Extração de Entidades de Produtos Utilizando Técnicas de *Few-Shot Learning*

Aleson G. S. Chaves [*] **Fernanda C. e Silva** [*] **Danton D. Ferreira** [**]
Bruno H. G. Barbosa [**] **Sinval T. Nascimento** [***]

[*] *Programa de Pós-Graduação em Engenharia de Sistemas e Automação, Universidade Federal de Lavras, MG*
[**] *Departamento de Automática, Universidade Federal de Lavras, MG, (e-mail: aleson.chaves@estudante.ufla.br, fernanda.silva10@estudante.ufla.br, danton@ufla.br, brunohb@ufla.br)*
[***] *Omnilogic Inteligência, (e-mail: sinval@omnilogic.com.br).*

Abstract: The interpretation of unstructured data in textual format is a research area that can be applied to electronic commerce, which has generated a large amount of natural language data. With the growth of marketplaces, new products are added daily by several different retailers, and there may be classes in the product databases that have only a few samples. Obtaining classifiers that consider new classes with few samples is a complex task, whether due to the computational cost of retraining existing models to include these classes, or due to their low number of samples. In this sense, Few Shot Learning (FSL) techniques are promising options. That said, this paper compared the FSL classifiers Matching Networks and Siamese Neural Network in a marketplace's product classification problem, with 34 classes and 394 samples in total. These classifiers were compared with the K-Nearest Neighbors (KNN) algorithm and Principal Component Analysis was applied for database size reduction. The FSL algorithms implemented performed better than the KNN method in the leave-one-out cross validation test, showing an accuracy of 96.85%, even considering classes with a low number of samples.

Resumo: A interpretação de dados não estruturados no formato textual é uma área de pesquisa que pode ser aplicada para o comércio eletrônico, que tem gerado uma grande quantidade de dados em linguagem natural. Com o crescimento dos *marketplaces*, novos produtos são adicionados diariamente por diferentes lojistas, e podem existir classes de produtos que tenham poucas amostras nas bases de dados. A obtenção de classificadores que considerem novas classes com poucas amostras é uma tarefa complexa, seja pelo custo computacional de retreinar os modelos existentes para contemplar essas classes, ou pela baixa quantidade de amostras delas. Nesse sentido, técnicas de *Few Shot Learning* (FSL) são opções promissoras. Isso posto, este trabalho comparou os classificadores FSL *Matching Networks* e Redes Neurais Siamesas no problema de classificação de produtos de um *marketplace*, com 34 classes e 394 amostras. Esses classificadores foram comparados com o algoritmo *K-Nearest Neighbors* (KNN) e foi aplicada a Análise de Componentes Principais para redução de dimensão do banco de dados. Os algoritmos FSL obtiveram desempenho superior ao KNN no teste de validação cruzada *leave-one-out*, apresentando acurácia de 96,85%, mesmo considerando classes com baixo número de amostras.

Keywords: Natural Language Processing; E-commerce; Machine Learning; Few-Shot Learning; Siamese Neural Network; Matching Networks; Artificial Intelligence

Palavras-chaves: Processamento de Linguagem Natural; Comércio Eletrônico; Aprendizado de Máquina; Few-Shot Learning; Redes Neurais Siamesas; Redes Matching; Inteligência Artificial

1. INTRODUÇÃO

Com o desenvolvimento dos recursos computacionais e do acesso à internet por grande parte da população, houve um aumento significativo do comércio eletrônico (Ristoski et al., 2018). O número de consumidores e vendedores adeptos ao modelo de mercado *on-line* tem crescido, o que gera um aumento no número de ofertas nas plataformas, com uma grande variedade de produtos. Novas ofertas são adicionados diariamente em grandes plataformas de mercado como *Amazon* e Magazine Luiza (Krishnan and Amarthaluri, 2019), criando uma grande quantidade de dados em formato textual (Zheng et al., 2019). A área de pesquisa que estuda os problemas da geração e interpretação da linguagem humana pelas máquinas é o Processamento de Linguagem Natural (NLP, do inglês *Natural Language Processing*), uma subárea da ciência da computação e inteligência artificial (e Silva et al., 2020).

