Desempenho vs. Reconhecimento no OSS Global: Uma Análise sobre o Reconhecimento de Desenvolvedores de Países Emergentes no GitHub

Ana Júlia Teixeira[1],
Marcella Chaves[1]

[1]Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
R. Dom José Gaspar, 500 - Coração Eucarístico, Belo Horizonte - MG, 30535-901

1. Introdução

A participação de desenvolvedores de países emergentes, como Brasil e Índia, em projetos de software de código aberto (OSS) tem recebido crescente atenção em estudos recentes. Pesquisas como [Barbosa et al. 2022] apontam para uma presença cada vez maior desses países no ecossistema global de OSS. Entretanto, grande parte dessas análises enfatiza o volume de contribuições. Permanecem abertas questões fundamentais: em que medida esses profissionais estão envolvidos em funções centrais de coordenação e decisão nos projetos? Ou sua atuação se concentra, predominantemente, em tarefas mais periféricas?

A subvalorização sistemática desses profissionais já é documentada no mercado tradicional. Segundo o [HackerRank 2020], desenvolvedores americanos ganham quase três vezes mais ($109.167/ano) que seus pares indianos ($38.229/ano). Esta disparidade não apenas persiste, mas se expande para novos contextos geográficos. O mercado de trabalho remoto tem testemunhado a emergência de outros países emergentes, como o Brasil, competindo por posições similares às tradicionalmente ocupadas por desenvolvedores indianos, evidenciando a institucionalização de hierarquias salariais baseadas em geografia ao invés de mérito técnico.

O diferencial desta pesquisa reside em fornecer evidências quantitativas e objetivas para investigar se esse padrão de subvalorização se replica no ambiente OSS.

2. Objetivo

Investigar se desenvolvedores de países emergentes, especificamente do Brasil, são subvalorizados em termos de reconhecimento e influência em projetos de software de código aberto internacionais, mesmo quando apresentam desempenho e participação comparáveis ou superiores aos de desenvolvedores de países desenvolvidos, especificamente da Alemanha, caracterizando assim o fenômeno da "mão de obra barata" no ecossistema global de software.

3. Metodologia

3.1. Visão Geral do Pipeline de Coleta

A metodologia de coleta de dados foi estruturada em um pipeline sequencial de cinco scripts principais, projetados para capturar diferentes dimensões de atividade e colaboração em repositórios de código aberto do GitHub. O pipeline segue uma arquitetura de dependências rígida, onde cada script subsequente depende dos dados gerados pelos anteriores, garantindo consistência e integridade dos dados coletados.

3.2. Seleção de Repositórios (script3.py)

Objetivo: Identificar e selecionar repositórios representativos para análise comparativa entre desenvolvedores de países emergentes e desenvolvidos.

Metodologia: O script utiliza a API de busca do GitHub (`/search/repositories`) para identificar os 200 repositórios mais populares (ranqueados por número de estrelas). Para cada repositório selecionado, o sistema busca de forma exaustiva todos os contribuidores através de paginação completa da API (`/repos/{owner}/{repo}/contributors`). Para cada contribuidor identificado, o script coleta informações do perfil (`/users/{login}`) e aplica algoritmos de geocodificação para determinar o país de origem baseado no campo `location` do perfil. O repositório só será incluído na seleção se possuir um ou mais contribuidores dos países alvo.

Países-alvo: Brasil, Índia, Alemanha e Estados Unidos foram selecionados como representantes de economias emergentes (Brasil, Índia) e desenvolvidas (Alemanha, Estados Unidos).

3.3. Identificação Completa de Países (script2.py)

Objetivo: Expandir a identificação geográfica para todos os contribuidores dos repositórios selecionados.

Metodologia: O script processa todos os repositórios identificados no Script 1 e realiza uma coleta exaustiva de todos os contribuidores de cada repositório. A identificação de países utiliza uma abordagem híbrida de três etapas:

1. **Mapeamento direto**: Busca em dicionário de aliases para termos comuns (“Brasil”, “USA”, “”).
2. **Identificação regional**: Reconhecimento de estados/províncias e cidades (ex.: “SP”, “California”, “Mumbai”).
3. **Geocodificação**: Uso da API Nominatim (OpenStreetMap) para localizações complexas, seguido de normalização via biblioteca `pycountry`.

3.4. Métricas de Repositórios (script1.py)

Objetivo: Coletar métricas quantitativas detalhadas dos repositórios para caracterização do contexto de desenvolvimento.

Metodologia: Para cada repositório selecionado, o script coleta métricas através de múltiplos endpoints da API GitHub:

- Metadados básicos (`/repos/{owner}/{repo}`)
- Contagem de pull requests (`/repos/{owner}/{repo}/pulls`)
- Histórico de commits com paginação completa
- Métricas de atividade (issues, releases, colaboradores)
- Cálculo de métricas derivadas (tempo médio de resposta, tempo até merge)

Métricas coletadas: 20 métricas quantitativas incluindo estrelas, forks, commits, PRs (abertos/mergeados), tempo de resposta em issues, periodicidade de releases, e contagem de mantenedores ativos.

