From Text to Barcode: Inferring Product Identifiers in Electronic Invoices with Missing Information

Carlos Filipe de Castro Lemos[1],
Brucce Neves dos Santos[1],
Ricardo Marcondes Marcacini[1]

[1] Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP)

{filipelemos,ricardo.marcacini}@usp.br, {brucce.neves}@alumni.usp.br

Abstract. *Electronic invoices are valuable sources of information for public administration, enabling price monitoring, fraud detection, and greater transparency. A recurring challenge, however, is the absence or inconsistency of structured product identifiers, such as barcodes (GTINs), which are essential for comparing products across transactions. Often, only short, noisy, and unstandardized textual descriptions are available. This work proposes a hybrid strategy for identifier inference, combining high-precision string-matching with interpretable machine learning models based on vectorized text representations. Results show that while string-matching is accurate, its coverage is limited; supervised classifiers expand coverage effectively, especially when using character-level n-grams. The proposed approach is integrated into an open-source fiscal mining tool, leveraging simple and efficient methods suitable for large-scale data processing.*

Resumo. *Notas fiscais eletrônicas são fontes valiosas de informação para a administração pública, permitindo o monitoramento de preços, detecção de fraudes e maior transparência. Um desafio recorrente, porém, é a ausência ou inconsistência de identificadores estruturados, como códigos de barras (GTINs), essenciais para comparar produtos entre transações. Frequentemente, apenas descrições textuais curtas, ruidosas e despadronizadas estão disponíveis. Este trabalho propõe uma estratégia híbrida para inferência de identificadores, combinando string-matching de alta precisão com modelos interpretáveis de aprendizado de máquina baseados em representações vetoriais. Os resultados mostram que o string-matching é preciso, mas limitado em cobertura, enquanto classificadores supervisionados ampliam a abrangência, especialmente com n-gramas de caracteres. A abordagem foi incorporada a uma ferramenta open-source para mineração fiscal, adotando métodos simples e eficientes, adequados ao processamento de dados em larga escala.*

1. Introdução

Em aplicações do mundo real, é comum a ocorrência de bases de dados heterogêneas [Rahm and Do 2000], caracterizadas por informações inconsistentes, com conteúdo genérico ou incompleto [Fan et al. 2014]. Essa realidade configura um dos principais desafios para o Aprendizado de Máquina na Indústria 4.0 [Xie et al. 2025], exigindo

estratégias robustas que garantam confiabilidade e escalabilidade no processamento e classificação de grandes volumes de dados [Chen et al. 2014, Gandomi and Haider 2015].

Um exemplo concreto desse cenário ocorre na análise de Notas Fiscais Eletrônicas (NFe), pois cada produto deveria ser identificado por uma numeração única e precisa (*Global Trade Item Number* - GTIN) capaz de identificá-lo inequivocamente. No entanto, existe uma grande quantidades de itens que possuem descrições vagas e imprecisas ou que sequer apresentam GTIN.

Com efeito, a ausência ou inconsistência dessas informaçoes acarreta prejuízos coletivos. Isso ocorre porque as secretarias da fazenda estaduais e os tribunais de contas utilizam os dados das NFe para, respectivamente, monitorar preços e detectar sobrepreços em compras públicas. Nesse caso, sem a veracidade dos dados, existe uma grande dificuldade na identificação e na precificação dos produtos, inviabilizando a comparação direta entre itens semelhantes. Consequentemente, as estatísticas de preços médios são calculadas de forma equivocada, o que resulta no comprometimento não apenas dos esforços de controle e fiscalização tributária [de Angeli Neto and Martinez 2016, da Cunha Panis et al. 2022], em caso de fraudes licitatórias [OECD 2017], mas também na elaboração de planejamentos públicos e privados mais eficientes [Araújo et al. 2023].

Diante dessas limitações, surge a necessidade de etapas de pós-processamento e da adoção de métodos automáticos capazes de estimar o GTIN com base nas informações textuais disponíveis. Assim, diversas técnicas podem ser empregadas na tentativa de classificar NFe desprovidas de GTIN, variando desde abordagens simples como *string-matching* até métodos baseados em aprendizado de máquina.

Nesse contexto, o *string-matching* destaca-se como uma solução intuitiva, pois verifica a identidade entre as descrições de produtos de uma nota fiscal eletrônica com correlatos na base de dados de referência, mas sua eficácia é limitada. Além disso, embora existam estudos abrangentes sobre *string matching* [Zhang 2022], a complexidade da tarefa permanece elevada devido a fatores como descrições curtas e não padronizadas, baixa densidade semântica e alta cardinalidade de classes (frequentemente na ordem de centenas ou milhares de GTINs). Esses desafios comprometem a eficácia de abordagens simplistas, especialmente em contextos caracterizados por grandes volumes de dados.

