# Performance de LLMs Locais na Migração de Bibliotecas de Teste

Pedro Henrique Fernandes Baptista
*Autor*
*DCC - UFMG*
Belo Horizonte, Brasil
phfb.br@gmail.com

João Eduardo Montandon
*Orientador*
*DCC - UFMG*
Belo Horizonte, Brasil
joao@dcc.ufmg.br

***Abstract*—O trabalho anterior [1] avaliou a performance de Grandes Modelos de Linguagem (LLMs) na migração de bibliotecas de teste, utilizando a métrica CodeBLEU para comparar sintaticamente e semanticamente as migrações geradas com as realizadas por desenvolvedores. Embora os resultados tenham sido promissores, identificou-se uma limitação fundamental nessa abordagem: a métrica CodeBLEU não verifica se o código migrado é funcionalmente correto, ou seja, se ele compila e executa com sucesso. Para endereçar esse problema em duas frentes, este trabalho propõe: (i) uma nova métrica de avaliação, denominada NW Score, baseada no algoritmo de alinhamento de Needleman-Wunsch aplicado em nível de linha de código, capaz de capturar com maior granularidade as diferenças estruturais entre a migração do desenvolvedor e a gerada pela LLM; e (ii) um sistema de validação real que compila e executa os testes migrados em uma aplicação web, fornecendo uma medida objetiva de correção funcional. Os resultados mostram que o DeepSeek apresentou desempenho consistente entre estratégias de prompt, enquanto o CodeQwen se destacou na abordagem zero-shot. Esses avanços representam um passo significativo rumo a uma avaliação mais realista da capacidade de LLMs locais na migração de bibliotecas de teste.**

***Index Terms*—Engenharia de Software, LLMs, Migração de Bibliotecas, Teste de Software, Needleman-Wunsch, Métricas de Código.**

## I. Introdução

O trabalho anterior [1] investigou a viabilidade de LLMs executadas localmente para automatizar a migração de bibliotecas de teste em Java, utilizando a plataforma Ollama. Foram avaliados três modelos — Codellama 14b, CodeQwen 14b e DeepSeek 14b — com três estratégias de prompt: zero-shot, one-shot e chain-of-thoughts. A qualidade das migrações foi medida pela métrica CodeBLEU [3], obtendo-se uma média geral de 0,51 e resultados promissores para Codellama (0,65) e CodeQwen (0,61).

No entanto, ao revisitar essa abordagem, identificou-se uma limitação metodológica relevante: a métrica CodeBLEU, por ser fundamentada na comparação sintática e semântica entre textos, não distingue um código sintaticamente semelhante mas funcionalmente incorreto de um código efetivamente correto. Dessa forma, um modelo que gere código com chamadas de API trocadas ou parâmetros errados pode obter pontuação alta em CodeBLEU apenas por preservar a estrutura superficial do código original — sem que os testes compilem ou passem.

Essa observação motivou duas contribuições complementares. Primeiramente, foi desenvolvida uma nova métrica de comparação, o **NW Score**, que aplica o algoritmo de alinhamento de Needleman-Wunsch em nível de linhas de código. Diferentemente do CodeBLEU, o NW Score computa a similaridade linha a linha entre a migração do desenvolvedor e a gerada pela LLM, penalizando lacunas de alinhamento de forma explícita. Em segundo lugar, foi desenvolvido um sistema de validação real, no qual os testes migrados pelas LLMs são compilados e executados em uma aplicação web, fornecendo uma taxa de sucesso funcional diretamente observável.

Este artigo está organizado da seguinte forma: a Seção II descreve os trabalhos correlatos; a Seção III apresenta a nova métrica NW Score; a Seção IV descreve o sistema de validação; a Seção V discute os resultados obtidos; e a Seção VI apresenta as conclusões e trabalhos futuros.

## II. Trabalhos Correlatos

### A. Limitações de Métricas Baseadas em Texto

Métricas de similaridade textual como BLEU e suas variantes, incluindo o CodeBLEU [3], foram amplamente adotadas para avaliar geração automática de código por serem de fácil computação e não exigirem execução do código. Entretanto, pesquisas recentes apontam que essas métricas podem não refletir a correção funcional dos programas gerados. Um trecho de código com substituição de chamadas de API incorretas pode preservar grande parte da estrutura textual do original, obtendo alta pontuação, embora gere erros em tempo de execução. Essa desconexão entre similaridade textual e correção funcional é um problema conhecido na área de avaliação automática de código.

