Aprendizado *few shot learning* com redes neurais siamesas aplicado na classificação para *E-commerce*

Fernanda C. e Silva [*] **Danton D. Ferreira, Bruno H. G. Barbosa** [**] **Paulo R. Silva** [***] **Sinval T. Nascimento** [***]

[*] *Programa de Pós-graduação em Engenharia de Sistemas e Automação, Universidade Federal de Lavras (e-mail: costa.fecs@gmail.com)*
[**] *Departamento de Automática, Universidade Universidade Federal de Lavras, CP 3037, 37200-900, Lavras/MG (e-mails: brunohb@ufla.br, danton@ufla.br).*
[***] *Omnilogic Inteligência S/A, Belo Horizonte/MG (e-mails: paulo.silva@omnilogic.ai, sinval@omnilogic.ai).*

Abstract: Applying artificial intelligence to e-commerce is a trend, however, algorithms that deal with small databases are lacking. This work proposes a model with Siamese neural network in a few shot learning context to classify new samples from an online store. In order to choose the best representative sample to be used in the Siamese network, the random choice and centroid calculation approaches were tested. In addition, an ensemble structure of Siamese network considering 3 and 5 representants was also proposed. Finally, the k-Means algorithm was used to calculate the centroids of the clusters of each class and use them as representative samples in the ensemble. The best result was with the centroid, improving performance from 93% to over 98% accuracy. Different structures were also proposed for the internal network, the best being with three hidden layers and applying the DropOut technique. In addition, it is possible to develop other alternatives to choose representatives and improve feature extractors, improving data quality, being useful to reduce the size of the analyzed data to reduce the complexity of the network in order to reduce costs with the use of servers and data storage in the cloud.

Resumo: O uso de inteligência artificial aplicado ao comércio eletrônico é uma tendência, porém faltam algoritmos que lidem com pequenas bases de dados. Neste trabalho é proposto um modelo com redes neurais siamesas num contexto de aprendizado *few shot learning* para classificação de novas amostras de uma loja *online*. Para a escolha do melhor representante de cada classe, a ser usado na rede siamesa, foram testadas as abordagens de escolha aleatória e cálculo do centroide. Uma estrutura de *ensembles* de representantes, com 3 e 5 elementos, foi também proposta. Finalmente, foi usado o algoritmo *k-Means* para calcular os centroides dos *clusters* de cada classe e usá-los como representantes no *ensemble*. O melhor resultado foi com o centroide, melhorando o desempenho de 93% para mais de 98% de acerto. Também foram propostas diferentes estruturas para a rede interna, sendo a melhor com três camadas ocultas e aplicando a técnica de *DropOut*. Além disso, pode ser desenvolvidas outras alternativas para escolha de representantes e o aprimoramento dos extratores de características, melhorando a qualidade dos dados, sendo vantajoso reduzir a dimensão dos dados analisados para diminuir a complexidade da rede visando a redução de custos com uso de servidores e armazenamento de dados na nuvem.

Keywords: One shot learning; Siamese neural networks; Natural language processing; Classification; Machine learning; E-commerce.

Palavras-chaves: One shot learning. Redes neurais Siamesas. Processamento de linguagem natural. Classificação. Aprendizado de máquina. Comércio eletrônico.

1. INTRODUÇÃO

Com a pandemia causada pelo novo coronavírus (COVID-19), os hábitos de consumo da população mundial foram alterados, uma vez que o isolamento social foi adotado como forma de contenção do vírus (Kim, 2020). Em seu trabalho, Kim (2020) relata como a pandemia potencializou o uso do comércio eletrônico (*E-commerce*, em inglês), uma vez que as pessoas não se sentiam seguras de se dirigirem até as lojas físicas para fazerem suas compras.

Diante deste crescimento no número de usuários e nas inovações tecnológicas que têm tornado as transações feitas pela *internet* cada vez mais seguras, as empresas têm investido na divulgação de seus produtos em *marketplaces*, uma espécie de loja virtual, que permite o cadastro de vendedores e seus produtos. O fato de cada vendedor ter a possibilidade de cadastrar seu produto com a descrição que melhor o atenda, cria um problema para a classificação automática destes produtos em cada nicho. Se um produto é classificado no nicho errado, os consumidores podem não encontrá-lo, causando uma experiência de compra ruim e, em alguns casos, até a desistência do uso da plataforma de vendas (Chen and Wang, 2013).

Um *marketplace* pode utilizar algoritmos com inteligência artificial para fazer a categorização de anúncios cadastrados na plataforma de vendas. Entretanto, novas categorias podem surgir a todo momento, uma vez que produtos e serviços são lançados frequentemente. Assim sendo, é comum que existam classes com poucas amostras, o que torna complicado o processo de treinamento do modelo utilizado, pois ele precisaria ser retreinado a cada nova categoria incluída na loja virtual.

