Zero-Shot Product Title Normalization with LLMs: A Multi-Model Evaluation

Humberto Nunes de Lira[1], **Rômulo Henrique Nascimento Duarte**[1], **Beatriz Almeida de Souza Silva**[2], **Brenno Henrique Alves da Silva Costa**[1], **Lucas Rabay Butcher**[1], **Fernando d'Ávila Lins Bezerra Cavalcanti Filho**[1], **Adriano Dias Jerônimo Filho**[1]

[1] Universidade Federal da Paraíba (UFPB)
João Pessoa – PB – Brasil

[2]Centro Universitário de João Pessoa (UNIPÊ)
João Pessoa – PB – Brasil

humberto.nunes@academico.ufpb.br, romulohenri000@gmail.com

beatrizalmeidabss@gmail.com, brennohdev@gmail.com

lucasrabaybutcher@gmail.com, fernando.d.avila.filho@gmail.com

adrianojeronimofilho@hotmail.com

Abstract. *Product title normalization is a critical data quality challenge in large-scale e-commerce catalogs, where independent sellers introduce severe inconsistencies that hinder deduplication and entity matching. We present TAIL (Title Attribute Inference Layer), a zero-shot LLM pipeline for attribute extraction, evaluated across three cost-efficient LLMs on a dataset of 1,200 pairs from a Brazilian e-commerce catalog. Pairwise matching performance depends strongly on the extraction model, ranging from 0.44 to 0.78 F1, with only the best model outperforming the TF-IDF baselines (+18.7 F1 points, 95% CI [+1.1, +37.3]), while normalization constancy varies more modestly. Model choice is as critical to deployment as the pipeline design.*

Resumo. *A normalização de títulos de produtos é um desafio crítico de qualidade de dados em catálogos de e-commerce, onde vendedores independentes introduzem inconsistências severas que dificultam deduplicação e entity matching. Apresentamos o TAIL (Title Attribute Inference Layer), pipeline zero-shot baseado em LLM para extração de atributos, avaliado sobre três LLMs em um dataset de 1.200 pares anotados de um catálogo brasileiro de e-commerce. O desempenho de matching par-a-par depende do modelo de extração, variando de 0,44 a 0,78 F1, com apenas o melhor superando os baselines TF-IDF (+18,7 pontos de F1, IC 95% [+1,1; +37,3]), enquanto a constância varia de forma mais modesta. A escolha do modelo é tão crítica quanto o desenho do pipeline.*

1. Introdução

A normalização de títulos de produtos em plataformas de e-commerce representa um desafio fundamental de qualidade de dados. Em marketplaces de grande escala, catálogos são alimentados por milhares de vendedores independentes sem padronização, de modo

que um único produto pode aparecer sob dezenas de variações superficiais, como *Coca-Cola 2L*, *Coca Cola 2 litros* ou *Cola Coca 2l*, sem compartilhar nenhum identificador comum além de sua descrição textual [More 2016]. Campos estruturados como GTIN e NCM, embora presentes, são preenchidos manualmente e mostram-se pouco confiáveis no contexto estudado, um *dump* real do catálogo brasileiro do Mercado Livre (MeLi) com 20 milhões de títulos bilíngues, o que motiva o foco na normalização baseada exclusivamente no título. Em nosso conjunto anotado, apenas 12,9% dos pares sinalizados como candidatos a *match* por similaridade textual ($\tau \geq 0{,}70$) correspondem, de fato, ao mesmo produto, o que evidencia a insuficiência da similaridade de string como critério isolado de decisão.

Apresentamos o TAIL (*Title Attribute Inference Layer*), um pipeline *zero-shot* que extrai atributos estruturados de títulos em texto livre (marca, modelo, tipo de produto, capacidade, voltagem, cor, material e quantidade) para produzir um identificador canônico estável, robusto a variações superficiais. Trabalhos recentes demonstraram que LLMs como GPT-3.5 e GPT-4 alcançam resultados expressivos em extração e normalização de atributos de produtos, superando modelos de linguagem pré-treinados menores em até 10% de F1 [Brinkmann et al. 2024]. Permanece pouco explorado, no entanto, até que ponto esse ganho depende do modelo específico utilizado, em vez de ser uma propriedade do pipeline de extração em si. Este trabalho avalia a portabilidade do pipeline proposto, com pré-normalização simbólica inspirada em [Li et al. 2020] e *batch prompting* [Fan et al. 2024], buscando responder: *o desempenho do pipeline de normalização é uma propriedade do desenho da solução, ou depende fortemente do modelo de extração escolhido?* Avaliamos três modelos da categoria econômica de seus respectivos laboratórios, Gemini 3.1 Flash-Lite (Google), GPT-5.4-mini (OpenAI) e DeepSeek V4 Flash (DeepSeek, em modo não-pensante), segundo três medidas detalhadas na Seção de Metodologia: constância por grupo, F1 par-a-par com *blocking* e estabilidade entre execuções repetidas.