Um problema comum nas bases de dados dos *marketplaces* é a ocorrência de classes que contém poucas amostras, o que dificulta o processo de treinamento de classificadores tradicionais, por exemplo, aqueles baseados em Redes Neu-

rais Artificiais. A obtenção de classificadores de produtos que levem em consideração novas classes com poucas amostras é uma tarefa complexa, seja pelo custo computacional de retreinar os modelos existentes de forma a contemplar as novas classes, ou seja pela baixa quantidade de amostras dessas novas classes. Como existem classes novas que não passaram pelo treinamento do classificador em operação, é necessário o re-treino do classificador com toda a base de dados, incluindo essas novas classes.

Os erros de classificação provocam, dentre outros problemas, a categorização incorreta de produtos nos *marketplaces*, podendo causar experiências desagradáveis no processo de compra. A falta de padrão pode fazer com que novas ofertas não apareçam ou apareçam na página do produto errado, devido a não correspondência dos mesmos produtos de vendedores distintos (Ristoski et al., 2018).

Nesse sentido, técnicas de *Few Shot Learning* (FSL) são opções promissoras por serem projetadas especificamente para lidar com treinamento de modelos com pequenas quantidades de amostras por classe (Jadon and Garg, 2020). Existem vários algoritmos de aprendizado FSL, como: *Matching Networks* (MN) (Vinyals et al., 2016), *Model Agnostic Meta-Learning* (MAML) (Finn et al., 2017), Redes Siamesas (Koch et al., 2015) e *Graph Neural Networks* (GNN) (Garcia and Bruna, 2018).

As Redes Siamesas são boas alternativas para lidar com problemas de reconhecimento de padrões a partir de uma ou poucas amostras (Koch et al., 2015). Essa rede pode ser usada para medir a similaridade entre duas entradas e determinar se o par de amostras pertence a uma mesma classe ou não (Bromley et al., 1994). As Redes *Matching* aprendem a mapear os dados de entrada e utilizam tais dados para fazer predição por meio de um mecanismo de atenção aplicado ao cálculo de similaridade do cosseno entre os representantes e o alvo (Vinyals et al., 2016).

Isso posto, este trabalho tem por objetivo propor e comparar o desempenho das Redes Siamesas e Redes *Matching* para extração de entidades de produtos de um *marketplace*. Esses classificadores foram comparados também com o algoritmo *K-Nearest Neighbors* (KNN), que foi utilizado como *baseline* do projeto. Ademais, foi analisado o desempenho dos algoritmos após a redução de dimensão das amostras do banco de dados, utilizando o método de transformação das características *Principal Component Analysis* (PCA) (Krishnan and Dutta, 2018).

As abordagens propostas têm como benefício manter uma boa acurácia dos classificadores em operação nos *marketplaces*, mesmo com a chegada de novas classes, o que mantém a plataforma de *marketplace* bem categorizada, com as especificações de produto organizadas e anúncios padronizados em uma estrutura de texto que atenda as necessidades ao consumidor e facilita o processo de busca e compra de produtos. Também podem reduzir o uso do servidor em nuvem, disponibilizando o mesmo para uso em outras aplicações ou reduzindo o valor do contrato.

2. MATERIAIS E MÉTODOS

2.1 Base de Dados

Os códigos computacionais foram implementados em linguagem *Python*. A base de dados utilizada foi cedida pela empresa parceira deste trabalho e é composta por amostras com textos não estruturados, provenientes de uma plataforma de comércio eletrônico. A base de dados contém 394 ofertas divididas em 34 classes, como mostrado na Tabela 1. Conforme pode ser observado, trata-se de uma base bastante desbalanceada e que possui poucas amostras.

Tabela 1. Base de Dados de Classes Novas

Descrição da Base	Valores
Número de amostras	394
Número de classes	34
Classes com 1 Amostra	13
Classes com 2 amostras	5
Classes com 3 a 8 amostras	10
Classes com 20 a 60 amostras	5
Classes com 113 amostras	1

Cada oferta em linguagem natural é transformada em um vetor de dados numéricos com 1200 valores, que são as características de cada oferta que serão usadas nos classificadores *few-shot*. Esses valores são extraídos por um modelo que encontra-se em operação e que foi previamente treinado com amostras de classes diferentes daquelas utilizadas neste trabalho. Ou seja, a partir de um classificador (Rede Neural Artificial) treinado previamente com um número grande de classes e amostras, são extraídas características (saídas dos neurônios da última camada intermediária da rede neural), em um processo que pode ser interpretado como *transfer learning*.

