Energy-Efficient Horizontal Scaling in Kubernetes with Digital Twins

Franklin T. P. Correia[1,2], **Marcelo A. C. Fernandes**[1,2,3]

[1] InovAI Lab – nPITI/IMD – Universidade Federal do Rio Grande do Norte (UFRN)
Natal – RN – Brazil

[2]Leading Advanced Technologies Center of Excellence (LANCE) – nPITI/IMD
UFRN – Natal – RN – Brazil

[3]Department of Computer Engineering and Automation (DCA) – UFRN
Natal – RN – Brazil

franklin.correia.101@ufrn.edu.br, mfernandes@dca.ufrn.br

Abstract. *This work proposes a machine learning-based digital twin to estimate the energy consumption of Kubernetes clusters from operational metrics. The methodology integrates CPU utilization, network traffic, and the number of pods, incorporating temporal dependencies through an autoregressive structure of depth M. The models were trained under load and configuration scenarios distinct from those used in the evaluation, and the reference power was measured by a dedicated IoT sensor. The results show that the temporal history consistently improves predictive performance, reaching an R^2 of 0.73 and MAE of 3.29 W in scenarios unseen during training, with statistically equivalent performance across the three evaluated models. The model is presented as the prediction component of an energy-oriented horizontal scaling policy, whose integration and closed-loop evaluation constitute the next stage of this work.*

Resumo. *Este trabalho propõe um gêmeo digital baseado em aprendizado de máquina para estimar o consumo energético de clusters Kubernetes a partir de métricas operacionais. A metodologia integra utilização de CPU, tráfego de rede e número de pods, incorporando dependências temporais por meio de uma estrutura autorregressiva de profundidade M. Os modelos foram treinados em cenários de carga e configuração distintos dos usados na avaliação, e a potência de referência foi medida por um sensor IoT dedicado. Os resultados mostram que o histórico temporal melhora consistentemente o desempenho preditivo, alcançando $R^2 = 0{,}73$ e MAE de 3,29 W em cenários não observados no treinamento, com desempenho estatisticamente equivalente entre os três modelos avaliados. O modelo é apresentado como o componente de predição de uma política de escalonamento horizontal orientada a energia, cuja integração e avaliação em malha fechada constituem a etapa seguinte deste trabalho.*

1. Introdução

A crescente adoção de arquiteturas baseadas em microserviços e computação em nuvem tem impulsionado o uso de plataformas de orquestração como o Kubernetes (K8s), especialmente em aplicações distribuídas e de larga escala. Nesse contexto, o escalonamento

horizontal dinâmico torna-se essencial para garantir desempenho e disponibilidade frente a variações na carga de trabalho [Silva et al. 2024, Da Silva et al. 2025, Tran et al. 2022]. Entretanto, esse processo frequentemente desconsidera aspectos energéticos, resultando em uso ineficiente de recursos computacionais e aumento no consumo de energia em infraestruturas modernas. Esse cenário tem motivado o avanço do paradigma de *green computing*, que busca reduzir o impacto energético de sistemas computacionais sem comprometer sua eficiência operacional. Assim, torna-se necessário o desenvolvimento de abordagens que integrem desempenho e eficiência energética de forma conjunta, motivando o uso de modelos preditivos capazes de representar o comportamento dinâmico do sistema ao longo do tempo [Arivoli et al. 2025, Godhani and Bhat 2025].

No contexto do escalonamento horizontal em K8s, diferentes abordagens adaptam dinamicamente o número de réplicas a partir de métricas como utilização de CPU, tráfego de rede e latência. Métodos baseados em lógica fuzzy demonstram ganhos em eficiência e aderência a requisitos de qualidade de serviço [Silva et al. 2024, Li et al. 2021], enquanto redes neurais artificiais têm sido empregadas na previsão de carga e no suporte à decisão de escalonamento [Da Silva et al. 2025]. Estratégias baseadas em regras e ajuste fino do Horizontal Pod Autoscaler (HPA) também seguem sendo investigadas [Freitas et al. 2025, Augustyn et al. 2024]. Esses trabalhos evidenciam o potencial de abordagens inteligentes para melhorar a utilização de recursos, mas ainda apresentam limitações na incorporação explícita de aspectos energéticos e da dinâmica temporal do sistema.

Paralelamente, o conceito de gêmeos digitais (*digital twins*) tem ganhado destaque como uma abordagem promissora para modelagem e simulação de sistemas físicos e computacionais em tempo real. Gêmeos digitais permitem a criação de representações virtuais que evoluem dinamicamente com base em dados observados, possibilitando previsão, monitoramento e otimização de sistemas complexos [Ouedraogo et al. 2025, Nele et al. 2024, Barricelli et al. 2019]. Em ambientes computacionais, o conceito de gêmeos digitais tem sido aplicado para modelar e representar o comportamento de infraestruturas baseadas em Kubernetes, permitindo a análise de desempenho, confiabilidade e operação em larga escala [Borsatti et al. 2023, Borsatti et al. 2024, Wermann and Wickboldt 2025]. Essas abordagens viabilizam a criação de réplicas virtuais de aplicações e clusters, possibilitando monitoramento avançado, simulação de cenários e suporte à tomada de decisão. No entanto, ainda existem lacunas na integração de gêmeos digitais com políticas de escalonamento orientadas à eficiência energética em plataformas como o Kubernetes, especialmente quando se consideram a dinâmica temporal do sistema e a variabilidade da carga de trabalho.