Este trabalho situa-se no contexto de sistemas multiagentes aplicados a dados: o TAIL é concebido como componente autônomo de normalização, integrável em arquiteturas onde agentes especializados cooperam para manter a qualidade de catálogos. A avaliação multi-modelo informa uma decisão de design crítica para tais sistemas, qual modelo de extração delegar ao agente normalizador, e mostra que a variação de 34 pontos percentuais de F1 entre modelos da mesma faixa de custo torna essa seleção um ato de coordenação de sistema, não uma escolha de implementação interna ao agente (Seção 4.4). A resposta à pergunta de pesquisa é, portanto, negativa: a portabilidade do pipeline entre provedores não é garantida. As contribuições deste trabalho são: (i) um pipeline *zero-shot* de normalização de títulos que combina pré-normalização simbólica, extração via LLM e limpeza determinística, avaliado sobre catálogo real de e-commerce brasileiro; (ii) uma avaliação da portabilidade desse componente entre três modelos de extração sob o mesmo esquema de campos e conjunto de avaliação, informando decisões de design de sistemas multiagentes que delegam extração a LLMs heterogêneos, já que a escolha do modelo interno ao agente altera o F1 do sistema em 34,0 pontos percentuais; (iii) uma caracterização do não-determinismo de provedores de LLM sob temperatura zero, dimensão de instabilidade distinta da acurácia e relevante para agentes que produzem chaves persistentes; e (iv) um conjunto de 1.200 pares de títulos anotados manualmente, com o protocolo de amostragem que o produziu, publicamente disponíveis.

2. Related Work

2.1. Normalização e Extração de Atributos de Títulos de Produtos

A normalização de títulos é um passo crítico para a qualidade dos dados em e-commerce, especialmente quando as descrições são geradas por múltiplos vendedores sem padronização. O problema mais amplo de identificar registros que se referem à mesma entidade do mundo real, do qual a normalização de títulos é um caso particular, é estudado há décadas sob os nomes de *entity matching* e *record linkage*. More [More 2016] caracteriza o problema da extração de atributos em títulos curtos e ruidosos, destacando que o título é muitas vezes a única fonte confiável de informação estruturada em catálogos de larga escala. Gazzola et al. [Gazzola et al. 2025], no contexto da SBC/STIL, introduzem o AI-PAVE-Br, um sistema especializado para o cenário brasileiro que utiliza LLMs para extração de atributos a partir de título e descrição e disponibiliza um *Golden Set* anotado em português. Além da contextualização na literatura, realizamos uma comparação experimental sobre os 7.890 itens públicos desse conjunto, recomputando AI-PAVE-Br, o *baseline* da empresa terceira e o TAIL sob a mesma implementação de métrica; o protocolo e os resultados estão na Seção 4.2.1.

2.2. Uso de Large Language Models (LLMs) em E-commerce

A aplicação de LLMs transformou a extração de atributos de uma tarefa de classificação ou *tagging* para uma tarefa generativa. Brinkmann et al. [Brinkmann et al. 2024] demonstram a superioridade de modelos como GPT-3.5 e GPT-4 na normalização de atributos frente a modelos menores, levantando também questões sobre custos de API. Neres de Sousa et al. [Neres de Sousa et al. 2025], no contexto da SBC/SBBD, apresentam um agente autônomo guiado por LLM que extrai informações de e-commerce de forma dinâmica, exemplo da tendência nacional de automatizar extração de dados em domínios comerciais. No âmbito de sistemas multiagentes, a decomposição de tarefas de qualidade de dados em agentes especializados de limpeza, resolução de entidades e enriquecimento, coordenados por orquestração centralizada ou protocolos de comunicação, é um padrão recorrente [Neres de Sousa et al. 2025]. O TAIL, ao encapsular extração e normalização em uma interface determinística (título $\rightarrow$ nome canônico), atende ao requisito de modularidade esperado de um agente em tal arquitetura.

Zhang et al. [Zhang et al. 2025] conduzem uma avaliação sistemática de métodos baseados em modelos de linguagem para *cross-dataset entity matching*, sobre oito abordagens e 11 datasets de benchmark, e reportam variação de aproximadamente 21 pontos de F1 médio entre modelos comerciais avaliados sob o mesmo formato de *prompt*. Nossos resultados estendem essa observação em duas direções. Primeiro, a variação que observamos ocorre entre modelos da mesma faixa de custo, sob *prompt*, esquema de campos e conjunto de avaliação idênticos, isolando o modelo de extração como variável manipulada. Segundo, enquanto aqueles autores quantificam a variância decorrente da perturbação deliberada da entrada, variando a ordem de serialização dos atributos entre execuções, observamos divergência entre execuções com entrada estritamente idêntica e temperatura zero (Seção 4.3), o que caracteriza não-determinismo do provedor e não sensibilidade à serialização. A lacuna que este trabalho busca preencher é específica: avaliar comparativamente múltiplos LLMs dentro do mesmo pipeline *zero-shot*, mantendo constantes os componentes do pipeline e variando apenas o modelo de extração.

3. Metodologia

Esta seção descreve a base de dados e o pipeline, incluindo os tratamentos e interações com as LLMs.

3.1. Base de dados

A base de dados utilizada é um *dump* real de catálogo de e-commerce do Mercado Livre (MeLi), contendo aproximadamente 20 milhões de títulos de produtos em português e espanhol. Como discutido na Introdução, o título é a única coluna do catálogo consistentemente preenchida e semanticamente significativa, o que motiva seu uso isolado como fonte de sinal para a extração de atributos. Para os experimentos reportados na Seção de Resultados, filtramos o subconjunto em português e amostramos um *pool* de 30.000 títulos (semente fixa, `random_state=42`), a partir do qual construímos o dataset de validação rotulado descrito a seguir.

