Engenharia e Avaliação de Features para Extração de Informação em Notas Fiscais

Eduardo Darrazão[1], **Victor Amorim**[1], **Krerley Oliveira**[2], **Luiz Gomes-Jr**[1]

[1]Departamento de Informática – Universidade Tecnológica Federal do Paraná (UTFPR)

[2]Laboratório de Estatística e Ciência dos Dados (LED)
Instituto de Matemática – Universidade Federal de Alagoas (UFAL)

{darrazao,vcardoso}@alunos.utfpr.edu.br, krerley@mat.ufal.br, lcjunior@utfpr.edu.br

Abstract. *The correct identification of the elements present in electronic invoices is important for various issues related to government oversight and transparency, such as identifying overpricing in public bids. However, this identification is a challenge due to the diversity of products, as well as variations and errors in filling out the information. This article applies state-of-the-art techniques to evaluate the best feature sets to apply to Brazilian electronic invoices. The tests used data from invoices issued in Piauí in 2021.*

Resumo. *A correta identificação dos elementos constantes em notas fiscais eletrônicas é importante para diversas questões de fiscalização e transparência governamental, como identificação de sobrepreço em licitações públicas. Porém, esta identificação é um desafio tanto pela diversidade de produtos, quanto pelas variações e erros de preenchimento. Este artigo aplica técnicas de estado-da-arte em extração de informação para avaliar os melhores conjuntos de features para se aplicar em notas fiscais eletrônicas brasileiras. Os testes utilizaram dados de notas fiscais de medicamentos emitidas no Piauí em 2021.*

1. Introdução

Todos os dias, milhões de empresas e pessoas físicas de todo o Brasil realizam operações de compra, venda e transporte de mercadorias. Nessas operações, há a obrigação de emissão de notas fiscais eletrônicas, armazenados pelo poder público para diversas finalidades. Fraudes tributárias geram ao ano perdas de cerca de 400 bilhões de reais, conforme o Instituto Brasileiro de Planejamento e Tributação. O ICMS representa uma parte importante delas [F. N. de Oliveira 2020].

Para possibilitar a automatização da análise das notas emitidas, é preciso identificar adequadamente os produtos negociados. Porém, grande parte da informação útil para identificação de produtos está no campo de descrição textual, que é escrito de modo livre pelo emissor da nota. No caso das notas fiscais eletrônicas relacionadas a medicamentos, o texto presente nas suas descrições contém subclasses hierárquicas que representam a composição e apresentação dos medicamentos, podendo incluir: o nome comercial ou genérico do medicamento, a dosagem, a quantidade do produto e a forma farmacêutica. Esses conteúdos podem ser escritos de forma reduzida, com terminologias diferentes para um mesmo elemento, com siglas, com ou sem caracteres especiais. Além disso, pode haver mal-preenchimento ou, até mesmo, itens que não são medicamentos.

Para a correta identificação das partes que compõem uma determinada descrição de nota fiscal eletrônica, é necessário elencar conjuntos de *features* que dão maior explicabilidade a elas, que permitam distingui-las bem uma das outras e identificar a qual tipo/classe ela pertence. Portanto, este trabalho visa à aplicação de engenharia de *features* para determinar conjuntos de *features* que possam ter maior relevância para a identificação de elementos de notas fiscais eletrônicas conforme algumas métricas determinadas pelos autores e explicadas na seção pertinente.

2. Fundamentos e Trabalhos Correlatos

2.1. Extração de Informação e *Sequence Labeling*

Extração de Informação é uma tarefa de processamento de linguagem natural focada em processar texto não estruturado para obter dados estruturados [Seymore and Rosenfeld 1999]. Diversas técnicas podem ser utilizadas nesta tarefa, dependendo do tipo e da estruturação dos dados de entrada. Por exemplo, quando os dados possuem uma ordem previsível, é possível usar técnicas não supervisionadas, como a proposta em [Cortez et al. 2011]. Quando a ordem é importante porém apresenta alto grau de variação (como no caso de notas fiscais), algoritmos de *sequence labeling* tendem a ser adotados [Seymore and Rosenfeld 1999].