3.5. script_prs_data.py

Funcionalidade: Este script coleta dados de *Pull Requests* (PRs) de repositórios GitHub. Ele pode operar em dois modos:

- **Modo Real:** Usa a API do GitHub para coletar dados reais.
- **Modo Simulação:** Gera dados sintéticos para teste.

Métricas Coletadas:

- `repo_name`: Nome do repositório.
- `pr_number`: Número do PR.
- `author`: Autor do PR.
- `reviewers_requested`: Revisores solicitados (separados por vírgula).
- `opened_at`: Data/hora de abertura do PR.
- `merged_at`: Data/hora de merge (se aplicável).
- `closed_at`: Data/hora de fechamento (se aplicável).

Arquivo de Saída: `prs_raw.csv`

Características Técnicas:

- Uso de `ThreadPoolExecutor` para processamento paralelo.
- Implementa controle de taxa (*rate limiting*) e lógica de repetição (*retry logic*).
- Suporte a múltiplos tokens de API para contornar limites de requisição.
- Coleta dados paginados (até 100 PRs por página).

3.6. script_user_metrics.py

Funcionalidade: Script mais complexo que coleta métricas detalhadas de desenvolvedores utilizando programação assíncrona para maior desempenho. Analisa a atividade de usuários específicos em repositórios GitHub.

Métricas Coletadas:

Métricas de Pull Requests:

- `prs_opened`: Número total de PRs abertos pelo usuário.
- `prs_merged`: Número de PRs aceitos (merged).
- `pr_accept_rate`: Taxa de aceitação de PRs (%).
- `avg_time_to_merge`: Tempo médio em dias para merge.
- `pr_requested_as_reviewer_rate`: Taxa de solicitações como revisor (%).

Métricas de Atividade:

- `commits_total`: Total de commits do usuário no repositório.
- `issues_opened`: Total de issues abertas pelo usuário.
- `contribution_period`: Período de contribuição em dias.
- `activity_frequency`: Frequência de atividade (commits/dia).

Métricas de Popularidade:

- `stars_own_repos`: Total de estrelas nos repositórios próprios do usuário.

Informações Gerais:

- `login`: Nome de usuário no GitHub.

- `profile_url`: URL do perfil.
- `location`: Localização declarada.
- `country`: País identificado.
- `repo_name` e `repo_url`: Repositório analisado.

Arquivos Utilizados:

- *Entrada:* `users_countries.csv`
- *Saída:* `users_metrics.csv`

Considerações Técnicas

Gerenciamento de Rate Limits: Todos os scripts implementam rotação automática entre tokens de autenticação GitHub e sistema de retry inteligente com backoff exponencial.

Integridade de Dados: Pipeline com verificação de dependências rígida — execução fora da ordem resulta em falhas controladas com mensagens de erro específicas.

Escalabilidade: Uso extensivo de processamento paralelo/assíncrono adaptado às limitações de cada endpoint da API GitHub.

Recuperação de Falhas: Salvamento incremental em todos os scripts de longa duração, permitindo retomada de coleta após interrupções.

4. Dificuldades

A principal dificuldade enfrentada foi a coleta de dados via API REST do GitHub. O volume de dados necessário ultrapassa amplamente o limite de requisições por token, obrigando o desenvolvimento de scripts que operassem com múltiplos tokens simultaneamente (14 no total). Mesmo assim, e trabalhando com amostra reduzida, as limitações da API representaram um obstáculo significativo.

Além disso, o tempo de execução completo dos scripts ultrapassou 78 horas de processamento contínuo.

Outra dificuldade relevante foi dominar a ferramenta Looker Studio para visualização dos dados.

5. Resultados

Estados Unidos vs Índia

Research Questions (RQs)

RQ1: Comparação de Desempenho

Como se compara o desempenho de desenvolvedores de países emergentes (Índia) ao de desenvolvedores de países desenvolvidos (Estados Unidos) em projetos internacionais de OSS?

RQ2: Estrutura das Interações Sociais e Colaborativas

Como se estruturam as interações sociais e colaborativas entre desenvolvedores de países emergentes e desenvolvidos em projetos OSS, e de que forma essas dinâmicas refletem padrões de influência, colaboração, reconhecimento e integração nas comunidades de software livre?

Hipóteses

Hipótese H1 (RQ1 - Desempenho)

Desenvolvedores de países emergentes (Índia) apresentam menor volume absoluto de contribuições, mas demonstram maior eficiência (produtividade por desenvolvedor) comparado aos desenvolvedores de países desenvolvidos (EUA), indicando diferentes padrões de contribuição: quantidade versus produtividade relativa.

Premissa: Embora numericamente inferiores, desenvolvedores indianos compensam com maior produtividade individual, sugerindo otimização de recursos e maior dedicação por contribuinte.

