UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Uso de entidades nomeadas para reconhecimento de marcas em descrições de produtos

Trabalho de Conclusão de Curso

Ian Luan Fontes de Oliveira

São Cristóvão – Sergipe

2022

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

DEPARTAMENTO DE COMPUTAÇÃO

Ian Luan Fontes de Oliveira

Uso de entidades nomeadas para reconhecimento de marcas em descrições de produtos

Trabalho de Conclusão de Curso submetido ao Departamento de Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador(a): Thiago Dias Bispo
Coorientador(a): Leonardo Nogueira Matos

São Cristóvão – Sergipe

2022

Resumo

Os comparadores de preço são *softwares* que fornecem informações sobre produtos e preços, possibilitando buscas e comparações, a fim de facilitar a decisão sobre o melhor local para realizar as compras, atuando como auxiliadores nesse processo. Um desses comparadores é o LudiiPrice, um aplicativo para dispositivos móveis que permite a busca, a comparação e o monitoramento de preços de produtos, cuja base de dados é populada através de colaboração, onde os usuários fornecem os dados a partir da leitura do *QR Code* das notas fiscais e através de descrições de produtos extraídos na internet. O objetivo deste trabalho é treinar um modelo de reconhecimento de entidades nomeadas para extrair as marcas em descrições de produtos, visando adicionar ao LudiiPrice uma maior confiabilidade na identificação dos produtos, melhorando os processos de recomendação e comparação deles. Para isso, foi criada uma base de dados contendo 16112 produtos, extraídos de três *e-commerces* diferentes. Ela foi utilizada para treinar e avaliar um modelo de aprendizado de máquina, desenvolvido para extrair as marcas de descrições de produtos, com o qual obtivemos a *f1-score* de 91% para o reconhecimento do início da marca, 77% para o do meio da marca e 82% para o do final da marca. Além disso, foi desenvolvido o novo sistema de sugestão de produtos que utiliza as entidades nomeadas além da similaridade por cosseno, obtendo uma taxa de acerto de 94%, 17.1% maior que a do sistema anterior, que utilizava apenas a similaridade por cosseno.

Palavras-chave: Reconhecimento de entidades nomeadas. Reconhecimento de marcas. Aprendizado profundo. Processamento de linguagem natural

Abstract

Price comparators are softwares that provide information about products and prices, enabling searches and comparisons, in order to ease the decision of the best place to shop, acting as helpers in this process. One of those comparators is Ludiiprice, an application for mobile devices that allows search, comparison and monitoring of product prices, that has its database populated through users' collaboration, in which the users provide the data from the invoices QR Code reading and through the description of products extracted from the internet. This paper's goal is to train a named entity recognition model to extract the brands on the products' description, with the goal of increasing the product identification confiability in LudiiPrice, improving the recommendation and comparison processes between products. To achieve this, we built a dataset of 16112 products, extracted from three different e-commerces. It was used to train and evaluate a machine learning model, designed to extract the brands of product descriptions, with which we obtained the f1-score of 91% for the recognition of the beginning of the brands, 77% for the middle of the brands and 82% for the end of the brands. Furthermore, we have developed a new product suggestion system that uses the named entities in addition to cosine similarity, obtaining a hit hate 94%, which is 17.1% higher than that of the previous system, which used only cosine similarity.

Keywords: *Named entity recognition. Brand recognition. Deep learning. Natural language processing.*

Lista de ilustrações

Figura 1 – Redes neurais recorrentes X *feed-forward* 14
Figura 2 – Matriz de confusão 14
Figura 3 – Etapas da metodologia 18
Figura 4 – Palavras que formam um produto dispostas em linha 19
Figura 5 – Features e a redução da F1 score ao remover a feature 20
Figura 6 – OpenTag Architecture: BiLSTM-CRF with Attention 22
Figura 7 – Explicação para a decisão das rotulagens 23
Figura 8 – Comparação de desempenho entre o BiLSTM, BiLSTM-CRF e *Open Tag* . 24
Figura 9 – Trecho da base de dados criada 25
Figura 10 – Modelo criado no banco de dados do LudiiPrice 29
Figura 11 – Documentação do *endpoint* para obter produto recomendado 30
Figura 12 – Similaridade por cosseno dos pares de produtos 40

Lista de tabelas

Tabela 1 – Quantidade de trabalhos encontrados por base de pesquisa 17
Tabela 2 – Comparação entre CRF e o SP 21
Tabela 3 – Resultados do modelo para cada tag 31
Tabela 4 – Resultados do modelo para identificação da marca completa 31
Tabela 5 – Comparação entre os resultados do modelo desenvolvido e do modelo *baseline* 32
Tabela 6 – Resultados do extrator de marcas utilizando 10-fold cross-validation 32
Tabela 7 – Quantidade de acertos e erros na sugestão de produtos por sistema de sugestão 33
Tabela 8 – Resultados do extrator de marcas utilizando a técnica de bigramas (SKIP-GRAM) 39
Tabela 9 – Resultados do extrator de marcas utilizando a técnica de bigramas (CBOW) 39

Lista de abreviaturas e siglas

REN	Reconhecimento de entidades nomeadas
CRF	*Conditional Random Fields*
LSTM	*Long Short-Term Memory*
BiLSTM	*Bidirectional Long Short-Term Memory*
SP	*Structured Perceptron*
RNN	Redes neurais recorrentes
NF-e	Nota fiscal eletrônica
PLN	Processamento de linguagem natural
API	*Application Programming Interface*
TCC	Trabalho de conclusão de curso

Sumário

1 Introdução 9
1.1 Objetivos 10
1.1.1 Objetivo geral 10
1.1.2 Objetivos específicos 10
1.2 Metodologia 10
1.2.1 Criação da base de dados 10
1.2.2 Treinamento do modelo 11
1.2.3 Construção do novo sistema de sugestão de produtos 11
1.3 Estrutura do Documento 11

2 Fundamentação Teórica 12
2.1 Reconhecimento de Entidades Nomeadas 12
2.1.1 Conditional Random Fields (CRF) 12
2.1.2 Long Short-Term Memory (LSTM) 13
2.2 Métricas 14
2.2.1 *Accuracy* 15
2.2.2 *Precision* 15
2.2.3 *Recall* 15
2.2.4 *F1-score* 16

3 Trabalhos Relacionados 17
3.1 Trabalhos Científicos 17
3.1.1 Reconhecimento de entidades nomeadas em itens de produto da nota fiscal eletrônica 18
3.1.1.1 Limpeza e Tratamento de Dados 18
3.1.1.2 Amostragem 18
3.1.1.3 Classificação Manual 19
3.1.1.4 Treinamento e validação 19
3.1.1.5 Resultados 19
3.1.2 *Attribute Extraction from Product Titles in eCommerce* 19
3.1.2.1 Rotulagem de Sequência 20
3.1.2.2 *Features* 20
3.1.2.3 Pós Processamento 21
3.1.2.4 Resultados 21
3.1.3 *OpenTag: Open Attribute Value Extraction from Product Profiles* 21
3.1.3.1 Rotulagem de Sequência 21

3.1.3.2 Arquitetura . . . 22
3.1.3.3 Mecanismo de Atenção . . . 22
3.1.3.4 Resultados . . . 22

4 Método . . . 25
4.1 Criação da base de dados . . . 25
4.1.1 *Web scrapping* . . . 26
4.1.2 Rotulação manual . . . 27
4.1.3 Rotulação da sequência . . . 27
4.2 Extração de marcas . . . 27
4.2.1 Validação com *K-fold cross-validation* . . . 28
4.3 Melhoria na sugestão de produtos utilizando entidades nomeadas . . . 28
4.3.1 População do banco de dados . . . 28
4.3.2 Filtro por entidades nomeadas . . . 29
4.3.3 Construção da API . . . 30

5 Resultados Obtidos . . . 31
5.1 Resultados do modelo extrator de marcas . . . 31
5.2 Resultados do extrator de marcas com *K-fold cross-validation* . . . 32
5.3 Melhoria na sugestão de produtos utilizando entidades nomeadas . . . 33

6 Considerações Finais . . . 35

Referências . . . 36

Apêndices 37

APÊNDICE A Experimentos . . . 38
A.1 Experimento do extrator de marcas com bigramas . . . 38
A.2 Experimento de igualdade de produtos . . . 39

1 Introdução

A decisão sobre o local ideal para realizar as compras não é uma tarefa fácil. Isso porque vários fatores são levados em consideração: o tamanho da loja, a disponibilidade de produtos, as filas, se é varejo ou atacado, a localização e, o mais importante, o preço. Um levantamento feito pela PROTESTE com diversos supermercados por todo o país mostra que é possível fazer economia de até 29% ao optar pelo estabelecimento com melhor preço (PROTESTE, 2015). Visando facilitar essa escolha, foram desenvolvidos os comparadores de preço, que são *softwares* de busca e comparação que fornecem informações sobre produto e sobre preço, de forma rápida e fácil. Eles funcionam como auxiliadores de compras e podem aumentar o número de locais de compra considerados, ao mesmo tempo em que reduz significativamente o tempo de pesquisa e os custos (PASSYN; DIRIKER; SETTLE, 2013).