3.1.1. Construção da amostra rotulada de validação

Para dimensionar a amostra, aplicamos $n = Z^2 p(1-p)/E^2$ com $Z = 1{,}96$, $p = 0{,}5$ e $E = 0{,}05$, obtendo $n \approx 385$; convertido em amostra total pela taxa de positivos preditos no piloto ($r \approx 0{,}34$), via $N = n/r$, isso define um dataset de aproximadamente 1.200 pares. Esse dimensionamento sustenta a representatividade do conjunto quanto à proporção geral entre pares equivalentes e não equivalentes, mas não a precisão de métricas condicionadas na classe positiva: a anotação resultou em 30 positivos entre os 1.200, taxa de 2,5% bem inferior à do piloto, de modo que a incerteza de *recall* e F1 excede a margem de 5% e é reportada por intervalos de confiança em todas as tabelas.

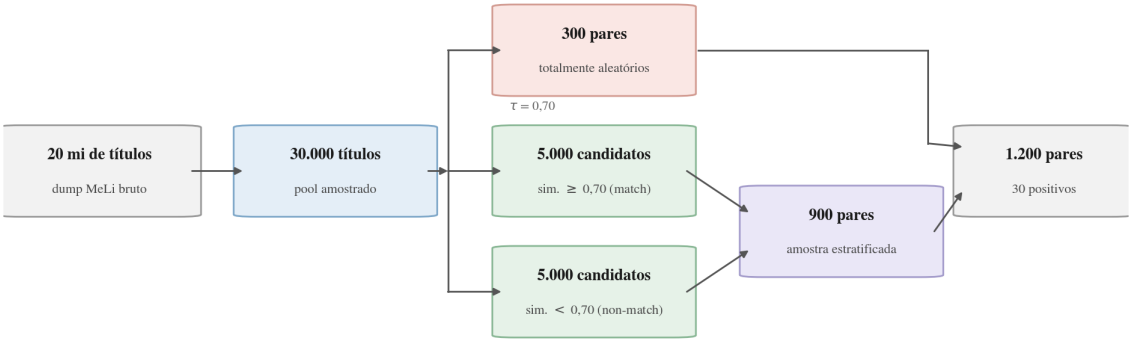

Figura 1. Pipeline de construção do dataset de 1.200 pares rotulados, combinando amostragem aleatória e estratificada por similaridade textual a partir de um pool de 30.000 títulos.

Uma amostragem puramente aleatória resultaria em um conjunto quase inteiramente trivial, já que a maioria dos pares de títulos distintos é trivialmente diferente. Combinamos então duas estratégias sobre o *pool*, resumidas na Figura 1: 300 pares totalmente aleatórios, como referência não enviesada da distribuição natural de similaridade, e 900 pares extraídos por amostragem estratificada de dois conjuntos de 5.000 candidatos separados pelo limiar de similaridade de string (`difflib.SequenceMatcher`, $\tau = 0{,}70$), priorizando pares da mesma categoria quando essa informação está disponível.

A estratificação preserva a proporção observada entre os dois conjuntos, em vez de impor um balanceamento artificial de 50/50 que não refletiria a distribuição real do problema. Entre os pares selecionados, 224 são candidatos a *match* (similaridade média de 0,77) e 676 a *non-match* (0,36). Os 300 pares aleatórios não contêm nenhum positivo e o estrato de candidatos a *match* concentra 29 dos 30, o que confirma empiricamente a necessidade da estratificação. A rotulagem foi conduzida pelos sete autores segundo critério comum definido previamente (*label* = 1 se os títulos descrevessem o mesmo SKU), com cada par anotado por um único anotador, casos ambíguos resolvidos por consulta ao grupo e independência em relação ao `candidate_label`, que é apenas critério de amostragem. Pares equivalentes não sorteados não integram o *ground truth*, de modo que as métricas reportadas são condicionadas ao conjunto de pares candidatos gerado.

3.2. Processamento pré-LLM

O pipeline aplica duas fases de processamento determinístico, antes e depois do envio à LLM. A pré-normalização usa expressões regulares para reduzir variações superficiais nos títulos antes de chamar o modelo, inspirada na normalização de *spans* do Ditto [Li et al. 2020]. As regras seguem um princípio conservador (preferível não normalizar a normalizar incorretamente), motivado por uma regressão em que a regra de quilogramas capturava erroneamente resoluções de tela (*"Smart TV 4K"* $\rightarrow$ *"Smart TV 4kg"*); por isso, unidades de massa exigem ao menos duas letras (`kg`, `kilo`, `grama`, `gr`), nunca a letra isolada, o que evita também colisões com designações como *5G* e *8GB*. As demais regras cobrem volume e fração (*"1/2 litro"*, *"0,5L"* $\rightarrow$ *"500ml"*), voltagem e potência (*"bi-volt"* $\rightarrow$ *"bivolt"*; *"110 Volts"* $\rightarrow$ *"110v"*; *"1000 W"* $\rightarrow$ *"1000w"*) e espaçamento excessivo; a etapa de pós-processamento (Seção 3.4) aplica transformações análogas sobre os campos já extraídos. Além das regras de reescrita, a etapa pré-LLM aplica um filtro de similaridade baseado em distância de edição ponderada por tipo de caractere, com custos assimétricos: substituições envolvendo dígitos custam duas ordens de grandeza mais que substituições entre letras (10,0 contra 0,1), e inserções ou remoções de dígito custam 8,0 contra 0,2 para letras. Essa ponderação codifica a premissa de que um dígito divergente indica tipicamente um produto distinto (*"1L"* contra *"2L"*), enquanto variações ortográficas e abreviações constituem ruído de superfície a ser resolvido pela etapa de extração.