2.2 Few-Shot Learning

O processo de aprendizagem com a utilização de poucos dados é uma tarefa complexa para as redes neurais tradicionais (Jadon and Garg, 2020). Para o caso de uma pequena base de dados, um algoritmo simples como o KNN pode ter desempenho melhor do que uma rede neural *Multi-Layer Preceptron* (MLP), caso se tenha um bom pré-processamento que leve a uma boa separação entre as classes e a escolha de uma boa métrica para o cálculo da distância (Jadon and Garg, 2020).

As redes de aprendizado profundo conseguem um bom desempenho em uma variedade de tarefas, porém necessitam de uma grande quantidade de dados para aprender (Zheng et al., 2019). No entanto, existem bases de dados com apenas algumas poucas amostras por classes e isso dificulta o processo de treinamento das redes neurais. Neste caso, é necessário implementar métodos de aprendizagem de máquinas específicos para redes neurais que sejam capazes de realizar o treinamento quando se tem poucas amostras por classe. Estes métodos são conhecidos como *Few-Shot Learning* (Wang et al., 2020).

2.3 K-Vizinhos Próximos

O k-vizinhos mais próximos (KNN) é um dos mais simples e tradicionais classificadores utilizados em reconhecimento

de padrões, cujo desempenho é competitivo em relação aos mais complexos classificadores da literatura. A grande vantagem do KNN é que ele não depende de treinamento pois a sua tarefa se resume em identificar, em um conjunto rotulado, os k objetos mais semelhantes (com menores distâncias) da amostra não rotulada (Trstenjak et al., 2014). Portanto, o KNN é uma opção interessante para um conjunto de dados com pequena quantidade de amostras. De fato, o KNN pode ser considerado uma técnica de *few-shot learning* uma vez que, a partir do momento que uma ou mais amostras de uma classe nova estiver disponível, tal amostra pode ser prontamente adicionada no conjunto de dados e poderá ser usada na classificação de outra amostra de sua mesma classe.

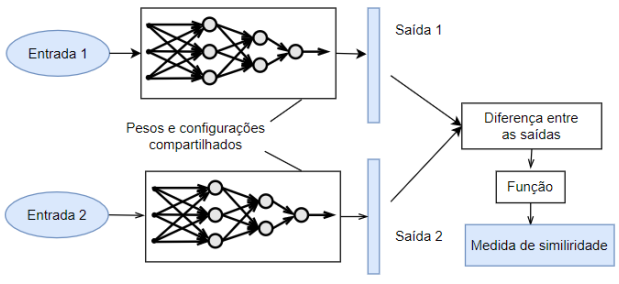
O algoritmo KNN foi implementado como classificador *few-shot* para o banco de dados da Seção 2.1. As características da nova oferta a ser classificada são comparadas (medida de similaridade) com as características das amostras já existentes no banco de dados, e o algoritmo KNN é então implementado.

Para os testes, as classes que possuíam apenas uma amostra foram colocadas na base de treino apenas para tornar o problema mais complexo. As que possuíam duas amostras, uma foi usada para teste (*Leave-One-Out*) e a outra como representante. Nas classes com três ou mais amostras, uma foi utilizada para teste (*Leave-One-Out*) e as demais utilizadas como representantes ou para o cálculo do centroide que foi utilizado como representante da respectiva classe. As distâncias utilizadas foram: *Euclidiana (ED), cosseno (CD), Manhattan (MD), Hassanat (Has-D), Lorentzian (LD)* e *Clark (Cla-D)*. Também foram testados diferentes valores do parâmetro k.

2.4 Redes Siamesas

A Rede Neural Siamesa foi apresentada no trabalho de Bromley et al. (1994), fazendo a comparação de assinaturas escritas em um *tablet*. Esse tipo de rede neural de treinamento supervisionado pode ser usada para medir a similaridade entre duas entradas e, portanto, pode determinar se um par de amostras pertence a uma mesma classe ou não. A Figura 1 apresenta a arquitetura de uma rede siamesa. Ela é formada por duas sub-redes idênticas, que possuem a mesma estrutura e são configuradas com os mesmos pesos e parâmetros.