Assim, este trabalho tem como objetivo avaliar a eficácia de diferentes abordagens para a inferência de GTINs a partir das descrições de produtos em NFe. A investigação é realizada no contexto do desenvolvimento contínuo da ferramenta *open-source* NFeMiner[1], que agrega métodos para mineração, análise e enriquecimento de dados fiscais.

A proposta inclui a implementação de um módulo específico para estimativa de GTINs, voltado à classificação de NFe que não possuem esse identificador ou apresentam códigos inconsistentes. O estudo busca quantificar as limitações do *string-matching* (soluções simples e intuitivas), bem como complementar a classificação com algoritmos de aprendizado de máquina aplicados a diferentes representações textuais das NFe. Além disso, este trabalho também investiga o impacto da representação textual no desempenho dos classificadores, a escalabilidade computacional das soluções propostas e explora possíveis combinações entre métodos com foco em robustez e cobertura.

[1] `https://github.com/LABIC-ICMC-USP/NFeMiner`

O restante do artigo está organizado em uma estrutura de seções. Na Seção 2 são apresentados os materiais e métodos utilizados, incluindo a formulação do problema, a descrição das abordagens avaliadas, os critérios de avaliação e as características da base de dados. A Seção 3 apresenta os resultados obtidos, com análises quantitativas e qualitativas sobre o desempenho das diferentes estratégias, incluindo aspectos de acurácia, ambiguidade, cobertura e eficiência computacional. Por fim, a Seção 4 traz as conclusões do estudo e indica possíveis direções para trabalhos futuros.

2. Materiais e Métodos

Esta seção descreve os materiais e métodos do estudo, detalhando-se os procedimentos utilizados no desenvolvimento e na avaliação do experimento. Na Seção 2.1, apresenta-se a formulação do problema, especificamente com a definição dos objetivos e particularidades da tarefa de predição do GTIN em Notas Fiscais Eletrônicas. Na Seção 2.2, são descritas as abordagens para quantificar e avaliar a classificação. Na Seção 2.3, discrimina-se os critérios de avaliação dos resultados. Por fim, na Seção 2.4, descreve-se o *pipeline*, as ferramentas e as demais configurações experimentais.

2.1. Formulação do Problema

Seja $\mathcal{N} = \{n_1, n_2, \ldots, n_T\}$ o conjunto total de NFe disponíveis em que cada nota $n_i \in \mathcal{N}$ possui, entre outros atributos, a descrição textual de um produto. A partir desse conjunto, define-se um subconjunto $\mathcal{R} \subset \mathcal{N}$ contendo apenas as notas com códigos GTIN considerados confiáveis, conforme validados por especialistas da área.

Cada nota fiscal $n_i \in \mathcal{R}$ é representada como um par (x_i, y_i), onde $x_i \in \mathcal{X}$ corresponde à descrição textual do produto (pré-processada e normalizada), e $y_i \in \mathcal{Y} \subset \mathbb{N}$ representa o respectivo código GTIN. O conjunto supervisionado de treinamento é, portanto, definido conforme a Equação 1.

$$\mathcal{D} = \{(x_i, y_i) \mid n_i \in \mathcal{R}\} \tag{1}$$

O objetivo deste trabalho é avaliar estratégias para uma função de mapeamento f, conforme definido na Equação 2, capaz de inferir o GTIN mais provável a partir de uma descrição textual de produto extraída de uma nota fiscal que não possui GTIN confiável:

$$f : \mathcal{X} \to \mathcal{Y} \tag{2}$$

Assim, dado um novo exemplo $x \in \mathcal{X}$, proveniente de uma nota $n \in \mathcal{N} \setminus \mathcal{R}$, o objetivo é predizer o GTIN mais provável $\hat{y} \in \mathcal{Y}$, conforme a Equação 3.

$$\hat{y} = f(x) \tag{3}$$

A função de predição f pode ser construída por diferentes estratégias. Uma possibilidade é a abordagem baseada em *string matching*, na qual f realiza a comparação da descrição x com as descrições presentes no conjunto de referência $\mathcal{D}$. Outra possibilidade é o uso de algoritmos supervisionados de aprendizado de máquina, nos quais f é induzida a partir de representações vetoriais geradas para os textos.

De forma geral, conforme ilustrado na Tabela 1, o problema é intensificado por dois fatores principais:

1. Alta cardinalidade das classes: por exemplo, o conjunto de dados usado neste trabalho possui 890 GTINs distintos em um total de 64.550 notas fiscais.
2. Representatividade informacional reduzida: as descrições dos produtos possuem restrições em termos qualidade, envolvendo caracteres maiúsculos e minúsculos, bem como diversas abreviações e unidades de medidas.