3.3. Extração de atributos via LLM

A extração estruturada de atributos é realizada em modo *zero-shot*: o prompt enviado ao modelo contém apenas a definição da tarefa, o esquema de campos esperado (*marca*, *modelo*, *tipo_produto*, *capacidade_volume*, *voltagem*, *cor*, *material*, *quantidade* e *outros_atributos*) e um conjunto de regras textuais de normalização e desambiguação semântica, por exemplo, unificação de sinônimos de tipo de produto (*"lavadora"* $\rightarrow$ *"máquina de lavar"*) e de grafias alternativas de marca (*"Coca Cola"* $\rightarrow$ *"Coca-Cola"*), sem nenhum exemplo de entrada/saída incluído no prompt. Essa escolha segue a constatação de Brinkmann et al. [Brinkmann et al. 2023] de que regras de *matching* em linguagem natural no prompt já produzem ganhos expressivos de F1 com custo mínimo, dispensando a necessidade de exemplos para guiar o formato de saída. O prompt completo de sistema, o código do pipeline de inferência e avaliação e o dataset anotado de 1.200 pares estão disponíveis publicamente em `https://github.com/TailUFPB/invoice-data-normalizer`. O mesmo *prompt* de sistema, o mesmo

esquema de campos, as mesmas etapas determinísticas de pré e pós-processamento, `temperature=0` e o mesmo limite de *tokens* de saída são aplicados aos três modelos de extração. A única assimetria está na imposição de formato de saída: Gemini e DeepSeek executam com saída estruturada obrigatória (`response_mime_type` e `response_format`), enquanto no GPT-5.4-mini esse modo teve de ser desabilitado, pois sob formato de objeto obrigatório o modelo passou a preencher apenas o primeiro item do lote, produzindo extração vazia em cerca de 80% dos títulos; para ele, a conformidade é solicitada por instrução textual, com o mesmo *fallback* de *parsing*. Essa incompatibilidade é ela própria uma manifestação da falta de portabilidade investigada aqui: saída estruturada não é um recurso equivalente entre provedores. A comparação com o GPT-5.4-mini deve, portanto, ser lida como comparação entre integrações viáveis, e não como isolamento estrito do modelo; a comparação entre Gemini e DeepSeek ocorre sob configuração integralmente equivalente. Para reduzir o número de chamadas à API, os títulos são processados em *batch prompting* [Fan et al. 2024]: cada chamada recebe um lote de 5 títulos (`BATCH_SIZE=5`), numerados explicitamente no prompt, e o modelo retorna um único array JSON contendo um objeto por título, na mesma ordem recebida, com *fallback* de *parsing* por expressão regular em caso de falha.

3.4. Processamento pós-LLM

Após a extração estruturada via LLM, uma segunda camada de limpeza determinística, simétrica à pré-normalização, mas operando sobre os campos já extraídos (*marca*, *voltagem*, *capacidade_volume*, entre outros) em vez do título bruto, atua como *safety net* contra inconsistências de formatação introduzidas pelo próprio modelo. Essa etapa padroniza espaçamento e capitalização (*" samsung " → "Samsung"*), converte unidades de volume e massa para uma base canônica única, mililitros e gramas (*"536 litros" → "536000ml"*), e normaliza notações equivalentes de voltagem (*"Bi-Volt"*, *"220 Volts" → "bivolt"*, *"220v"*). A partir dos campos limpos, o pipeline produz o **nome canônico** (*display_name*), construído pela concatenação de tipo de produto, marca, cor e demais atributos (modelo, material, voltagem e outros), seguidos de quantidade e capacidade. É essa representação que as métricas de constância e de F1 par-a-par consomem, produzida independentemente para cada modelo a partir do mesmo título e esquema. A estabilidade dessa representação depende da combinação de três componentes do Pipeline v2.1: regras de *matching* semânticas no prompt [Brinkmann et al. 2023], a pré-normalização simbólica do título (Seção 3.2) [Li et al. 2020] e *batch prompting* [Fan et al. 2024]. Cabe registrar uma consequência dessa construção. Como o nome canônico agrega todos os atributos extraídos, dois títulos do mesmo produto em que apenas um menciona determinado atributo produzem chaves distintas, ainda que a extração esteja correta em ambos os lados. Esse comportamento é discutido junto aos resultados de constância (Seção 4.1) e delimita uma classe de erro atribuível ao critério de construção da chave, e não à capacidade de extração do modelo.