Figura 1. Arquitetura simplificada de uma rede siamesa



Fonte: Do Autor (2021)

Para o treinamento da rede são necessários pares de amostras da mesma classe, que terão valor de alvo 1, e pares de amostras de classes diferentes que terão valor de alvo 0. Cada sub-rede recebe as entradas e elas produzem um vetor na saída, em um espaço multidimensional de similaridade semântica, criado pela rede siamesa. Esse espaço tem a mesma dimensão do número de neurônios de saída das sub-redes.

O neurônio que faz a junção dos vetores de saída das sub-redes também é responsável por medir a distância entre eles. A distância é usada para determinação do valor de similaridade, que pode ser usado para definir se as entradas da rede são de uma mesma classe ou não, de acordo com os limites estabelecidos para que as amostras sejam consideradas similares. Esse valor de similaridade é utilizado pela rede para calcular a perda, que será empregada na atualização dos pesos das sub-redes via *backpropagation*. A perda é calculada pela diferença entre o valor de similaridade calculado e o valor do alvo de cada par de amostras.

Após o treino da rede siamesa, o conceito *few-shot learning* pode ser aplicado diretamente, pois a princípio a rede aprendeu a distinguir amostras de classes diferentes ou a identificar amostras pertencentes à mesma classe. No problema analisado neste trabalho, a aplicação das redes siamesas em *few-shot learning* será feita para classificação de novas amostras de produtos que tenham poucas amostras e não são conhecidas pelo classificador em operação, assim como realizado para o classificador KNN.

As sub-redes utilizadas na construção da rede siamesa foram propostas com uma única camada intermediária, com 300 neurônios. Após a camada intermediária há uma camada que calcula a distância ponto a ponto entre as saídas das duas sub-redes. A saída dessa camada alimenta uma camada de saída com um único neurônio que calcula o valor de similaridade entre as entradas.

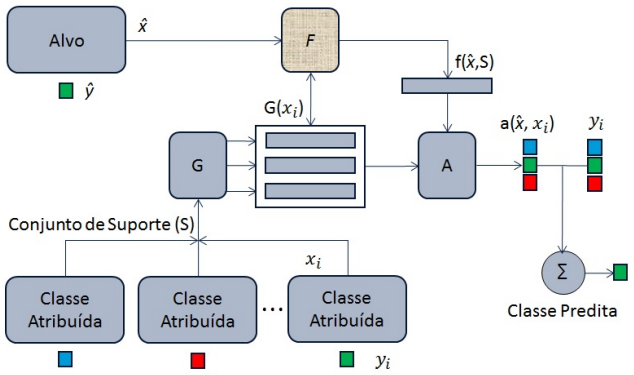
2.5 Redes Matching

As redes *matching* aprendem a mapear uma pequena base de dados de treino e teste em um mesmo espaço de *embeddings* (tarefa *few-shot*). Elas são treinadas para aprender uma representação com os *embeddings* da pequena base de dados de treino de forma adequada, em busca da minimização dos erros do cálculo de similaridade entre o alvo e os representantes de classe. As redes *matching* funcionam com a ideia de um KNN diferenciável com medida de similaridade do cosseno. Além dos *encoders* que formam os *embeddings* estas redes utilizam um mecanismo de atenção pela aplicação da função *softmax* na distância do cosseno.

As redes *matching* desempenham bem a tarefa *few-shot* para classes não vistas no treinamento (Vinyals et al., 2016), sendo implementadas em cinco principais etapas: (i) Extrator de *embeddings* G, que forma o *encoder* com os dados das amostras; (ii) *Embeddings* de contexto completo F, cujo objetivo é obter o contexto entre as amostras, quando houver, e é formado por uma rede LSTM bidirecional (o uso é opcional); (iii) cálculo de similaridade com a função de distância cosseno, c; (iv) mecanismo de atenção que é dado pela aplicação da função *softmax* na saída da função de cálculo de similaridade c; e (v) cálculo da função de perda entropia cruzada (Jadon and Garg, 2020).

A arquitetura com as etapas das redes *matching* pode ser vista na Figura 2. O conjunto de suporte (S) é o conjunto de dados de entrada, como se fossem os representantes (vizinhos) do KNN sendo que x_i é o vetor de dados, y_i são os rótulos. O alvo $\hat{x}$ é a amostra a ser predita.