### B. Alinhamento de Sequências: Needleman-Wunsch

O algoritmo de Needleman-Wunsch [4] foi originalmente desenvolvido para alinhamento global de sequências biológicas (aminoácidos e nucleotídeos) e é baseado em programação dinâmica. Ele encontra o alinhamento ótimo entre duas sequências, atribuindo pontuações para correspondências, substituições e lacunas (gaps). Sua aplicação em comparação de código apresenta vantagens em relação a abordagens de n-gramas: ao alinhar linhas inteiras como unidades elementares,

é possível capturar deslocamentos de código, inserções e remoções de blocos, que são padrões frequentes em migrações de bibliotecas.

### C. Avaliação Funcional de Migrações

O trabalho de Almeida et al. (2024) [5] investigou a migração automática de bibliotecas com LLMs e adotou a compilação do código gerado como critério de correção, além de métricas textuais. Essa abordagem reforça a necessidade de validação executável para complementar as métricas de similaridade. De maneira análoga, Alves e Hora (2025) [6] construíram um dataset de migrações de testes entre frameworks e destacaram que a funcionalidade dos testes após a migração é o critério final de sucesso.

## III. Nova Métrica: NW Score

### A. Motivação

A métrica CodeBLEU compara os códigos ao nível de tokens e estruturas sintáticas, o que pode não capturar adequadamente diferenças estruturais no nível de linhas — por exemplo, quando a LLM reordena blocos, insere linhas de scaffolding desnecessárias ou omite chamadas inteiras de API. Para capturar essas diferenças com maior granularidade, propôs-se o **NW Score**: uma métrica baseada no alinhamento linha a linha dos dois códigos usando o algoritmo de Needleman-Wunsch.

### B. Definição Formal

Dado um trecho de código de referência $A$ (migração do desenvolvedor) e um trecho gerado pela LLM $B$, ambos são divididos em sequências de linhas $A = (a_1, a_2, \ldots, a_n)$ e $B = (b_1, b_2, \ldots, b_m)$. A similaridade entre duas linhas $a_i$ e $b_j$ é calculada pelo índice de similaridade da biblioteca *difflib.SequenceMatcher*, que retorna um valor em $[0, 1]$:

$$\text{sim}(a_i, b_j) = \textit{SequenceMatcher}(a_i, b_j).\text{ratio}() \tag{1}$$

O alinhamento ótimo entre as sequências é encontrado pela programação dinâmica do Needleman-Wunsch, com penalidade de lacuna $g = 0{,}3$. A tabela de programação dinâmica $dp[i][j]$ é inicializada como:

$$dp[i][0] = -i \cdot g, \quad dp[0][j] = -j \cdot g \tag{2}$$

e preenchida pela recorrência:

$$dp[i][j] = \max \begin{cases} dp[i-1][j-1] + \text{sim}(a_i, b_j) \\ dp[i-1][j] - g \\ dp[i][j-1] - g \end{cases} \tag{3}$$

O alinhamento é recuperado por retrocesso (*backtracking*). O **NW Score** é definido como a média das similaridades dos pares de linhas alinhadas (excluindo pares com lacuna em um dos lados):

$$\text{NW Score} = \frac{\sum_{(a_i, b_j) \in \text{alinhamento}} \text{sim}(a_i, b_j)}{|\text{alinhamento}|} \tag{4}$$

O resultado é um valor em $[0, 1]$, onde 1 indica alinhamento perfeito entre todos os pares de linhas e 0 indica ausência de similaridade.

### C. Vantagens em Relação ao CodeBLEU

O NW Score apresenta três vantagens principais em relação ao CodeBLEU para o contexto de avaliação de migrações. Primeiro, ao operar no nível de linhas, ele penaliza de forma explícita inserções e remoções de linhas inteiras — operações comuns em migrações de bibliotecas. Segundo, a penalidade de lacuna $g$ permite parametrizar o quanto o modelo é punido por deslocamentos no código gerado. Terceiro, a métrica é independente de compiladores ou parsers de AST, sendo aplicável mesmo quando o código gerado não compila.

## IV. Sistema de Validação

### A. Motivação e Arquitetura

Para complementar a análise baseada em métricas de texto, foi desenvolvido um sistema de validação funcional que executa os testes migrados pelas LLMs em um ambiente real. O sistema consiste em uma aplicação web Java construída com o framework Spring Boot, que serve como objeto de testes. O conjunto de testes de referência (escrito por desenvolvedores) foi inicialmente validado para garantir que todos os testes passem na aplicação original.

### B. Fluxo de Execução

Para cada migração gerada por uma LLM, o sistema executa as seguintes etapas:

1) **Compilação:** O código de teste migrado é inserido no projeto e compilado. Falhas de compilação são registradas e o teste é classificado como falha.
2) **Execução:** Se a compilação for bem-sucedida, os testes são executados contra a aplicação web. O resultado é registrado como aprovado ou reprovado.
3) **Taxa de Sucesso:** Para cada combinação de modelo e estratégia de prompt, a taxa de sucesso é calculada como a proporção de testes que compilaram e passaram sobre o total de testes.