Diante desse cenário, este trabalho propõe uma abordagem baseada em gêmeos digitais e aprendizado de máquina para estimativa do consumo energético em ambientes K8s, com foco na incorporação de dependências temporais por meio de estruturas autoregressivas. O modelo proposto utiliza métricas de CPU, rede e número de pods para prever a potência consumida, servindo como base para futuras estratégias de escalonamento energeticamente eficientes. Os resultados experimentais demonstram que a inclusão de histórico temporal melhora significativamente o desempenho preditivo, alcançando valores de R^2 de até 0,73, com desempenho equivalente entre RF, XGB e SVR, evidenciando a eficácia da abordagem na modelagem da dinâmica energética do sistema.

2. Metodologia

2.1. Coleta de Dados

A coleta de dados foi realizada a partir da integração de múltiplas fontes, representando tanto o comportamento lógico do sistema quanto seu consumo energético real. Conforme ilustrado na Figura 1, o ambiente experimental é composto por um cluster Kubernetes (K8s) executado em uma máquina dedicada, denominada neste trabalho de Servidor K8s (SK8s), no qual uma aplicação é submetida a diferentes níveis de carga gerados pelo aplicativo Apache JMeter, executado em uma segunda máquina, denominada Host. Nesse contexto, a cada n-ésimo instante de tempo, o sistema de monitoramento Prometheus, executado no cluster K8s, é responsável pela coleta de métricas de telemetria, incluindo utilização de CPU ($\mathbf{m}^{\mathrm{cpu}}(n)$), tráfego de rede ($\mathbf{m}^{\mathrm{rp}}(n)$) e informações relacionadas ao número de pods ativos ($m^{\mathrm{p}}(n)$). Em paralelo, o JMeter induz variações controladas na carga de trabalho por meio da simulação de múltiplos usuários concorrentes, sendo o número de usuários associados ao i-ésimo experimento denotado por N_k.

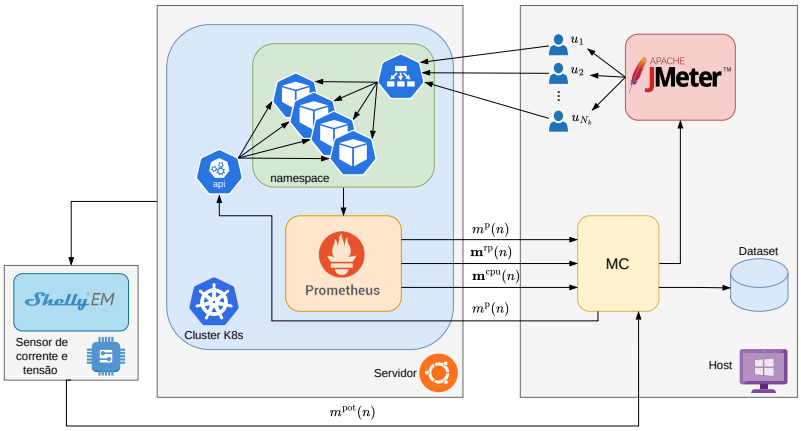


Figura 1. Esquema geral do processo de coleta e integração de dados para construção do dataset.

As métricas de utilização de CPU são representadas pelo vetor $\mathbf{m}^{\mathrm{cpu}}(n)$, definido como

$$\mathbf{m}^{\mathrm{cpu}}(n) = \left[m_{\mathrm{pr}}^{\mathrm{cpu}}(n),\, m_{\mathrm{ef}}^{\mathrm{cpu}}(n),\, m_{\mathrm{pp}}^{\mathrm{cpu}}(n)\right], \tag{1}$$

onde $m_{\mathrm{pr}}^{\mathrm{cpu}}(n)$ corresponde à utilização percentual de CPU no instante n, assumindo valores no intervalo $[0, 1]$, $m_{\mathrm{ef}}^{\mathrm{cpu}}(n)$ representa a carga computacional efetiva do sistema, definida como o produto entre a utilização de CPU e o número de pods ativos, e $m_{\mathrm{pp}}^{\mathrm{cpu}}(n)$ denota a carga média de CPU por pod, obtida pela razão entre a utilização de CPU e o número de pods. Essas variáveis permitem capturar diferentes perspectivas da utilização computacional do sistema, incluindo sua intensidade global, escalabilidade e distribuição entre instâncias. As métricas de tráfego de rede são representadas pelo vetor $\mathbf{m}^{\mathrm{rp}}(n)$, definido como

$$\mathbf{m}^{\mathrm{rp}}(n) = \left[m_{\mathrm{tx}}^{\mathrm{rp}}(n),\, m_{\mathrm{rx}}^{\mathrm{rp}}(n)\right], \tag{2}$$

onde $m_{\mathrm{tx}}^{\mathrm{rp}}(n)$ corresponde à taxa de dados transmitidos pela interface de rede no instante n, enquanto $m_{\mathrm{rx}}^{\mathrm{rp}}(n)$ representa a taxa de dados recebidos, ambos expressos em bytes por segundo. Essas métricas permitem caracterizar a intensidade do tráfego de rede do sistema, refletindo o comportamento de comunicação associado à carga de trabalho.

Paralelamente, o consumo energético do SK8s é medido diretamente por um sensor IoT (Shelly EM), permitindo a obtenção de valores reais de potência elétrica, também a cada n-ésimo instante de tempo, associados ao estado do sistema ($m^{\mathrm{pot}}(n)$). O sensor IoT comunica-se com o Host por meio de uma rede sem fio dedicada, garantindo a coleta contínua das medições elétricas. Adicionalmente, as requisições de carga entre o Host e o SK8s são realizadas por meio de uma rede dedicada, assegurando isolamento e controle experimental durante os testes. As métricas do sistema são coletadas a cada 5 segundos por módulos específicos de monitoramento, os quais são orquestrados por um módulo central denominado Módulo de Controle (MC). O MC é responsável por gerenciar todo o ciclo experimental, incluindo a inicialização, execução e finalização dos experimentos, bem como a sincronização entre as diferentes fontes de dados. Foram realizados um total de $K = 27$ experimentos, considerando diferentes combinações de carga de trabalho e configuração do sistema.

