RESOLUÇÃO INCREMENTAL DE ENTIDADES NA PLATAFORMA LATTES: INTEGRAÇÃO DE NOVOS LOTES SEM REPROCESSAR A BASE

Daniel Oliveira Paixão[1] - engsisdaniel.profissional@gmail.com
Henrique Sena Silva[1] - uni.hsilva@gmail.com
Luiz Fayme Soares Pereira[1] - luiz.fayme.adm@gmail.com
Renê Rodrigues Veloso[1] - rene.veloso@unimontes.br
Jairo Francisco de Souza[2] - jairo.souza@ufjf.br
[1]Universidade Estadual de Montes Claros (UNIMONTES), Montes Claros, MG, Brasil
[2]Universidade Federal de Juiz de Fora (UFJF), Juiz de Fora, MG, Brasil

Resumo. *A Plataforma Lattes é uma das principais fontes de informação sobre a produção científica nacional, e a inserção descentralizada dos dados faz com que uma mesma entidade apareça em múltiplos registros. Deduplicar essa base tem custo elevado, e reexecutá-la a cada atualização é inviável, pois o esforço se repete sobre todo o conjunto acumulado. Apresentamos um algoritmo de resolução incremental de entidades que integra um novo lote a uma base já deduplicada sem reprocessá-la: o lote é primeiro deduplicado isoladamente e, em seguida, cada cluster resultante é confrontado com a base por consulta a um índice de vizinhos mais próximos sobre os embeddings dos representantes. Sobre 1.000.000 de projetos de pesquisa do Lattes, em cinco lotes de 200.000 registros, o tempo da integração incremental variou 18%, ao passo que o do reprocessamento integral cresceu de 7.257,34 s a 30.800,37 s; somadas as etapas, foi 5,81 vezes mais rápida, com acurácia de decisão entre 0,9926 e 0,9686. O custo de atualização passa a ser função do tamanho do lote, e não da base.*

Palavras-chave: *Deduplicação. Embeddings. Clusterização. Escalabilidade.*

1. Introdução

A Plataforma Lattes, em específico o Currículo Lattes, registra a vida acadêmica dos pesquisadores brasileiros e é uma das principais fontes de informação sobre a produção científica nacional. O currículo é exigido nos pedidos de fomento à pesquisa no país (CNPQ, 2026), e seus dados alimentam a avaliação de programas de pós-graduação (Silva et al., 2017) e os estudos cienciométricos (Mena-Chalco & CESAR JUNIOR, 2009). Como essas análises pressupõem registros consistentes, a qualidade da base condiciona sua confiabilidade.

A duplicação de registros decorre, em grande medida, da forma como os próprios usuários inserem seus dados (de Souza & Souza, 2017): um mesmo produto é informado de maneiras

distintas pelos pesquisadores envolvidos, com variações na grafia de nomes e na formatação de títulos. Deduplicar a base inteira, porém, tem custo elevado. Como ilustrado na avaliação da Seção 4, o reprocessamento de um milhão de registros demandou 8,56 h. Como a base está em constante atualização, reexecutar a deduplicação a cada lote é inviável: o esforço se repete sobre os registros já consolidados.

Este artigo apresenta um algoritmo que integra um novo conjunto de registros a uma base já deduplicada sem reprocessá-la. O problema não é trivial, pois um registro novo pode duplicar tanto um registro da base quanto outro do próprio conjunto de entrada. A abordagem se insere na resolução incremental de entidades (Christophides et al., 2020), em que o custo de atualização deve ser proporcional ao volume de novos dados, e não ao tamanho da base (Araújo et al., 2022).

2. Formulação do Problema

Esta seção fixa a notação e as premissas necessárias. A abordagem adotada não substitui o algoritmo de deduplicação: opera sobre o que ele produz, os *clusters* da base e os *embeddings* de seus representantes.