Sequence Labeling é uma tarefa de aprendizado de máquina que envolve a atribuição de rótulos a elementos em uma sequência de texto, sendo uma função importante em processamento de linguagem natural. Os modelos usados para *Sequence Labeling* têm como objetivo identificar padrões ou entidades específicas do texto e extrair informações a partir disso. Alguns exemplos de aplicações para os modelos são análise de *part-of-speech* (POS) e *Named Entity Recognition* (NER). Dois algoritmos amplamente conhecidos para resolver esse tipo de problemas são o Hidden Markov Models (HMMs) e Conditional Random Fields (CRFs).

2.2. Conditional Random Fields (CRF)

O CRF é um modelo probabilístico baseado em um grafo não-direcionado cujos vértices são as observações da sequência de dados e os estados; as arestas são as probabilidades referentes às relações entre esses elementos [John Lafferty 2001]. Um exemplo de grafo pode ser visto na (Figura 1), em que ϕ e ϕ' representam a probabilidade relacionada à conexão entre estados subsequentes e à conexão entre um estado e a observação relacionada a ele, respectivamente.

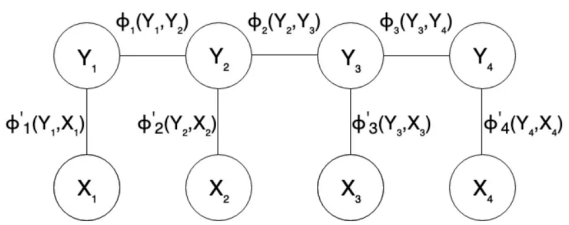

Figure 1. Exemplo de grafo de CRF

O CRF um modelo discriminativo e, portanto, sua probabilidade é determinada por probabilidade condicional. O CRF diferencia-se do HMM (Hidden Markov Model)

com relação à capacidade de dinamizar as probabilidades na medida que uma observação aparece em um lugar da sequência e que vértices não-subsequentes que representam estados podem ter uma aresta entre eles. Além disso, o HMM é um modelo generativo, o que implica que sua probabilidade é dada por probabilidade conjunta.

O CRF utiliza informação contextual para ampliar a quantidade de informações para prever um determinado rótulo. Para isso, utiliza-se também dos rótulos de observações anteriores na sequência. Em teoria, o CRF pode ter quantas conexões forem necessárias. Porém, a maneira mais simples e comum de representação do grafo do CRF é aquela em que os vértices que representam os estados formam uma cadeia simples de primeira-ordem [Weijun FU 2009].

Como o CRF recebe como entrada uma sequência de dados e utiliza informações contextuais com relação a observações anteriores, *Feature Functions* são utilizadas para prever o rótulo de determinada observação. Elas têm como objetivo representar alguma característica da observação com relação à sequência de dados. No treinamento, quaisquer *Feature Functions* que sejam pertinentes podem ser utilizadas, o que flexibiliza o modelo. Para determinar a probabilidade de determinado estado dada uma observação, utiliza-se como base a função F, descrita por:

$$F(X, y_{i-1}, y_i, i) = \sum_{i=1}^{n} w_i f_i(X, y_{i-1}, y_i, i) \tag{1}$$

em que X é o conjunto dos dados de entrada, i a posição da observação sendo analisada no momento, y_{i-1} o último estado e y_i o estado atual, f_i é uma *Feature Function* e w_i é a ponderação relativa a ela. A ponderação de cada um desses valores é o peso relativo que se estabelece para determinar a importância de cada função. A probabilidade P de y um determinado estado y dada uma observação x é determinada por

$$P\left(y \mid x\right) = \frac{1}{Z(x)} e^{\sum_{i=1}^{n} F(X, y_{i-1}, y_i, i)} \tag{2}$$

em que Z(x) é o fator de normalização.

O treinamento de um modelo CRF segue a seguinte sequência de etapas: definição das *Feature Functions*, inicialização dos pesos de cada *Feature Function* para valores aleatórios e uso de algoritmo, geralmente Gradient Descent, para convergir os valores dos pesos de acordo com os dados de treinamento. Neste artigo utilizamos o CRF para modelar o problema de extração de informação de notas fiscais. O objetivo é identificar as *features* (ou *feature functions*, na definição formal) que mais ajudam na tarefa.