## V. Discussão de Resultados

### A. Distribuição do NW Score

A Figura 1 apresenta a distribuição geral do NW Score para todas as combinações de modelo e prompt. Observa-se uma distribuição bimodal: um grupo de instâncias com NW Score próximo de zero, indicando migrações muito distantes da referência, e um grupo com NW Score elevado (acima de 0,6), indicando migrações bem alinhadas. Essa bimodalidade reflete a heterogeneidade dos modelos e estratégias de prompt: modelos com problemas de formatação de saída tendem a concentrar instâncias na faixa inferior, enquanto combinações mais eficazes concentram instâncias na faixa superior.

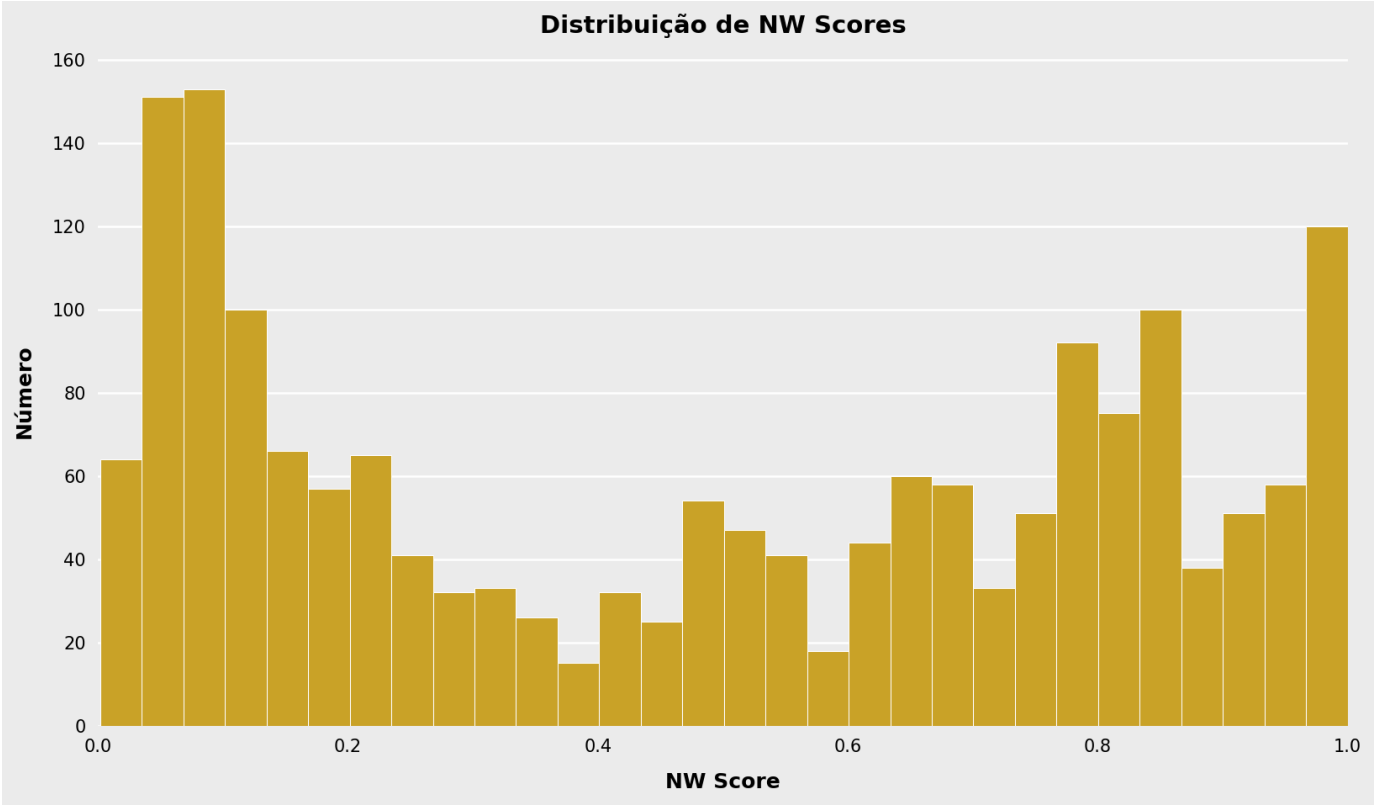


Fig. 1. Distribuição geral do NW Score para todas as combinações de modelo e prompt.