2.1 Notação

Seja $\mathcal{R}$ o universo de registros e $B \subset \mathcal{R}$ a base consolidada. Cada registro r possui k campos-chave textuais $c_1(r), \ldots, c_k(r)$, e um modelo de *embeddings* ε mapeia cada campo em um vetor de dimensão d, de modo que $E(r) = \big(\varepsilon(c_1(r)), \ldots, \varepsilon(c_k(r))\big)$. A proximidade entre dois registros é a similaridade do cosseno $\mathrm{sim}_i(r, r') = \cos\big(\varepsilon(c_i(r)),\, \varepsilon(c_i(r'))\big)$ em cada campo, agregada por uma função σ, de modo que $S(r, r') = \sigma\big(\mathrm{sim}_1(r, r'), \ldots, \mathrm{sim}_k(r, r')\big)$. Mantemos σ genérica, pois a abordagem não depende de sua forma particular; um critério válido é exigir $\mathrm{sim}_i(r, r') \geq \tau_i$ para todo i. Denota-se por $\mathrm{match}(r, r')$ o predicado que indica se r e r' são correspondentes. Uma partição $\mathcal{C}$ é dita uma clusterização por entidade quando cada C_j agrupa exatamente os registros de uma mesma entidade do mundo real, e $\rho(C_j) \in C_j$ denota seu representante. Por fim, $\Delta \subset \mathcal{R}$ é o lote a incorporar, tratado como conjunto e não como sequência de inserções.

2.2 Premissas

P1. Deduplicação por componentes conexos. Parte-se de um algoritmo D que devolve uma clusterização por entidade, com critério de correspondência baseado nas similaridades sim_i agregadas por σ, e que construa os *clusters* como componentes conexos do grafo não direcionado $G(X)$, cujas arestas ligam os pares de X que satisfazem match: se $\mathrm{match}(r_1, r_2)$ e $\mathrm{match}(r_2, r_3)$ valem, os três registros pertencem ao mesmo *cluster*, ainda que $\mathrm{match}(r_1, r_3)$ seja falso. Toma-se D como o mesmo algoritmo, com idênticos parâmetros, na base e nos lotes, para que o resultado seja comparável ao do reprocessamento completo.

P2. Representantes e *embeddings* materializados. A saída de D sobre a base é $\mathcal{C}_B = D(B)$. Pressupõe-se uma função ρ que elege um representante por *cluster* e que os *embeddings* $E(\rho(C))$ estejam materializados. O representante é a interface pela qual um *cluster* é comparado a novos registros, o que reduz as comparações potenciais de $|B|$ para $|\mathcal{C}_B|$. Decorre de P1 que um registro do lote pode corresponder a um membro não representante sem o ser ao representante; restringir a comparação a este é uma aproximação deliberada, cuja divergência a Seção 4 quantifica.

P3. Recuperação de candidatos. Supõe-se disponível uma função NN (do inglês *nearest neighbors*) que, dada $E(r)$, devolve um conjunto reduzido de representantes candidatos, por um índice com custo de consulta sublinear no tamanho da base, como *HNSW* (Malkov & Yashunin, 2016) ou *IVF* (Johnson et al., 2017). Como σ é arbitrária, o critério de correspondência não é, em geral, redutível à proximidade em um único espaço vetorial. A consulta se estrutura, por isso, em duas etapas: o índice recupera candidatos plausíveis pela proximidade em um campo indexado, e match, que incorpora σ sobre os k campos, é avaliado exaustivamente apenas sobre esse conjunto.

P4. Inserção com atualização do representante. Requer-se, por fim, uma função Insert que incorpore um registro a um *cluster* e reavalie o representante segundo ρ. Quando este muda, os *embeddings* materializados e o índice de P3 são atualizados.

2.3 Definição do problema

Dada a base B com sua clusterização $\mathcal{C}_B$ e um lote Δ, o problema consiste em obter, com custo em função do tamanho do lote, a clusterização $\mathcal{C}_{B\cup\Delta}$, sem avaliar $D(B \cup \Delta)$. Ele não se reduz a consultar cada registro de Δ contra $\mathcal{C}_B$ isoladamente: registros do lote podem corresponder a *clusters* já existentes, mas também entre si, formando entidades sem contrapartida na base.

3. Abordagem Proposta

3.1 Estratégia de processamento do lote

O lote é primeiro deduplicado isoladamente, aplicando-se D sobre Δ para obter $\mathcal{C}_\Delta = D(\Delta)$, o que resolve as correspondências internas antes de qualquer interação com a base. Somente então cada *cluster* de $\mathcal{C}_\Delta$ é confrontado com $\mathcal{C}_B$, por uma consulta ao índice a partir dos *embeddings* de seu representante, de modo que o número de consultas é proporcional a $|\mathcal{C}_\Delta|$, e não a $|\Delta|$.