2.3. Trabalhos Correlatos

Existem diversos trabalhos em torno de análises de documentos fiscais a fim de fiscalizar e evitar fraudes, que diferenciam-se tanto em técnica quanto em tipos de documentos. No geral, são trabalhos que focam na detecção de fraudes bancárias ou transações empresariais, não necessariamente utilizando dados de notas fiscais, mas, sim, dados de contabilidade geral.

No âmbito governamental, o Tribunal de Contas da União (TCU) destaca-se no desenvolvimento de projetos utilizando Ciência de Dados e Inteligência Artificial para

colaborar na prevenção e detecção de fraude e corrupção[Veras Carvalho Menezes 2022], tais como: ALICE (Analisador de Licitações, Contratos e Editais); MONICA (Monitoramento Integrado para o Controle de Aquisições); SOFIA (Sistema de Orientação sobre Fatos e Indícios para o Auditor); ADELE (Análise da Disputa em Licitações Eletrônicas).

ALICE [Veras Carvalho Menezes 2022], entre outros sistemas, busca auxiliar os órgãos competentes no controle das contas públicas. Segundo o TCU (2017) (apud [Veras Carvalho Menezes 2022]), o sistema ALICE 'tem possibilitado a avaliação tempestiva e automatizada de editais de licitação e atas de pregão, com a identificação de indícios de irregularidades, fraudes, desvios e desperdícios de recursos públicos, possibilitando ações de controle mais eficientes e efetivas'. O sistema ALICE é aplicado sobre uma quantidade massiva de dados de naturezas diferentes, inclusive produtos hospitalares. A autora discute a necessidade de conseguir estruturar esses dados. O objetivo deste artigo é auxiliar este tipo de demanda, podendo melhorar a atuação de sistemas como o ALICE e ampliá-los para novas áreas.

Com o propósito de averiguar produtos com sobrepreços e facilitar o controle e prevenção de despesas públicas indevidas, [Pereira 2020] propõe redes heterogêneas para classificação de produtos em notas fiscais eletrônicas de compras públicas. Esse trabalho apresenta um objetivo similar ao nosso, contudo, emprega técnicas diferentes: *word embedding*, redes neurais, modelagem do conjunto de dados em uma rede bipartida, e utiliza um algoritmo indutivo supervisionado baseado em redes heterogêneas bipartidas para classificação de texto. Trabalhos como este nos ajudam a ampliar o escopo do problema, bem como demonstra outros métodos de tentar resolvê-lo. É possível então cogitar que a união das estratégias aqui propostas às colocadas pelo autor possa propiciar resultados ainda melhores.

Considerando outras aplicações de *Sequence Labeling*, a Conference on Computational Natural Language Learning (CoNLL) é uma conferência anual que promove a pesquisa em aprendizado de máquina para processamento de linguagem natural. Em 2003 foram abordadas metodologias de avaliação para a tarefa de *Named Entity Recognition* [Tjong Kim Sang and De Meulder 2003], utilizando diferentes técnicas como modelo de entropia máxima e Hidden Markov Models. Embora não seja exatamente a mesma tarefa que realizamos, é possível encontrar semelhanças nos métodos empregados, como a utilização de arquétipos para a anotação de dados (Seção 3.1) e separação das features em grupos (Seção 3.3).

A Yet Another Sequence Tagger (YASET) é uma ferramenta de *Sequence Labeling* projetada para textos de biomedicina. Tourille et al. [Tourille et al. 2018] apresentam a ferramenta, que é avaliada em tarefas de *part-of-speech* e *Named Entity Recognition*. O algoritmo é construído em múltiplas camadas, misturando redes neurais com Conditional Random Fields (CRFs), e utilizando diferentes estratégias de construção de *features* (como *word embedding*, por exemplo) e otimização do modelo. Apesar da diferença de complexidade entre o modelo aqui proposto, podemos verificar que novamente há semelhanças em estratégias empregadas, como a utilização de arquétipos para as anotações.