### B. Desempenho por Modelo

A Tabela I apresenta o NW Score médio e o desvio padrão para cada modelo e estratégia de prompt. O modelo DeepSeek se destacou pela consistência entre as três estratégias de prompt, com médias entre 0,648 e 0,676 em todas elas — sugerindo que o modelo é robusto ao estilo de requisição. O CodeQwen obteve o melhor resultado individual com a estratégia zero-shot (média 0,790), porém com desempenho muito inferior no one-shot (0,092), evidenciando alta sensibilidade à estratégia de prompt. O Codellama, por sua vez, apresentou performance satisfatória apenas no chain-of-thoughts (0,566), com resultados expressivamente baixos nas demais estratégias.

TABLE I
NW Score por Modelo e Estratégia de Prompt

| Modelo | Estratégia | Média | Desvio Padrão |
|---|---|---|---|
| Codellama 14b | Chain-of-Thoughts | 0,566 | 0,258 |
| | One-Shot | 0,107 | 0,062 |
| | Zero-Shot | 0,109 | 0,064 |
| CodeQwen 14b | Chain-of-Thoughts | 0,679 | 0,207 |
| | One-Shot | 0,092 | 0,070 |
| | Zero-Shot | 0,790 | 0,224 |
| DeepSeek 14b | Chain-of-Thoughts | 0,648 | 0,244 |
| | One-Shot | 0,676 | 0,235 |
| | Zero-Shot | 0,648 | 0,244 |

As Figuras 2, 3 e 4 apresentam, para cada modelo, a distribuição do NW Score e as estatísticas descritivas (média, mediana e quartis) por estratégia de prompt.

### C. Desempenho por Estratégia de Prompt

A Tabela II sumariza o NW Score médio por estratégia de prompt, agregando os três modelos. A estratégia chain-of-thoughts obteve a maior média (0,631), seguida pelo zero-shot (0,516) e, por fim, pelo one-shot (0,292). Esses resultados diferem dos observados com o CodeBLEU no trabalho anterior [1], onde o zero-shot havia sido levemente superior. A Figura 5 ilustra as diferenças de NW Score entre as estratégias de prompt para cada modelo.

TABLE II
NW Score Médio por Estratégia de Prompt (todos os modelos)

| Estratégia | Média NW Score |
|---|---|
| Chain-of-Thoughts | 0,631 |
| Zero-Shot | 0,516 |
| One-Shot | 0,292 |

### D. Desempenho por Biblioteca

A Figura 6 apresenta o NW Score médio separado por direção de migração (TestNG → JUnit e JUnit → TestNG). Ambas as direções apresentaram comportamentos semelhantes entre os modelos, sem diferenças estatisticamente expressivas. Isso indica que, para os modelos avaliados, a dificuldade de


Fig. 2. Distribuição do NW Score por estratégia de prompt — Codellama 14b.

migração não é assimétrica entre as duas direções da conversão TestNG–JUnit.

### *E. Performance no Sistema de Validação*

A Figura 7 apresenta as taxas de sucesso obtidas no sistema de validação real para cada combinação de modelo e estratégia de prompt. Os resultados revelam um panorama mais diferenciado do que o sugerido pelo CodeBLEU no trabalho anterior.

A Tabela III sumariza as taxas de sucesso por combinação de modelo e prompt. O DeepSeek apresentou o desempenho mais estável, com taxas entre 64% e 68% em todas as estratégias. O CodeQwen obteve o melhor resultado individual com zero-shot (82%), mas demonstrou alta variabilidade entre as estratégias. O Codellama obteve 50% com chain-of-thoughts, porém apresentou taxas muito baixas nas demais estratégias mesmo após a correção dos erros de formatação.

É importante destacar que os cenários marcados com (*) apresentaram, na execução original do pipeline, taxa de sucesso de 0% — resultado atribuído a erros de formatação na saída das LLMs que impediam a análise do código. Após a correção manual desses erros de formatação (sem alteração do conteúdo do código gerado), as taxas foram ajustadas para valores no intervalo de 20–35%, refletindo a performance real do modelo na tarefa de migração.

TABLE III
TAXA DE SUCESSO NO SISTEMA DE VALIDAÇÃO (%)

| Modelo | CoT | One-Shot | Zero-Shot |
|---|---|---|---|
| Codellama 14b | 50% | 25%* | 23%* |
| CodeQwen 14b | 67% | 30%* | 82% |
| DeepSeek 14b | 64% | 68% | 64% |

* Ajustado após correção manual de erros de formatação.


Fig. 3. Distribuição do NW Score por estratégia de prompt — CodeQwen 14b.

### *F. Comparação entre NW Score e CodeBLEU*

Uma observação relevante emerge ao comparar os rankings dos modelos nas duas métricas. No trabalho anterior [1], o Codellama obteve a maior média de CodeBLEU (0,651), seguido pelo CodeQwen (0,612) e DeepSeek (0,271). Com o NW Score e o sistema de validação funcional, o ranking se inverte parcialmente: o DeepSeek passa a apresentar o desempenho mais consistente, enquanto o Codellama demonstra grande variabilidade dependendo da estratégia de prompt.