3.2 Consolidação de múltiplas correspondências

Sejam C_a e C_b *clusters* distintos de $\mathcal{C}_B$, com representantes a e b, e seja r o representante de $C_r \in \mathcal{C}_\Delta$ tal que $\mathrm{match}(a, r)$ e $\mathrm{match}(r, b)$ valham. Serem distintos significa que não há caminho entre a e b em $G(B)$; contudo, em $G(B \cup \Delta)$, as arestas (a, r) e (r, b) estabelecem o caminho $a - r - b$, e o reprocessamento integral agruparia os três em um único *cluster*. Portanto, quando mais de um representante da base satisfaz match com r, todos os *clusters* associados devem ser consolidados com o do lote. Como a consolidação pode alterar o representante do *cluster* resultante (P4), e este pode satisfazer match com *clusters* que o anterior não alcançava, o procedimento deve ser reiterado até que nenhuma nova correspondência seja detectada.

3.3 Formalização dos algoritmos

O Algoritmo 1 (IntegraLote) organiza o procedimento em três etapas: a deduplicação interna do lote (linha 1), a recuperação dos candidatos de cada *cluster* pelo índice (linhas 2 a 4, P3) e a resolução (linhas 5 a 14), em que match é avaliado sobre esses candidatos para obter M, o conjunto de *clusters* da base que correspondem a r. Quando M é vazio, as linhas 9 e 10

incorporam C_r como novo *cluster*, atualizando o índice (P4); caso contrário, a linha 12 delega a fusão a CONSOLIDA.

No Algoritmo 2, os registros de cada $C' \in M$ são incorporados a C por Insert e C' é removido de $\mathcal{C}_B$ e do índice. Como Insert pode alterar o representante (P4), a entrada do novo é inserida e a consulta repetida a partir dele, enquanto houver novas correspondências.

Algoritmo 1 INTEGRALOTE($\mathcal{C}_B$, Δ)

Entrada: clusterização $\mathcal{C}_B$ da base consolidada, com representantes $\rho(C)$ e *embeddings* $E(\rho(C))$ materializados para todo $C \in \mathcal{C}_B$, e índice NN construído sobre $\{E(\rho(C)) : C \in \mathcal{C}_B\}$ (P2, P3); lote Δ de novos registros
Saída: clusterização atualizada $\mathcal{C}_{B \cup \Delta}$

1: $\mathcal{C}_\Delta \leftarrow D(\Delta)$ ▷ resolve as correspondências internas ao lote (P1) antes de tocar a base
2: **for** each $C_r \in \mathcal{C}_\Delta$ **do**
3: $\quad Cands[C_r] \leftarrow \mathrm{NN}(E(\rho(C_r)))$ ▷ recuperação sobre os representantes de $\mathcal{C}_B$ (P3)
4: **end for**
5: **for** each $C_r \in \mathcal{C}_\Delta$ **do**
6: $\quad r \leftarrow \rho(C_r)$
7: $\quad M \leftarrow \{\, C \in \mathcal{C}_B : \rho(C) \in Cands[C_r]$ e $\mathrm{match}(\rho(C), r) \,\}$
8: $\quad$ **if** $M = \emptyset$ **then**
9: $\quad\quad \mathcal{C}_B \leftarrow \mathcal{C}_B \cup \{C_r\}$
10: $\quad\quad$ ATUALIZAÍNDICE(insere $E(\rho(C_r))$)
11: $\quad$ **else**
12: $\quad\quad \mathcal{C}_B \leftarrow$ CONSOLIDA($\mathcal{C}_B, C_r, M$)
13: $\quad$ **end if**
14: **end for**
15: **return** $\mathcal{C}_B$

Algoritmo 2 CONSOLIDA($\mathcal{C}_B$, C, M)

Entrada: clusterização corrente $\mathcal{C}_B$; cluster C a incorporar; conjunto não vazio $M \subset \mathcal{C}_B$ de clusters correspondentes a $\rho(C)$
Saída: $\mathcal{C}_B$ com C e os clusters de M fundidos em um único cluster