Figura 2. Arquitetura das redes *Matching*



Fonte: Do Autor (2021)

Os dados do conjunto de suporte formados são encaminhados para o *encoder* G, que aprende uma nova representação para as amostras. Depois os dados passam pelo *embedding* de contexto completo F, ou seja, a saída de G e o alvo $\hat{x}$ passam pela rede BILSTM (Vinyals et al., 2016). Conforme a Figura 2, na sequência de F pode ser observada a equação $f(\hat{x}, S)$ que é a nova representação dos dados do conjunto de suporte (S) e do alvo $\hat{x}$ obtida por F.

O próximo passo antes da aplicação da atenção é o cálculo de similaridade utilizando a função do cosseno. A similaridade é calculada entre a nova representação do conjunto de suporte (x_i) e o alvo $\hat{x}$. O mecanismo de atenção A é obtido pela aplicação da função *softmax* na saída da função de similaridade do cosseno conforme a Equação 1, em que c é a função que executa o cálculo de similaridade com a distância do cosseno:

$$a(\hat{x}, x_i) = \frac{e^{c(f(\hat{x}),g(x_i))}}{\sum_{j=1}^{k} e^{c(f(\hat{x}),g(x_j))}}. \quad (1)$$

A predição é dada pela Equação 2 que implementa a combinação linear da *softmax* da camada de atenção com o vetor *one hot encoded* dos rótulos y_i:

$$P(\hat{y}|\hat{x}, S) = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i. \quad (2)$$

Esta equação é uma combinação linear de probabilidades que determina a qual classe o alvo pertence. A função de perda normalmente utilizada é a entropia cruzada.

O *encoder* G foi implementado como uma rede MLP com apenas uma camada intermediária com 100 neurônios e função *tanh*, configuração esta definida após testes preliminares na base de dados. Na parametrização das redes *matching*, além dos parâmetros tradicionais tal como os existentes na MLP, existem dois parâmetros a serem ajustados, esses são chamados "amostras por classe" e lote (*batch*). Em cada posição do lote é realizado um teste de similaridade entre o conjunto de suporte e o alvo $\hat{x}$ com a distância do cosseno. O parâmetro "amostras por classe" é bem intuitivo, pois são os números de amostras de cada uma das classes que vão compor o lote, é algo similar ao número de representantes do KNN. A atualização dos pesos acontece após o término do cálculo de cada um dos lotes. A acurácia é dada pela média dos lotes.

As redes *Matching* foram implementadas empregando a função de perda de entropia cruzada e o otimizador *Adam* com taxa de aprendizagem de 0,001 e 0,0001 para 307 e 1200 características respectivamente. Foram utilizadas 34 "classes por conjunto de suporte" e 2 "amostras por classe". Número de épocas de treinamento foram 50 e 80 para 307 e 1200 características respectivamente e 10 lotes. O *embedding* de contexto completo F não foi utilizado. (Os parâmetros foram ajustados experimentalmente usando o banco de dados de treino, com a seguinte faixa de variação: "amostras por classe" (2 a 111), lotes (10 a 20), taxa de aprendizagem (0,01 a 0,0001). No caso da MLP as faixas de variações dos parâmetros da camada intermediária foram as seguintes: camadas (1 a 5), neurônios (10 a 307), função não linear (*tanh* e *relu*)).

Para a avaliação de desempenho dos algoritmos aqui implementados foi utilizada a medida de acurácia, obtida no método de validação cruzada *Leave-One-Out* (SCHREIBER et al., 2017). Esse método é aconselhável para este trabalho, pois a base de dados possui poucas amostras.

3. ANÁLISE E DISCUSSÃO DOS RESULTADOS

3.1 O Algoritmo KNN

Os experimentos com o KNN foram iniciados com um vizinho, $k = 1$, ou um representante (centroide) por classe para as 34 classes e 394 amostras da base de dados. O uso de vários representantes por classe gerou melhor resultado do que apenas o uso do centroide, como mostrado na Tabela 2, juntamente com os resultados obtidos para diferentes medidas de similaridade. As distâncias *Manhattan (MD), Hassanat (Has-D)* e *Lorentzian (LD)* apresentaram os melhores resultados, com 96,33% de acerto, seguidas pela distância cosseno, com 95,80% de acerto.