Um desses comparadores é o LudiiPrice, um aplicativo para dispositivos móveis desenvolvido por dois alunos de TCC em 2020, ambos orientados pelos mesmos orientadores deste trabalho. Desde então, o aplicativo continua sendo aprimorado através de outros TCC. No LudiiPrice é possível buscar, comparar e monitorar preços de produtos, e sua base é populada por um método colaborativo, onde os usuários fornecem os dados dos produtos a partir da leitura do *QR Code* das notas fiscais emitidas nas compras. Para manter essa base atualizada, seria necessária uma grande quantidade de usuários colaborando frequentemente e, por isso, passou-se a utilizar, também, os dados obtidos através da extração de descrições e preços de produtos na internet, presentes principalmente nos *e-commerces*.

Como um dos aprimoramentos do LudiiPrice, o aplicativo passou a ser considerado um sistema de recomendação, cuja lógica de comparação de produtos estabelece a semelhança entre suas descrições tão somente através da presença ou ausência de certas sub-palavras, sem nenhum tipo de comparação direta das marcas, unidade de medida e tamanho dos produtos em questão. A consequência disso é que, em alguns casos, produtos que não são iguais acabam sendo considerados semelhantes, e produtos iguais mas com diferenças sutis nas descrições

acabam sendo considerados dissimilares (OLIVEIRA, 2022). Isso pode ser superado através do reconhecimento de elementos existentes nas descrições dos produtos antes da avaliação da semelhança entre os textos, complementando essa comparação. Esses elementos que serão reconhecidos são denominados entidades nomeadas.

Este trabalho busca integrar ao LudiiPrice uma maior confiabilidade na identificação dos produtos, através da utilização de entidades nomeadas para o reconhecimento de marcas em descrições de produto. Conhecer a marca de um item possibilita um entendimento mais completo dele, melhorando a sugestão e a comparação de produtos. Neste documento serão avaliados alguns métodos para reconhecimento de entidades nomeadas (REN) e será detalhado o desenvolvimento de um modelo de aprendizado profundo que identifique as marcas nas descrições dos produtos.

1.1 Objetivos

1.1.1 Objetivo geral

Treinar um modelo de classificação para o reconhecimento de marcas em descrições de produtos. Essa funcionalidade permitirá uma melhoria na recomendação de produtos associados, já que a identificação da marca aumenta a confiabilidade na comparação de produtos.

1.1.2 Objetivos específicos

- Criação de uma base de dados composta pelos títulos de produtos, suas marcas e seus rótulos.
- Treinamento de um modelo para reconhecimento de entidades nomeadas.
- Agregação do extrator de marcas na lógica de sugestão de produtos do LudiiPrice.

1.2 Metodologia

A metodologia utilizada neste trabalho é constituída pelas etapas de criação da base de dados, treinamento do modelo e utilização de entidades nomeadas na lógica de sugestão de produtos do LudiiPrice, descritas nas seções a seguir.

1.2.1 Criação da base de dados

Para a construção da base foram utilizados os títulos de produtos dos *e-commerces* do Atacadão[1], Mercantil Rodrigues[2] e Makro[3], obtidos anteriormente com *web scrapping* de seus

[1] Disponível em http://www.atacadao.com.br.
[2] Disponível em https://www.delivery.mercantilrodrigues.com.br.
[3] Disponível em http://www.delivery.makro.com.br.

websites. A base final é composta pelos títulos dos produtos, as marcas e os rótulos criados com a estratégia BIOE, que será detalhada no decorrer deste trabalho.

1.2.2 Treinamento do modelo

O Modelo utilizado foi desenvolvido baseado no trabalho científico de Zheng et al. (2018), visto que ele obteve bons resultados, explica a metodologia de forma detalhada e possui implementações não oficiais que disponibilizam o código fonte em repositórios públicos, favorecendo a reprodução. Esse trabalho está detalhado na seção 3.1.3, apresentando toda a arquitetura que será adotada no desenvolvimento do modelo aqui treinado.

1.2.3 Construção do novo sistema de sugestão de produtos

Para o novo sistema de sugestão foi construída uma API tal que, dado um certo produto, ela retorna outro produto como sugestão. Diferentemente do antigo sistema que buscava apenas o mais semelhante considerando a descrição do produto, nossa API utiliza as entidades nomeadas de marca, quantidade e unidade como filtros e depois calcula a similaridade, tornando a comparação mais precisa.

1.3 Estrutura do Documento

Este documento está estruturado em capítulos, são eles:

- Capítulo 1 - Introdução: Corresponde ao capítulo atual, que apresenta os objetivos deste trabalho;
- Capítulo 2 - Fundamentação Teórica: Apresenta conceitos importantes para o entendimento do trabalho;
- Capítulo 3 - Trabalhos Relacionados: Cita os trabalhos que possuem objetivos semelhantes ao deste documento, explicando a proposta, as técnicas utilizadas e os resultados obtidos;
- Capítulo 4 - Método: Detalha os métodos utilizados para alcançar os objetivos propostos pelo documento;
- Capítulo 5 - Resultados: Apresenta os resultados obtidos de cada etapa do trabalho;
- Capítulo 6 - Considerações finais: Apresenta as considerações finais e os trabalhos futuros.

2

Fundamentação Teórica

Neste capítulo serão apresentados os principais conceitos necessários ao entendimento das técnicas e motivações deste trabalho.

2.1 Reconhecimento de Entidades Nomeadas

O reconhecimento de entidades nomeadas (REN) é uma sub área de estudo no campo de extração de informação, que tem como objetivo identificar entidades nomeadas e classificá-las dentro de um conjunto de categorias pré-definidas, como pessoa, organização, local e marca (AMARAL; VIEIRA, 2013).

O REN é uma técnica amplamente utilizada em Processamento de Linguagem Natural (PLN), que é uma área de pesquisa e de aplicação que explora como os computadores podem ser usados para processar e manipular texto ou discurso em linguagem natural para fazer coisas úteis. Tradução automática e reconhecimento de voz são exemplos comuns de aplicações de PLN (CHOWDHURY, 2003).

Serão abordadas, nessa dissertação, dois modelos bastante utilizados para o REN, que são *Conditional Random Fields* (CRF) e o *Long Short-Term Memory* (LSTM).

2.1.1 Conditional Random Fields (CRF)

Conditional Random Fields (CRF) é um modelo matemático probabilístico que tem, como objetivo etiquetar e segmentar dados sequenciais. Ou seja, dada uma sentença, o objetivo é atribuir a cada palavra da sentença sua etiqueta adequada, como por exemplo, dizer se a palavra é um substantivo, um artigo, etc. O CRF realiza essa tarefa com base em uma abordagem condicional, que visa etiquetar uma nova sequência de observação x, selecionando a sequência de rótulo y que aumente a probabilidade condicional p(y|x) (AMARAL; VIEIRA, 2013).

Agora, imagine um cenário cuja entrada é um título de um produto e o objetivo é reconhecer a marca no título. Uma abordagem natural seria tratar qualquer palavra como um possível alvo e realizar a rotulação de cada uma. Porém, sofreria dos seguintes problemas. (1) Problema de dimensionamento de rótulo: este método não dimensiona bem com milhares de potenciais valores para qualquer atributo. (2) Mundo fechado: não pode descobrir qualquer novo valor fora do conjunto de rótulos nos dados de treinamento; (3) Independência entre as etiquetas: trata cada atributo-valor independente do outro, assim, ignorando qualquer dependência entre eles (ZHENG et al., 2018). A fim de solucionar esses problemas, entram em cena algumas estratégias de rotulação de sequência, como a BIOE.

BIOE é a estratégia de rotulação de sequência mais popular (ZHENG et al., 2018). Seguindo com o cenário descrito no parágrafo anterior, caso o objetivo fosse encontrar a marca, utilizando a estratégia BIOE, a palavra que representa o início da marca receberia a etiqueta **B**, as palavras que estão dentro da marca (em casos de marcas compostas) receberiam o **I**, as palavras que não fazem parte da marca receberiam o **O**, e a palavra que representa o final da marca receberia o **E**. Portanto, teríamos para cada entrada a seguir as seguintes saídas:

entrada 1: Café solúvel **Santa Clara** 300g forte.

saída 1: OO**BE**OO.

entrada 2: Notebook **Samsung** 15.6 8gb 1TB.

saída 2: O**B**OOO.

entrada 3: Vinho **Quinta do Morgado** 1L tinto.

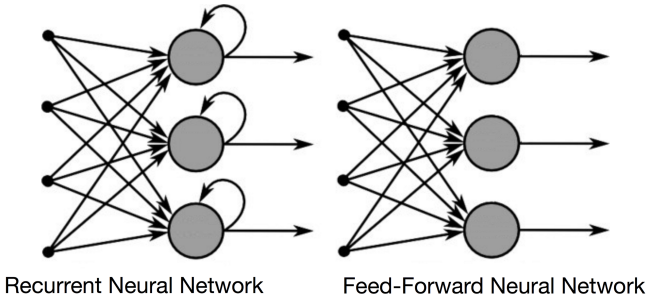
saída 3: O**BIE**OO.

2.1.2 Long Short-Term Memory (LSTM)

Redes neurais recorrentes (RNN) contêm ciclos que alimentam as ativações da rede a partir de uma etapa de tempo anterior. Essas ativações são armazenadas nos estados internos da rede que, em princípio, pode conter informações contextuais temporais de longo prazo. Este mecanismo permite que as RNNs explorem uma janela contextual que muda dinamicamente ao longo do histórico de sequência de entrada, em vez de uma estática como na janela de tamanho fixo usada com redes neurais *feed-forward* (SAK; SENIOR; BEAUFAYS, 2014). Sendo assim, diferente das redes *feed-forward*, as RNNs não são amnésicas em relação ao passado e possuem noção de ordem de tempo.

