Mitigating Concept Drift in Job Execution Time Prediction Models

Bernardo Gallo[1], Lúcia Drummond[1], José Viterbo[1]

Universidade Federal Fluminense, Brazil
bgallo@id.uff.br
lucia@ic.uff.br
viterbo@ic.uff.br

Abstract. This work investigates the mitigation of Concept Drift in machine learning models applied to job execution time prediction in High Performance Computing (HPC) systems. The study assesses the impact of a periodic evaluation strategy with retraining in a machine learning model using SLURM logs from a Petrobras supercomputer. Our experiments demonstrate that Concept Drift significantly impacts the reliability of machine learning models over time. We show that a strategy of daily model evaluation and retraining, triggered when the Mean Absolute Percentage Error (MAPE) exceeds 150%, effectively mitigates Concept Drift. This approach significantly reduces the average MAPE in different periods.The results underscore the necessity of daily evaluations to maintain acceptable predictive model performance in such dynamic settings. This study highlights a practical solution for mitigating Concept Drift, thereby enhancing the applicability of ML models in HPC systems and showcasing their potential to optimize processes in critical sectors like oil and gas.

CCS Concepts: • **Computing methodologies** → **Classification and regression trees**.

Keywords: Concept Drift, Machine Learning, Classification Tree, C4.5

1. INTRODUÇÃO

No setor de Petróleo e Gás, simulações de reservatórios são cruciais para a tomada de decisões estratégicas e redução de riscos no desenvolvimento de campos petrolíferos [Mustafiz and Islam 2008]. Essas simulações, que analisam a dinâmica de fluidos em reservatórios, demandam o uso de sistemas de computação de alto desempenho (HPC) devido ao grande volume de dados e complexidade dos cálculos [Portella et al. 2022]. Em HPC, escalonadores de recursos (como SLURM, Torque e PBS) otimizam o uso dos recursos computacionais, alocando *jobs* com base em suas necessidades para maximizar métricas de interesse do sistema e minimizar o tempo de espera. A previsão precisa do tempo de execução de cada *job* permite que esses sistemas aloquem recursos de forma mais eficiente, evitando ociosidade e implementando políticas de escalonamento sofisticadas, como backfilling, Shortest Job First (SJF) e Shortest Area First (SAF).

Conhecer antecipadamente a duração de cada *job* permite que os sistemas de gerenciamento de recursos, como o SLURM [Yoo et al. 2003] no caso da Petrobras, aloquem de maneira mais eficiente os recursos computacionais, minimizando tempos de espera e evitando ociosidade dos nós de processamento. Isso resulta em uma utilização mais eficiente dos recursos disponíveis, aumentando a produtividade e reduzindo custos operacionais. Além disso, a previsão do tempo de execução permite a implementação de políticas de escalonamento mais sofisticadas, como o *backfilling* [Mu'alem and Feitelson 2001], que insere *jobs* menores em lacunas de tempo que seriam desperdiçadas, e as SJF e SAF. O SJF prioriza os *jobs* mais curtos com o objetivo de otimizar o *throughput* do sistema, contudo

essa heurística sofre de *starvation*, quando é possível um *job* esperar tempo indeterminado na fila. A SAF prioriza os *jobs* que utilizam menos recursos computacionais por menos tempo, também com o objetivo de maximar o *throughput*, assim como o SJF sofre de *starvation*. A capacidade de prever com precisão a duração dos *jobs* também facilita o planejamento a longo prazo e a gestão de cargas de trabalho em cenários de pico, garantindo um desempenho consistente e confiável dos sistemas de HPC [Carastan-Santos et al. 2019].