Todos os testes de estresse tiveram duração de 600 segundos. A carga de trabalho foi definida a partir da variação do número de usuários concorrentes, denotado por N_k para o k-ésimo experimento, assumindo os valores de 200, 350, 400, 510 e 600 usuários. Para os cenários com 200, 400 e 600 usuários concorrentes, foi realizada uma variação mais ampla no número de pods, considerando as configurações de 1, 4, 8, 12, 16, 20 e 24 pods. Já para os cenários intermediários, com 350 e 510 usuários concorrentes, foram consideradas configurações específicas de 3, 6 e 18 pods, permitindo avaliar o comportamento do sistema em condições não observadas durante parte dos experimentos e contribuindo para a análise de generalização dos modelos. Os dados das três fontes foram integrados em um *dataset* unificado, que representa o estado do sistema a cada n-ésimo instante. Como sensor IoT, Prometheus e JMeter operam com relógios independentes, essa integração exige tratamento explícito de dessincronização: a série do sensor IoT é adotada como referência temporal, e as demais fontes são associadas a ela por vizinho mais próximo, com tolerância de $\pm 2{,}0$ s para as métricas do Prometheus e $\pm 1{,}5$ s para as do JMeter. Amostras cuja fonte correspondente não apresenta leitura dentro da tolerância são descartadas, sem imputação por médias ou interpolação, de modo a não introduzir informação artificial na base. No conjunto de treinamento, 3 das 2688 amostras ($0{,}1\%$) não apresentaram correspondência com o Prometheus e 148 ($5{,}5\%$) não apresentaram correspondência com o JMeter; no conjunto de teste, essas proporções foram de 0% e $5{,}5\%$, respectivamente. Após o descarte, os conjuntos resultaram em 2540 e 726 amostras.

O SK8s é equipado com processador AMD Ryzen 5 5600G (12 núcleos, 3,3–4,4 GHz), 32 GB de RAM DDR4 3200 MHz e Ubuntu Server 22.04.5. O Host utiliza um Intel Core Ultra 9 185H (16 núcleos, 2,3–5,1 GHz), 32 GB de RAM DDR5 e Windows 11 Home.

2.2. Modelo de Gêmeo Digital Proposto

O modelo de gêmeo digital (GD) proposto tem como objetivo estimar dinamicamente o consumo energético do sistema a partir de seu estado operacional e de sua evolução temporal, fornecendo essa estimativa a um módulo de política de escalonamento (MPE). Conforme ilustrado na Figura 2, o GD é estruturado como um modelo preditivo baseado em aprendizado de máquina, capaz de inferir a potência estimada $\tilde{m}^{\mathrm{pot}}(n)$ a cada n-ésimo instante de tempo, a partir de informações observadas do sistema.

O modelo utiliza como base as métricas instantâneas $\mathbf{m}^{\mathrm{cpu}}(n)$, $\mathbf{m}^{\mathrm{rp}}(n)$ e $m^{\mathrm{p}}(n)$,

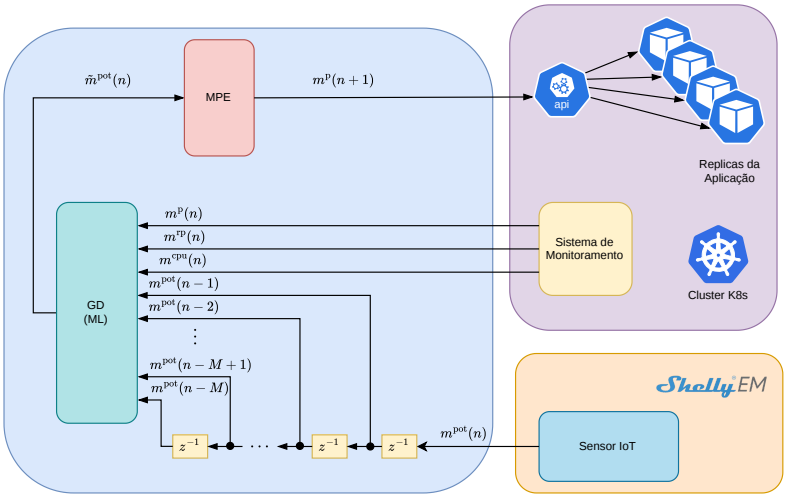


Figura 2. Estrutura do GD proposto. O histórico temporal é composto pelas medições de potência do sensor IoT, propagadas pela cadeia de operadores de atraso z^{-1}. A saída $\tilde{m}^{\mathrm{pot}}(n)$ alimenta o MPE, que define o número de réplicas $m^{\mathrm{p}}(n+1)$ do próximo instante.

juntamente com um histórico temporal da potência efetivamente medida pelo sensor IoT, representado por $m^{\mathrm{pot}}(n-1), m^{\mathrm{pot}}(n-2), \ldots, m^{\mathrm{pot}}(n-M)$. Esse mecanismo confere ao modelo uma natureza autorregressiva, permitindo capturar a dinâmica temporal e a inércia energética do sistema. A implementação dessa memória temporal é realizada por meio de operadores de atraso discretos (z^{-1}), inicializados a cada experimento com as M primeiras medições disponíveis do sensor; as amostras correspondentes a essa fase de aquecimento são descartadas do conjunto de dados, de modo que nenhuma estimativa é produzida sem histórico completo.