A *Long Short-Term Memory* (LSTM) é uma arquitetura específica de rede neural recorrente (RNN) que foi projetada para modelar sequências temporais e suas dependências de longo alcance com mais precisão do que as RNNs convencionais. A LSTM contém unidades especiais chamadas blocos de memória na camada oculta recorrente. Os blocos de memória contêm células capazes de armazenar o estado temporal da rede, além de unidades multiplicativas

Figura 1 – Redes neurais recorrentes X *feed-forward*



Deep Learning Book, 2022. Disponível em: <https://www.deeplearningbook.com.br/>. Acesso em: 12 de abril. 2022.

especiais chamadas de *gates* (portões) para controlar o fluxo de informações. Cada bloco de memória na arquitetura original contém um portão de entrada e um portão de saída. O primeiro controla o fluxo de ativações de entrada na célula de memória. O último controla o fluxo de ativações de saída da célula para o resto da rede (SAK; SENIOR; BEAUFAYS, 2014).

LSTMs bidirecionais são uma melhoria em relação ao LSTM, pois capturam tanto as características passadas quanto as do futuro, através de estados para a frente e para trás, respectivamente. Em tarefas de *sequence tagging*, muitas vezes precisamos considerar tanto o contexto esquerdo quanto direito em conjunto para criar um modelo melhor. Assim, dois LSTM's são usados, um com a sequência padrão e o outro com a sequência invertida. A saída final é a concatenação dos dois estados ocultos (ZHENG et al., 2018).

2.2 Métricas

Durante o desenvolvimento de modelos de aprendizado de máquina, faz-se necessário etapas de avaliação, a fim de entender a qualidade do modelo. Essa avaliação se dá com base em diferentes métricas que demonstram o desempenho do modelo em determinadas situações. Algumas das métricas mais comuns em problemas de classificação são *accuracy*, *precision*, *recall* e *f1-score*. O cálculo dessas métricas está relacionado com a matriz de confusão, que é uma tabela que exibe a distribuição dos dados em relação às suas classes reais e suas classes previstas.

Figura 2 – Matriz de confusão

		PREVISTO	
		Sim	**Não**
REAL	**Sim**	Verdadeiro Positivo TP	Falso Negativo FN
	Não	Falso Positivo FP	Verdadeiro Negativo TN

2.2.1 *Accuracy*

A acurácia é a métrica mais simples dentre as quatro citadas: ela é o valor de previsões corretas em relação ao total de previsões.

$$Accuracy = \frac{\text{previsões corretas}}{\text{total de previsões}}$$

Sua fórmula também pode ser derivada da matriz de confusão:

$$Accuracy = \frac{\text{VerdadeirosPositivos} + \text{VerdadeirosNegativos}}{\text{Total}}$$

Deve-se ter cuidado pois a *accuracy* pode causar uma falsa sensação de qualidade do modelo, principalmente se houver um desbalanceamento na quantidade de exemplos de cada classe dentro de um problema de classificação. Por exemplo: se nos dados existirem 90 palavras que não são marcas, representadas pela classe O, e 10 palavras que são marcas, representadas pela marca B, caso o modelo consiga prever corretamente as 90 palavras do tipo O, mas nenhuma do tipo B, a *accuracy* ainda seria de 90%, o que poderia ser visto como um modelo de alta qualidade. Porém, sua capacidade de prever marcas foi de 0%, e se o foco do modelo é identificar as marcas, tem-se um modelo de baixa qualidade.

2.2.2 *Precision*

A *precision* observa o desempenho do modelo em relação à sua taxa de acerto ao afirmar que determinado dado pertence a uma classe. Ou seja, a precisão visa responder a seguinte pergunta: dos registros que o modelo classificou como positivo, quantos desses estavam corretos?

$$Precision = \frac{\text{VerdadeirosPositivos}}{\text{VerdadeirosPositivos} + \text{FalsosPositivos}}$$

Essa métrica é recomendada quando falsos positivos forem mais impactantes do que falsos negativos. Por exemplo, ao classificar um investimento como bom ou ruim, julgar um investimento ruim como bom pode ser pior do que classificar investimentos bons como ruins.

2.2.3 *Recall*

A *recall* mede a habilidade do modelo no reconhecimento de todos os exemplos da base de dados que são positivos. Sua fórmula é a seguinte:

$$Recall = \frac{\text{VerdadeirosPositivos}}{\text{VerdadeirosPositivos} + \text{FalsosNegativos}}$$

Essa métrica é recomendada quando falsos negativos forem mais impactantes do que falsos positivos. Por exemplo, ao classificar se um tumor é maligno ou não, julgar um tumor maligno como não maligno é pior do que classificar um tumor benigno como maligno.

2.2.4 *F1-score*

A *f1-score* é uma média harmônica que busca medir o grau de concordância entre as medidas de *recall* e *precision*. Sua fórmula é a seguinte:

$$\text{F1-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

3

Trabalhos Relacionados

Este capítulo tem como objetivo citar trabalhos com propósitos semelhantes ao deste documento. Aqui, estão apresentados os artigos que compartilham a mesma finalidade: realizar o reconhecimento de atributos em produtos, sendo que a maioria deles tem, como foco, o atributo **marca** e, como *corpus*, o título do produto. As buscas foram efetuadas nas bases da *Scopus*, *Web of Science* e *IEEE*.

Para a pesquisa, foram usadas as seguintes *strings* de busca: "*brand AND (detection OR recognition) AND products*", "(*named AND entity AND e-commerce AND brand*)"e, a que retornou os resultados mais significativos, "*brand AND (extraction OR detection) AND product AND title*". Na Tabela 1 a seguir encontra-se a quantidade de trabalhos retornados por base de busca, utilizando a última *string*.

Tabela 1 – Quantidade de trabalhos encontrados por base de pesquisa

Base	**Quantidade de trabalhos encontrados**
Scopus	10
Web of Science	6
IEEE	1

A seção 3.1, descrita a seguir, apresenta os trabalhos científicos encontrados durante a busca de anterioridade que são mais relevantes para este documento. Nelas, estão discutidas as propostas e técnicas utilizadas por cada trabalho, além dos resultados obtidos por eles.

3.1 Trabalhos Científicos

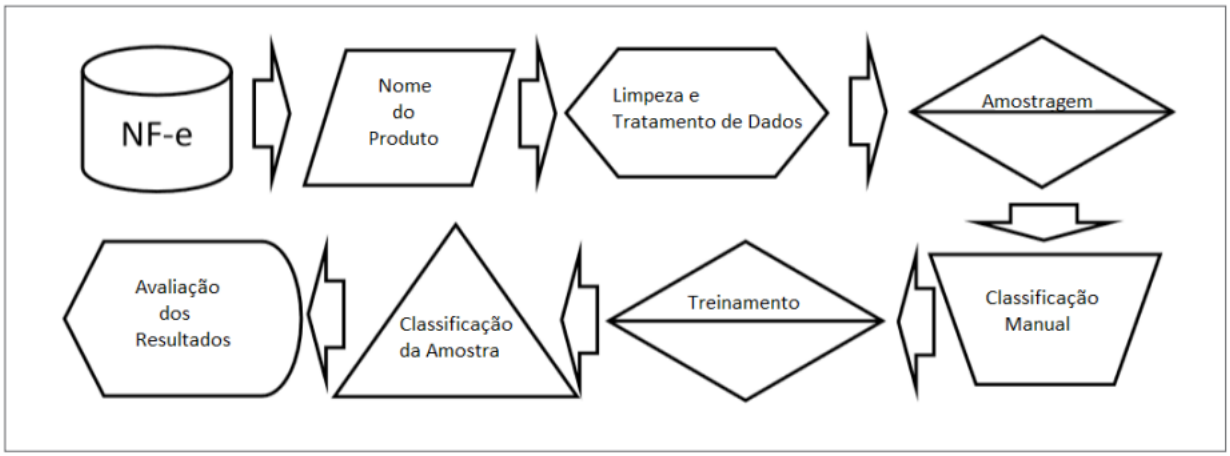
Nesta seção são apresentados três trabalhos científicos. Cada subseção representa um trabalho diferente, detalhando sua metologia, seus objetivos e seus resultados alcançados.

3.1.1 Reconhecimento de entidades nomeadas em itens de produto da nota fiscal eletrônica

Este trabalho tem como objetivo identificar produtos similares para estimar variações anormais de preços, que podem ser indícios de comportamento malicioso ou ilegal na venda de produtos. Além disso, pretende melhorar o aproveitamento das informações contidas em notas fiscais eletrônicas (NF-e). Para resolver esses problemas, utiliza o reconhecimento de entidades nomeadas para processar os descritivos dos produtos.

Em Ribeiro et al. (2018), os autores propõe a utilização de *Conditional Random Field* e uma metodologia em etapas que começa com a coleta de títulos de produtos das notas fiscais, seguida das técnicas de limpeza e tratamento de dados, amostragem, classificação, treinamento e finaliza com avaliação dos resultados. Essas etapas estão detalhadas a seguir.