As mudanças nas características dos dados de entrada de um classificador, em comparação com os dados utilizados durante o treinamento, são um fenômeno denominado *Concept Drift*. Essas mudanças podem manifestar-se de várias formas, incluindo alterações nas distribuições dos atributos, mudanças nas relações entre variáveis de entrada e saída, ou ainda modificações na disponibilidade e qualidade dos dados.[Gama et al. 2009] Tais variações são comuns em ambientes dinâmicos onde os dados estão em constante evolução, como monitoramento de infraestruturas críticas e sistemas de HPC. A ocorrência de *Concept Drift* pode ter consequências significativas para o desempenho dos modelos de ML. Modelos que não são adaptados para lidar com mudanças nos dados correm o risco de se tornar obsoletos, gerando previsões imprecisas e potencialmente prejudiciais. Por exemplo, em um sistema de detecção de fraudes financeiras, uma mudança nos padrões de transações legítimas pode levar a um aumento no número de falsos positivos, sobrecarregando equipes de revisão e gerando custos desnecessários.[Žliobaitė 2010]

Para mitigar os efeitos do *Concept Drift* e garantir a continuidade e a confiabilidade dos sistemas de ML, diversas abordagens têm sido propostas na literatura. Essas abordagens variam desde métodos simples de monitoramento de métricas de desempenho do modelo, até técnicas avançadas de re-treinamento contínuo e algoritmos de aprendizado online capazes de se adaptar em tempo real às mudanças nos dados [Gama et al. 2009]. A detecção precoce do *concept drift* é crucial, permitindo que intervenções corretivas sejam aplicadas antes que o desempenho do modelo se deteriore significativamente. Entre as técnicas mais comuns estão os testes estatísticos para monitoramento de distribuições de dados, métodos de *ensemble learning* para adaptação contínua, e *frameworks* de aprendizado semi-supervisionado que combinam dados rotulados e não rotulados para ajuste dinâmico dos modelos [Bifet 2010].

Em trabalhos anteriores [Nunes et al. 2025], avaliou-se o melhor modelo para o mesmo *dataset*, utilizando uma janela de treinamento de um ano (junho de 2022 a maio de 2023) e uma janela de teste de um mês (junho de 2023). No presente artigo, ampliam-se os experimentos para os meses subsequentes de 2023, de modo a investigar o comportamento temporal do modelo selecionado.Além disso, visando mitigar o *concept drift* em modelos de *machine learning* aplicados à predição de tempo de execução de jobs (TEJ) em sistemas de alto desempenho, propomos uma estratégia de reavaliação contínua e re-treinamento sempre que a métrica de desempenho ultrapassar um determinado *threshold*.

Este estudo teve três objetivos principais:

(1) Mensurar, por meio de dados históricos e experimentos, o impacto do *concept drift* nos *logs* da Petrobras;
(2) Determinar a frequência mínima necessária para reavaliação dos modelos de predição, de forma a mitigar o *concept drift*;
(3) Avaliar o desempenho preditivo após a adoção da estratégia proposta.

Para tanto, foram coletados e analisados dados históricos referentes ao período de junho a dezembro de 2023, empregando a métrica de Erro Percentual Absoluto Médio (MAPE) para comparar a precisão das previsões em uma ampla variação de tempos de execução.

O restante deste trabalho está organizado da seguinte maneira: a Seção 2 apresenta alguns artigos que também abordam a questão do *Concept Drift* e a previsão de tempo de execução de *job*. Na Seção 3 são apresentados os dados e o modelo que foram usados na análise do *Concept Drift*. A Seção 4

apresenta os resultados dos testes realizados. Na Seção 5 são apresentadas as conclusões, as limitações do trabalho e trabalhos futuros.

2. TRABALHOS RELACIONADOS

Em um trabalho anterior [Nunes et al. 2025], explorou-se o uso de ML para prever o TEJ das simulações de reservatórios de petróleo da Petrobras. Utilizou-se um log de produção e foi proposto um preditor de TEJ em duas etapas baseado no classificador J48. A avaliação experimental demonstra que o modelo J48 de duas etapas supera o modelo J48 original, alcançando maior precisão, pontuação kappa e pontuação F1. Além disso, explorou-se outros modelos de ML, como métodos de regressão, incluiu-se métricas adicionais, como Erro Absoluto Médio (MAE) e MAPE, para comparar o desempenho de modelos de regressão e classificação. Os resultados mostraram que o modelo J48 de duas etapas também alcança os menores valores de MAE e MAPE.