A cada instante de tempo, o modelo gera uma estimativa $\tilde{m}^{\mathrm{pot}}(n)$ da potência instantânea a partir do estado operacional corrente e do histórico medido. Cabe destacar que o histórico é composto por medições do sensor, e não por estimativas anteriores do próprio modelo: no cenário de operação pretendido, o sensor Shelly EM fornece a leitura de potência a cada ciclo de 5 s, de modo que $m^{\mathrm{pot}}(n-1)$ está disponível no instante n. Essa sincronização contínua com o sistema físico é o que caracteriza o GD. Para que o MPE avalie uma configuração hipotética de $m^{\mathrm{p}}(n+1)$, o histórico medido permanece disponível e apenas a variável de decisão é alterada, de modo que a estimativa contrafactual é obtida em um único passo. Já a propagação por múltiplos passos, em que as próprias estimativas realimentam o histórico, introduz acúmulo de erro e constitui uma das etapas em andamento deste trabalho.

A saída do GD é utilizada pelo MPE, responsável por determinar a quantidade de réplicas da aplicação, representada por $m^{\mathrm{p}}(n+1)$, a ser adotada no próximo instante. Dessa forma, o modelo proposto integra predição energética e tomada de decisão, permitindo que o escalonamento horizontal considere explicitamente o impacto energético das configurações do sistema. Neste artigo, a análise experimental concentra-se na avaliação do desempenho do GD na estimação do consumo energético, considerando diferentes configurações e profundidades temporais. A integração completa com o MPE é apresentada como uma perspectiva futura, destacando o potencial da abordagem para aplicações

em sistemas autônomos e energeticamente eficientes.

2.3. Treinamento dos Modelos

Para a modelagem do consumo energético, foram utilizados três algoritmos de aprendizado de máquina amplamente empregados em tarefas de regressão: Random Forest (RF), Extreme Gradient Boosting (XGB) e Support Vector Regression (SVR). Esses modelos foram selecionados devido à sua capacidade de capturar relações não lineares e lidar com dados heterogêneos, características relevantes para a modelagem do comportamento energético em ambientes Kubernetes.

A avaliação baseia-se em um conjunto de teste inteiramente disjunto do conjunto de treinamento, composto por cenários de carga e configurações de pods não observados durante o treinamento. Essa separação por cenário, em vez de uma partição aleatória das amostras, evita o vazamento temporal que ocorreria caso amostras adjacentes que compartilham grande parte do vetor de histórico fossem distribuídas entre treino e validação. A avaliação foi realizada por meio das métricas erro absoluto médio (MAE), raiz do erro quadrático médio (RMSE) e coeficiente de determinação (R^2), permitindo analisar tanto a magnitude dos erros quanto a capacidade explicativa dos modelos.

Para a construção dos conjuntos de treinamento e teste, foram utilizados os cenários com 200, 400 e 600 usuários concorrentes, nos quais foi considerada uma variação mais ampla no número de pods, com configurações de 1, 4, 8, 12, 16, 20 e 24 pods. Já os cenários intermediários, com 350 e 510 usuários concorrentes, foram utilizados como conjunto de teste, considerando configurações específicas de 3, 6 e 18 pods, permitindo avaliar a capacidade de generalização dos modelos em condições não observadas durante o treinamento. Ao final, o conjunto de treinamento foi composto por 2540 amostras, enquanto o conjunto de teste totalizou 726 amostras. Adicionalmente, foram conduzidos experimentos variando a profundidade do histórico temporal M incorporado nas features autorregressivas, permitindo investigar o impacto da informação temporal na capacidade preditiva. Essa análise é particularmente relevante no contexto do modelo de gêmeo digital proposto, uma vez que a inclusão de estados passados influencia diretamente a modelagem da dinâmica e da inércia energética do sistema.

Tabela 1. Configuração dos hiperparâmetros dos modelos avaliados

Modelo	Hiperparâmetros
Random Forest	$n_estimators = 800$, $max_depth = 18$, $min_samples_leaf = 2$
XGBoost	$n_estimators = 800$, $learning_rate = 0.03$, $max_depth = 8$, $colsample_bytree = 0.8$, $reg_alpha = 0$, $reg_lambda = 1$
SVR	$kernel = \text{rbf}$, $C = 20$, $\epsilon = 0.5$, $gamma = \text{scale}$

3. Resultados

3.1. Desempenho Geral dos Modelos

A Tabela 2 apresenta o desempenho dos modelos para diferentes configurações de histórico temporal, considerando as métricas erro absoluto médio (MAE), raiz do erro quadrático médio (RMSE) e coeficiente de determinação (R^2).

Observa-se melhora acentuada até $M \approx 8$, a partir do qual as métricas se estabilizam: entre $M = 8$ e $M = 32$ a variação de R^2 do RF é inferior a $0{,}02$. Todos os modelos

Tabela 2. Desempenho no conjunto de teste em função da profundidade do histórico temporal M (MAE e RMSE em watts). Para referência, o preditor de persistência $\tilde{m}^{\mathrm{pot}}(n) = m^{\mathrm{pot}}(n-1)$ obtém MAE=4,28 W, RMSE=6,36 W e R^2=0,40.