1: **while** $M \neq \emptyset$ **do**
2: $\quad$ **for** each $C' \in M$ **do**
3: $\quad\quad$ **for** each $x \in C'$ **do**
4: $\quad\quad\quad C \leftarrow \mathrm{Insert}(C, x)$ ▷ P4: incorpora registro a registro, reavaliando ρ
5: $\quad\quad$ **end for**
6: $\quad\quad \mathcal{C}_B \leftarrow \mathcal{C}_B \setminus \{C'\}$
7: $\quad\quad$ ATUALIZAÍNDICE(remove $E(\rho(C'))$)
8: $\quad$ **end for**
9: $\quad$ ATUALIZAÍNDICE(insere $E(\rho(C))$)
10: $\quad Cands \leftarrow \mathrm{NN}(E(\rho(C)))$
11: $\quad M \leftarrow \{\, C' \in \mathcal{C}_B \setminus \{C\} : \rho(C') \in Cands$ e $\mathrm{match}(\rho(C'), \rho(C)) \,\}$
12: **end while**
13: **return** $\mathcal{C}_B \cup \{C\}$

4. Resultados

4.1 Base de avaliação e protocolo de medição

O conjunto de avaliação é composto por 1.000.000 de projetos de pesquisa extraídos da base real do Lattes, dividido em cinco partições de 200.000 registros. O algoritmo D opera sobre dois campos-chave, título e integrantes do projeto, cujos *embeddings* são produzidos pelo modelo *all-MiniLM-L6-v2* (Reimers & Gurevych, 2019), de 384 dimensões. A função σ é a conjunção de dois limiares independentes: $\mathrm{sim}_{\text{título}} \geq 0{,}935$ e $\mathrm{sim}_{\text{integrantes}} \geq 0{,}700$. A recuperação devolve

os 50 vizinhos mais próximos, e o índice é um *HNSW* (Malkov & Yashunin, 2016) com $M = 32$ e $efSearch = 128$, provido pelo *FAISS* (Johnson et al., 2017).

Para medir a eficácia, toma-se como referência a clusterização que o reprocessamento integral produziria, isto é, a aplicação direta de D sobre $B \cup \Delta$. Como o Algoritmo 1 toma exatamente uma decisão por *cluster* de $\mathcal{C}_\Delta$, incorporá-lo como novo ou fundi-lo a um existente, adota-se como métrica a acurácia de decisão por *cluster* do lote. São acertos a *nova correta* (não havia contrapartida, e o *cluster* entrou como novo) e a *fusão exata* (o resultado coincide com o da referência); são erros a *fusão incompleta* (fusão com parte do conjunto correto), a *fusão perdida* (havia contrapartida e o *cluster* entrou como novo) e a *fusão indevida* (uniram-se *clusters* que a referência manteve separados).

4.2 Custo e acurácia de decisão

Encontram-se na Tabela 1, por etapa, o tempo de cada procedimento e a acurácia de decisão.

Tabela 1: Integração incremental contra reprocessamento integral, por etapa. Lotes de 200.000 registros; *acum.* é o total da base após a integração. Na última linha, os tempos são somas e a acurácia é a média das etapas.

		tempo (s)			concordância
k	acum.	integral	increm.	razão	decisão
2	400.000	7.257,34	2.905,75	2,50	0,9926
3	600.000	13.202,90	2.958,71	4,46	0,9856
4	800.000	21.379,86	3.214,77	6,65	0,9772
5	1.000.000	30.800,37	3.432,34	8,97	0,9686
total / média		72.640,48	12.511,57	5,81	0,9810

O comportamento confirma a propriedade pretendida: o tempo do reprocessamento integral cresce de 7.257,34 s para 30.800,37 s à medida que a base passa de 400.000 para 1.000.000 de registros, ao passo que o da integração incremental varia 18%, de 2.905,75 s a 3.432,34 s, para lotes constantes. A razão cresce de 2,50 na etapa 2 para 8,97 na etapa 5, e o tempo total das quatro etapas é 5,81 vezes menor.