Tabela 1. Exemplo de descrições e suas respectivas GTIN de NFe.

Descrição	GTIN
carne bovina contra file s osso congelado	0000000000017
carne bovina costela 5porcento de gord cong	0000000000017
costela bovina	0000000000017
carne bovina cha de dentro sem osso congela	—
carne bovina lagarto redondo s osso congela	—
carne casada bovina	0000000000031
carne bovina bife	0000000000031
carne bovina paleta	0000000000031
carne bovina agulha	—
carne bovina guizado	—

Por outro lado, os objetivos específicos deste estudo são: (i) avaliar e quantificar as métricas de desempenho associadas à tarefa de predição do código GTIN; e (ii) propor uma solução híbrida que combine a abordagem baseada em *string-matching* com técnicas de aprendizado de máquina, aplicada a registros de NFe que não contenham o código GTIN explicitamente preenchido ou apresentem essa informação de forma inconsistente.

2.2. Abordagens Avaliadas

Foram avaliadas duas estratégias para a abordagem do problema de predição do código GTIN: (i) a metodologia baseada em *string-matching* e (ii) a utilização de algoritmos de aprendizado de máquina. Essas metodologias são formalmente apresentadas, respectivamente, nas Seções 2.2.1 e 2.2.2.

2.2.1. *String-matching*

A primeira estratégia consiste na aplicação de *string-matching*. Essa técnica é baseada na comparação direta entre cadeias de caracteres. A ideia central é que, caso uma descrição textual de uma nota fiscal sem GTIN seja idêntica a outra previamente conhecida e associada a um GTIN, é possível atribuir esse rótulo de forma imediata.

Propõe-se a utilização de uma estrutura de mapeamento no seguinte formato: $f(\text{descrição da NFe}) = \text{GTIN}$. Essa abordagem é viável porque os softwares e as empresas tendem a padronizar as descrições dos produtos, resultando na geração de registros de confiança, ou seja, descrições textuais que se repetem com uma frequência mensurável. Esse cenário é ideal para o processamento, pois simplifica o fluxo de trabalho, tornando as operações computacionais mais rápidas, eficientes, explicáveis e escaláveis.

No entanto, essa técnica possui limitações, uma vez que descrições semelhantes podem variar por pequenos detalhes, como abreviações, erros de digitação ou omissões, o que compromete a efetividade da correspondência exata. Além disso, uma mesma descrição pode, em determinadas situações, estar associada a múltiplos GTINs que possuem numerações diferentes (em uma relação de um para muitos), o que acarreta erros de classificação. Esclareça-se que, neste trabalho, isso é definido como ambiguidade.

Dessa forma, a análise experimental com *string-matching* foi guiada pelos seguintes questionamentos dentro dos tópicos abaixo:

1. GTINs confiáveis: qual é o impacto da definição de um número mínimo de repetições para que uma descrição seja considerada confiável e qual é o impacto sobre a taxa de acerto da classificação?
2. Taxa de ambiguidade: como se comporta a ambiguidade dos registros de confiança à medida que o limiar mínimo de repetição é elevado?
3. Taxa de cobertura: qual é a porcentagem do conjunto de dados que potencialmente pode ser classificada com base em descrições que atendem ao critério de confiança e não são ambíguas?

Essas questões são fundamentais, especialmente a taxa de cobertura, pois, se a abordagem por *string-matching* se mostrar eficiente e suficiente, ela pode, em tese, dispensar o uso de métodos mais complexos como os de aprendizado de máquina.

2.2.2. Algoritmos de Aprendizado de Máquina

A segunda estratégia investigada consiste no emprego de técnicas de aprendizado supervisionado, aplicadas aos registros que não foram classificados pela abordagem de *string-matching*. Esse cenário ocorre, por exemplo, quando os registros têm frequência inferior ao limite de confiança, as GTINs possuem descrições ambíguas ou variações linguísticas.

Nesse cenário, os algoritmos de aprendizado de máquina são capazes de aprender padrões latentes a partir de dados rotulados. Essa característica torna-os úteis quando a técnica de *string-matching* falha ou retorna possibilidades concorrentes. Portanto, é uma alternativa robusta para predição do código GTIN em cenários complexos.

Neste trabalho, foi adotado o modelo de espaço vetorial para representar as descrições textuais, explorando três formas distintas de tokenização: por palavras (*WORDS*), por caracteres (*CHAR*) e por n-gramas de caracteres (*NGRAMS*), com $n \in \{1, 2, ..., 5\}$. Para cada uma dessas formas, foram utilizadas duas técnicas clássicas de vetorização: contagem de frequências (TF) e *Term Frequency-Inverse Document Frequency* (TFIDF). Adicionalmente, é necessário esclarecer que as siglas entre parênteses correspondem aos identificadores utilizados nos resultados experimentais.