M	RF			XGB			SVR		
	MAE	**RMSE**	R^2	**MAE**	**RMSE**	R^2	**MAE**	**RMSE**	R^2
1	4.04	5.59	0.543	4.01	5.40	0.573	4.06	5.83	0.503
2	3.75	5.28	0.593	3.69	5.10	0.620	3.51	5.13	0.615
3	3.73	5.24	0.599	3.80	5.22	0.602	3.50	5.28	0.593
4	3.58	4.93	0.644	3.66	4.93	0.644	3.18	4.21	0.740
6	3.39	4.37	0.715	3.43	4.43	0.707	3.16	4.11	0.747
8	3.34	4.32	0.722	3.50	4.52	0.695	3.17	4.11	0.748
10	3.28	4.23	0.733	3.34	4.32	0.722	3.20	4.18	0.739
12	3.28	4.25	0.732	3.36	4.37	0.716	3.26	4.28	0.727
16	**3.29**	**4.26**	**0.730**	**3.33**	**4.31**	**0.724**	**3.30**	**4.35**	**0.719**
20	3.33	4.29	0.727	3.37	4.37	0.717	3.32	4.38	0.715
24	3.32	4.28	0.729	3.43	4.43	0.710	3.37	4.39	0.715
32	3.32	4.27	0.733	3.38	4.34	0.724	3.43	4.44	0.711

superam com folga o preditor de persistência ($R^2 = 0{,}40$), o que indica que o ganho não decorre apenas da inércia energética. Em $M = 16$, o teste de Wilcoxon pareado sobre os erros absolutos não indica diferença significativa entre os três modelos ($p > 0{,}05$), e os intervalos de confiança de 95% por *bootstrap* se sobrepõem amplamente: $0{,}730$ [$0{,}692$–$0{,}763$], $0{,}724$ [$0{,}686$–$0{,}757$] e $0{,}719$ [$0{,}676$–$0{,}757$] para RF, XGB e SVR. A profundidade M, e não a escolha do algoritmo, é portanto o fator determinante. As análises seguintes adotam $M = 16$, no interior do platô; a inferência do RF nessa configuração requer $30{,}3$ ms por amostra, ou $0{,}61\%$ do ciclo de 5 s, desprezível frente à faixa de 49 a 121 W medida no cluster.

3.2. Análise dos Resultados para o Modelo com RF

A Figura 3 apresenta a evolução temporal da potência real $m^{\mathrm{pot}}(n)$ e da potência estimada $\tilde{m}^{\mathrm{pot}}(n)$, em watts, ao longo das amostras do conjunto de teste, considerando o modelo RF com profundidade temporal $M = 16$. O eixo horizontal corresponde ao índice da amostra, resultante da concatenação dos seis experimentos de validação (3, 6 e 18 pods sob 350 e 510 usuários concorrentes) amostrados a cada 5 s, de modo que as descontinuidades observadas correspondem a transições entre experimentos. Observa-se que o modelo é capaz de acompanhar de forma consistente as variações dinâmicas do sistema, incluindo mudanças abruptas associadas à transição entre cenários de operação. A proximidade entre as curvas indica alta capacidade de ajuste, mesmo em regiões com maior variabilidade, evidenciando a eficácia da incorporação de histórico temporal na modelagem do consumo energético.

A relação entre os valores reais e preditos é apresentada na Figura 4a, onde se observa uma forte concentração dos pontos ao redor da reta identidade ($y = x$). Esse comportamento indica elevada correlação entre as variáveis, refletindo a capacidade do modelo em capturar a dinâmica do consumo energético. Pequenas dispersões são observadas principalmente em regiões de menor potência, sugerindo maior dificuldade de modelagem em condições específicas de operação, possivelmente associadas a transições rápidas de carga. A Figura 4b apresenta a distribuição do erro absoluto em função do número de pods. Observa-se que, embora exista variabilidade nos erros, as medianas

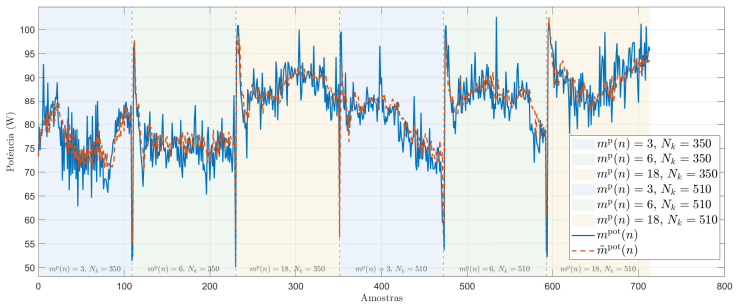
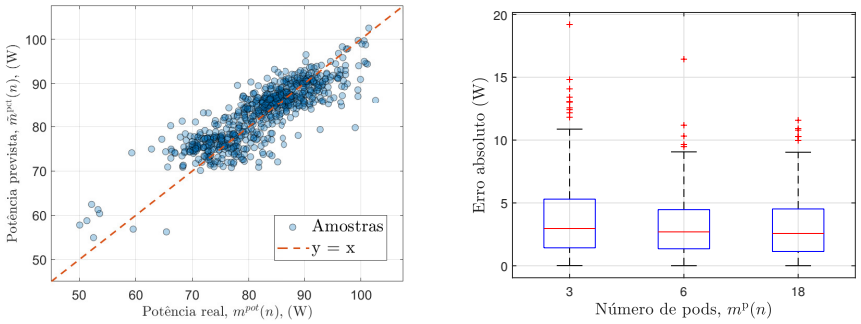


Figura 3. Potência real $m^{\mathrm{pot}}(n)$ e potência estimada $\tilde{m}^{\mathrm{pot}}(n)$ ao longo do tempo para o conjunto de teste para o modelo de RF.

permanecem relativamente baixas e próximas entre os diferentes cenários, indicando consistência do modelo. Além disso, a presença de outliers, principalmente para menores valores de pods, sugere que cenários com menor paralelismo podem apresentar maior sensibilidade a flutuações na carga de trabalho.



