et chaque cellule LSTM !
RMSE par période estime la précision de prédiction pour chaque cellule.

Peut-on appliquer votre approche dans un système réel?
En fait, notre approche MEAP est déjà un système réel.

Pourquoi vous n'avez pas comparé votre approche avec d'autres dans la littérature !
L'implémentation d'autre approche nécessite plus de temps, ce qui est difficile pour notre cas.

Quel test montre que votre approche améliore l'utilisation de CPU?
Dans la section tests, nous avons deux sous-sections: 1) pour évaluer le modèle de prédiction LSTM en utilisant la méthode RMSE, 2) pour évaluer l'approche MEAP.

Est-ce que vous avez implémenté toutes les contraintes dans le modèle mathématique?
oui, nous avons considéré toutes les contraintes dans le modèle mathématique.

Expliquez chaque contrainte dans le modèle mathématique
- contrainte 1: limitation du nombre de replicas: cette contrainte est proposée pour n'est pas affecter les autres microservices.
- chevauchement : pour assurer qu'un pod appartient à un seul service.
- Équilibrage de charge pour assurer qu'il y a équilibrage de charge après la mise à l'échelle.

Explique la fonction objective dans le modèle mathématique
La fonction objective consiste à minimiser la consommation de CPU et le nombre de fois que la consommation de CPU Réelle arrive au seuil de mise à l'échelle.

Expliquez le seuil de confiance dans le modèle mathématique
Le seuil de confiance permet d'assurer qu'il y a vraiment un dépassement de seuil de mise à l'échelle, ceci vient du fait que parfois la prédiction peut n'est pas donner une valeur correcte.

Est-ce que vous avez considéré la qualité de service (QoS) ?
Nous avons considéré que, l'amélioration de la consommation de CPU revient à respecter la QoS.