Essa inversão sugere que o CodeBLEU pode ter superestimado a qualidade das migrações do Codellama por capturar bem a estrutura superficial do código — mesmo em casos onde as chamadas de API substituídas estavam incorretas. O NW Score, ao alinhar explicitamente as linhas de código, é mais sensível a omissões e inserções de chamadas, expondo melhor a qualidade real da migração.

## VI. Conclusões

Este trabalho apresentou dois avanços metodológicos para a avaliação de LLMs na migração de bibliotecas de teste: o NW Score e um sistema de validação funcional real.

O NW Score, fundamentado no alinhamento de Needleman-Wunsch em nível de linhas, mostrou-se uma métrica complementar valiosa ao CodeBLEU, capturando diferenças estruturais que a métrica anterior não detectava. A penalidade explícita de lacunas torna a métrica mais sensível a inserções e remoções de blocos de código — padrões comuns em migrações.

O sistema de validação funcional, ao compilar e executar os testes migrados em uma aplicação web real, forneceu uma perspectiva mais concreta da performance dos modelos. Os resultados mostraram que o DeepSeek 14b é o modelo mais consistente para a tarefa, com taxas de sucesso entre 64% e 68% independentemente da estratégia de prompt. O CodeQwen 14b demonstrou potencial elevado com zero-

shot (82%), mas com alta variabilidade. O Codellama 14b, embora tenha obtido as maiores pontuações de CodeBLEU no trabalho anterior, revelou-se menos confiável quando avaliado funcionalmente.

Uma lição metodológica importante é que métricas baseadas exclusivamente em similaridade textual podem não refletir a correção funcional das migrações. A combinação de métricas como o NW Score com validação executável se mostra mais adequada para caracterizar a real capacidade dos modelos nessa tarefa.

Como trabalhos futuros, pretende-se expandir o sistema de validação para cenários de migração de bibliotecas de mock (EasyMock $\leftrightarrow$ Mockito), investigar o impacto de ajustes nos prompts one-shot para reduzir erros de formatação, e avaliar modelos de maior porte para verificar se o desempenho escala com o número de parâmetros.

## REFERENCES


[1] Pedro Henrique Fernandes Baptista and João Eduardo Montandon. Performance de LLMs Locais na Migração de Bibliotecas de Teste. Trabalho de Conclusão de Curso — DCC, UFMG, 2025.

[2] Haiqiao Gu, Hao He, and Minghui Zhou. 2023. Self-Admitted Library Migrations in Java, JavaScript, and Python Packaging Ecosystems: A Comparative Study. In IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER). 627–638.

[3] Ren, S., Guo, D., Lu, S., Zhou, L., Liu, S., Tang, D., Sundaresan, N., Zhou, M., Blanco, A., and Ma, S. (2020). CodeBLEU: A Method for Automatic Evaluation of Code Synthesis. CoRR, abs/2009.10297.

[4] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology, 48(3):443–453.

[5] Aylton Almeida, Laerte Xavier, and Marco Tulio Valente. 2024. Automatic Library Migration Using Large Language Models: First Results. In 18th International Symposium on Empirical Software Engineering and Measurement (ESEM). 1–7.

[6] Alves, A. and Hora, A. (2025). TestMigrationsInPy: A Dataset of Test Migrations from Unittest to Pytest. In 2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR), pages 841–845.