(a) Dispersão entre $m^{\mathrm{pot}}(n)$ e $\tilde{m}^{\mathrm{pot}}(n)$.

(b) Distribuição do erro absoluto.

Figura 4. Análise do desempenho do modelo RF no conjunto de teste.

De forma geral, os resultados demonstram que o modelo RF com $M = 16$ apresenta elevado desempenho preditivo no conjunto de teste, sendo capaz de capturar tanto o comportamento estacionário quanto as transições dinâmicas do sistema. Esses resultados reforçam a importância da incorporação de informações temporais na modelagem do consumo energético e validam a abordagem baseada em gêmeos digitais proposta neste trabalho.

3.3. Análise Comparativa com SVR e XGB

As Figuras 5 e 6 apresentam a evolução temporal das estimativas ao longo do conjunto de teste para SVR e XGB, respectivamente. Ambos os modelos acompanham as transições entre experimentos e o comportamento estacionário dentro de cada cenário, sendo as maiores discrepâncias concentradas nos instantes imediatamente posteriores às mudanças de configuração, quando o histórico autorregressivo ainda reflete o cenário anterior.

A Figura 7 relaciona a potência medida $m^{\mathrm{pot}}(n)$ à estimada $\tilde{m}^{\mathrm{pot}}(n)$ no conjunto de teste, para os modelos SVR e XGB com $M = 16$. Ambos concentram-se em torno da reta identidade, com dispersão mais acentuada nas faixas de menor potência — comportamento também observado no RF e associado a transições rápidas de carga.

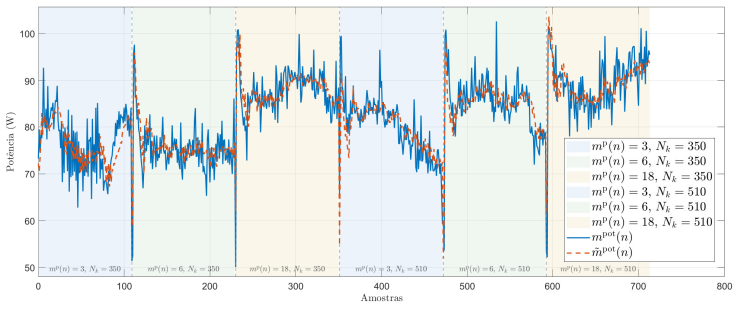


Figura 5. Potência real $m^{\mathrm{pot}}(n)$ e estimada $\tilde{m}^{\mathrm{pot}}(n)$, em watts, ao longo das amostras do conjunto de teste para o modelo SVR ($M = 16$).

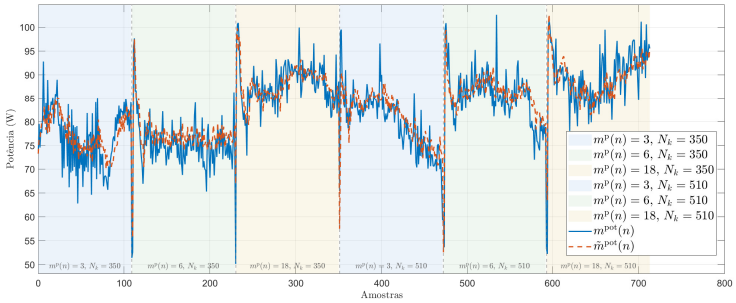
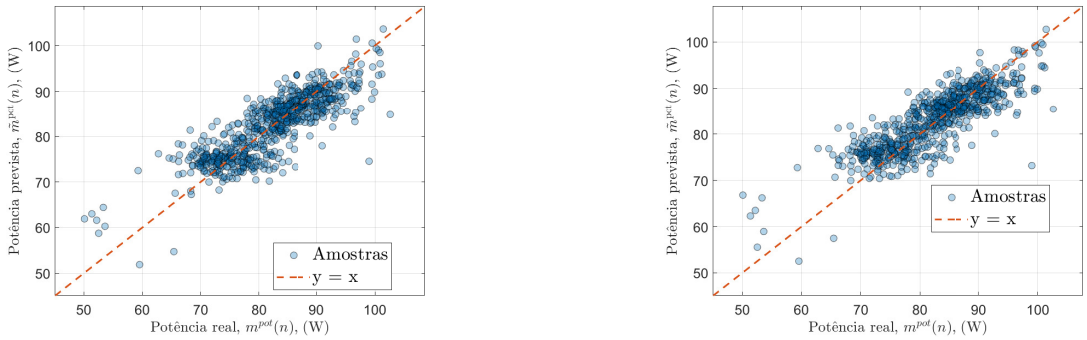


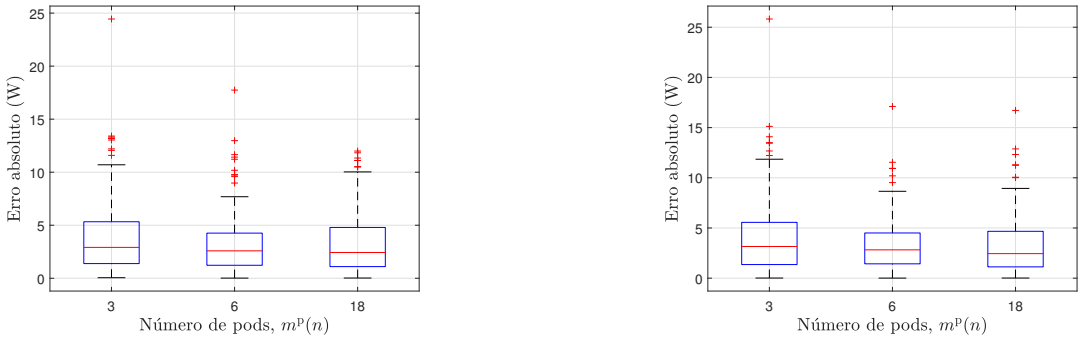
Figura 6. Potência real $m^{\mathrm{pot}}(n)$ e estimada $\tilde{m}^{\mathrm{pot}}(n)$, em watts, ao longo das amostras do conjunto de teste para o modelo XGB ($M = 16$).