O *Concept Drift* também tem sido estudado em contextos de aplicação prática, onde a manutenção da acurácia dos modelos em ambientes em constante mudança é crucial. Gama et al.(2014) apresentaram uma pesquisa detalhada sobre a aplicação de técnicas de *Concept Drift* em diversos domínios, como sistemas de detecção de fraude, diagnóstico médico e monitoramento de redes de sensores. O trabalho destaca como a natureza do *drift* pode variar significativamente de um domínio para outro, exigindo abordagens personalizadas e uma avaliação cuidadosa dos métodos de adaptação empregados.

Um estudo de Barddal et al.(2017) focou na aplicação de técnicas para mitigação de *Concept Drift* em sistemas de recomendação. O trabalho propõe o uso de algoritmos adaptativos que ajustam as recomendações com base nas mudanças de interesse e comportamento dos usuários ao longo do tempo. A pesquisa demonstra que a consideração do *Concept Drift* pode melhorar significativamente a relevância das recomendações, especialmente em plataformas dinâmicas como serviços de *streaming* e *e-commerce*.

O crescimento exponencial dos sistemas de HPC destacou a necessidade de previsões precisas de tempo de execução para otimizar a utilização de recursos e reduzir ineficiências. Estimativas incorretas do TEJ, geralmente fornecidas pelos usuários, frequentemente resultam em atrasos ou na interrupção prematura de tarefas. O artigo de Menear et al.(2023) aborda esses desafios, propondo uma abordagem metodológica para a previsão de tempo de execução com base em uma extensa pesquisa e em um conjunto de dados com mais de 11 milhões de *jobs* do supercomputador *petascale* Eagle, do Laboratório Nacional de Energia Renovável (NREL).

O estudo destaca a distribuição enviesada dos tempos de execução, identifica inconsistências em técnicas de previsão (divisão de dados, seleção de características e avaliação de modelos) e propõe uma abordagem orientada por dados, utilizando características numéricas e categóricas com codificação de rótulos. O artigo sugere uma divisão de dados consciente do tempo, modelos como XGBoost retreinados diariamente com janela de 100 dias para ótimos resultados, e o uso de métricas de erro granulares baseadas no tempo de execução real.

3. MATERIAIS E MÉTODOS

O conjunto de dados explorado de *jobs* de simulação de reservatórios de petróleo , originado dos *logs* do SLURM de um supercomputador, abrange as atividades diárias de mais de 300 engenheiros da Petrobras. O conjunto de dados cobre mais de 6 milhões de registros coletados ao longo de três anos e pode ser utilizado para desenvolver modelos preditivos baseados em ML para tempo de execução de *jobs*, entre outras possibilidades. No entanto, como essa infraestrutura de HPC passou por diversas atualizações durante esses três anos, nesse trabalho, o período de análise foi limitado a um ponto estável da infraestrutura.

Portanto, o conjunto de dados bruto compreende os *jobs* gerenciados pelo Slurm de 1º de junho

de 2022 até 31 de dezembro de 2023. Ele contém 3.621.661 registros de *jobs*, dos quais 1.083.224 são de 2022 e 2.538.437 são de 2023. A Tabela I resume o tipo, nome e descrição dos campos usados no modelo. Uma técnica de filtragem foi aplicada ao conjunto de dados bruto considerando as seguintes regras:

—descartar *jobs* com status de erro
—descartar *jobs* de teste
—descartar *jobs* não relacionados a *scripts* específicos

O conjunto de dados de treinamento varia para cada teste no presente trabalho, contudo o conjunto de treinamento inicial consiste em 1.174.186 *jobs* de 1º de junho de 2022 a 30 de maio de 2023 (89,94%), e o conjunto de dados de teste consiste em 131.266 *jobs* de junho de 2023 (10,06%).

Table I: Tipo, nome, e descrições dos 46 campos disponíveis nos logs do Slurm.