Figura 3 – Etapas da metodologia



Fonte: Ribeiro et al. (2018)

3.1.1.1 Limpeza e Tratamento de Dados

Após obter os títulos de produtos das notas fiscais, é iniciada a etapa de limpeza e tratamento. Aqui, são realizados alguns processos como a transformação do corpus em caracteres maiúsculos, remoção de acentos, remoção de palavras sem relevância e a separação da expressão relativa a um item de NF-e de forma que a disposição de palavras esteja em linhas, conectadas pela coluna idx, representando o item.

3.1.1.2 Amostragem

Aqui foram gerados conjuntos com 1000 ocorrências de cada um dos seguintes grupos: produtos de limpeza, chocolates, carne de frango, tubos e conexões e um conjunto de 1000 ocorrências aleatórias de produtos.

Figura 4 – Palavras que formam um produto dispostas em linha

Idx	Item de NF-e
0	frango
0	congelado
1	peito
1	frango
2	coxa
2	de
2	frango
3	frango
3	inteiro

Fonte: Ribeiro et al. (2018)

3.1.1.3 Classificação Manual

Cada conjunto de produtos passou por uma subdivisão na qual 10% do conjunto (100 ocorrências) foi encaminhado para que técnicos classificassem as palavras, atribuindo a cada uma delas uma entidade pré-determinada.

3.1.1.4 Treinamento e validação

Para o treinamento, foram usadas duas técnicas: *Hold-out Validation*, em que cada grupo foi separado em 20% dos registros para teste e 80% para treino e *Cross Fold Validation*, em que os grupos foram separados em 10 *folds* e foi feita a validação cruzada com as ocorrências.

3.1.1.5 Resultados

Como resultado, os autores constataram que a técnica CRF funciona melhor quanto mais segmentado for o grupo de produtos a classificar. Por exemplo: a acurácia em Tubos e conexões foi de 96%, com as duas técnicas de treinamento, enquanto que o grupo aleatório obteve 52% com o *Hold-out* e 45% com *Cross Fold*.

3.1.2 *Attribute Extraction from Product Titles in eCommerce*

Este trabalho tem como objetivo a extração de atributos a partir de títulos de produtos, a fim de melhorar a experiência de busca dentro de um *e-commerce*. Ao buscar por um produto em alguma loja *online* geralmente é exibido ao usuário diversos filtros que se alteram de acordo com o produto. Esses filtros podem ser essenciais para ligar o comprador ao produto, visto que o volume de produtos disputando espeço na *web* é enorme.

Em More (2016), os autores relatam que os atributos dos itens não podem ser obrigatórios, pois o cadastro de um item deve ser rápido e prático. Portanto, analisar o título pode coletar

informações cruciais para o processo de busca. Além disso, os atributos também são importantes para veicular campanhas publicitárias e por necessidade de conformidade com a lei. Neste trabalho serão comparadas duas implementações diferentes: uma utilizando *Structured Perceptron* (SP) e a outra *Conditional Random Field*.

3.1.2.1 Rotulagem de Sequência

Os autores utilizam uma abordagem de rotulagem de sequência, a fim de eliminar a necessidade de nomear um atributo para cada palavra do título do produto. Nessa abordagem, cada *token* (palavra) recebe uma etiqueta de um conjunto de etiquetas e o objetivo passa a ser de prever a etiqueta de cada palavra do título.

A estratégia utilizada aqui é a BIO. Nela, cada palavra do título pode receber B, caso seja o início de um atributo, I, caso seja a parte interna de um atributo e O, caso não faça parte de um atributo. Dessa forma, se o atributo buscado fosse marca e o título fosse "Café solúvel Santa Clara 30g forte", a sequência rotulada seria **OOBIOO**.

3.1.2.2 *Features*

Aqui são usados três grupos de *features*: as características, correspondentes à análise do *token* (composição, *letter case*, tamanho), as de localização, relacionadas à posição do *token* (quantidade de *tokens* antes, quantidade de *tokens* depois) e as de contexto, relacionadas à vizinhança (identidade dos *tokens* anteriores e dos seguintes, se o *token* anterior ou próximo é capitalizado e as sequências contendo o *token* atual e o anterior, e o *token* atual e o próximo).

Figura 5 – Features e a redução da F1 score ao remover a feature

Feature function	ΔF_1 (LCCRF)	ΔF_1 (SP)
(w_{-1}, w_0)	0.577	0.675
(w_0, w_1)	0.423	0.443
w_0	0.220	0.181
w_{-1} lemma	0.177	0.079
(w_{-2}, w_{-1})	0.207	0.272
$w_1[0]$ is a digit	0.146	0.120
w_0 consists only of letters	0.142	0.132
First token in title	0.134	0.049
w_{-1}	0.072	0.029
w_0 contains hyphen	0.071	-
$w_0[0]$ is uppercase	0.066	0.027
Number of characters in w_0	0.064	0.096
$w_{-1} =$ by	0.061	-
w_0 lemma	0.056	0.095
i (token position)	0.037	-

Fonte: More (2016)

3.1.2.3 Pós Processamento

Para realizar pós processamento, os autores utilizaram técnicas como dicionários de normalização, *blacklists* e *feedback* manual. Os dicionários de normalização servem para evitar que o atributo se repita nos filtros com formatos diferentes, por exemplo: St. Clara e Santa Clara são a mesma marca, e ter essas duas opções no filtro de marcas, por exemplo, prejudicaria a experiência. Já a *blacklist* tem como principal função a eliminação imediata de termos conhecidos por não ser um atributo. Por fim, os *feedbacks* manuais servem para alimentar os dicionários de normalização. Após um atributo não normalizado se repetir mais de 30 vezes, é enviado para validação manual que decide se entrará para o dicionário de normalização.

3.1.2.4 Resultados

Para o treinamento foi usado um *dataset* com 61,374 títulos de produtos, utilizando *10-fold Cross Validation*. Para medir o desempenho foi utilizado o valor da precisão (em inglês, *precision*) e da cobertura (em inglês, *recall*). A *label Accuracy* é relacionada a quantidade de etiquetas previstas corretamente.

Tabela 2 – Comparação entre CRF e o SP

	Precision (%)	**Recall (%)**	**Label Accuracy (%)**
LCCRF	91.94±0.25	92.21±0.25	98.44±0.04
SP	91.98±0.21	92.18±0.22	98.44±0.03

3.1.3 *OpenTag: Open Attribute Value Extraction from Product Profiles*

Este trabalho tem como objetivo a criação de um extrator de atributos de perfis de produtos em um cenário de *Open World Assumption*, a fim de conseguir extrair atributos sem vocabulários pré-definidos, podendo reconhecer uma marca nova, ou seja, que nunca foi citada anteriormente. Esse trabalho é o mais significativo entre os resultados buscados, pois além de utilizar técnicas de *deep learning*, dispensando configurações de *features* manuais, explica detalhadamente a metologia e possui algumas implementações não oficiais com o código fonte disponibilizado em repositórios públicos, favorecendo a reprodução. Por causa disso, junto à afinidade entre as propostas e aos resultados obtidos (listados a seguir), o Zheng et al. (2018) está sendo usado como *baseline* para o nosso trabalho.

3.1.3.1 Rotulagem de Sequência

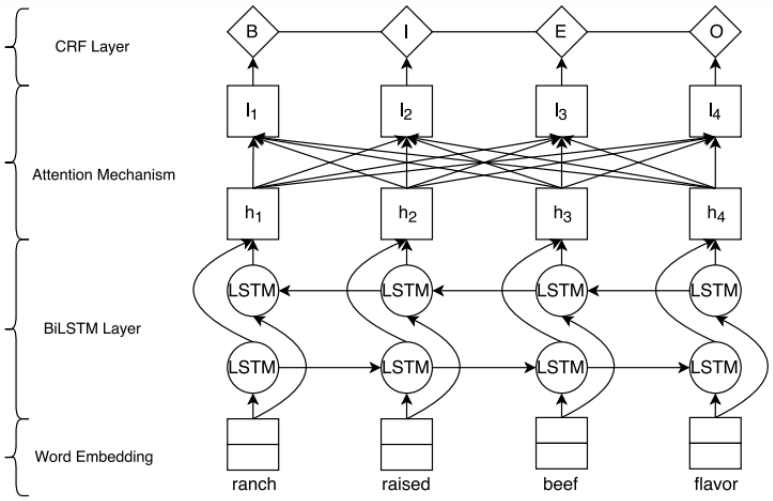
Em Zheng et al. (2018), os autores também utilizam uma abordagem de rotulagem de sequência, semelhante à utilizada em More (2016), descrita na Seção 3.1.2.1. Porém, a estratégia utilizada aqui é a BIOE e não a BIO. Nesta técnica, cada palavra do título pode receber B, caso seja o início de um atributo, I, caso seja a parte interna de um atributo, O, caso não faça parte de um atributo e E, caso seja o final do atributo. Dessa forma, considerando que o atributo de

interesse seja a marca e o título seja "Café solúvel Santa Clara 30g forte", a sequência rotulada seria **OOBEOO**.

3.1.3.2 Arquitetura

Os autores propõem uma arquitetura que tem os *word embeddings* das palavras como entrada para o modelo BiLSTM, seguido por uma cama de Atenção e finalizando com uma camada de CRF.