A acurácia de decisão, entre 0,9926 e 0,9686, mostra que o algoritmo decide corretamente sobre cerca de 98% dos *clusters* do lote, com queda monotônica ao longo das etapas. A divergência é fortemente assimétrica: as fusões indevidas somam 9 ocorrências em 752.266 decisões nas quatro etapas, três ordens de grandeza abaixo das fusões incompletas e das perdidas. Da etapa 2 à etapa 5, a fusão incompleta multiplica-se por 6,9 e a perdida por 3,1, diferença consistente com a natureza de cada aproximação: a fusão incompleta decorre de P2 e, como cada fusão aumenta o tamanho médio dos *clusters*, cresce a cada etapa a chance de a correspondência verdadeira residir em um membro não representante; já a fusão perdida decorre de P3 e depende da cobertura da recuperação de candidatos, que se degrada mais lentamente.

5. Conclusão

Este trabalho apresentou um algoritmo para a integração incremental de novos lotes a uma base já deduplicada, sem reprocessar o conjunto acumulado. A avaliação sobre 1.000.000 de

projetos de pesquisa da Plataforma Lattes mostrou que o tempo da integração incremental cresceu 18%, ao passo que o do reprocessamento integral cresceu de 7.257,34 s a 30.800,37 s. Somadas as quatro etapas, a integração foi 5,81 vezes mais rápida. A acurácia de decisão manteve-se entre 0,9926 e 0,9686, com divergência fortemente assimétrica: o procedimento deixa de realizar parte das fusões que o reprocessamento realizaria, e raramente produz uma fusão falsa.

Como limitação, as premissas P2 e P3 constituem aproximações, ao restringirem a comparação ao representante e a recuperação aos candidatos do índice aproximado; juntas reduzem o custo, mas são a origem dos erros observados. Como trabalhos futuros, propomos relaxar essas aproximações de forma controlada: quanto a P2, materializar os *embeddings* de até m registros por *cluster* mantém o índice limitado a $m|\mathcal{C}_B|$ entradas, preservando a redução frente a $|B|$ a menos de um fator constante e permitindo caracterizar o compromisso entre custo e acurácia em função de m; quanto a P3, o mesmo estudo pode variar a cobertura da recuperação de candidatos.

Agradecimentos

Os autores agradecem à FAPEMIG (Fundação de Amparo à Pesquisa do Estado de Minas Gerais) pelo apoio (processos APQ-04805-24 e APQ-02313-25).

Declaração de uso de IA

A ferramenta de inteligência artificial generativa *Claude* (Anthropic) foi utilizada no apoio à redação e à revisão do texto. Todo o conteúdo científico e as conclusões são de responsabilidade integral dos autores.

REFERÊNCIAS

T. B. Araújo, K. Stefanidis, C. E. S. Pires, J. Nummenmaa, e T. P. d. Nóbrega. Incremental entity blocking over heterogeneous streaming data. *Information*, 13(12):568, 2022. doi: 10.3390/info13120568.

V. Christophides, V. Efthymiou, T. Palpanas, G. Papadakis, e K. Stefanidis. An overview of end-to-end entity resolution for big data. *ACM Computing Surveys*, 53(6):1–42, 2020. doi: 10.1145/3418896.

CNPQ. Plataforma lattes. https://www.gov.br/cnpq/pt-br/acesso-a-informacao/acoes-e-programas/plataforma-lattes, 2026. Conselho Nacional de Desenvolvimento Científico e Tecnológico. Acesso em: 12 jul. 2026.

M. de Souza e R. R. Souza. Fluxo informacional para redução de dados duplicados na Plataforma Lattes. In *Anais do XVIII Encontro Nacional de Pesquisa em Ciência da Informação (ENANCIB)*, Marília, SP, Brasil, 2017.

J. Johnson, M. Douze, e H. Jégou. Billion-scale similarity search with GPUs. arXiv:1702.08734, 2017.

Y. A. Malkov e D. A. Yashunin. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. arXiv:1603.09320, 2016. Publicado posteriormente em IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018.

J. P. Mena-Chalco e R. M. CESAR JUNIOR. scriptLattes: an open-source knowledge extraction system from the Lattes platform. *Journal of the Brazilian Computer Society*, 15(4):31–39, 2009. doi: 10.1007/BF03194511.

N. Reimers e I. Gurevych. Sentence-BERT: sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019. doi: 10.18653/v1/D19-1410.

T. H. P. Silva, A. H. F. Laender, C. A. DAVIS JR., A. P. C. d. Silva, e M. M. Moro. A profile analysis of the top Brazilian Computer Science graduate programs. *Scientometrics*, 113(1):237–255, 2017. doi: 10.1007/s11192-017-2462-3.