Tipo	Nome	Descrição
	account	Conta associada com o *job*
	groupname	Nome do grupo associado com o *job*.
	job_name	Nome do *job*.
	nodes	Nós que executaram o *job*.
	qos	Prioridade de escalonamento do *job*.
	script	Script Batch submetido para o *job*.
	tres_alloc	Quantidade de recursos alocado para o *job*.
	tres_req	Quantidade de recurso pedido pelo *job*.
	username	Usuário que submeteu o *job*.
	work_dir	Caminho do diretório de trabalho (Working directory) do script Batch executado pelo *job*.
Nominal	@eligible	*Timestamp* do momento que o *job* ficou pronto para executar.
	@end	*Timestamp* do momento que o *job* terminou.
	@start	*Timestamp* do momento que *job* começou.
	@submit	*Timestamp* do momento que o *job* foi submetido.
	jobid	Identificador para o *job*.
Ordinal	cpus_per_task	Número de Núcleos de CPU por tarefa alocada para o *job*.
	elapsed	Tempo de execução do *job* em segundos.
	ntasks	Número de processos paralelos executado pelo *job*
Discrete	ntasks_per_node	Número de tarefas por nó alocado para o *job*.
	ntasks_per_tres	Número de tarefas por recurso alocado para o *job*.
	queue_wait	Tempo em segundos que o *job* esperou na fila antes de começar a executar.
	time_limit	Tempo limite para execução do *job* em segundos.
	total_cpus	Número de Núcleos de CPU alocados para o *job*.
	total_nodes	Números de nós computacionais alocados para o *job*.
Continuous	cpu_hours	Tempo de execução, em horas, multiplicado pelo número de núcleos de CPU alocados.

3.1 Algoritmo de treinamento

O modelo preditivo utilizado foi o J48 (duas etapas), o J48 é uma implementação do C4.5 [Quinlan 1993] em java presente na ferramenta WEKA [Frank et al. 2010], selecionado por sua eficácia comprovada na literatura, e pelos bons resultados encontrados anteriormente para a predição de TEJ nos *logs* da Petrobras[Nunes et al. 2023; Nunes et al. 2025]. Os modelos foram treinados e avaliados utilizando um conjunto de dados particionado em janelas de treinamento e teste, conforme descrito a seguir: Partindo do período de treinamento de um ano, junho de 2022 a maio de 2023, foi utilizado para treinamento, utilizando o *10-fold-cross-validation* para validação, os modelos gerados então foram testados para os meses seguintes, junho, julho, agosto, etc.

Por um número muito desbalanceado de *jobs* para o atributo *job_elapsed_class*, uma estratégia de classificação em duas etapas foi usada para lidar com a maioria dos *jobs* que durem até 1800 segundos(classe zero). No primeiro passo, um modelo m_0 foi treinado considerando apenas duas classes: classe zero e classe não-zero,onde a última agrega os *jobs* das classes 1 a 14. Para o segundo passo, um modelo m_1 foi treinado considerando classes 1 a 14. Os modelos treinados foram usados para previsão de *jobs* usando o seguinte critério: se um *job* é classificado por m_0 como pertencendo à classe zero, sua classificação definitiva é determinada como zero. Porém, se um *job* é classificado

por m_0 como sendo da classe não-zero, esse mesmo *job* é avaliado por m_1, que então determinará sua classificação definitiva.

3.2 Métrica de Avaliação

Ao avaliar a acurácia de modelos preditivos em diversas áreas, o MAE e o MAPE emergem como métricas de desempenho cruciais. O MAE oferece uma medida direta da magnitude do erro nas unidades originais dos dados. Em contrapartida, o MAPE expressa o erro como uma porcentagem do valor real, proporcionando uma interpretação relativa da precisão, útil para comparar previsões em diferentes escalas.

Como no caso estudado o tempo dos *jobs* varia de alguns minutos para dias, foi escolhido o MAPE. Para os modelos classificadores utilizou-se o menor valor da categoria para medir o MAPE. O MAPE é calculado da seguinte maneira:

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100$$

Onde:

—n: número total de observações;
—y_i: valor real da i-ésima observação ($y_i \neq 0$);
—$\hat{y}_i$: valor predito pelo modelo para a i-ésima observação.