3. Metodologia

O problema que abordamos baseia-se em tentar classificar partes de descrições de produtos médicos em notas fiscais em 7 categorias distintas, sem interseção. Além disso, as descrições não necessariamente contêm todas as categorias distribuídas em suas partes. As categorias consideradas são:

1. Produto: produto vendido;
2. Disposição: como o produto está disposto (comprimido, cápsula, solução...);
3. Embalagem: como o produto é embalado (caixa, ampola);
4. Quantidade: quantos produtos foram vendidos;
5. Concentração: concentração do produto (em miligramas, porcentagem...);
6. Conteúdo: referente ao peso/quantidade/volume do produto;
7. Adicional: adicional que acompanha o produto (seringas e copos, por exemplo).

Para o treinamento do modelo, selecionamos aleatoriamente um conjunto de descrições e realizamos suas respectivas anotações (Seção 3.1). Em seguida, construímos um modelo de CRF para realizar a tarefa de *Sequence Labeling* (Seção 3.2), modelamos as *features* (Seção 3.3) e otimizamos os hiper-parâmetros do modelo utilizando o conjunto de todas as *features*. Com o modelo completo pronto, construímos 5 novos modelos variando os conjuntos de *features* – mantendo os mesmos hiper-parâmetros definidos na etapa anterior. Desta forma, cada modelo pôde ser avaliado (Seção 3.4) individualmente e posteriormente comparado com os demais. Por fim, avaliamos como diferentes *features* impactam na qualidade dos modelos.

3.1. Origem e processamento dos dados

Os dados das descrições de notas fiscais eletrônicas foram disponibilizados no escopo do projeto 1776 PROMAT/UFAL-SEFAZ-PI, desenvolvido pelo LED/UFAL. A base de dados inclui 1.000.506 descrições de itens de notas fiscais eletrônicas referentes à venda de medicamentos ao longo do ano de 2021.

A partir dos dados descritos acima, selecionamos aleatoriamente 200 descrições e manualmente realizamos as anotações com a ferramenta Doccano[1]. Existem alguns tipos de arquétipos para codificar os rótulos utilizados em tarefas de anotações de textos, os quais impactam diretamente a qualidade dos modelos de *Sequence Labeling* [Alshammari and Alanazi 2021]. Optamos por utilizar o esquema BILOU, pois, durante o processo de anotação manual, percebemos que as diferentes categorias possuem uma grande variação de comprimento de palavras e de ordem de aparição, e essa modelagem possibilita capturar tais variações com mais facilidade.

Nesta codificação BILOU são adicionados prefixos para cada uma das 7 categorias originais, sendo eles: B (beginning): para a primeira palavra de uma entidade nomeada; I (inside): para palavras dentro de uma entidade nomeada; L (last): para a última palavra de uma entidade nomeada; O (outside): para palavras que não fazem parte de uma entidade nomeada; U (unit): para entidades nomeadas que consistem em uma única palavra.

[1]https://doccano.github.io/doccano/

3.2. Algoritmo de Sequence Labeling

Para podermos usufruir de algoritmos de validação cruzada e otimização de hiper-parâmetros disponíveis pela biblioteca Scikit-Learn[2], usamos a implementação do CRF da biblioteca sklearn-crfsuite[3]. Dentre os parâmetros do modelo, ajustamos os seguintes:

1. Algoritmo:
    - **lbfgs**: Gradient descent utilizando o método L-BFGS;
    - **l2sgd**: Stochastic Gradient Descent com regularização L2;
    - **ap**: Averaged Perceptron;
    - **pa**: Passive Aggressive;
    - **arow**: Adaptive Regularization Of Weight Vector (AROW).
2. Frequência mínima (float): se maior que 0, define a frequência de corte de uma *feature*, ou seja, irá ignorar as *features* com uma frequência menor que a definida;
3. Todas possíveis transações (bool): se verdadeiro, o modelo gerará *features* de transição que não ocorrem nos dados.
4. C1 (float): coeficiente da regularização Lasso (L1). É útil para os casos em que existem muitas *features* diferentes. Ajuda a identificar as *features* mais importantes e a reduzir o *overfitting*;
5. C2 (float): coeficiente da regularização Ridge (L2). Também é útil para quando existem muitas *features*. Aumenta a probabilidade do modelo ter pequenos coeficientes para todas as *features* em vez de alguns poucos com altos coeficientes. Dessa forma, o modelo torna-se mais robusto e melhora-se a sua capacidade de generalização.