Hipótese H2 (RQ2 - Interações Sociais)

As interações revelam uma estrutura dual no ecossistema OSS: existe meritocracia técnica (taxa de aceitação similar de PRs) mas persistem desigualdades de capital social, com desenvolvedores emergentes sendo tecnicamente integrados porém socialmente periféricos, não alcançando os mesmos níveis de influência, visibilidade e reconhecimento.

Premissa: O OSS é simultaneamente aberto tecnicamente e fechado socialmente, criando barreiras estruturais relacionadas a redes, língua, fusos horários e capital social pré-existente.

Análises e Resultados

Dados Gerais da Amostra

- 135 repositórios de projetos internacionais OSS
- 7.427 desenvolvedores totais
- Estados Unidos: 5.569 desenvolvedores (75%)
- Índia: 1.858 desenvolvedores (25%)

Resultados para RQ1 - Análise de Desempenho

Volume Absoluto de Contribuições

Pull Requests Merged:

- Países Desenvolvidos (EUA): 160.324 PRs merged
- Países Emergentes (Índia): 52.357 PRs merged
- Ratio: 3:1 em favor dos EUA

Distribuição proporcional consistente (75/25):

- PRs Merged: 75% EUA, 25% Índia
- Reviews: 75% EUA, 25% Índia
- Stars recebidas: 75% EUA, 25% Índia
- Commits: 75% EUA, 25% Índia

Descoberta Chave - Eficiência Superior da Índia

Métricas de Eficiência (PRs por colaborador):

- Desenvolvedores Emergentes (Índia): 467,5 PRs/colaborador
- Desenvolvedores Desenvolvidos (EUA): 177,2 PRs/colaborador
- Vantagem da Índia: 2,6 vezes mais eficiente

Interpretação RQ1: Os dados revelam um padrão claro: desenvolvedores indianos são significativamente mais eficientes que seus pares americanos. Embora o volume absoluto seja três vezes menor, a produtividade por desenvolvedor é 2,6 vezes superior, indicando otimização de recursos e maior dedicação individual.

Resultados para RQ2 - Análise de Interações Sociais

Reconhecimento da Comunidade

- Diferença de apenas 21% em favor dos países desenvolvidos.
- Reconhecimento muito mais equilibrado que o volume de produção.
- A comunidade OSS reconhece valor nas contribuições indianas de forma relativamente proporcional.

Padrões de Engajamento

- Stars Count (reconhecimento passivo): favorece países desenvolvidos.
- Fork Ratio (engajamento ativo): quase igual entre países.
- Implicação: desenvolvedores emergentes inspiram ação na mesma proporção, indicando influência técnica real.

Integração na Comunidade

- Taxa de aceitação de PRs similar nos dois grupos (70–75%).
- Não há discriminação evidente no processo de merge.
- A principal barreira está em abrir PRs, não em tê-las aceitas.

Análise Espacial da Comunidade (Scatter Plot)

- Desenvolvedores dos EUA ocupam todo o espectro, incluindo quadrantes superiores de alto reconhecimento.
- Desenvolvedores da Índia concentram-se na nuvem central, com poucos outliers.

Medianas Revelam Disparidade:

- Activity Score: EUA superior.
- Stars: EUA superior.

Conclusão Visual: Existe segregação espacial onde desenvolvedores americanos ocupam posições de maior visibilidade, enquanto indianos, mesmo eficientes, permanecem em zonas de menor capital social.

Teto de Vidro Social Identificado: Alta eficiência não se traduz automaticamente em alto reconhecimento para desenvolvedores emergentes.

Conclusões

H1 (RQ1 - Desempenho): Comprovada Integralmente

Evidências Confirmatórias:

- Menor volume absoluto confirmado (3:1 ratio).
- Maior eficiência comprovada (2,6x superior).
- Diferentes padrões de contribuição: quantidade (EUA) vs produtividade relativa (Índia).

Conclusão H1: Hipótese completamente validada. Desenvolvedores indianos compensam a menor quantidade absoluta com produtividade individual extraordinariamente superior.

H2 (RQ2 - Interações Sociais): Comprovada com Nuances

Evidências da Estrutura Dual:

Meritocracia Técnica Confirmada:

- Taxa de aceitação de PRs similar (70–75%).
- Fork ratio equilibrado.
- Reconhecimento técnico proporcional.

Desigualdade de Capital Social Confirmada:

- Segregação espacial na comunidade visualizada.
- Concentração em projetos de alto impacto limitada.
- ”Teto de vidro” social identificado.
- Menor reconhecimento médio apesar da eficiência.

Conclusão H2: Hipótese validada com precisão. O ecossistema OSS demonstra ser simultaneamente aberto tecnicamente e fechado socialmente.

References

Barbosa, A., Santana, C. R. B., and Baltes, S. (2022). The geography of open source software: Evidence from GitHub. *Technological Forecasting and Social Change*, 176:121478.

HackerRank (2020). 2020 developer skills report. Technical report, HackerRank.