3.5. Métricas de avaliação

Adotamos duas métricas principais, cada uma com variante exata e ponderada, e reportamos complementarmente a estabilidade da extração entre execuções repetidas (Seção 4.3), medida como a fração de títulos cujo nome canônico é idêntico em três execuções independentes sob configuração inalterada. Todos os intervalos de confiança são de 95%, por *bootstrap* não paramétrico com 10.000 reamostragens.

Constância por grupo. Mede a estabilidade do nome canônico para títulos conhecidamente equivalentes. Grupos são formados pela união dos pares com *label* = 1 via *union-find*; para um grupo de tamanho n com normalizações $\{x_1, \ldots, x_n\}$, a constância exata é $\max_v |\{i : x_i = v\}|/n$, a fração que coincide com a normalização mais frequente. Como essa definição penaliza o grupo inteiro por divergências triviais de formatação, definimos a constância ponderada, que substitui a igualdade exata por uma distância de edição ponderada por tipo de caractere (dígitos custam $100\times$ mais que letras, espaços metade de uma letra, demais símbolos $5\times$), normalizada pelo comprimento da maior string, com limiar 1,0 e agrupamento por *union-find*. A ponderação codifica que um dígito incorreto tipicamente indica produto diferente (*"500ml"* contra *"600ml"*), enquanto uma letra a mais indica ruído. Reportamos a média das duas variantes sobre os 26 grupos formados, que totalizam 56 títulos.

F1 par-a-par com *blocking*. Avalia diretamente a decisão de *entity matching*. Após *post_normalize* (remoção de vírgulas antes de dígitos e de hifens entre letra e dígito, *"sx-2"* $\rightarrow$ *"sx2"*), avaliamos duas variantes: no **exact match** dois títulos são o mesmo produto se seus nomes canônicos são idênticos; na variante **weighted**, extraímos os tokens numéricos com unidade e descartamos o par somente quando ambos os títulos têm tokens *e* esses conjuntos conflitam (nenhum é subconjunto do outro), aceitando-o, caso contrário, se a similaridade de sequência (`difflib.SequenceMatcher`) for $\geq 0{,}90$. O *blocking* corrige uma falha da similaridade pura: títulos muito similares podem descrever produtos distintos que diferem em um único atributo numérico. Precisão e *recall* têm bases distintas: a precisão é condicionada aos pares preditos como positivos por cada método, e o *recall* aos 30 positivos anotados, todos oriundos do conjunto de candidatos descrito na Seção 3.1.1.

4. Resultados e discussões

Os experimentos foram conduzidos sobre o dataset de 1.200 pares anotados manualmente, combinando amostragem aleatória e estratificada por similaridade textual a partir de um pool de 30.000 títulos do dataset MeLi. Avaliamos o pipeline através de constância por grupo e F1 par-a-par com *blocking*, aplicadas independentemente a três modelos de extração: Gemini 3.1 Flash-Lite, GPT-5.4-mini e DeepSeek V4 Flash.

4.1. Constância por grupo

Tabela 1. Comportamento por modelo de extração: constância por grupo (26 grupos, 56 títulos) e estabilidade entre três execuções idênticas (2.267 títulos únicos). IC 95% por *bootstrap*, 10.000 reamostragens.

Modelo	Const. exata	Const. ponderada	Extr. idênt.	F1 (méd. $\pm$ dp)
Gemini 3.1 Flash-Lite	0,680 [0,596; 0,769]	0,872 [0,782; 0,949]	100,0%	$0{,}784 \pm 0{,}000$
GPT-5.4-mini	0,628 [0,551; 0,712]	0,827 [0,731; 0,923]	52,5%	$0{,}458 \pm 0{,}020$
DeepSeek V4 Flash	0,686 [0,596; 0,776]	0,885 [0,808; 0,962]	72,3%	$0{,}494 \pm 0{,}044$

A união dos pares com *label* = 1 produz 26 grupos de produtos equivalentes com pelo menos dois títulos. A Tabela 1 resume a constância média obtida nesses grupos, nas variantes exata e ponderada, para os três modelos avaliados. Os três modelos apresentam constância estatisticamente indistinguível: os intervalos de confiança se sobrepõem amplamente em ambas as variantes. O DeepSeek V4 Flash obtém os maiores valores médios

e o GPT-5.4-mini os menores, mas a magnitude dessas diferenças não é sustentada pela incerteza da estimativa. Esse resultado contrasta com o F1 par-a-par (Seção 4.2), em que os mesmos três modelos diferem em 34,0 pontos percentuais e o DeepSeek V4 Flash apresenta o pior desempenho; o contraste é discutido na Seção 4.4. Em todos os modelos a constância ponderada supera a exata, indicando que parte considerável da instabilidade observada na variante exata decorre de divergências superficiais de formatação, e não de extrações semanticamente distintas.

Os grupos que permanecem com baixa constância na variante ponderada revelam dois padrões dominantes, consistentes entre os modelos. O primeiro, e majoritário, corresponde a títulos em que apenas um dos lados menciona determinado atributo (*"carburador cht"* contra *"carburador cht 1.6"*): a extração está correta em ambos, e a divergência decorre do critério de construção do nome canônico, que agrega todos os atributos extraídos (Seção 3.4). O segundo corresponde a divergências semânticas legítimas: nos três modelos, as normalizações de dois cartões de memória Sandisk de 64 GB preservam a distinção entre os subtipos *classe 10* e *micro sd xc*, resultando em constância ponderada de 0,50. Esse é o comportamento desejado caso os títulos descrevam produtos distintos, o que sugere que parte do erro remanescente é atribuível ao rótulo de equivalência adotado na anotação, e não ao pipeline.