Figura 6 – OpenTag Architecture: BiLSTM-CRF with Attention



Fonte: Zheng et al. (2018)

3.1.3.3 Mecanismo de Atenção

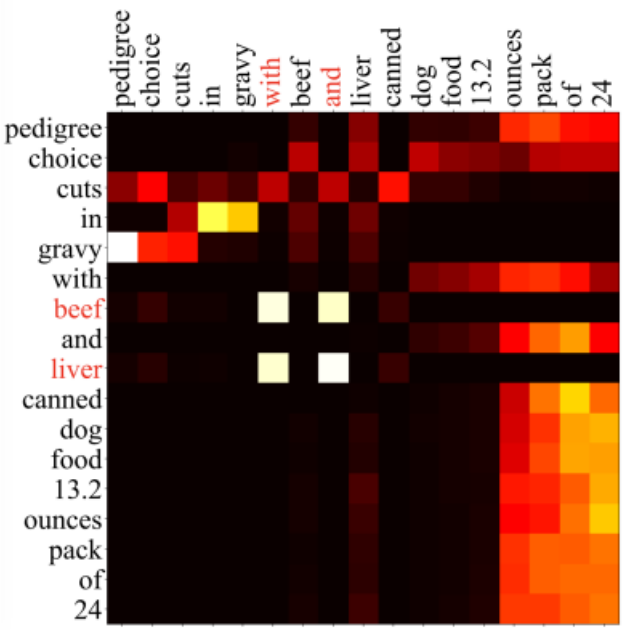
Como proposto por Vaswani et al. (2017), os mecanismos de atenção permitem calcular a influência das palavras vizinhas em relação a um certa palavra da sentença e compor sua representação usando a representação de seus vizinhos. Dentre outras vantagens, a Atenção possibilita um maior entendimento do processo de classificação de modelos de *machine learning*, razão pela qual Zheng et al. (2018) a utilizou.

Os autores descrevem esse mecanismo como uma forma de entender o modelo, limpando a imagem de uma caixa preta. Assim, o objetivo dessa técnica é destacar o que realmente importa para o modelo, filtrando os estados que são passados da camada BiLSTM para o mecanismo de atenção. O mapa de calor exibido na Figura 7 aponta a importância que uma palavra teve para a outra no contexto de vizinhança (quanto mais claro, maior a relação).

3.1.3.4 Resultados

Para a construção do modelo foi utilizada a biblioteca TensorFlow, com o Keras, e o *word embedding* GloVe de 100 dimensões. Os autores realizaram os testes com grupos diferentes

Figura 7 – Explicação para a decisão das rotulagens



Fonte: Zheng et al. (2018)

de produtos e, para cada grupo, buscava um atributo distinto, já que não faz sentido buscar o atributo **sabor** em câmeras, por exemplo. Também foram comparados os desempenhos entre o modelo BiLSTM, o modelo BiLSTM-CRF e o modelo *Open Tag*, sendo o último o modelo vencedor. Os resultados podem ser vistos na Figura 8.

Figura 8 – Comparação de desempenho entre o BiLSTM, BiLSTM-CRF e *Open Tag*

Datasets/Attribute	Models	Precision	Recall	Fscore
Dog Food: Title	BiLSTM	83.5	85.4	84.5
Attribute: Flavor	BiLSTM-CRF	83.8	85.0	84.4
	OpenTag	**86.6**	**85.9**	**86.3**
Camera: Title	BiLSTM	94.7	88.8	91.8
Attribute: Brand	BiLSTM-CRF	91.9	**93.8**	92.9
	OpenTag	**94.9**	93.4	**94.1**
Detergent: Title	BiLSTM	81.3	82.2	81.7
Attribute: Scent	BiLSTM-CRF	**85.1**	82.6	83.8
	OpenTag	84.5	**88.2**	**86.4**
Dog Food: Description	BiLSTM	57.3	58.6	58
Attribute: Flavor	BiLSTM-CRF	62.4	51.5	56.9
	OpenTag	**64.2**	**60.2**	**62.2**
Dog Food: Bullet	BiLSTM	93.2	94.2	93.7
Attribute: Flavor	BiLSTM-CRF	94.3	94.6	94.5
	OpenTag	**95.7**	**95.7**	**95.7**
Dog Food: Title	BiLSTM	71.2	67.4	69.3
Multi Attribute:	BiLSTM-CRF	72.9	67.3	70.1
Brand, Flavor, Capacity	OpenTag	**76.0**	**68.1**	**72.1**

Fonte: Zheng et al. (2018)

4 Método

4.1 Criação da base de dados

Devido à falta de bases de produtos rotulados seguindo o padrão BIOE em português, fez-se necessário a criação de uma base própria. Conforme descrito na Seção 1.2.1, para a construção dessa base foram utilizadas três fontes de dados: as descrições dos produtos do Atacadão[1] e do Mercantil Rodrigues[2], obtidas anteriormente com *web scrapping* de seus *websites* e as descrições dos produtos do Makro[3], também obtidas de seu *website*. A base final é composta pelas descrições dos produtos, as marcas e os rótulos criados com a estratégia BIOE, como pode ser visto na Figura 9 a seguir.

Figura 9 – Trecho da base de dados criada

product	brand	tag
aperitivo agave ouro garrafa tequilero 750ml	tequilero	OOOOBO
flocos de milho domilho - 500g	domilho	OOOBOO
canjica branca siamar - 500g	siamar	OOBOO
flocos para purê de batata romariz - 1kg	romariz	OOOOOBOO
frango inteiro seara assa fácil temperado - 2,3kg	seara	OOBOOOOO
espeto peito frango jundiai - 2kg	jundiai	OOOBOO
meio asa frango da granja bandeja 600g	da granja	OOOBEOO

Além da etapa de *web scrapping*, para criar essa base foram realizadas etapas que consistem em: rotular manualmente os produtos que não possuíam marca e gerar os rótulos de cada descrição de produto com a ténica BIOE. Todas as três etapas estão detalhadas nas subseções a seguir.

[1] Disponível em http://www.atacadao.com.br/.
[2] Disponível em https://www.delivery.mercantilrodrigues.com.br.
[3] Disponível em https://delivery.makro.com.br/.

4.1.1 *Web scrapping*

Para a realização dessa etapa foi utilizada a linguagem de programação Python[4] com a biblioteca Scrapy[5], criada para facilitar esse processo de coleta de dados da internet. Para obter os produtos do Makro foi necessário realizar a extração em três níveis: setores, categorias e produtos. A URL inicial é da página de setores. Lá estão, por exemplo, os setores de bebidas, mercearia e limpeza. Para cada um desses setores existem as categorias como, por exemplo, para o setor de bebidas, as categorias de vinhos, refrigerantes e águas. Por fim, em cada uma dessas categorias existe a lista de produtos. Portanto foi necessário instruir o Scrapy a iterar todos os setores, em cada setor iterar todas a categorias e em cada categoria coletar as informações de todos os produtos. Nos níveis de setor e de categoria os elementos exibidos no site são iguais, portanto foi possível reutilizar o mesmo código para ambos, que pode ser visto no Código 1. O código utilizado para extração das informações dos produtos pode ser observado no Código 2.

Código 1 – Código utilizado para extração dos setores e das categorias

```
1  def parse_departments(self, response):
2      LINK_SELECTOR = '.styles__Li-sc-1l2f31k-2 a::attr(href)'
3      BASE_URL =
           'https://delivery.makro.com.br/produtos/makro-carlos-lisdegno'
4
5      links = response.css(LINK_SELECTOR).getall()
6      links = [BASE_URL + link[1:] for link in links]
7
8      yield from response.follow_all(links, self.parse_categories)
```

Código 2 – Código utilizado para extração das informações dos produtos

```
1  def parse_products(self, response):
2      PRODUCT_SELECTOR =
           '.sc-jSgupP.styles__Container-sc-1qx2rbh-0.gIhqDC.cbHmbg'
3      TITLE_SELECTOR = '.sc-eCssSg.cDIEmL span::text'
4      BRAND_SELECTOR = '.sc-eCssSg.eSrsuJ span::text'
5
6      for product in response.css(PRODUCT_SELECTOR):
7          new_product = {
8              'title':
                   product.css(TITLE_SELECTOR).extract_first().lower(),
9              'brand':
                   product.css(BRAND_SELECTOR).extract_first().lower()
10         }
11
```

[4] Disponível em https://www.python.org.
[5] Disponível em https://scrapy.org.

```
12          if self.is_brand_in_title(new_product['brand'],
                new_product['title']):
13              self.products.append(new_product)
```

4.1.2 Rotulação manual

Durante o processo de extração dos dados do Makro com *web scrapping*, obtivemos as descrições dos produtos contendo a marca e também a marca separada, já que o *website* disponibiliza essas duas informações separadamente. Porém, percebemos que em alguns produtos a marca destacada estava ausente ou com algum erro que tornava-a diferente da marca existente na descrição, o que atrapalharia a rotulação de sequência. Sendo assim, foi necessário ajustar manualmente esses dados adicionando as marcas ausentes ou corrigindo as marcas incorretas para que não atrapalhasse a eficiência do modelo posteriormente.

4.1.3 Rotulação da sequência

Uma vez que foram obtidos os produtos e suas marcas, foram geradas as respectivas sequências rotuladas usando a estratégia BIOE. Para tal, identificamos manualmente a marca dentro do produto e, para cada palavra, demarcamos se consistia do início (B), meio (I) ou final (E) da marca. Todas as demais palavras que não faziam parte da marca foram classificadas com a *tag* O.