Por outro lado, os algoritmos de classificação que foram utilizados como *benchmark* porque representam abordagens clássicas consolidadas na literatura de aprendizado de máquina [Han et al. 2011]. Cada um deles oferece características distintas que permitem avaliar o desempenho do sistema sob diferentes perspectivas: métodos baseados em distância, como o 1-NN, oferecem simplicidade e robustez [Cover and Hart 1967, Hastie et al. 2009]; algoritmos probabilísticos, como o *Naive*

Bayes, proporcionam modelos rápidos e eficientes, mesmo com conjuntos de dados extensos [Domingos and Pazzani 1997, Zhang 2004]; enquanto que técnicas baseadas em conjunto, como o *Random Forest*, são reconhecidas por sua capacidade de lidar com dados complexos e por sua elevada acurácia [Breiman 2001, Fernández-Delgado et al. 2014]. Dessa forma, a utilização dessas técnicas possibilitam uma avaliação mais completa e comparativa dos resultados e estabelecer uma base mais sólida para futuras análises.

2.3. Métodos de Avaliação

As técnicas de avaliação envolveram três eixos principais: a qualidade das classificações, a eficiência computacional e a robustez dos resultados em relação à significância estatística. No caso da abordagem de *string-matching*, a taxa de ambiguidade identificou a frequência de descrições não confiáveis, enquanto a taxa de cobertura mensurou a proporção de registros classificáveis dentro do limite de confiança.

Quanto aos algoritmos de aprendizado de máquina, a eficiência foi aferida com base no tempo médio de treinamento, que corresponde ao período necessário para a criação dos vetores de representação das descrições de entrada, e no tempo médio de classificação. Também foi considerado o tempo total, definido como a soma das duas etapas. Esses aspectos são fundamentais para avaliar a escalabilidade das soluções propostas. A qualidade das classificações foi analisada por meio de métricas como a acurácia, que expressa a proporção de classificações corretas, e a *F1-Macro*, que avalia o equilíbrio entre precisão e revocação em cenários multiclasse.

Por fim, a robustez e a reprodutibilidade dos resultados foram asseguradas mediante 50 execuções com diferentes sementes aleatórias e validação cruzada estratificada, garantindo a significância estatística das análises realizadas.

2.4. Configurações Experimentais

Para a realização do experimento, foram empregadas ferramentas tradicionais de análise de dados. Em termos de hardware, as execuções ocorreram em um sistema equipado com processador AMD Ryzen 7 5700X e placa gráfica NVIDIA GeForce RTX 3060. A seguir, as etapas do processo são descritas detalhadamente:

1. Definição das Partições e Configuração de Sementes Aleatórias: foram realizadas 50 execuções independentes, cada uma inicializada a partir de uma lista de sementes aleatórias com valores variando entre 10 e 300, controladas e rastreadas por meio da *seed* 42. Em cada execução, o conjunto original de dados foi particionado, de maneira estratificada com relação ao atributo GTIN, em dois subconjuntos: treinamento (80%) e teste (20%).
2. Seleção de Descrições Confiáveis: esta etapa teve como objetivo extrair medidas numéricas para a análise do comportamento das taxas de acurácia, ambiguidade e cobertura, em função da variável livre associada aos registros considerados confiáveis. Para tanto, estabeleceu-se o limiar variável, no intervalo de 1 a 301, que determinou a frequência mínima para que os pares (<GTIN>, <descrição original>) fossem considerados confiáveis. Assim, apenas os pares cuja frequência fosse superior a esse limiar foram mantidos para as etapas subsequentes da análise.

3. Filtragem para Eliminação de Ambiguidades: posteriormente, aplicaram-se operações de diferença entre conjuntos de *strings*, com o objetivo de extrair descrições exclusivas por GTIN, de modo a evitar classificações ambíguas. Esse procedimento foi fundamental para garantir que cada GTIN fosse associada a descrições exclusivas, mesmo que não-únicas, daquele GTIN. A partir desse processo, foi possível construir a estrutura de mapeamento com base no conjunto de treinamento e, em seguida, utilizá-la tanto para a classificação do conjunto de teste quanto para a extração das métricas de avaliação pertinentes.
4. Algoritmos de Aprendizado de Máquina: foram aplicados como estratégia complementar aos casos remanescentes ao *string-matching*. Inicialmente, foram realizados procedimentos de remoção de registros duplicados e, em seguida, os dados foram submetidos à validação cruzada com *10-fold*. Nesse momento, foram utilizadas diferentes abordagens de classificação: algoritmos baseados em distância (com $k = 1$), métodos de conjuntos baseados em modelos de árvores (com quantidade de estimadores $= 100$) e modelos probabilísticos.