4.2. F1 par-a-par

Tabela 2. F1 par-a-par sobre 1.200 pares anotados (30 positivos), por modelo de extração. TP/FP/FN em contagens absolutas.

Modelo	Método	TP	FP	FN	Precisão	*Recall*	F1
Gemini 3.1 Flash-Lite	Exact match	14	1	16	0,933	0,467	0,622
	Weighted	20	1	10	**0,952**	0,667	**0,784**
GPT-5.4-mini	Exact match	10	8	20	0,556	0,333	0,417
	Weighted	13	15	17	0,464	0,433	0,448
DeepSeek V4 Flash	Exact match	13	21	17	0,382	0,433	0,406
	Weighted	16	26	14	0,381	0,533	0,444
Baseline TF-IDF	palavra	20	17	10	0,541	0,667	0,597
	caractere	19	18	11	0,514	0,633	0,567

A Tabela 2 reporta precisão, *recall* e F1 sobre os 1.200 pares anotados (30 positivos), para as duas variantes de decisão descritas na Metodologia e para os três modelos de extração, comparados a baselines TF-IDF de palavra (unigramas e bigramas; limiar 0,608) e de caracteres (limiar 0,7789). Ambos os limiares foram selecionados por varredura sobre os rótulos do próprio conjunto de avaliação, condição que favorece os baselines lexicais; o *exact match* do TAIL, em contraste, não possui limiar ajustável. O Gemini 3.1 Flash-Lite é o único modelo cujo melhor resultado (*Weighted*) supera os dois baselines TF-IDF, e a natureza dessa vantagem está nas contagens absolutas: ele e o melhor baseline recuperam o mesmo número de pares positivos (20 dos 30, *recall* idêntico), mas o baseline o faz

ao custo de 17 falsos positivos, contra 1 do pipeline proposto. O ganho não decorre, portanto, de maior capacidade de recuperação, e sim da rejeição de pares textualmente similares que descrevem produtos distintos. O *blocking* por tokens numéricos é o que produz esse efeito: para o Gemini, ele eleva o *recall* sem custo de precisão, e os seis pares adicionais recuperados são todos verdadeiros positivos. Nos demais modelos, o mesmo relaxamento admite majoritariamente falsos positivos e derruba a precisão do GPT-5.4-mini; o DeepSeek V4 Flash é o único cujo *exact match* já apresenta precisão inferior à dos baselines lexicais, e seu melhor F1 fica abaixo de ambos.

Tabela 3. Comparações pareadas contra Gemini 3.1 Flash-Lite (*Weighted*, F1 = 0,784)

Comparação	ΔF1 (p.p.)	IC 95%	McNemar (p)
vs. GPT-5.4-mini (Weighted)	$+33{,}6$	$[+16{,}8; +52{,}4]$	$1{,}9 \times 10^{-5}$
vs. DeepSeek V4 Flash (Weighted)	$+34{,}0$	$[+21{,}1; +48{,}5]$	$< 10^{-5}$
vs. TF-IDF palavra	$+18{,}7$	$[+1{,}1; +37{,}3]$	$0{,}0037$
vs. TF-IDF caractere	$+21{,}7$	$[+5{,}3; +39{,}4]$	$0{,}0009$

A Tabela 3 compara as demais variantes à melhor configuração do Gemini 3.1 Flash-Lite, com ICs obtidos por *bootstrap* pareado agrupado (10.000 reamostragens sobre 1.196 agrupamentos), preservando a dependência entre pares e a correlação de dificuldade entre métodos. Nenhum dos intervalos inclui zero, e todos os testes de McNemar rejeitam a hipótese de desempenho equivalente ao nível de 5%. As diferenças em relação aos demais modelos de extração são substancialmente mais amplas e mais precisamente estimadas que as diferenças em relação aos baselines lexicais, cujos limites inferiores são de $+1{,}1$ e $+5{,}3$ pontos percentuais. A vantagem sobre os baselines é, portanto, consistente mas de magnitude menos firmemente estabelecida, condição agravada pelo fato de seus limiares terem sido ajustados no próprio conjunto de avaliação. Não incluímos Ditto [Li et al. 2020] na Tabela 2 porque é um método supervisionado, baseado em *fine-tuning* sobre pares rotulados, enquanto o TAIL opera em regime *zero-shot*: a comparação não seria controlada sob o mesmo regime de aprendizagem. Além disso, os 30 pares positivos deste conjunto são volume insuficiente para ajustar um *cross-encoder* e, simultaneamente, manter uma avaliação independente, o que exigiria uma nova coleção rotulada com divisão própria de treino, validação e teste.