Além dos parâmetros fornecidos pela biblioteca, criamos um que atua diretamente com as *features*: tamanho da janela. Ele varia entre 1, 3 e 5, e diz respeito a quanto cada um dos tokens[4] irá agregar informação dos tokens vizinhos. Caso 1, será considerado apenas o token em si, caso 3, serão agregadas as *features* referentes aos tokens posterior e anterior, etc. Assim sendo, extraímos as *features* de cada um dos tokens (Seção 3.3) e separamos os dados em 80% para treino e 20% para teste.

Após modelarmos as *features*, geramos um modelo inicial utilizando o conjunto completo das *features* e otimizamos os hiper-parâmetros. Para tal, aplicamos o algoritmo GridSearchCV da biblioteca Scikit-Learn para avaliar todos os parâmetros citados e encontrar o que produz melhores resultados. Assim, obtivemos o seguinte: Algoritmo: lbfgs; Frequência mínima: 0; Todas possíveis transações: Verdadeiro; C1: 0.3; C2: 0.5; Tamanho da janela: 3. Com os parâmetros definidos, retreinamos o modelo inicial e geramos 4 novos, variando o conjunto de *features* entre eles e mantendo o mesmo conjunto de dados.

3.3. Conjuntos de Features

Para identificar os tipos de *features* que mais contribuem com a tarefa, agrupamos as *features* nos seguintes conjuntos:

1. Geral:

[2]https://scikit-learn.org/

[3]https://sklearn-crfsuite.readthedocs.io/

[4]Entende-se por token cada parte de uma descrição

- Bias, para lidar com a tendência geral do modelo, que ocorre quando algumas categorias/rótulos ocorrem com mais frequência que outros. Essa *feature* captura a probabilidade geral de cada categoria;
- o token em si em letra minúscula;
- adiciona-se a *feature* CDD caso o token esteja no começo da descrição;
- adiciona-se a *feature* FDD caso o token esteja no fim da descrição;
- posição relativa do token na descrição (posição do token divido pela quantidade de tokens na descrição).

2. Dicionário:
   - se está ou não no dicionário de concentração/conteúdo (MG, ML, %, UI);
   - se está ou não no dicionário de disposição (COMP, REV, CP, CAP, SOL, FR, INJ, SF);
   - se está ou não no dicionário de quantidade (C/, UN);
   - se está ou não no dicionario de embalagem (CX, AMP, BL).
3. Morfologia - Estrutura:
   - os dois primeiros caracteres do token (prefixo);
   - os três primeiros caracteres do token (prefixo);
   - os três últimos caracteres do token (sufixo);
   - se o token é ou não uma palavra composta (contém '-');
   - quantidade de caracteres do token.
4. Morfologia - Apresentação:
   - se possui ou não números;
   - se o token só contém números;
   - quantidade de números;
   - quantidade de vogais;
   - quantidade de caracteres especiais (!, @, #, $,%,&,*,(,),-,_,+,=,/, \);
   - quantidade relativa[5] de números;
   - quantidade relativa de vogais;
   - quantidade relativa de caracteres especiais.

Deste modo, podemos facilmente treinar modelos utilizando diferentes conjuntos de *features* e analisar o quão eficaz e eficiente é cada um.

3.4. Testes implementados e avaliação da qualidade

Para identificar os melhores conjuntos de *features*, geramos seis modelos ao todo: um com todas as possíveis *features*, um para cada um dos quatro grupos de *features* e o último com os dois grupos de maior destaque (os que obtiveram melhores resultados nos seus respectivos modelos). Avaliamos cada modelo de três formas:

1. A partir da biblioteca SEQeval[6], uma ferramenta de avaliação de qualidade de *Sequence Labeling* com suporte a diferentes arquétipos de codificação de rótulos;
2. Validação cruzada implementada pela biblioteca Scikit-Learn;
3. Investigando o que o modelo aprendeu através da biblioteca ELI5[7], que possui ferramentas úteis para interpretar modelos complexos e identificar as características mais importantes para cada modelo.