(a) SVR. **(b) XGB.**

Figura 7. Dispersão entre $m^{\mathrm{pot}}(n)$ e $\tilde{m}^{\mathrm{pot}}(n)$ no conjunto de teste para os modelos SVR e XGB ($M = 16$).

A Figura 8 apresenta a distribuição do erro absoluto agrupada pelo número de pods do cenário ($m^{\mathrm{p}}(n) \in \{3, 6, 18\}$ no conjunto de teste), com o erro em watts no eixo vertical. Em ambos os modelos as medianas permanecem baixas e homogêneas entre configurações, e os *outliers* concentram-se nos cenários de menor paralelismo, sugerindo maior sensibilidade a flutuações da carga quando há menos réplicas distribuindo o processamento. Esse padrão é o mesmo verificado para o RF na Figura 4b, o que é coerente com a equivalência estatística reportada na Seção 3.1: nenhum dos três modelos apresenta assinatura de erro qualitativamente distinta.



(a) SVR. **(b) XGB.**

Figura 8. Distribuição do erro absoluto em função do número de pods $m^{\mathrm{p}}(n)$ no conjunto de teste, para os modelos SVR e XGB ($M = 16$).

A diferença mais relevante entre os modelos não está na precisão, mas na sensibilidade à profundidade do histórico: conforme a Tabela 2, o SVR degrada progressivamente para $M > 8$, enquanto RF e XGB permanecem estáveis até $M = 32$. Esse comportamento sugere que o SVR é mais suscetível ao aumento de dimensionalidade do vetor de entrada, aspecto relevante na escolha do modelo caso o gêmeo digital opere com históricos mais longos.

3.4. Posicionamento em Relação ao Estado da Arte

A Tabela 3 contextualiza a abordagem proposta em relação aos principais trabalhos relacionados, organizados em duas linhas de pesquisa: escalonamento horizontal em Kubernetes e estimativa de consumo energético em ambientes de contêineres.

Tabela 3. Comparação qualitativa com o estado da arte. H = horizontal; GD = gêmeo digital; HT = histórico temporal; ✓ = sim; – = não abordado. † política de escalonamento especificada, porém não avaliada experimentalmente neste trabalho.

Trabalho	Abordagem	Esc.	Energia	GD	HT / Métricas reportadas
Escalonamento horizontal em Kubernetes					
Silva et al. [Silva et al. 2024]	Lógica Fuzzy (FHS)	H	–	–	– / CPU, réplicas, SLA
Da Silva et al. [Da Silva et al. 2025]	ANN (ANN-HS)	H	–	–	– / réplicas, CPU
Freitas et al. [Freitas et al. 2025]	HPA baseado em CPU	H	–	–	– / latência, throughput
Augustyn et al. [Augustyn et al. 2024]	HPA tuning	H	–	–	– / latência, SLA
Borsatti et al. [Borsatti et al. 2024]	KubeTwin	H	–	✓	– / qualitativo
Wermann & Wickboldt [Wermann and Wickboldt 2025]	KTwin (serverless)	H	–	✓	– / qualitativo
Godhani & Bhat [Godhani and Bhat 2025]	Green AI (K8s)	H	✓	–	– / eficiência de inferência
Estimativa de consumo energético em contêineres					
Algarni et al. [Algarni et al. 2024]	AR/ARIMA/ETS (Docker)	–	✓	–	– / MAPE por contêiner
Saad et al. [Saad et al. 2025]	XGBoost federado (Kepler/K8s)	–	✓	–	– / MAE relativo (%)
Este trabalho	**RF/XGB/SVR autorregressivo**	**H†**	✓	✓	✓/ R^2=0,73; MAE=3,29 W; RMSE=4,26 W

Os trabalhos focados em escalonamento horizontal em K8s [Silva et al. 2024, Da Silva et al. 2025, Freitas et al. 2025, Augustyn et al. 2024] adaptam dinamicamente o número de réplicas com base em métricas operacionais como CPU e latência, mas não incorporam estimativa de potência elétrica. As abordagens baseadas em GD para K8s [Borsatti et al. 2024, Wermann and Wickboldt 2025] avançam na representação virtual de clusters, porém sem integrar modelos preditivos de consumo energético. Por outro lado, os trabalhos que estimam energia em contêineres operam com métricas relativas ou em benchmarks padronizados (e.g., SPECPower), adotando granularidade e contextos

experimentais distintos dos adotados neste trabalho, o que inviabiliza uma comparação numérica direta das métricas de regressão. Essa heterogeneidade reflete a lacuna identificada: nenhum trabalho anterior combina modelagem por GD e incorporação de histórico temporal com avaliação quantitativa por R^2, MAE e RMSE em ambiente Kubernetes real, tendo como referência a potência medida por sensor físico. A abordagem proposta preenche essa lacuna, obtendo $R^2 = 0{,}73$ em cenários de carga não observados durante o treinamento, e constitui a base sobre a qual a política de escalonamento orientada a energia será construída.