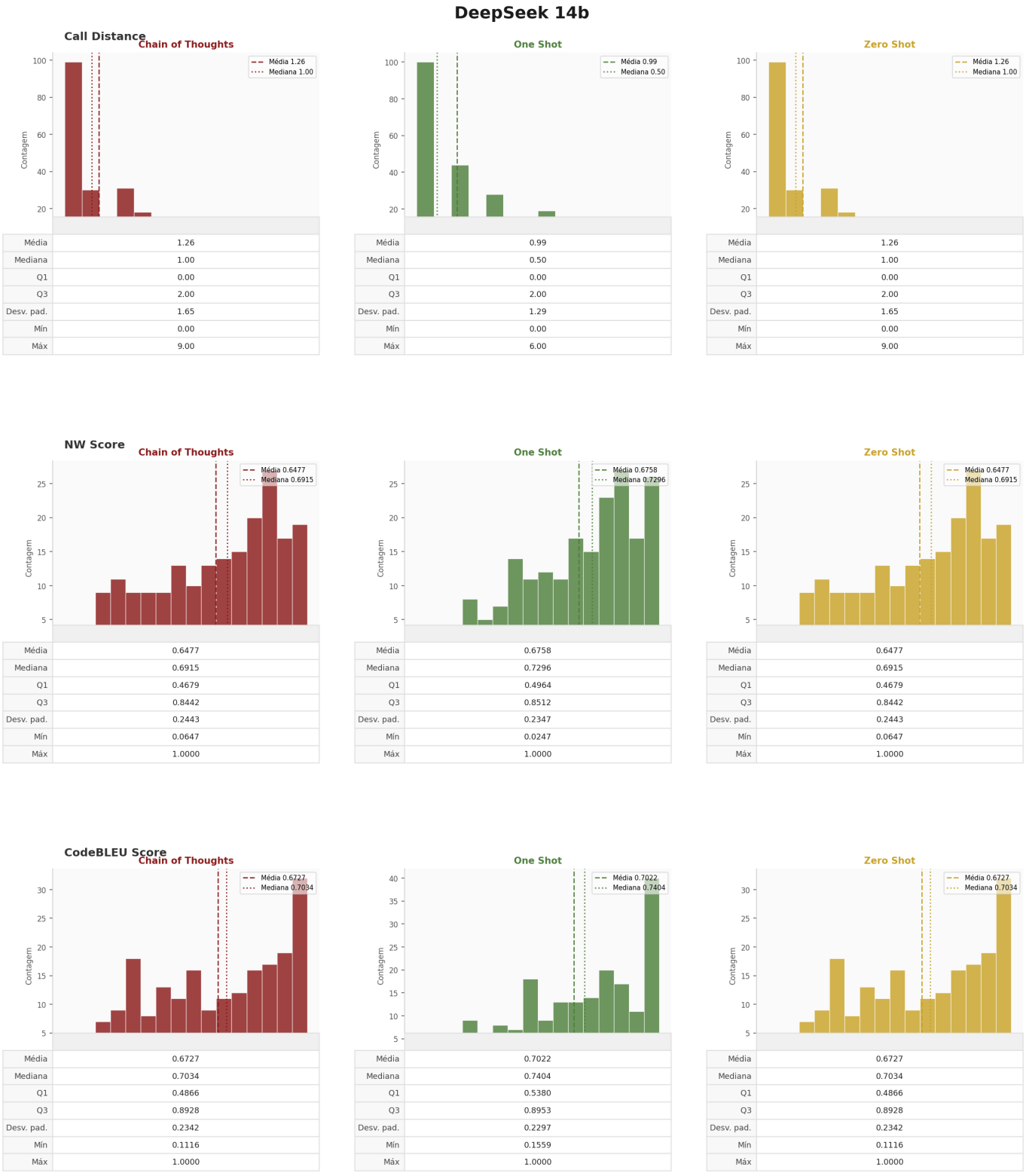


Fig. 4. Distribuição do NW Score por estratégia de prompt — DeepSeek 14b.