[5]Calcula-se a quantidade do item em questão dividido pela quantidade de caracteres no token.
[6]https://github.com/chakki-works/seqeval
[7]https://eli5.readthedocs.io/

O SEQeval dispõe de dois modos de avaliação: padrão e estrito. Eles diferenciam-se na maneira como calculam as métricas: no primeiro, basta acertar a categoria do token para validar como correta a previsão do modelo, e, no segundo, valida-se a previsão como correta apenas quando se acerta a categoria e o prefixo do arquétipo. Para o BILOU, apenas o modo estrito está disponível, portanto esse foi o modo utilizado. Para cada modelo avaliado pelo SEQeval, são calculadas a acurácia geral do modelo e a média simples da precisão, do recall e do F1-score para cada uma das sete categorias (ou seja, é calculado para cada uma das variantes[8] de cada categoria e apresentada a média simples dessa categoria). Para cada métrica, são apresentadas três médias: micro, macro e a com peso. Nossa avaliação foi baseada na média macro, a qual foi escolhida por tratar todas as categorias igualmente, independente do suporte de cada uma. Quanto às métricas utilizadas, podemos interpretá-las da seguinte forma:

- Acurácia: mede a proporção de previsões corretas em relação ao total de previsões feitas;
- Precisão: mede a proporção de verdadeiros positivos em relação ao total de classificações positivas realizadas;
- Recall: mede a proporção de verdadeiros positivos em relação ao número total de tokens positivos;
- F1-Score: um modo de resumir as métricas precisão e recall a uma só, equivalente à média harmônica entre elas;
- Suporte: quantidade de observações.

A validação cruzada foi realizada entre 10 subconjuntos dos dados de teste por meio do K-fold, no qual avaliamos a acurácia e o desvio padrão de cada modelo. Na parte da análise do que cada modelo aprendeu, observamos quais as *features* tiveram maior impacto e as probabilidades de transições entre diferentes categorias.

Portanto cada uma das formas servirá para salientar diferentes aspectos dos modelos: o SEQeval permitirá avaliar o desempenho para cada categoria; a validação cruzada viabilizará a comparação dos modelos como um todo, uma vez que testará cada modelo com diferentes combinações de descrições para treino e teste, calculando uma acurácia média; e com o auxílio do ELI5 poderemos inspecionar a estrutura dos modelos, como os coeficientes de cada *feature*.

4. Resultados e discussão

A Tabela 1 mostra os resultados obtidos pela biblioteca SEQeval. As linhas descrevem cada tipo de métrica e as colunas organizam os modelos construídos.

Table 1. Médias macro das categorias dos modelos avaliados pelo SEQeval.

SEQeval	Todas features	Geral	Dicionário	Morf. Estrutura	Morf. Apresentação	Morfologias
Acurácia	0.895	0.695	0.68	0.8	0.815	0.87
Precisão	0.95	0.72	0.72	0.82	0.7	0.85
Recall	0.79	0.59	0.63	0.7	0.65	0.76
F1-Score	0.84	0.6	0.64	0.73	0.67	0.79

A Tabela 2 mostra os resultados obtidos pela avaliação cruzada dos modelos. As linhas descrevem cada tipo de métrica e as colunas organizam os modelos construídos.

[8]Cada categoria vira cinco no arquétipo BILOU, levando uma das letras como prefixo.

Table 2. Médias das acurácias e o desvio padrão dos modelos avaliados pela validação cruzada.

CV	Todas features	Geral	Dicionário	Morf. Estrutura	Morf. Apresentação	Morfologias
Acurácia	0.836	0.649	0.656	0.747	0.772	0.81
Desvio Padrão	0.046	0.046	0.06	0.05	0.03	0.048

A Tabela 3 mostra os resultados para cada categoria do modelo com todas as *features* obtidos pela biblioteca SEQeval. As linhas descrevem cada tipo de métrica e as colunas organizam as categorias.

Table 3. Resultados das métricas para cada categoria no modelo completo com todas as features.

Todas features	Produto	Disposição	Embalagem	Quantidade	Concentração	Conteúdo	Adicional
Precisão	0.93	0.93	1.00	0.88	0.93	1.00	1.00
Recall	0.93	0.90	0.88	0.91	1.00	0.57	0.38
F1-Score	0.93	0.91	0.93	0.89	0.96	0.73	0.55
Suporte	43	29	8	23	37	7	8

Podemos perceber que o modelo com apenas as *features* dos dois conjuntos de morfologia alcançaram acurácia muito próxima à do modelo com todas as *features* (menos de 3% na média), indicando que elas produzem uma maior explicabilidade para as categorias dos tokens, diferentemente dos conjuntos Geral e Dicionário, com médias de diferença em torno de 18%.