4.2.1. Benchmark no Golden Set do AI-PAVE-Br

Tabela 4. Benchmark no Golden Set do AI-PAVE-Br, esquema alinhado (7.890 itens, 18.677 células)

Sistema	Precisão	Recall	F1	Cobertura
AI-PAVE-Br — Gemini 1.5 Flash	0,778	0,602	**0,679**	77,7%
Baseline empresa terceira	0,851	0,529	0,652	66,8%
TAIL — Gemini 3.1 Flash-Lite	0,845	0,537	0,657	65,1%

Executamos adicionalmente o TAIL com Gemini 3.1 Flash-Lite, na configuração da Seção 3.3, sobre os 7.890 itens do *Golden Set* público do AI-PAVE-Br, utilizando so-

mente o título como entrada. Os três sistemas foram recomputados sobre os mesmos itens e com a mesma implementação de métrica; os valores publicados no trabalho original não são reproduzidos aqui, pois o subconjunto exato de avaliação não pôde ser reconstruído. A Tabela 4 reporta o recorte de esquema alinhado, definido a priori pelos campos que o TAIL representa, com equivalência de unidade aplicada igualmente aos três sistemas. A comparação é assimétrica na entrada: o sistema de referência recebe título e descrição, enquanto o TAIL recebe apenas o título. Sob essa desvantagem, o TAIL fica 2,2 pontos percentuais de F1 abaixo dele e supera o *baseline* industrial, com precisão superior e cobertura menor: Os atributos ausentes do título não são preenchidos incorretamente, apenas deixam de ser preenchidos. Considerando todos os atributos mapeados (23.765 células), o F1 cai para 0,566, o que expõe a limitação de um esquema fixo *cross-categoria*.

4.3. Estabilidade entre execuções

As APIs de LLM não garantem saída determinística, mesmo com `temperature=0`. Para quantificar essa fonte de incerteza, executamos o pipeline completo três vezes para cada modelo, sob configuração inalterada e sobre os mesmos títulos; as duas últimas colunas da Tabela 1 reportam a fração de títulos cujo nome canônico é idêntico nas três execuções e a variação do F1 entre elas. Apenas o Gemini 3.1 Flash-Lite apresenta comportamento determinístico: as três execuções produzem nomes canônicos idênticos para todos os títulos, e o desvio padrão de todas as métricas par-a-par é nulo. Os demais modelos divergem em 27,7% e 47,5% dos títulos entre execuções, apesar da configuração de temperatura zero. Para uma aplicação de normalização, cujo produto é uma chave de deduplicação persistente, essa instabilidade tem natureza distinta de um erro de acurácia: o reprocessamento do mesmo catálogo produziria chaves diferentes para os mesmos produtos, comprometendo a deduplicação incremental mesmo sem alteração no catálogo de origem. Os resultados de constância e os da Tabela 2 correspondem à primeira execução de cada modelo.

4.4. Discussão

A amplitude de 34,0 pontos percentuais de F1 entre modelos da mesma faixa de custo ocorre sob *prompt*, esquema de campos e conjunto de avaliação idênticos, o que isola o modelo de extração como variável manipulada, condição distinta da de Zhang et al. [Zhang et al. 2025], que comparam arquiteturas e métodos sobre múltiplos datasets. Isso qualifica a leitura corrente sobre LLMs em extração de atributos de e-commerce [Brinkmann et al. 2024, Gazzola et al. 2025]: o ganho sobre baselines puramente lexicais não é propriedade do desenho do pipeline, mas da capacidade de extração do modelo escolhido, e se manifesta por precisão, não por recuperação (Seção 4.2). Constância e *matching* capturam aspectos distintos da qualidade da normalização, e é por isso que o DeepSeek V4 Flash lidera uma métrica e fica em último na outra: um modelo pode produzir nomes canônicos internamente consistentes para um mesmo produto sem que coincidam com os de outros títulos do mesmo produto sob o critério de similaridade adotado. A fração de erro que se repete de forma análoga nos três modelos parece ser propriedade do dataset e da definição de equivalência adotada na rotulagem, não do pipeline. Do ponto de vista de sistemas multiagentes, esses resultados informam três decisões de design. A amplitude entre modelos da mesma faixa de custo torna a seleção do modelo uma decisão de roteamento consequente, viabilizando *delegação adaptativa*: um orquestrador pode

reservar o modelo mais capaz aos títulos em que o modelo barato produz extração incompleta. A complementaridade entre constância e F1 par-a-par sustenta um *ensemble* com agente juiz no paradigma *LLM-as-judge* [Zheng et al. 2023], ou, em variante mais simples, consenso por votação majoritária entre execuções, que atenua o não-determinismo sem exigir determinismo do provedor. Por fim, a constância por grupo é computável sem rótulos de referência quando o sistema já dispõe de títulos agrupados como equivalentes, podendo operar como sinal de monitoramento que aciona reprocessamento seletivo por um modelo mais capaz.