3. Resultados e Discussões

Nesta seção, são apresentados e discutidos os resultados obtidos. A Seção 3.1 descreve o conjunto de dados utilizado. Em seguida, a Seção 3.2 apresenta os resultados referentes à abordagem de *string-matching*. A Seção 3.3 aborda o desempenho dos algoritmos de aprendizado de máquina. A Seção 3.4 analisa as métricas de desempenho por classificador e vetorizador. Por fim, na Seção 3.5, é feita comparação dos tempos de processamento.

3.1. Conjunto de Dados

O conjunto de dados do experimento é composto por 64550 registros, sendo certo que possuíam dois atributos, conforme ilustrado na Tabela 1:

1. Descrição textual: atributo textual que trazia como conteúdo as descrições dos produtos nas NFe. No conjunto de dados, existem 2739 descrições únicas.
2. GTIN: identificador único para produtos comercializados globalmente. Esses identificadores eram representados por uma pequena cadeia de caracteres, exclusivamente numéricos. No conjunto de dados, existem 890 numerações únicas de código de barras dos produtos.

3.2. Resultados relacionados ao String-Matching

A Tabela 2 apresenta as estatísticas descritivas médias das 50 execuções da abordagem de *string-matching*, aplicada ao conjunto de teste após o treinamento. A Figura 1 complementa esses dados com *boxplots* das distribuições e a evolução das métricas.

Tabela 2. Estatísticas descritivas das métricas de avaliação do *string-matching*. Valores menores são desejáveis para a taxa de ambiguidade, enquanto que, mais altos, indicam melhor acurária e taxa de cobertura

	Média	**Desvio Padrão**	**Min**	**25%**	**50%**	**75%**	**Max**
Taxa de Ambiguidade	0.133	0.054	0.057	0.081	0.130	0.175	0.279
Acurácia	0.979	0.009	0.967	0.973	0.976	0.989	0.999
Taxa de Cobertura	0.632	0.069	0.523	0.570	0.626	0.690	0.799

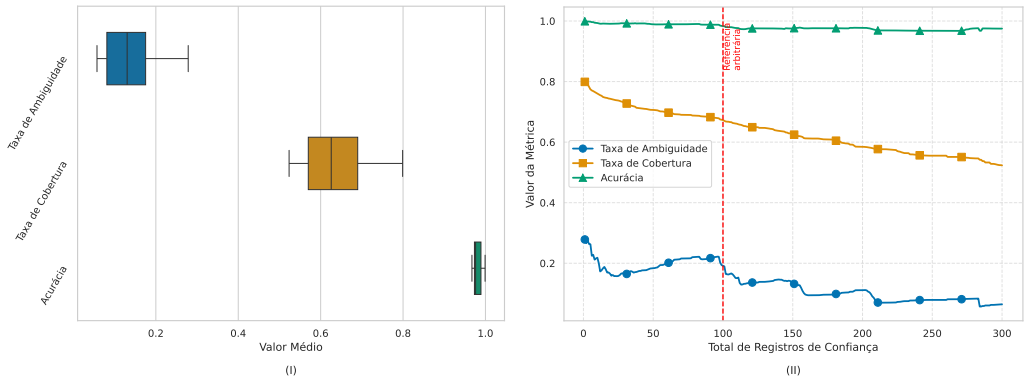

Figura 1. As métricas Taxa de Ambiguidade (menor é melhor), Taxa de Cobertura e Acurácia (maiores são melhores) são apresentadas em (I) por meio de boxplots das médias das 50 execuções e em (II) pela evolução dessas métricas ao longo dos experimentos.

A taxa média de ambiguidade foi de aproximadamente 13,3 $\pm$ 5,4%, com variação entre 5,7% (mínimo) e 27,9% (máximo). Esses resultados indicam a presença de ambiguidades em todos os níveis de confiança, reforçando a necessidade de mecanismos de controle para mitigar classificações incorretas.

No que se refere à acurácia, observou-se uma média elevada, em torno de 97,9 $\pm$ 0.9%, com valores variando entre 96,7% (mínimo) e 99,9% (máximo). Tal resultado evidencia a alta precisão do método, enquanto que a baixa variabilidade dos resultados sugere que as intervenções de confiança contribuíram para manter um desempenho consistente em diferentes níveis de confiança.

Por fim, a taxa de cobertura apresentou média de 63,2 $\pm$ 6,9%, com amplitude entre 52,3% e 79,9%. Esses dados indicam que, embora o método tenha se mostrado bastante preciso quando aplicável, sua cobertura média permaneceu relativamente limitada. Mesmo no cenário de máxima cobertura, não foi possível abranger a totalidade dos registros do conjunto de dados. Assim, independentemente da quantidade de registros de confiança, conclui-se que o método baseado em *string-matching* requer técnicas complementares para ampliar sua cobertura.