4.2 Extração de marcas

O modelo extrator de marcas utilizado é uma implementação não oficial do trabalho de Zheng et al. (2018), disponibilizada no repositório lumiqai/UOI-1806.01264[6]. Após isso, seguindo todas as configurações propostas pelo *baseline*, o modelo foi treinado e avaliado utilizando a base de dados desenvolvida na seção 4.1 e seguindo duas técnicas de divisão do *dataset* entre treino e teste: *hold-out*, com a divisão da base em 80% para treino e 20% para teste e *K-fold cross-validation*, com K=10, através da biblioteca Sklearn[7]. Todo o código relacionado ao modelo que utilizamos como base foi documentado e pode ser acessado através do Google Colab[8]. Nesse Colab também estão disponíveis os *word embeddings* e os *datasets* necessários para o treinamento do modelo, estando assim pronto para uso. A fim de entender o desempenho do modelo em um cenário mais prático com diferentes conjuntos de dados, foi realizado um experimento utilizando a técnica de *K-fold cross-validation*, detalhado na subseção 4.2.1 a seguir:

[6] Disponível em https://github.com/lumiqai/UOI-1806.01264.
[7] Disponível em https://scikit-learn.org/.
[8] Disponível em https://colab.research.google.com/drive/1VbY7nZPm-yp6fEOQWeHv3B1zGRtxArKm?usp=sharing.

4.2.1 Validação com *K-fold cross-validation*

Ainda utilizando a biblioteca Sklearn, o conjunto de dados foi separado em 10 subconjuntos (*folds*) de mesmo tamanho. Após isso, o modelo foi treinado e avaliado dez vezes, sendo que em cada uma delas um subconjunto diferente era utilizado como *dataset* de teste enquanto os outros nove eram utilizados como *dataset* de treino, conforme determina a validação cruzada. Os resultados da avaliação de cada turno foram armazenados e, no final, o desempenho do modelo foi calculado através da média e do desvio padrão.

4.3 Melhoria na sugestão de produtos utilizando entidades nomeadas

Até o momento, o sistema de recomendação de produtos do LudiiPrice consistia na busca de produtos similares por similaridade de cosseno: dado um produto, retornava o produto com maior similaridade entre as descrições (OLIVEIRA, 2022). O objetivo é que um mesmo produto possa ser encontrado em diferentes supermercados para que estes possam ser comparados, influenciando na escolha de local de compra do cliente. Porém, apesar de entidades como marca terem um alto impacto nas diferenças de preços, produtos semelhantes com marcas diferentes podem apresentar uma similaridade de cosseno ainda muito alta. Também percebemos que características como quantidade e unidade podem passar despercebidas no processo de similaridade por cosseno, embora seja um detalhe importante para a comparação entre produtos.

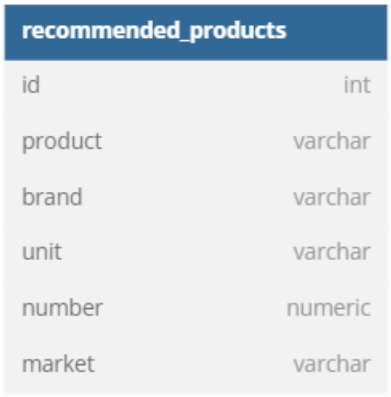
Visto isso, o objetivo desse aprimoramento na recomendação é utilizar as entidades nomeadas de marca, quantidade e unidade como fatores importantes nesse processo de sugestão. Para isso, foram necessárias 3 etapas: Popular o banco de dados do LudiiPrice com a base de dados criada; criar o filtro por entidades nomeadas com definição de prioridades e construir a API que, dado um produto, retorne seu produto recomendado. As etapas estão detalhadas a seguir:

4.3.1 População do banco de dados

A priori, foi criado um novo modelo no banco de dados do LudiiPrice contendo os seguintes campos: id, descrição do produto, marca, unidade, quantidade e o supermercado do qual o produto foi extraído.

Após isso, foram extraídas as marcas, quantidade e unidade de todos os 16112 produtos existentes na base criada e depois foram inseridos no banco de dados. Para a extração das marcas foi utilizado o modelo de aprendizado de máquina desenvolvido neste trabalho e para a extração da quantidade e da unidade foram utilizadas expressões regulares, já que estas são características mais fáceis de serem extraídas, descartando a necessidade de um outro modelo de *machine learning*. Para a conexão com o banco de dados e para a inserção dos registros, foi utilizada a

Figura 10 – Modelo criado no banco de dados do LudiiPrice



biblioteca SQLAlchemy[9], para Python.

4.3.2 Filtro por entidades nomeadas

O objetivo é filtrar por produtos que possuem mesma marca, quantidade e unidade e, após isso, obter o produto mais semelhante por similaridade de cosseno. Porém, existem situações em que o filtro não retornará nenhum dado, e por isso é necessário ir para filtros menos específicos. Dessa forma, foram definidos 4 níveis de prioridade do filtro:

Prioridade 1: Marca, unidade e quantidade iguais. Aqui é o cenário ideal: obter a semelhança apenas de produtos que possuem as mesmas três entidades nomeadas. Caso nenhum dado seja retornado, é realizado um novo filtro com a prioridade 2.

Prioridade 2: Caso não existam produtos com as mesmas três entidades nomeadas, será realizado o filtro apenas pela marca. Isso porque produtos com mesma marca têm maior semelhança do que produtos de mesma quantidade e unidade e marcas diferentes. Caso não existam produtos com a mesma marca, será realizado o filtro com a prioridade 3.

Prioridade 3: Nesse filtro será buscado produtos que possuem mesma quantidade e mesma unidade. Isso porque, caso não existam produtos com a mesma marca, filtrar pela quantidade e unidade pode ajudar a afunilar produtos com maior chance de semelhança. Se também não existir produtos com mesma quantidade e unidade, será realizado o filtro de prioridade 4.

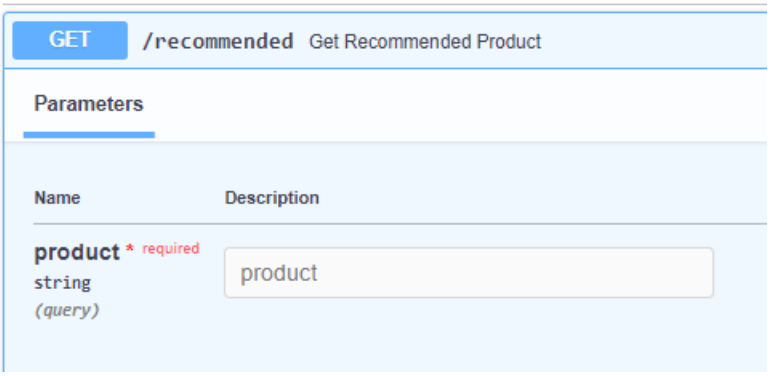
Prioridade 4: Aqui todos os produtos da base é utilizado para a busca do produto mais semelhante, ou seja, o que possui a maior similaridade por cosseno.

[9] Disponível em https://www.sqlalchemy.org.

4.3.3 Construção da API

Para a construção da API foi utilizado a biblioteca FastAPI[10], para o Python, a fim de criar o servidor e as rotas necessárias, junto à biblioteca SQLAlchemy para realizar as interações com o banco de dados do LudiiPrice. O foco inicial dessa API é apenas retornar um produto recomendado com base nas definições de filtro citadas ao longo desta seção, portanto foi criado um *endpoint* do tipo *GET*, o qual deve ser chamado passando como *query paramater* uma descrição de produto. Ao acessar essa rota, é extraída a marca, a unidade e a quantidade do produto de entrada. Depois, é realizado o filtro da base de dados, explicado na subseção anterior e é retornado um objeto JSON contendo o produto recomendado e a similaridade com o produto da entrada.

Figura 11 – Documentação do *endpoint* para obter produto recomendado



[10] Disponível em https://fastapi.tiangolo.com.

5 Resultados Obtidos

Este capítulo apresenta os resultados obtidos. Como destacado anteriormente, nosso *baseline* é o *Open Tag* (ZHENG et al., 2018) e uma implementação não oficial deste trabalho, disponibilizada no repositório lumiqai/UOI-1806.01264[1].

5.1 Resultados do modelo extrator de marcas

Com o modelo treinado e avaliado com a técnica de *hold-out*, sendo 80% dos dados para treino e 20% para teste, obtivemos os seguintes resultados presentes na Tabela 3 a seguir.

Tabela 3 – Resultados do modelo para cada tag

Tags	**Accuracy**	**Precision**	**Recall**	**F1 Score**
O	0.971	0.978	0.987	0.983
B	0.974	0.922	0.882	0.902
I	0.999	0.902	0.840	0.870
E	0.992	0.826	0.762	0.793

Como visto na tabela, o modelo obteve um bom desempenho, alcançando valores abaixo de 0.8 apenas na *tag* E, referente ao final da marca. Verifica-se que a maior quantidade de erros cometidos pelo modelo aconteceu ao tentar extrair uma marca não vista na fase de treinamento. Esses foram os resultados individuais de cada *tag*, mas analisando o modelo e sua capacidade de identificar a marca completa, obtivemos os seguintes resultados:

Tabela 4 – Resultados do modelo para identificação da marca completa

Accuracy	**Precision**	**Recall**	**F1 Score**
0.957	0.860	0.818	0.839

[1] Disponível em https://github.com/lumiqai/UOI-1806.01264.