3.3 Monitoramento de performance e Re-treinamento

Esse trabalho propõe uma estratégia para lidar com o *Concept Drift*, aplicada a modelos de predição de TEJ de simulação de reservatórios executados no sistema de HPC da Petrobras. Dado um modelo inicial C_n, ele será avaliado a cada t_{av} dias para detecção do *drift*. Caso o MAPE tenha excedido um Q_{th}, é executado um novo treinamento que produzirá um modelo C_{n+1}, repetindo-se o ciclo. O presente trabalho procura encontrar um t_{av} razoável.

4. RESULTADOS EXPERIMENTAIS

A presente seção analisa e discute os resultados dos experimentos realizados para avaliar a estratégia de mitigação *Concept Drift* em modelos de predição de TEJ em sistemas de HPC. Os experimentos focaram em determinar a periodicidade necessária para a avaliação do modelo J48 (duas etapas) e no impacto da estratégia proposta na acurácia das previsões.

4.1 Avaliação da periodicidade de avaliação

Para determinar o melhor t_{av} foram treinados modelos J48, em um esquema de predição de duas etapas, descrito anteriormente, e avaliados de forma semanal com o objetivo de determinar a ocorrência do *Concept Drift*. Uma vez que identificou-se o drift, treinou-se novos modelos quando o MAPE era maior que 150%. Usou-se 150% como valor para o Q_{th} por pragmatismo; esse valor não necessariamente é o ótimo. O gráfico da Figura 1 compara o resultado do MAPE para o modelo sem nenhuma mitigação do *Concept Drift* com a estratégia de avaliação e re-treinamento periódico com um $t_{av} = 7$ durante o período de 7 de Junho até 11 de outubro. Pode-se perceber que quando o MAPE do modelo atinge 156.15% de MAPE em 19 de Julho, se o modelo é re-treinado com os dados atualizados, ele atinge 89.72 na semana seguinte; caso contrário, o MAPE seria 168.43% para a mesma semana. Na média o MAPE do modelo sem mitigação é 164.21% enquanto a da estratégia proposta é de 133.08%, uma diferença significativa.