3.3. Resultados relacionados aos Algoritmos de Aprendizado de Máquina

Na Tabela 3 estão ilustradas as estatísticas descritivas relacionadas ao desempenho geral dos modelos de classificação (paradigma médio de referência), considerando as 50 execuções de treino e validação. Em relação às quantidades de registros, as métricas geradas pela validação cruzada 10-*fold* indicaram a utilização média de 16.282,47 $\pm$ 161,53 registros para o treinamento e de 1.809,35 $\pm$ 17,94 registros para a validação.

No que se refere à acurácia e a *F1-Macro*, os resultados apresentaram diferenças numéricas expressivas. A acurácia média obtida pelos modelos foi de 73,1 $\pm$ 11,8%, enquanto o *F1-Macro* apresentou uma média inferior, de 56,7 $\pm$ 25,4%. Esses resultados evidenciam uma taxa de acerto moderada, associada a uma elevada variabilidade nas taxas de precisão e revocação. Assim, conclui-se que os modelos são razoavelmente eficazes, demonstrando um bom desempenho em termos de acurácia, mas com certo desequilíbrio entre as taxas de revocação e precisão.

Tabela 3. Estatísticas descritivas das métricas de avaliação

	Média	Desvio Padrão	Min	25%	50%	75%	Max
Tempo de Treino	0.328	0.705	0.001	0.004	0.038	0.274	3.383
Tempo de Predição	0.149	0.194	0.002	0.008	0.095	0.325	0.880
Acurácia	0.731	0.118	0.373	0.671	0.769	0.826	0.852
F1-Macro	0.567	0.254	0.026	0.549	0.702	0.722	0.784
Registros de Treino	16282.469	161.526	15956.000	16167.000	16269.000	16436.000	16551.000
Registros de Validação	1809.351	17.937	1772.000	1796.000	1808.000	1826.000	1839.000

3.4. Desempenho Médio da Acurácia e do F1-Macro por Classificador e Vetorizador

Nas Tabelas 4 e 5 estão ilustrados, respectivamente, os desempenhos médios de acurácia e de *F1-Macro* obtidos a partir das diferentes combinações entre vetorizadores e classificadores, sob as mesmas condições experimentais. Ressalte-se que ambos os resultados correspondem às médias calculadas a partir das 50 execuções, realizadas com validação cruzada 10-*fold*. Ademais, esses resultados encontram-se sintetizados na Figura 2.

Tabela 4. Desempenho médio de acurácia por classificador e vetorizador

	TF-CHAR	TF-NGRAM	TF-WORD	TFIDF-CHAR	TFIDF-NGRAM	TFIDF-WORD
1-NN (GPU e threads)	0.769	0.773	0.762	0.768	0.773	0.761
Naive Bayes	0.389	0.558	0.540	0.531	0.638	0.660
Random Forest	0.836	0.840	0.830	0.836	0.840	0.830

Tabela 5. Desempenho médio de *F1-Macro* por classificador e vetorizador

	TF-CHAR	TF-NGRAM	TF-WORD	TFIDF-CHAR	TFIDF-NGRAM	TFIDF-WORD
1-NN (GPU e threads)	0.709	0.719	0.686	0.707	0.719	0.686
Naive Bayes	0.031	0.096	0.114	0.112	0.199	0.244
Random Forest	0.734	0.743	0.709	0.734	0.742	0.711

Na comparação entre classificadores, é possível verificar que o *Random Forest* apresentou os melhores desempenhos, com taxas de acurácia variando entre 83,0% e 84,0%, e *F1-Macro* entre 70,9% e 74,3%. No extremo oposto, o *Naive Bayes* registrou os piores resultados, com acurácia oscilando entre 38,9% e 66,0%, e *F1-Macro* entre 3,1% e 24,4%. Os algoritmos 1-NN exibiram desempenhos intermediários, com resultados idênticos entre si, como esperado, dado que correspondem ao mesmo método. Suas taxas situaram-se entre os extremos observados, o que é coerente com a literatura especializada.

No que tange à comparação entre vetorizadores, é possível observar resultados heterogêneos. Para os algoritmos baseados em distância, destacaram-se as representações geradas pelos vetorizadores TF-NGRAM e TFIDF-NGRAM, ambos atingindo acurácia e *F1-Macro* de 77,3%. Por sua vez, o classificador probabilístico obteve melhor desempenho com o vetorizador TFIDF-WORD, alcançando acurácia de 66,0% e *F1-Macro* de 24,4%. O classificador baseado em conjunto de árvores de decisão apresentou desempenhos mais uniformes: obteve sua melhor acurácia com TF-NGRAM e TFIDF-NGRAM (ambos com 84,0%), enquanto que o melhor *F1-Macro* foi alcançado com TF-NGRAM (74,4%), seguido de perto por TFIDF-NGRAM (74,2%) e, empatados, TF-CHAR e TFIDF-CHAR, ambos com 73,4%.