Como podemos ver, apesar de bons resultados, o menor valor foi produzido pela métrica de *recall*, que é a métrica que mais está relacionada com a quantidade de marcas que o modelo acabou não classificando, já que ela estabelece a relação entre a quantidade de marcas que o modelo classificou corretamente e a quantidade de marcas totais, diferentemente da *precision*, que está mais focada em saber quantas das marcas que o modelo classificou realmente estavam corretas, sendo a relação entre as marcas que o modelo classificou corretamente por todas as marcas que o modelo classificou, como visto na seção 2.2. E o motivo da *precision* ser maior que a *recall* se dá pelo fato de que é mais comum o modelo não encontrar uma marca do que classificar uma marca incorretamente.

Ao comparar os valores do desempenho do modelo ao classificar a marca completa com os valores obtidos no trabalho que é o nosso *baseline*, temos a seguinte tabela:

Tabela 5 – Comparação entre os resultados do modelo desenvolvido e do modelo *baseline*

	Precision	**Recall**	**F1 Score**
Modelo desenvolvido	0.860	0.818	0.839
Modelo baseline	0.949	0.934	0.941

Embora o modelo treinado neste trabalho e o modelo desenvolvido no nosso *baseline* não possam ser comparados diretamente, uma vez que não temos tudo que é necessário para uma reprodução fidedigna, como o *dataset* utilizado e o código original, por exemplo, podemos observar a viabilidade da aplicação do modelo com os mesmos parâmetros mas com um *dataset* diferente. Também é possível perceber que, assim como a nossa, a *recall* foi a menor métrica, como citado no parágrafo anterior.

5.2 Resultados do extrator de marcas com *K-fold cross-validation*

Com o modelo treinado e avaliado com a técnica *K-fold cross-validation*, com K=10, obtivemos os seguintes resultados exibidos na Tabela 6 a seguir:

Tabela 6 – Resultados do extrator de marcas utilizando 10-fold cross-validation

Tags	**Accuracy**	**Precision**	**Recall**	**F1 Score**
O	0.97 ± 0.00	0.98 ± 0.00	0.99 ± 0.00	0.98 ± 0.00
B	0.98 ± 0.00	0.92 ± 0.01	0.90 ± 0.01	0.91 ± 0.01
I	1.00 ± 0.00	0.82 ± 0.06	0.74 ± 0.06	0.77 ± 0.04
E	0.99 ± 0.00	0.84 ± 0.03	0.81 ± 0.03	0.82 ± 0.03

Comparando-se com os resultados da Tabela 3, houve um aumento na maioria das métricas. Uma possível razão dessa melhora se dá pelo objetivo proposto pela técnica do *10-fold*, que é verificar o desempenho considerando uma variação dos dados de treino e teste, expondo o modelo a casos mais raros ou cujo padrão seja mais complexo. Apesar da melhora de forma geral, houve uma piora na precisão e *recall* da *tag* I, referente ao meio da marca. Supomos que o

motivo dessa redução se dá pelo fato de que, devido à baixa quantidade de produtos cujas marcas possuem pelo menos três palavras, alguns *folds* podem ter ficado com poucos produtos com a *tag* I. Assim, quando algum desses *folds* é tomado como *dataset* de teste, qualquer erro se torna mais expressivo, gerando um resultado pior e reduzindo a média do resultado final.

Considerando que os valores alcançados na Tabela 6 são os mais confiáveis, utilizaremos esses resultados como os definitivos do modelo. Além disso, alguns experimentos foram realizados na tentativa de melhorar esses resultados e são demonstrados no Apêndice A.1.

5.3 Melhoria na sugestão de produtos utilizando entidades nomeadas

Para verificar se de fato houve melhoria na sugestão de produtos com o novo sistema, detalhado na seção 4.3, fizemos um experimento comparativo. Primeiro, buscamos em uma base de dados comprada, a mesma utilizada por Oliveira (2022), por produtos que também existissem na nossa base de dados criada, mas que estivessem descritos de forma diferente. Com esse processo manual foram encontrados 117 produtos, que compuseram a amostra deste experimento. Para cada um desses 117 produtos, buscamos uma sugestão do produto na nossa base utilizando apenas similaridade por cosseno (sistema de sugestão de produtos anterior) e a mesma sugestão porém utilizando a nossa API, representando o novo sistema de sugestão. Os resultados foram rotulados manualmente, observando se o produto sugerido era igual ao produto de entrada, apesar de descrições diferentes. Os resultados estão apresentados na Tabela 7 a seguir:

Tabela 7 – Quantidade de acertos e erros na sugestão de produtos por sistema de sugestão

	Similaridade por cosseno	**Novo sistema de sugestão**
Acertos	90	110
Erros	27	7
Total	117	117

Como visto acima, o sistema de sugestão anterior baseado apenas na similaridade por cosseno foi capaz de sugerir corretamente 90 dos 117 produtos buscados, enquanto a API desenvolvida sugeriu 110 produtos corretamente, resultando em um salto na taxa de acerto de 76,9% para 94,0%. Apesar desse primeiro experimento ter demonstrado um maior desempenho da API para esse conjunto de dados e uma efetividade dos filtros utilizados nela, para obter uma validação mais consistente sobre a superioridade dessa API faz-se necessário um experimento mais completo, com um conjunto de dados maior. A seguir, alguns exemplos de sugestões que apenas a API acertou:

Produto de entrada: Adoçante Maratá 100Ml

Sugestão do sistema antigo: Adoçante Magro 100ml
Similaridade por cosseno: 0.697

Sugestão do sistema novo Adoçante líquido frasco 100ml - maratá
Similaridade por cosseno: 0.608

Apesar do produto de entrada ter uma estrutura mais semelhante ao produto sugerido pelo sistema antigo, resultando em uma maior similaridade por cosseno, os produtos não são iguais, já que possuem marcas diferentes. Com isso, podemos ver que o filtro pela entidade nomeada de marca foi o principal fator para o sucesso da sugestão do novo sistema, que buscou o mais similar dentre os produtos de mesma marca. Abaixo, ilustramos mais um exemplo comparativo:

Produto de entrada: Leite De Coco Sococo Tradicional 500Ml

Sugestão do sistema antigo: Leite De Coco Sococo Tradicional 200ml
Similaridade por cosseno: 0.925

Sugestão do sistema novo Leite de coco tradicional sococo vidro 500ml
Similaridade por cosseno: 0.893

Aqui temos uma situação semelhante. Embora o produto de entrada tenha maior semelhança por cosseno com o produto sugerido pelo sistema antigo, suas quantidades são diferentes e, portanto, são produtos diferentes. O filtro pela entidade nomeada de quantidade fez com que o novo sistema de sugestão descartasse o produto de 200ml, levando a uma sugestão correta. Em alguns casos específicos os resultados não são como o esperado, e esses casos estão demonstrados no experimento descrito no Apêndice A.2.

6

Considerações Finais

Neste trabalho foi criada uma base de dados composta pelas descrições de produtos, suas marcas e seus rótulos, totalizando 16112 registros, obtidos de três *e-commerces* diferentes. Além disso, treinamos um modelo de aprendizado de máquina capaz de extrair as marcas de descrições de produtos, com base no nosso *baseline*, com o qual obtivemos a *f1-score* de 91% para o reconhecimento do início da marca, 77% para o do meio da marca e 82% para o do final da marca. O código do modelo foi documentado, está pronto para uso e disponível no Google Colab. Por fim, também foi construído um novo sistema de sugestão de produtos que utiliza entidades nomeadas além da similaridade por cosseno. Durante os experimentos, o novo sistema obteve uma taxa de acerto de 94%, enquanto o sistema anterior, que utilizava apenas similaridade por cosseno, obteve 76,9%. Para fornecer acesso ao novo sistema, foi desenvolvida uma API que, dado uma descrição de produto, retorna a melhor sugestão.

Como trabalho futuro temos: (i) A expansão dos experimentos com a nova API de sugestão de produtos, realizando os testes necessários com uma maior quantidade de dados, a fim de obter uma validação mais abrangente da efetividade dessa API em relação ao sistema de sugestão anterior; (ii) A integração dessa API desenvolvida com o ecossistema do LudiiPrice, possibilitando seu uso real dentro da aplicação e permitindo que os resultados deste trabalho alcancem o usuário final.

Referências

AMARAL, D. O. F. do; VIEIRA, R. O reconhecimento de entidades nomeadas por meio de conditional random fields para a língua portuguesa (named entity recognition with conditional random fields for the Portuguese language) [in Portuguese]. In: *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. [s.n.], 2013. Disponível em: <https://aclanthology.org/W13-4807>. Citado na página 12.

BOJANOWSKI, P. et al. Enriching word vectors with subword information. *CoRR*, abs/1607.04606, 2016. Disponível em: <http://arxiv.org/abs/1607.04606>. Citado na página 39.

CHOWDHURY, G. G. Natural language processing. *Annual Review of Information Science and Technology*, v. 37, n. 1, p. 51–89, 2003. Disponível em: <https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/aris.1440370103>. Citado na página 12.

MORE, A. Attribute extraction from product titles in ecommerce. *CoRR*, abs/1608.04670, 2016. Disponível em: <http://arxiv.org/abs/1608.04670>. Citado 3 vezes nas páginas 19, 20 e 21.