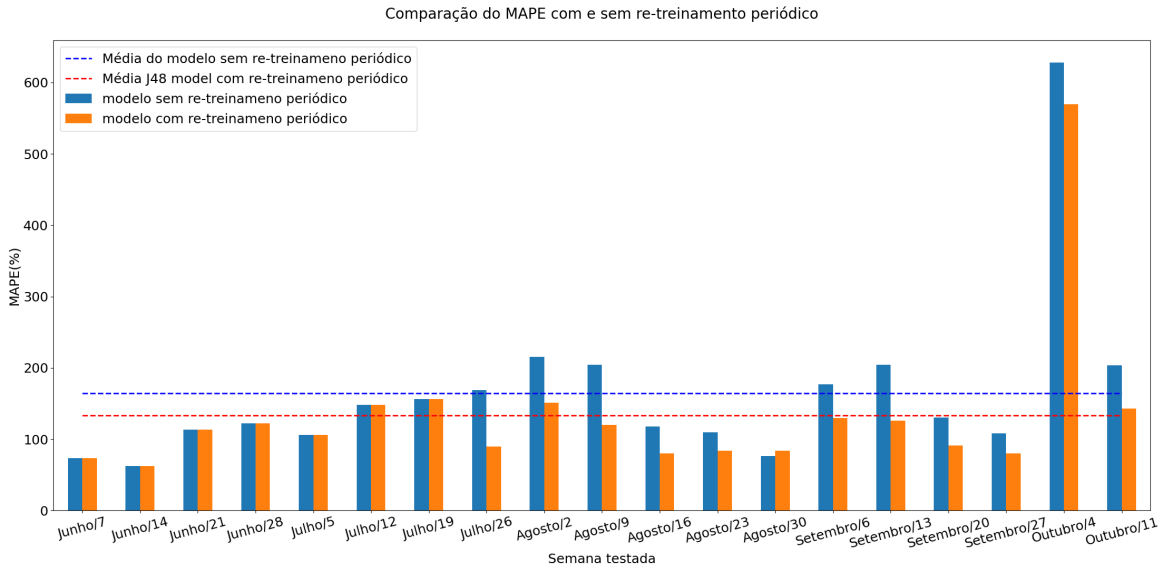

Fig. 1: Comparação do MAPE dos modelos com re-treinamento quando MAPE$\geq$ 150% e $t_{av} = 7$

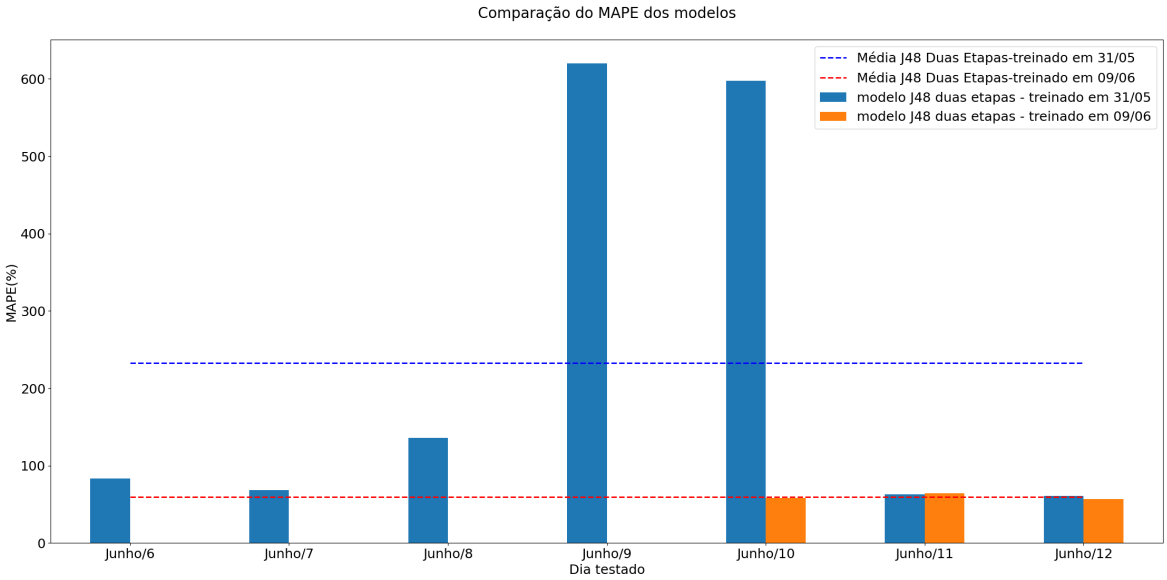

Fig. 2: Comparação do MAPE dos modelos com re-treinamento quando MAPE$\geq$ 150% e $t_{av} = 1$

Porém, os dados analisados demonstram variações diárias significativas no comportamento dos logs do SLURM da Petrobras, o que impacta diretamente a performance dos modelos preditivos. Como o gráfico da Figura 2 demonstra, os dados apresentam variações diárias significativas, a ponto do mesmo modelo apresentar um MAPE de 136.2% em 8 de junho e um MAPE de 619.7% no dia seguinte. Contudo, com os dados do dia 9 de junho, o modelo foi capaz de se adaptar, obtendo um MAPE de 58.11% no dia 10 de junho.