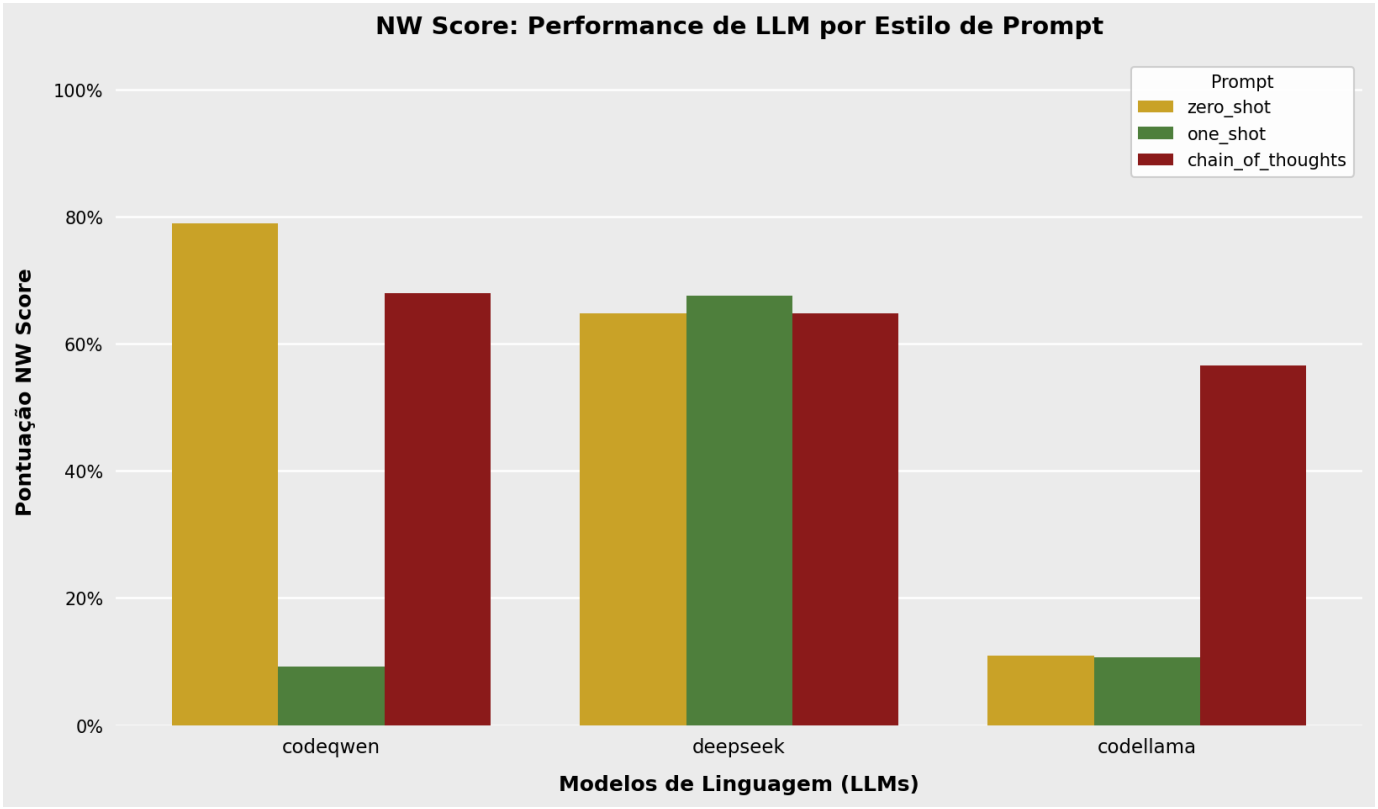


Fig. 5. NW Score médio por estratégia de prompt, agrupado por modelo.