5. Limitações e ameaças à validade

Validade interna. O conjunto de avaliação contém 30 pares positivos entre 1.200, o que limita a precisão das métricas condicionadas na classe positiva; a incerteza é reportada por intervalos de confiança em todas as tabelas. Como 29 dos 30 positivos provêm do estrato de candidatos a *match*, o *recall* reportado é condicionado à região de alta similaridade textual, e a amostra aleatória de 300 pares não contém positivos, não permitindo estimar a fração de equivalências descartada pelo limiar. A anotação foi distribuída entre os sete autores sem sobreposição de pares, o que impede a medição formal de concordância inter-anotador. **Validade externa.** A avaliação cobriu um único *marketplace* e o subconjunto em português, e os resultados não devem ser extrapolados para outros domínios, catálogos ou idiomas sem avaliação específica; a mesma ressalva aplica-se à substituição do modelo de extração, que exige reavaliação e não é automática. **Validade de construto.** A constância por grupo assume que os rótulos de equivalência são um *ground truth* perfeito, suposição que os casos discutidos na Seção 4.1 não sustentam: há pares rotulados como equivalentes cujos títulos descrevem subtipos distintos de produto, de modo que a métrica penaliza o pipeline por um comportamento correto. Soma-se a isso o fato de o nome canônico agregar todos os atributos extraídos, e de a contribuição isolada de cada componente não ter sido quantificada por ablação. **Custo.** O pipeline tem custo de inferência $O(n)$, mas o custo monetário é proporcional ao volume de *tokens* processados, e o efeito do *batch prompting* sobre o custo total não foi medido empiricamente.

6. Conclusão

Este trabalho apresentou o TAIL, um pipeline *zero-shot* baseado em LLM para normalização de títulos de produtos em e-commerce brasileiro, avaliado sobre três modelos de extração dentro da mesma arquitetura e sobre o mesmo dataset de 1.200 pares anotados manualmente. O Gemini 3.1 Flash-Lite obteve o melhor desempenho e foi o único a superar os baselines TF-IDF, principalmente por precisão: sobre os mesmos pares positivos recuperados pelo melhor baseline, reduziu os falsos positivos de 17 para 1. Esse desempenho depende fortemente do modelo de extração escolhido, em duas dimensões distintas: a acurácia, com amplitude de 34,0 pontos percentuais de F1 entre o melhor e o pior modelo, e a reprodutibilidade, já que apenas o Gemini 3.1 Flash-Lite produz extração idêntica em execuções repetidas, enquanto os demais divergem em 27,7% e 47,5% dos títulos. Para uma aplicação cujo produto é uma chave de deduplicação persistente, a segunda dimensão tem consequência prática distinta de um erro de acurácia. A fração de erro que se repete nos três modelos corresponde ao critério de construção do nome canônico e a divergências semânticas legítimas entre subtipos de produto, e não a falhas de extração, o que delimita um piso razoável para uma abordagem baseada exclusivamente no título; esse piso é competitivo, já que no *Golden Set* do AI-PAVE-Br

[Gazzola et al. 2025] o TAIL fica 2,2 pontos de F1 abaixo do sistema de referência recebendo apenas o título. Como trabalhos futuros, pretendemos investigar o uso de um LLM como *judge* [Zheng et al. 2023] na expansão do dataset de avaliação, conduzir um estudo de ablação isolando a contribuição de cada componente do pipeline, e comparar o TAIL a baselines supervisionados de *entity matching* sob uma coleção rotulada com divisão própria de treino e teste.

Declaração sobre o Uso de Inteligência Artificial

Durante a elaboração deste artigo, o modelo Claude Sonnet 4.6 (Anthropic) foi utilizado como ferramenta de assistência tecnológica. Seu uso restringiu-se ao aprimoramento do estilo de escrita acadêmica, à revisão ortogramatical e ao auxílio na formatação de comandos LaTeX.

Referências

Brinkmann, A., Baumann, N., and Bizer, C. (2024). Using LLMs for the extraction and normalization of product attribute values. In *Advances in Databases and Information Systems: 28th European Conference (ADBIS 2024)*, volume 14918 of *Lecture Notes in Computer Science*, pages 217–230. Springer Nature Switzerland.

Brinkmann, A., Shraga, R., and Bizer, C. (2023). Product attribute value extraction using large language models. *arXiv preprint arXiv:2310.12537*.

Fan, M., Han, X., Fan, J., Chai, C., Tang, N., Li, G., and Du, X. (2024). Cost-effective in-context learning for entity resolution: A design space exploration. In *Proceedings of the 40th IEEE International Conference on Data Engineering (ICDE)*, pages 3696–3709.

Gazzola, M., Souto, H. G., Silva, S., Peixoto, J. S., Siqueira, F., Morais, A. L. P. d., and Gomes, C. (2025). AI-PAVE-Br: Leveraging large language models for enhanced product attribute value extraction through a golden set approach. In *Anais do XVI Simpósio Brasileiro de Tecnologia da Informação e da Linguagem Humana (STIL)*, pages 149–160.

Li, Y., Li, J., Suhara, Y., Doan, A., and Tan, W.-C. (2020). Deep entity matching with pre-trained language models. *Proceedings of the VLDB Endowment*, 14(1):50–60.

More, A. (2016). Attribute extraction from product titles in ecommerce. *arXiv preprint arXiv:1608.04670*.

Neres de Sousa, J. V. C., Mingardo, L. M., Freire, C. E. T., Traina, A. J. M., and Junior, C. T. (2025). Agente autônomo guiado por LLM para extração de notícias. In *Anais do XL Simpósio Brasileiro de Bancos de Dados (SBBD)*, pages 278–288.

Zhang, Z., Groth, P., Calixto, I., and Schelter, S. (2025). A deep dive into cross-dataset entity matching with large and small language models. In *Proceedings of the 28th International Conference on Extending Database Technology (EDBT)*, pages 922–934.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 46595–46623.