Analisamos, então, qual(is) *feature(s)* de cada conjunto têm maior importância para o modelo com o uso a biblioteca ELI5 como apoio:

- Geral: o token em si;
- Dicionário: se está ou não no dicionário de concentração/conteúdo;
- Morfologia - Estrutura: os três primeiros caracteres do token (prefixo)/os três últimos caracteres do token (sufixo);
- Morfologia - Apresentação: se possui ou não números/quantidade relativa de números.

Quando verificamos o modelo com todas as *features*, é possível observar como as importâncias delas se sobressaem entre si (listado top *features* por ordem de relevância), e, mais importante: como *features* de conjuntos que não se saíram tão bem (geral e dicionário) são altamente relevantes para o modelo completo. Sugerindo que, apesar de isoladas não obterem bons resultados, servem de direcionamento para o modelo, possivelmente acelerando a tomada de decisão e resolvendo potenciais ambiguidades na classificação, resultando numa pequena melhoria da acurácia média (3%):

- se está ou não no dicionário de concentração/conteúdo;
- se possui ou não números;
- se está ou não no dicionário de disposição;
- quantidade de vogais;
- os três primeiros caracteres do token (prefixo);
- se está ou não no dicionário de embalagem;
- CDD caso o token esteja no começo da descrição.

Neste modelo examinamos também o desempenho em cada categoria (tabela 3). As categorias Embalagem, Conteúdo e Adicional tiveram uma precisão total, todavia as duas últimas tiveram um Recall muito baixo. Isso demonstra que, apesar de todos os tokens que foram classificados nestas categorias eram, de fato, destas categorias, houve muitos tokens que deveriam ser classificados nelas mas não os foram. Estes resultados sugerem que o modelo teve dificuldade em generalizar características destas categorias, talvez devido a baixa quantidade de observações.

Por fim, a análise das probabilidades de transições entre categorias reforça que os modelos conseguiram generalizar uma relação de ordem de aparição entre as diferentes categorias. Por exemplo, é comum um produto ser seguido por uma concentração, conquanto é bem pouco provável que ocorra uma concentração seguida por quantidade.

5. Conclusão

Este artigo avaliou os melhores tipos de features para serem usados em tarefas de extração de informação de notas fiscais eletrônicas brasileiras. Os resultados obtidos e discutidos acima podem ser usados como base para diversos trabalhos que demandem a estruturação do conteúdo de documentos fiscais.

References

Alshammari, N. and Alanazi, S. (2021). The impact of using different annotation schemes on named entity recognition. *Egyptian Informatics Journal*, 22(3):295–302.

Cortez, E., Oliveira, D., da Silva, A. S., de Moura, E. S., and Laender, A. H. F. (2011). Joint unsupervised structure discovery and information extraction. In *Proc. SIGMOD*.

F. N. de Oliveira, L. P. G. d. S. (2020). Estratégias para combater a sonegação fiscal: Um modelo para o icms baseado em redes neurais artificiais. *Revista de Gestão, Finanças e Contabilidade*, 10:42–64.

John Lafferty, Andrew McCallum, F. C. P. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. *Proc. ICML*.

Pereira, R. d. S. (2020). Redes heterogêneas para classificação de produtos em notas fiscais eletrônicas de compras públicas [tcc]. *CGU*.

Seymore, K. and Rosenfeld, R. (1999). Learning hidden markov model structure for information extraction.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. Conference on Natural Language Learning at HLT-NAACL*.

Tourille, J., Doutreligne, M., Ferret, O., Névéol, A., Paris, N., and Tannier, X. (2018). Evaluation of a sequence tagging tool for biomedical texts. In *Proc. International Workshop on Health Text Mining and Information Analysis*.

Veras Carvalho Menezes, A. P. (2022). Inteligência artificial para identificação de indícios de fraude e corrupção em compras públicas no tcu. *Revista Debates em Administração Pública – REDAP*, 3(2).

Weijun FU, L. L. (2009). A method and application of automatic term extraction using conditional random fields. *Proc. NLP-KE*.