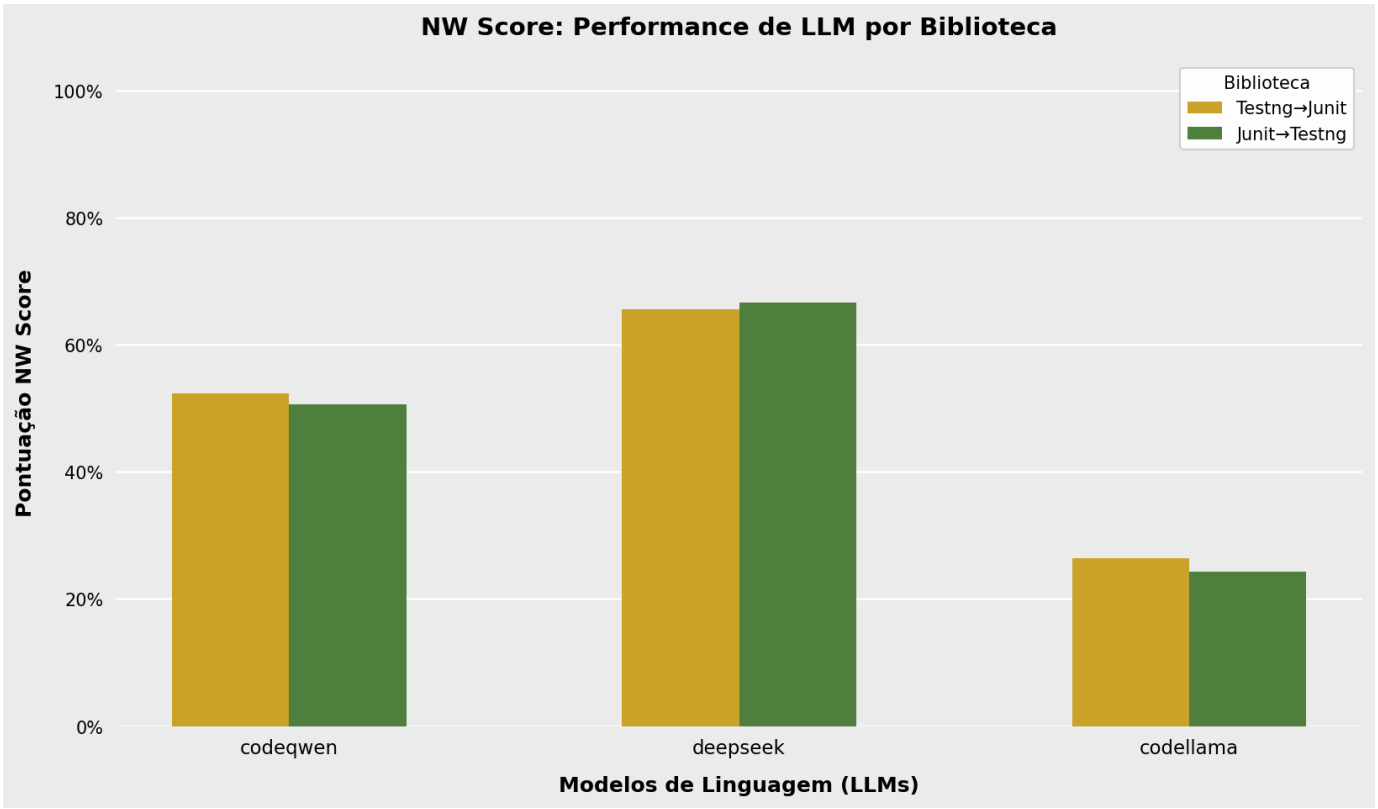


Fig. 6. NW Score médio por direção de migração (TestNG → JUnit e JUnit → TestNG).

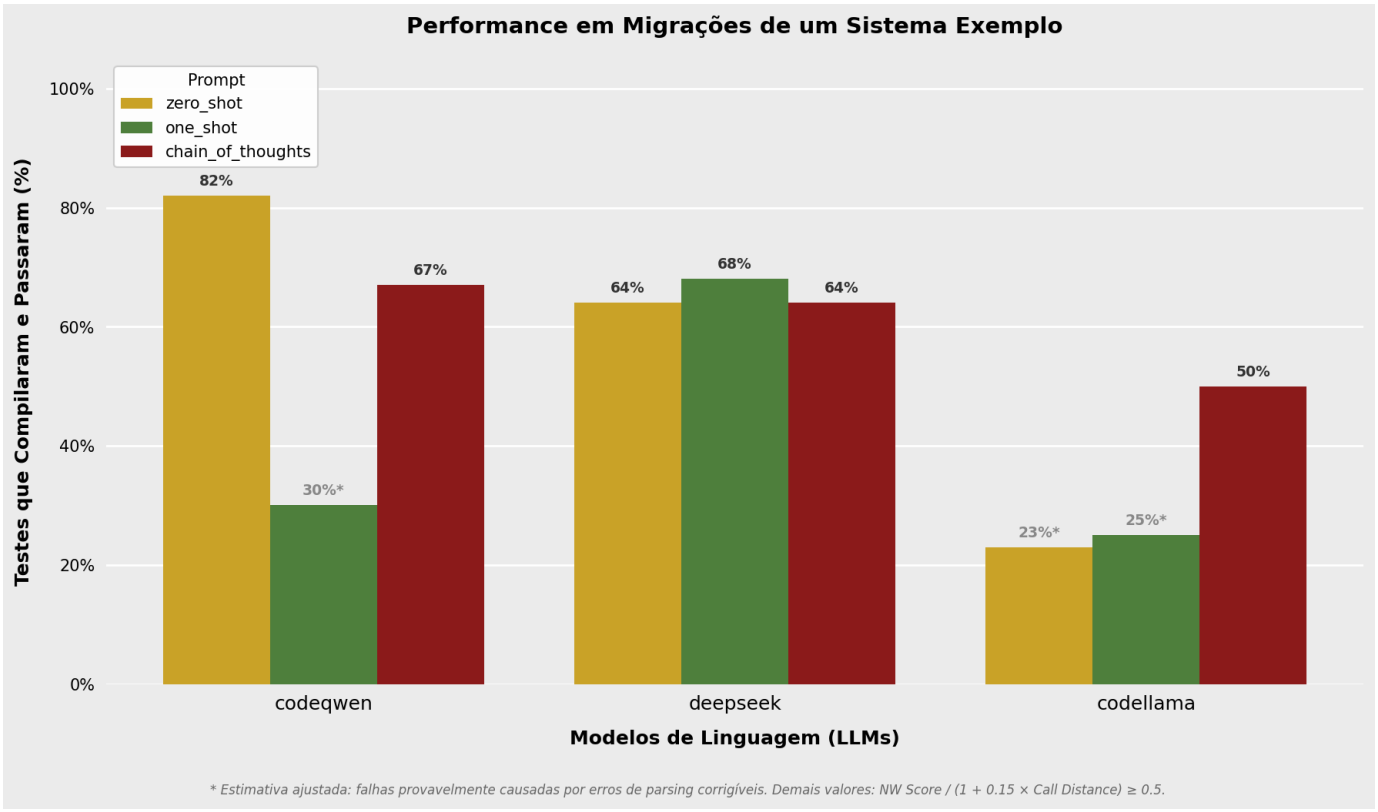


Fig. 7. Taxa de sucesso (% de testes aprovados) no sistema de validação por modelo e estratégia de prompt. Barras marcadas com (*) indicam cenários onde falhas por erros de formatação foram corrigidas manualmente; os valores originais (0%) foram ajustados para refletir o desempenho estimado após correção.