OLIVEIRA, W. L. Trabalho de conclusão de curso, *Uso de web scraping para mineração de produtos e preços em e-commerce*. 2022. Citado 3 vezes nas páginas 10, 28 e 33.

PASSYN, K.; DIRIKER, M.; SETTLE, R. Price comparison, price competition, and the effects of shopbots. *Journal of Business Economics Research*, v. 11, p. 401–416, 08 2013. Citado na página 9.

PROTESTE. *Supermercados: Preços Variam conforme localidade e com Pesquisa É possível economizar*. 2015. Disponível em: <https://www.proteste.org.br/suas-contas/supermercado/noticia/supermercados-compartivos-de-precos-entre-localidades-diferentes-economia>. Citado na página 9.

RIBEIRO, L. et al. Reconhecimento de entidades nomeadas em itens de produto da nota fiscal eletrônica. v. 36, p. 116–126, 01 2018. Citado 2 vezes nas páginas 18 e 19.

SAK, H.; SENIOR, A.; BEAUFAYS, F. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, p. 338–342, 01 2014. Citado 2 vezes nas páginas 13 e 14.

VASWANI, A. et al. Attention is all you need. *CoRR*, abs/1706.03762, 2017. Disponível em: <http://arxiv.org/abs/1706.03762>. Citado na página 22.

ZHENG, G. et al. Opentag: Open attribute value extraction from product profiles. *CoRR*, abs/1806.01264, 2018. Disponível em: <http://arxiv.org/abs/1806.01264>. Citado 9 vezes nas páginas 11, 13, 14, 21, 22, 23, 24, 27 e 31.

Apêndices

APÊNDICE A

Experimentos

A.1 Experimento do extrator de marcas com bigramas

Como citado na seção 5.1, a maioria dos erros cometidos pelo modelo acontece quando a marca não está presente na base treinamento, e isso se dá por dois motivos principais: realmente não existem produtos com essa marca ou a marca está presente mas escrita de forma diferente, com abreviações, por exemplo. Descrições de produtos com marcas abreviadas são comuns, principalmente nos itens de nota fiscal, que é uma fonte de dados importante para o objetivo desse trabalho e, pensando nisso, foi realizado um experimento com bigramas. Essa técnica consiste em rotular as sequências de dois caracteres consecutivos, diferente da abordagem tradicional, que consiste em rotular cada palavra. Assim, por hipótese, o modelo seria capaz de aprender partes das marcas, podendo reconhecê-las em produtos cujas marcas não estão normalizadas. Abaixo um exemplo de como o modelo poderia reconhecer a marca Santa Clara em um produto com marca abreviada:

Produto no *dataset* de treino:

Café Santa Clara 50g

Bigramas do produto acima:

'Ca' 'af' 'fé' 'é ' ' S' 'Sa' 'an' 'nt' 'ta' ' C' 'Cl' 'la' 'ar' 'ra' ' 5' '50' '0g'

Produto no *dataset* de teste:

Café Santa Cl 50g

Bigramas do produto acima:

'Ca' 'af' 'fé' 'é ' ' S' 'Sa' 'an' 'nt' 'ta' ' C' 'Cl' 'l ' ' 5' '50' '0g'

Na situação acima, seguindo a técnica sem bigramas, o modelo provavelmente não reconheceria o item de teste como um produto com marca composta, já que ele não treinou com nenhum produto de marca **Santa Cl** e não é comum marcas com esse tipo de estrutura. Já utilizando a técnica dos bigramas, supomos que o modelo reconheceria a segunda parte da marca já que ele foi exposto aos bigramas dela, sendo que **Santa Clara** e **Santa Cl** possuem 7 bigramas

iguais.

Para implementar essa técnica foi utilizado o FastText (BOJANOWSKI et al., 2016) para os *word embeddings* e os testes foram feitos tanto com o SKIP-GRAM quanto com o CBOW, ambos com 100 dimensões. Os resultados podem ser observados nas Tabelas 8 e 9 a seguir:

Tabela 8 – Resultados do extrator de marcas utilizando a técnica de bigramas (SKIP-GRAM)

Tags	**Accuracy**	**Precision**	**Recall**	**F1 Score**
O	0.9560	0.9636	0.9840	0.9737
B	0.9629	0.9028	0.8483	0.8747
I	0.9987	0.9516	0.3907	0.5539
E	0.9888	0.8329	0.5250	0.6441

Tabela 9 – Resultados do extrator de marcas utilizando a técnica de bigramas (CBOW)

Tags	**Accuracy**	**Precision**	**Recall**	**F1 Score**
O	0.9518	0.9671	0.9748	0.9709
B	0.9569	0.8587	0.8588	0.8587
I	0.9988	1	0.4238	0.5953
E	0.9885	0.7796	0.5625	0.6535

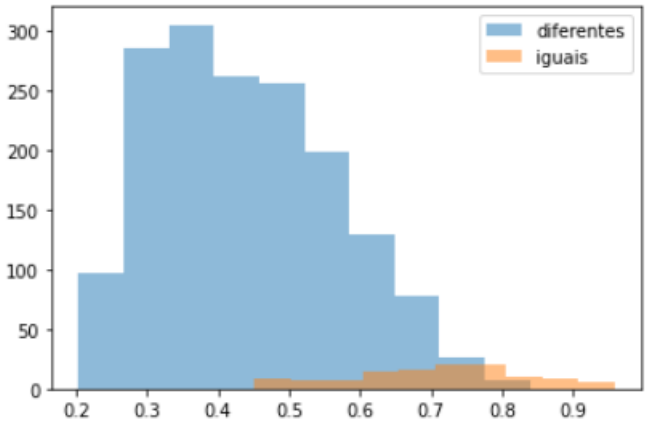
Apesar do modelo utilizando CBOW ter um desempenho melhor que o modelo com SKIP-GRAM, ambos desempenharam pior que o modelo anterior, sem a utilização da técnica de bigramas. Supomos que isso ocorreu pois os bigramas aumentaram muito a extensão de cada tag BIOE, dificultando um pouco mais o reconhecimento de padrões por parte do modelo com a base utilizada. Com esses resultados também é possível identificar uma grande diferença entre os valores de *precision* e *recall*, principalmente para as *tags* I e E. Isso acontece pois o modelo apresenta dificuldade em identificar o meio (I) e o final (E) da marca, gerando uma *recall* baixa para ambos. Porém, quando o modelo reconhece um meio ou final de marca dificilmente classifica-os de forma de errada, gerando uma *precision* alta.

A.2 Experimento de igualdade de produtos

Esse experimento tem como objetivo identificar se é possível afirmar que dois produtos de mesma marca, quantidade e unidade que possuíssem uma determinada similaridade por cosseno eram iguais. Para isso, buscamos a existência de um limiar de similaridade por cosseno em que dois produtos com mesmas entidades nomeadas (marca, quantidade e unidade) fossem considerados iguais. Como o modelo desenvolvido nesse trabalho apenas extrai marcas, foi construído um extrator de quantidade e unidade utilizando expressões regulares. Para o experimento, foram selecionados manualmente 1.762 pares de produtos que possuíam mesma marca, quantidade e unidade, mas com descrições diferentes. Após isso, foi rotulado manualmente quais desses pares eram de produtos iguais ou diferentes, que resultou em 117 pares de produtos e iguais e 1.645

pares de produtos diferentes. A distribuição da similaridade por cosseno pode ser vista na Figura a seguir:

Figura 12 – Similaridade por cosseno dos pares de produtos



Com esse experimento percebemos que não é possível estabelecer um limiar para afirmar com certeza a igualdade entre dois produtos com mesma marca, quantidade e unidade, já que mesmo com uma amostra pequena, alguns produtos diferentes obtiveram similaridades por cosseno altas, até 83%, e alguns produtos iguais obtiveram similaridades baixas, até 45%. Alguns exemplos desses pares de produtos estão a seguir:

Produtos diferentes mas com similaridade acima de 83%:

Produto 1: cerveja itaipava lata 350ml

Produto 2: Cerveja Itaipava Sem Alcool Lata 350ml

Similaridade: 0.83

Produto 1: Água Tônica Zero Açúcar Tônica Antarctica Lata 350ml

Produto 2: Tonica Agua Antarctica Lata 350ml

Similaridade: 0.82

Produto 1: Biscoito Amanteigado Chocolate Vitarella Turma do Treloso Pacote 100g

Produto 2: Biscoito Treloso Amanteigado Coco Vitarella 100g

Similaridade: 0.82

Esses casos acontecem principalmente quando produtos possuem mesma marca, quantidade e unidade, mas diferem entre si por alguma razão, como por exemplo sabor, se é zero açúcares ou não, se possui ou não álcool, entre outros.

Produtos iguais mas com similaridade acima de até 0.45%:

Produto 1: nectar de fruta sabor uva tetra pak 200ml - marata

Produto 2: Suco Marata Uva 200ml

Similaridade: 0.47

Produto 1: Café torrado e moído tradicional (em pó) almofada 250g - pilão

Produto 2: Café Pilao 250g

Similaridade: 0.45

Produto 1: bala sabor morango pacote 80g - fini

Produto 2: Fini Tubes Morango 80g

Similaridade: 0.45

Já esses casos acontecem por ter produtos iguais com descrições muito diferentes.