De modo específico, as combinações de TFIDF-NGRAM ou TF-NGRAM com o classificador *Random Forest* configuraram-se como as soluções mais eficazes. Em

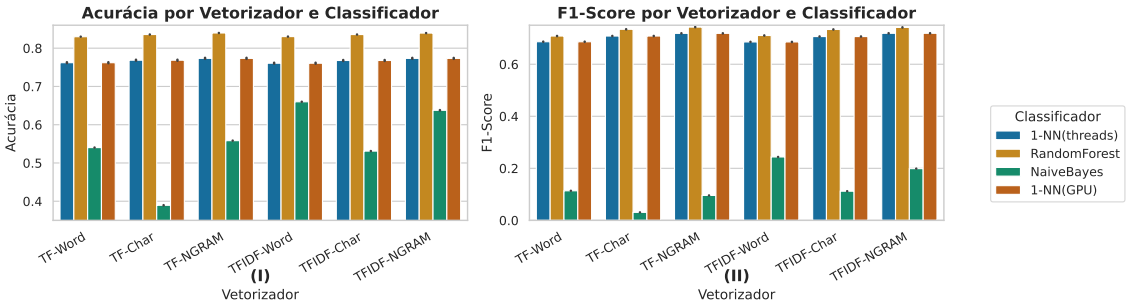

Figura 2. O gráfico mostra o comparativo das métricas acurácia e da *F1-Macro* dos classificadores por vetorizador

contraste, a combinação de TF-CHAR com *Naive Bayes* resultou no pior desempenho. Ressalte-se ainda que tais resultados são coerentes com as características do problema, uma vez que, dada a reduzida dimensionalidade das descrições, representações baseadas exclusivamente em palavras ou caracteres isolados tendem a ser menos adequadas do que aquelas fundamentadas em n-gramas de caracteres.

3.5. Comparativo dos Tempos de Treinamento, Classificação e Tempo Total

Na Tabela 3 está ilustrado os tempos médios de treinamento e de classificação, enquanto que a Figura 3 complementa essas informações, oferecendo uma visualização comparativa entre os diferentes algoritmos analisados.

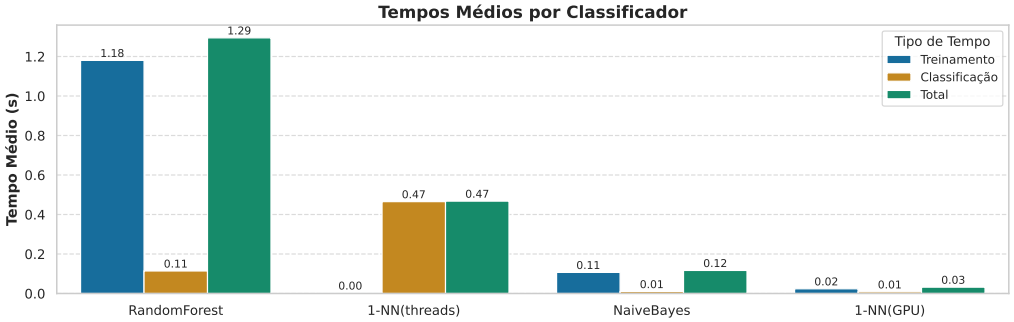

Figura 3. O gráfico compara o desempenho de tempo de treinamento, classificação e tempo total dos algoritmos de aprendizado de máquina nas 50 execuções com validação cruzada de *10-folds*

No que se refere ao tempo de treinamento, observou-se uma média de 0,328 ± 0,705 segundos, com valores variando entre 0,001 segundo (mínimo) e 3,383 segundos (máximo). A Figura 3 evidencia que o classificador *Random Forest* foi o que apresentou o maior tempo médio de treinamento (1,18 segundos), o que pode representar uma limitação para aplicações que demandem treinamento frequente em ambientes com restrições computacionais. Em contraste, os algoritmos 1-NN produziram tempos próximos de zero, em razão de adotarem o paradigma de *lazy learning* que não exige treinamento.