Tabela 2. Valores da acurácia (k=1) para várias distâncias, com uso de centroide ou não

Distâncias	MD	Has-D	LD	CD	ED	Cla-D
KNN	96,33	96,33	96,33	95,80	95,01	94,75
KNN (cent.)	81,36	77,16	80,05	82,41	84,78	76,11

A fim de encontrar o melhor número de k vizinhos, foram escolhidas as distâncias Cosseno, Euclidiana e *Manhattan*, e foram mantidas 21 classes para teste. Para as classes com números de amostras menor do que k, foram adicionadas cópias de suas amostras até se obter $k + 1$ amostras em cada classe. Os resultados encontrados com o teste *Leave-One-Out* são apresentados na Tabela 3.

Pode ser inferido que aumentar o número de vizinhos não foi uma estratégia interessante, pois o valor de $k = 1$ obteve os melhores resultados, o que pode ser justificado pelo fato de alguns *clusters* não serem tão bem definidos na base. A distância cosseno é interessante por possibilitar de forma mais direta a elaboração de um limiar geral de

decisão de classificação e será a distância utilizada para comparações no restante deste trabalho.

Tabela 3. Valores da acurácia do KNN ($k \geq 1$)

Valores (k)	k=1	k=3	k=4	k=5	k=6	k=7
Cosseno	95,80	94,49	93,96	93,44	93,18	90,81
Euclidiana	95,01	93,96	94,23	92,65	92,13	90,81
Manhatan	96,33	95,01	94,49	93,18	92,65	92,91

Com o objetivo de redução de dimensão do problema em estudo foi realizada a transformação das características com *Principal Component Analysis* (PCA).

O resultado do classificador KNN com $k = 1$ e distância cosseno, utilizando características reduzidas com PCA é apresentado na Tabela 4. A PCA apresentou um resultado interessante, mostrando que há bastante redundância de características, conseguindo uma redução de 74,41% (de 1.200 para 307 características) mantendo a mesma acurácia de 95,8% (99,9999% de variância explicada).

Os resultados obtidos com o algoritmo KNN foram usados para construção de um *baseline* para comparações com algoritmos de *Few-Shot Learning*. A utilização do algoritmo KNN se mostrou uma boa opção para a tarefa de classificação de produtos com poucas amostras, sem a necessidade de treinamento ou ajuste de parâmetros de modelos. O índice de acerto sempre superior a 90%, indica que a maioria das amostras estão bem localizadas em seus respectivos agrupamentos, validando o uso do extrator de características utilizado pelo método de *transfer learning*.

Tabela 4. Resultados do KNN ($k = 1$) com dimensão reduzida com PCA, para diferentes números de características (NC)

Variância(%)	100	98	95	90	85	80
NC	307	107	64	37	24	16
Resultado	95,80	95,54	95,28	94,23	93,70	92,39

3.2 Redes Siamesas

A estrutura utilizada na construção da rede siamesa foi proposta com uma única camada intermediária, em que o número de neurônios foi variado experimentalmente de 200 a 800, sendo que o melhor resultado foi com 300 neurônios nesta camada. O modelo utilizou o otimizador *Adam* com taxa de aprendizagem que foi variada de 0,0001 a 0,001, sendo 0,0006 o valor que com melhor acurácia.

Também foram realizados testes para encontrar o melhor representante das classes, uma vez que, na aplicação proposta, a rede siamesa irá comparar uma amostra de classe conhecida com uma nova amostra que não foi rotulada. Um teste *Leave-One-Out* adicional foi realizado, em que foi calculado um centroide para cada classe a partir da média dos dados de treino de cada classe. Esse centroide foi usado como representante da classe no teste, sendo fixado em uma das entradas da rede siamesa para cada classe.

Os resultados da acurácia e o tempo utilizado aproximado para realizar o treinamento e teste da rede siamesa (SN) utilizando as características da base com 1200 características e da base de 307 características obtidas com o PCA são mostrados na Tabela 5, juntamente com os valores para a rede siamesa com centroide (SNC), o KNN e redes *matching* (MN). Os valores das colunas Acc.(1200) e Tempo (1200) se referem aos resultados de acurácia e tempo de treinamento dos algoritmos para a base de 1200 características e as colunas Acc.(307) e Tempo (307) se referem aos resultados com a base reduzida pelo PCA.