Outrossim, o gráfico da Figura 3 apresenta resultados semelhantes, para um outro período de tempo. Um modelo treinado dia 29 de setembro apresentou um MAPE de 79.87% enquanto um modelo

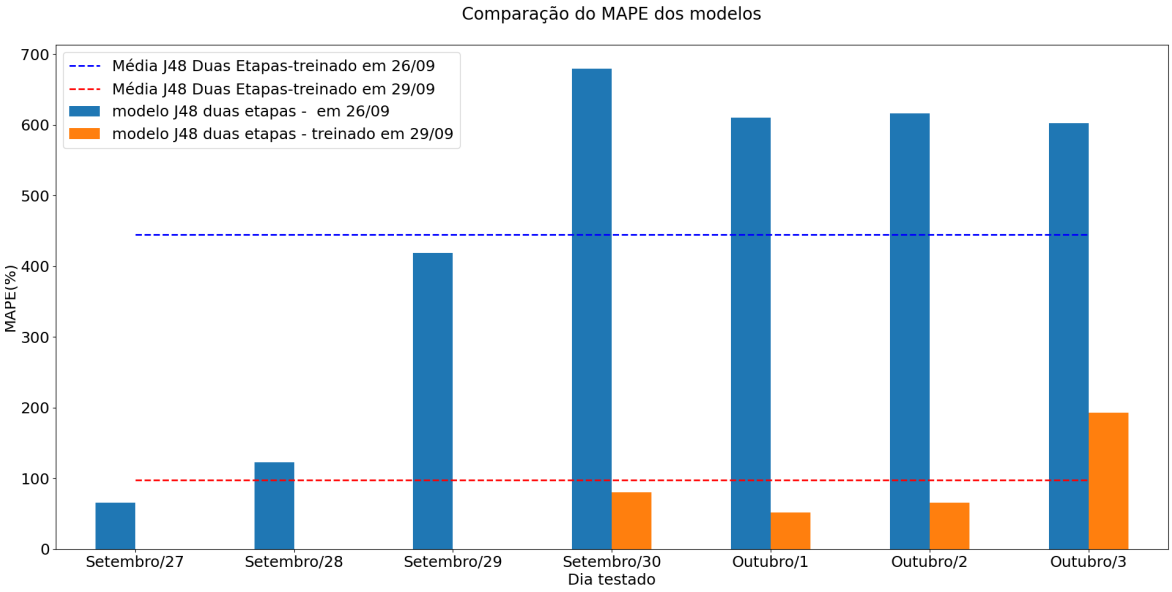

Fig. 3: Comparação do MAPE dos modelos com re-treinamento quando MAPE$\geq$ 150% e $t_{av} = 1$

treinado dia 26 de setembro apresentou 679.50%. Pode-se perceber, portanto, que a mudança nos dados dos *logs* do SLURM da Petrobras são diárias, necessitando um t_{av} de um dia. Esses resultados experimentais demonstram que a estratégia de avaliação diária e re-treinamento condicional, quando o MAPE supera 150%, é eficaz em mitigar o *Concept Drift* e em reduzir significativamente o MAPE médio em diferentes períodos sem incorrer em um custo computacional elevado. A necessidade da avaliação diária sublinha a natureza dinâmica dos ambientes de HPC, onde os padrões de carga de trabalho e o comportamento dos *jobs* podem mudar rapidamente.

5. CONCLUSÃO

A conclusão deste trabalho reflete a concretização de seus objetivos propostos e destaca as contribuições no campo do aprendizado de máquina aplicado à predição de TEJ em sistemas de HPC, abordando, de forma específica, a problemática do *Concept Drift*. O estudo analisou o comportamento dinâmico dos *logs* do SLURM da Petrobras, evidenciando como as mudanças no ambiente impactam a performance dos modelos preditivos.

Os resultados obtidos confirmam que a ocorrência de *Concept Drift* é um fator determinante para a degradação da confiabilidade dos modelos de ML ao longo do tempo. Os experimentos conduzidos pontuam a necessidade de avaliações diárias para preservar um desempenho aceitável dos modelos de predição. Com avaliações diárias e re-treinamentos quando o MAPE do modelo superava 150% foi possível diminuir significativamente o MAPE na média, em dois períodos diferentes.