Em relação ao tempo de classificação, a média foi de 0,149 ± 0,194 segundos, com uma amplitude de 0,002 a 0,880 segundos. Embora os tempos médios de classificação tenham se mantido em níveis considerados baixos, nota-se uma variação importante entre os

algoritmos. Destaca-se que o classificador 1-NN implementado com processamento paralelo via GPU apresentou um desempenho superior, com tempo médio de classificação de 0,01 segundo, significativamente inferior ao observado para a versão *thread* (0,47 segundo) e para o *Random Forest* (0,12 segundo). O *Naive Bayes*, por sua vez, também demonstrou elevada eficiência, com tempo médio de 0,01 segundo.

Considerando-se o tempo total de processamento, verifica-se que o 1-NN com suporte a GPU obteve o melhor desempenho global, com média de 0,03 segundo, superando substancialmente os demais algoritmos. Estes resultados indicam que, especialmente para cenários caracterizados por grandes volumes de dados e necessidade de classificações em larga escala, a combinação do 1-NN com aceleração por GPU representa a solução mais eficiente do ponto de vista computacional.

4. Conclusões e Trabalhos Futuros

Este trabalho concentrou-se na avaliação de técnicas clássicas e híbridas para a classificação de GTINs, organizadas conforme a complexidade necessária para lidar com os desafios impostos pelos dados. Inicialmente, o método de *string-matching* apresentou excelente acurácia e baixa taxa de ambiguidade, sendo capaz de classificar diretamente mais de 60% das notas fiscais. Contudo, a limitação na cobertura revelou-se um obstáculo significativo, evidenciando a necessidade de recorrer a algoritmos de aprendizado de máquina para ampliar a abrangência da classificação.

Nos experimentos envolvendo aprendizado supervisionado, observou-se uma considerável variação de desempenho entre as diferentes combinações de vetorizadores e classificadores, indicando que ambas as escolhas impactam diretamente as métricas de acurácia e *F1-Macro*. Nesse cenário, as representações baseadas em N-Gramas destacaram-se por sua maior robustez na captura de padrões relevantes, superando alternativas mais simples, fundamentadas apenas em palavras ou caracteres isolados.

Embora os resultados obtidos tenham sido globalmente satisfatórios, persiste uma parcela de notas que não pôde ser classificada de forma confiável, o que reforça a necessidade de explorar abordagens mais avançadas. Como perspectivas para trabalhos futuros, propõe-se a investigação de técnicas de enriquecimento semântico, a geração de atributos derivados (*feature augmentation*), o uso de *word embeddings*, bem como a adoção de modelos de *Large Language Models*, visando incrementar a capacidade preditiva e a generalização das soluções.

5. Agradecimentos

Os autores agradecem à Fundação de Apoio à Pesquisa do Estado de São Paulo (FAPESP), à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), à Secretaria da Fazenda do Estado do Rio Grande do Sul (SEFAZ-RS) e ao ICMC pelo apoio financeiro e técnico, que contribuiram para que este trabalho tenha sido realizado.

Referências

Araújo, L., Behr, A., and Schiavi, G. S. (2023). Adoção de business analytics na contabilidade. *Revista Contabilidade e Finanças – USP*, 34(93):e1771.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Chen, M., Mao, S., and Liu, Y. (2014). Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

da Cunha Panis, A., da Silva Filho Isidro, A., de Oliveira Carneiro, D. K., Montezano, L., Junior, P. C. R., and Sano, H. (2022). Inovação em compras públicas: Atividades e resultados no caso do robô alice da controladoria-geral da união. *Cadernos Gestão Pública e Cidadania*, 27(86):e83111.

de Angeli Neto, H. and Martinez, A. L. (2016). Nota fiscal de serviços eletrônica: uma análise dos impactos na arrecadação em municípios brasileiros. *Revista de Contabilidade e Organizações*, 10(26):49–62.

Domingos, P. and Pazzani, M. (1997). On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130.

Fan, J., Han, F., and Liu, H. (2014). Challenges of big data analysis. *National Science Review*, 1(2):293–314.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181.

Gandomi, A. and Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management*, 35(2):137–144.

Han, J., Kamber, M., and Pei, J. (2011). *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 3rd edition.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2nd edition.

OECD (2017). *Technology Tools to Tackle Tax Evasion and Tax Fraud*. OECD Publishing.

Rahm, E. and Do, H. H. (2000). Data cleaning: Problems and current approaches. *IEEE Data Engineering Bulletin*, 23(4):3–13.

Xie, J., Sun, L., and Zhao, Y. F. (2025). On the data quality and imbalance in machine learning-based design and manufacturing—a systematic review. *Engineering*, 45:105–131.

Zhang, H. (2004). The optimality of naive bayes. In *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference*, pages 562–567.

Zhang, Z. (2022). Review on string-matching algorithm. In *Proceedings of the 2022 International Conference on Science and Technology Ethics and Human Future (STEHF 2022)*, volume 144 of *SHS Web of Conferences*, pages 1–6. EDP Sciences.