Tabela 5. Desempenho dos classificadores

*Alg.	*Acc.(1200)	Tempo (1200)	Acc.(307)	Tempo (307)
SN	96,32%	3,2 horas	96,85%	2,3 horas
SNC	95,27%	2,1 horas	95,80%	1,0 horas
KNN	95,80%	NA*	95,80%	NA*
MN	96,32%	44,4 horas	96,85%	12,3 horas

*Abreviações: Acc.=acurácia em (%), Alg.=algoritmo, NA=não aplicável

É possível perceber que a rede siamesa superou os resultados obtidos pelo KNN para o teste *Leave-One-Out*. A abordagem com a base de dados reduzida apresenta melhor desempenho que a abordagem com as características originais. Em relação à escolha do melhor representante de cada classe com o centroide como candidato, o resultado obtido não superou os resultados obtidos com o KNN em nenhuma das duas bases de dados.

3.3 Redes Matching

As redes *matching* só conseguem realizar treinamento *Leave-One-Out* para classes que possuem 3 amostras ou mais. Uma amostra é separada para o teste e, no treinamento, o cálculo de similaridade é realizado, ou seja, são necessárias uma amostra para representante de classe e uma amostra para ser o alvo. Para aproveitar classes com somente 2 amostras para ajuste dos parâmetros da rede *encoder*, foi realizada uma cópia da amostra destas classes do conjunto de treino para ser o alvo.

Na tabela 5 (Algoritmo MN, Acc.(307)) pode ser observado o resultado das redes *matching* utilizando as 307 características obtidas por redução com PCA e, que levou a uma rede mais compacta (307 neurônios na entrada e na saída). Com esta configuração, o treinamento convergiu com maior acurácia e rapidez do que com as 1200 características originais (Algoritmo MN, Acc.(1200))(1200 neurônios na entrada e na saída), além de ter reduzido o tempo utilizado nos cálculos de similaridade, que interferem bastante no tempo de treinamento.

As redes *matching* conseguiram superar o KNN, e apresentaram a mesma acurácia da rede siamesa. Porém, as redes *matching* com duas "amostras por classe" apresentaram maior tempo de treinamento do que a rede siamesa. Ela também tem mais parâmetros e arquitetura mais complexa que a rede siamesa, logo é mais complicada de ser parametrizada e treinada. Desta forma, a rede siamesa é mais adequada para a aplicação.

Levando em consideração que as características extraídas das amostras utilizadas são de boa qualidade, e que a base de dados possuiu menos classes do que é utilizado em bases de dados da literatura para tarefa *few-shot* (por exemplo, o conjunto de dados *Omniglot* possui 1623 classes com 20 amostras cada (Lake et al., 2011) e o conjunto de dados *mini*ImageNet possui 100 classes com 600 amostras cada (Vinyals et al., 2016)), superar o KNN não é uma tarefa

simples, pois tanto a rede siamesa quanto o *encoder* da rede *matching* precisam ser bem treinados para obtenção de características ainda melhores do que aquelas já obtidas pelo extrator de características utilizado.

4. CONCLUSÃO

Os algoritmos de aprendizado *one/few shot learning* são usados em situações em que se dispõe de poucos dados de alguma classe. A maioria das aplicações de comércio eletrônico utilizam algoritmos de aprendizado profundo que apresentam bom desempenho somente nas grandes bases de dados. Então conseguir algoritmos que funcionem bem para bases de dados com poucas classes é um ganho, pois as classes recém chegadas não possuem amostras suficientes para o treinamento das redes neurais tradicionais. Neste sentido os algoritmos *few-shot learning* empregados são capazes de suprir esta lacuna deixada pelas redes tradicionais. Além do mais, este trabalho tem como diferencial aplicar as ferramentas *few-shot learning* ao processamento de linguagem natural (base obtida por *transfer learning*), visto que, na literatura a maioria das aplicações são para base de dados de imagens, tal como, *mini*ImageNet.

Foram implementadas as técnicas de *few-shot learning* Redes Siamesas e Redes *Matching*, sendo seus resultados comparados com o algoritmo KNN. Observou-se que as mesmas obtiveram desempenho satisfatório, conseguindo superar os resultados obtidos com o KNN. Com o aumento de classes a serem treinadas, espera-se que os seus benefícios fiquem ainda mais evidentes. Em relação aos algoritmos FSL, as redes siamesas são mais adequadas para a aplicação pois apresentaram o mesmo resultado que as redes *matching*, porém possuem menos parâmetros e apresentaram menor tempo empregado nas etapas de treino e teste. Além disso, as redes siamesas são menos complexas e mais fáceis de serem parametrizadas e treinadas do que as redes *matching*.