Mesmo com resultados promissores, é importante ressaltar que o presente trabalho possui algumas limitações. Entre elas está a generalização dos resultados; os experimentos foram feitos com um modelo e um *dataset*. Em estudos futuros, planeja-se explorar soluções alternativas, como algoritmos de aprendizado *online*, outros modelos de ML e o uso de dados não presentes nos *logs* do SLURM que podem ser benéficos e que foram coletados para o período de 2024.

Em suma, este trabalho oferece uma contribuição relevante ao investigar os impactos do *Concept Drift* e propõe uma solução prática para mitigá-los, fortalecendo a aplicabilidade de modelos de ML em sistemas de HPC e evidenciando o potencial dessas tecnologias para otimizar processos em setores

de grande importância, como o de petróleo e gás.

REFERENCES

Barddal, J. P., Gomes, H. M., Enembreck, F., and Pfahringer, B. A survey on feature drift adaptation: Definition, benchmark, challenges and future directions. *Journal of Systems and Software* vol. 127, pp. 278–294, 2017.

Bifet, A. Adaptive stream mining: Pattern learning and mining from evolving data streams. *Frontiers in Artificial Intelligence and Applications* vol. 207, pp. 1–212, 01, 2010.

Carastan-Santos, D., De Camargo, R. Y., Trystram, D., and Zrigui, S. One can only gain by replacing easy backfilling: A simple scheduling policies case study. In *2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. pp. 1–10, 2019.

Frank, E., Hall, M., Holmes, G., Kirkby, R., Pfahringer, B., Witten, I. H., and Trigg, L. pp. 1269–1277. In , *Weka-A Machine Learning Workbench for Data Mining*. Springer, pp. 1269–1277, 2010.

Gama, J. a., Sebastião, R., and Rodrigues, P. P. Issues in evaluation of stream learning algorithms. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '09. Association for Computing Machinery, New York, NY, USA, pp. 329–338, 2009.

Gama, J. a., Žliobaitundefined, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. A survey on concept drift adaptation. *ACM Comput. Surv.* 46 (4), mar, 2014.

Menear, K., Nag, A., Perr-Sauer, J., Lunacek, M., Potter, K., and Duplyakin, D. Mastering hpc runtime prediction: From observing patterns to a methodological approach. In *Practice and Experience in Advanced Research Computing 2023: Computing for the Common Good*. PEARC '23. Association for Computing Machinery, New York, NY, USA, pp. 75–85, 2023.

Mu'alem, A. and Feitelson, D. Utilization, predictability, workloads, and user runtime estimates in scheduling the IBM SP2 with backfilling. *IEEE Transactions on Parallel and Distributed Systems* 12 (6): 529–543, 2001.

Mustafiz, S. and Islam, M. R. State-of-the-art petroleum reservoir simulation. *Petroleum Science and Technology* 26 (10-11): 1303–1329, 2008.

Nunes, A. L., Gallo, B., Lopes, B., Portella, F. A., Viterbo, J., Drummond, L. M. A., Andrade, L., de Lima, M., Estrela, P. J. B., and Malini, R. Q. Two-step estimation strategy for predicting petroleum reservoir simulation jobs runtime on an hpc cluster. *Concurrency and Computation: Practice and Experience* 37 (4-5): e70026, 2025.

Nunes, A. L., Portella, F., Estrela, P., Malini, R., Lopes, B., Bittencourt, A., Leite, G., Coutinho, G., and Drummond, L. Prediction of Reservoir Simulation Jobs Times Using a Real-World SLURM Log. In *Anais do XXIV Simpósio em Sistemas Computacionais de Alto Desempenho*. SBC, Porto Alegre/RS, pp. 49–60, 2023.

Portella, F., Buchaca, D., Rodrigues, J. R., and Berral, J. L. TunaOil: A tuning algorithm strategy for reservoir simulation workloads. *Journal of Computational Science* vol. 63, 2022.

Quinlan, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, 1993.

Yoo, A. B., Jette, M. A., and Grondona, M. SLURM: Simple Linux Utility for Resource Management. In *Job Scheduling Strategies for Parallel Processing*. Springer, pp. 44–60, 2003.

Žliobaitė, I. Learning under concept drift: an overview, 2010.