4. Conclusões

Este trabalho apresentou uma abordagem baseada em gêmeos digitais e aprendizado de máquina para estimativa do consumo energético em ambientes Kubernetes, considerando explicitamente a dinâmica temporal do sistema. Os resultados demonstraram que a incorporação de histórico temporal por meio de estruturas autorregressivas melhora significativamente o desempenho preditivo, elevando o R^2 de $0{,}54$, com um único atraso, para valores próximos de $0{,}73$ a partir de $M \approx 8$, com RF, XGB e SVR apresentando desempenho estatisticamente equivalente uma vez fornecido histórico suficiente. A abordagem mostrou-se capaz de capturar tanto o comportamento estacionário quanto as transições dinâmicas do consumo energético, superando com folga o preditor de persistência adotado como referência. A avaliação, contudo, foi conduzida em um único servidor físico e sob configurações fixas de pods, e a política de escalonamento foi especificada mas não avaliada experimentalmente. Como trabalhos futuros, pretende-se integrar o GD ao MPE em tempo real, permitindo a adaptação dinâmica do número de pods com base em estimativas energéticas, e comparar a política resultante ao Horizontal Pod Autoscaler nativo do Kubernetes sob tráfego variável em ambiente de produção. Serão ainda investigadas abordagens mais avançadas, como aprendizado por reforço e modelos de aprendizado profundo, bem como o tratamento do acúmulo de erro na operação em malha fechada.

Agradecimentos

Os autores agradecem ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio e financiamento. Esta pesquisa integra o Instituto Nacional de Ciência e Tecnologia em Redes de Comunicação e Internet das Coisas Inteligente (ICoNIoT), financiado pelo CNPq (processo nº 405940/2022-0) e pela CAPES (Código de Financiamento nº 88887.954253/2024-00).

Referências

Algarni, A., Shah, I., Jehangiri, A. I., Ala'Anzy, M. A., and Ahmad, Z. (2024). Predictive energy management for docker containers in cloud computing: A time series analysis approach. *IEEE Access*, 12:52524–52538.

Arivoli, A., Kalaavathi, B., and Iong-Zong, C. J. (2025). Developing green computing awareness based on optimization techniques for environmental sustainability. In *Computational Intelligence in Sustainable Computing and Optimization*, pages 237–258. Elsevier.

Augustyn, D. R., Wyciślik, Ł., and Sojka, M. (2024). Tuning a kubernetes horizontal pod autoscaler for meeting performance and load demands in cloud deployments. *Applied Sciences*, 14(2):646.

Barricelli, B. R., Casiraghi, E., and Fogli, D. (2019). A survey on digital twin: Definitions, characteristics, applications, and design implications. *IEEE access*, 7:167653–167671.

Borsatti, D., Cerroni, W., Foschini, L., Grabarnik, G. Y., Manca, L., Poltronieri, F., Scotece, D., Shwartz, L., Stefanelli, C., Tortonesi, M., et al. (2024). Kubetwin: A digital twin framework for kubernetes deployments at scale. *IEEE Transactions on Network and Service Management*, 21(4):3889–3903.

Borsatti, D., Cerroni, W., Foschini, L., Grabarnik, G. Y., Poltronieri, F., Scotece, D., Shwartz, L., Stefanelli, C., Tortonesi, M., and Zaccarini, M. (2023). Modeling digital twins of kubernetes-based applications. In *2023 IEEE Symposium on Computers and Communications (ISCC)*, pages 219–224. IEEE.

Da Silva, L. M., Alves, P. V., Silva, S. N., and Fernandes, M. A. (2025). Adaptive horizontal scaling in kubernetes clusters with ann-based load forecasting. *Cluster Computing*, 28(3):176.

Freitas, Y. V., de A Melo, J. M., Queiroz, A. P. d. O., and Fernandes, A. M. (2025). Cpu-based horizontal pod autoscaling for distributed iiot control in kubernetes edge environments. In *2025 Symposium on Internet of Things (SIoT)*, pages 1–4. IEEE.

Godhani, P. and Bhat, A. (2025). Green ai: A dynamic framework for energy-efficient inference on kubernetes. In *2025 IEEE International Conference for Women in Innovation, Technology & Entrepreneurship (ICWITE)*, pages 1–6. IEEE.

Li, D. C., Huang, C.-T., Tseng, C.-W., and Chou, L.-D. (2021). Fuzzy-based microservice resource management platform for edge computing in the internet of things. *Sensors*, 21(11).

Nele, L., Mattera, G., Yap, E. W., Vozza, M., and Vespoli, S. (2024). Towards the application of machine learning in digital twin technology: a multi-scale review. *Discover Applied Sciences*, 6(10):502.

Ouedraogo, E. B., Hawbani, A., Wang, X., Liu, Z., Zhao, L., Al-qaness, M. A., and Alsamhi, S. H. (2025). Digital twin data management: a comprehensive review. *IEEE Transactions on Big Data*, 11(5):2224–2243.

Saad, Z., Yang, J., Leung, H., and Drew, S. (2025). Towards carbon-aware container orchestration: Predicting workload energy consumption with federated learning. *arXiv preprint arXiv:2510.03970*.

Silva, S. N., Goldbarg, M. A. S. d. S., Silva, L. M. D. d., and Fernandes, M. A. C. (2024). Application of fuzzy logic for horizontal scaling in kubernetes environments within the context of edge computing. *Future Internet*, 16(9).

Tran, M.-N., Vu, D.-D., and Kim, Y. (2022). A survey of autoscaling in kubernetes. In *2022 Thirteenth International Conference on Ubiquitous and Future Networks (ICUFN)*, pages 263–265. IEEE.

Wermann, A. G. and Wickboldt, J. A. (2025). Ktwin: A serverless kubernetes-based digital twin platform. *Computer Networks*, 259:111095.