Foi possível observar que há redundância nas características e a utilização de PCA reduziu bastante a dimensão das entradas, permitindo a obtenção de modelos mais parcimoniosos. É importante ressaltar que a aplicação de PCA não diminuiu o desempenho dos classificadores propostos, quando comparado ao desempenho em testes na base de dados sem redução de características.

Visando o melhor desempenho da rede siamesa, podem ser estudadas estratégias para escolha do melhor representante de cada classe, já que o uso do centroide não superou o desempenho da rede com os dados originais. Também é considerada a alteração na estrutura da rede siamesa, colocando mais camadas ou outro tipo de rede neural, além da mudança na escolha e formação dos pares de treinamento. Para as redes *matching* podem ser testadas outras configurações, como uso de *ensembles* e definição de representantes por classes. Também podem ser analisadas opções de FSL da literatura, como redes neurais de grafos.

AGRADECIMENTOS

Os autores agradecem à empresa *Omnilogic Inteligência* e à FAPEMIG por apoiarem este trabalho.

REFERÊNCIAS

Bromley, J., Guyon, I., LeCun, Y., Sackinger, E., and Shah, R. (1994). Signature verification using a "siamese" time delay neural network. *In Advances in neural information processing systems*, 737–744.

e Silva, F.C., de Sousa, R.H., Chaves, A.G.S., Barbosa, B.H.G., and Ferreira, D.D. (2020). Classificador fuzzy-genético aplicado ao processamento de linguagem natural. *Anais do XXIII Congresso Brasileiro de Automática*, 2.

Finn, C., Abbeel, P., and Levine, S. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *34th International Conference on Machine Learning, ICML 2017*, volume 3, 1856–1868.

Garcia, V. and Bruna, J. (2018). Few-Shot Learning With Graph Neural Networks. In *arXiv*, 1–13.

Jadon, S. and Garg, A. (2020). *Hands-On One-shot Learning with Python Learn*, volume 1.

Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. *Proceedings of the 32nd International Conference on Machine Learning*, 37.

Krishnan, A. and Amarthaluri, A. (2019). Large Scale Product Categorization using Structured and Unstructured Attributes. In *Conference on Knowledge Discovery and Data Mining, August 04–08, 2019, Anchorage, Alaska. ACM, New York, NY, USA*, 9.

Krishnan, M. and Dutta, D. (2018). A Study of Effectiveness of Principal Component Analysis on Different Data Sets. In *2017 IEEE International Conference on Computational Intelligence and Computing Research, ICCIC 2017*. IEEE. doi:10.1109/ICCIC.2017.8524329.

Lake, B.M., Salakhutdinov, R., Gross, J., and Tenenbaum, J.B. (2011). One shot learning of simple visual concepts. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, 6.

Ristoski, P., Petrovski, P., Mika, P., and Paulheim, H. (2018). A machine learning approach for product matching and categorization. *Semantic Web*, 9(5), 707–728.

SCHREIBER, J.N.C., BESKOW, A.L., MÜLLER, J.C.T., NARA, E.O.B., SILVA, J.I.D., and REUTER, J.W. (2017). Técnicas de validação de dados para sistemas inteligentes: Uma abordagem do Software SDbayes. In *XVII Colóquio Internacional de Gestão Universitária*, 1–18. doi:10.1017/CBO9781107415324.004.

Trstenjak, B., Mikac, S., and Donko, D. (2014). Knn with tf-idf based framework for text categorization. *Procedia Engineering*, 69, 1356–1364. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, 3637–3645.

Wang, Y., Yao, Q., Kwok, J., and Ni, L.M. (2020). Generalizing from a Few Examples: A Survey on Few-Shot Learning. *ACM Comput. Surv.*, 1(1), 1–34.

Zheng, Y., Wang, R., Yang, J., Xue, L., and Hu, M. (2019). Principal characteristic networks for few-shot learning. *Journal of Visual Communication and Image Representation*, 59, 563–573.