Entretanto, desenvolver algoritmos que consigam lidar com bases de dados em que uma ou mais classes possuem poucas amostras é uma tarefa difícil. Para lidar com tal desafio podem ser usados algoritmos que implementam técnicas de aprendizado *Few Shot Learning*, que fazem o treinamento de modelos em bancos de dados com classes que contém poucas amostras (Jadon and Garg, 2020). Aplicações desses algoritmos são encontradas em diversas áreas da literatura como o uso das *Matching Networks* no trabalho de Chaves et al. (2021) e as Redes Neurais Siamesas aplicadas no trabalho de Koch et al. (2015) para análise de imagens.

As redes siamesas são uma boa opção para lidar com o problema de classes desbalanceadas. Esse algoritmo realiza o cálculo de similaridade entre duas entradas e pode ser usado para determinar se tais amostras pertencem a mesma classe de acordo com a similaridade retornada pela rede Koch et al. (2015).

As redes siamesas possuem aplicações em diversas áreas, como no reconhecimento de voz, com o trabalho de Velez et al. (2018) que apresenta uma abordagem para identificação de voz em que a rede aprende a verificar se dois sinais de áudio são da mesma pessoa. Já no contexto de autenticação e identificação de condutores, de Souza et al. (2019) utilizam as redes siamesas em uma aplicação *one shot learning* para evitar retreinar os modelos a cada vez que um novo condutor for inserido no sistema.

Já na área de *E-commerce*, no trabalho de Sreepada and Patra (2020) foi proposto o uso das redes siamesas para um sistema de recomendação de produtos de uma plataforma de vendas *online*. O sistema de recomendações, inicialmente, coletou dados e agrupou as preferências dos usuários em diferentes *clusteres* e as redes siamesas foram usadas para aprender os interesses de cada usuário. Em seguida, um grupo específico de usuários recebia a recomendação de itens pouco populares na loja, de acordo com suas preferências, que foram observadas previamente pelo sistema desenvolvido. Após o modelo ser treinado, ele foi capaz de recomendar itens gerais e itens relevantes de cauda longa para cada usuário.

O trabalho apresentado neste artigo propõe a aplicação de um algoritmo *few shot learning* baseado em redes siamesas, para classificar amostras de uma plataforma de *E-commerce*. Como diferencial, tem-se a rede interna proposta com diferentes estruturas. Além disso foram propostos diferentes métodos para escolha do elemento representante enviado para uma das entradas da rede siamesa. A inovação do trabalho está na combinação da técnica de *few shot learning* com a rede siamesa usando *transfer learning* para um problema de *E-commerce*, no sentido de evitar o retreino de redes principais para classificação dos produtos.

O trabalho está organizado da seguinte forma: a Seção 2 apresenta os materiais utilizados, bem como a metodologia adotada para desenvolvimento do trabalho. A seção 3 apresenta as análises quantitativa e comparativa do sistema proposto. Por fim, a conclusão do trabalho de pesquisa com sugestões para futuros trabalhos estão descritas na seção 4.

2. MATERIAIS E MÉTODOS

2.1 Base de dados e recursos computacionais

A base de dados utilizada neste trabalho é proveniente de plataformas de comércio eletrônico e foi fornecida pela empresa parceira neste trabalho para desenvolvimento do projeto de extração de entidades em base de dados não estruturados. A base conta com 394 amostras, divididas em 34 classes, com distribuição de acordo com o mostrado na Tabela 1.

Tabela 1. Distribuição de amostras por classe

Amostras por classe	Número de classes
Menos de 3	18
De 3 a 8	10
De 20 a 60	5
Mais de 100	1

Cada amostra da base de dados é referente a um produto cadastrado no *marketplace*. Todas as amostras passam por um extrator de características desenvolvido pela empresa que usa técnicas de processamento de linguagem natural para transformar a descrição dos anúncios em valores numéricos que passam a representar este anúncio no espaço de características. Portanto, os dados enviados para o classificador proposto neste trabalho foram primeiramente processados por esse extrator, sendo que cada anúncio novo foi transformado em um vetor de 1200 números.

Os códigos computacionais foram implementados em linguagem *Python*, com utilização das bibliotecas *pandas*,

numpy e *TensorFlow* na maior parte do desenvolvimento. Esta linguagem foi escolhida por ser *open source* e por suas bibliotecas serem constantemente atualizadas, com correções e novas funcionalidades (Coelho and Richert, 2015). Foi utilizada a plataforma *Colab*, um produto do *Google Research*, que permite a execução de *scripts* com acesso a unidades de processamento gráfico (GPU) gratuitamente.

2.2 Algoritmos de inteligência artificial

Quando em um problema de categorização o classificador deve aprender tendo apenas uma ou poucas amostras de uma classe no treinamento, tem-se um problema de *one/few-shot learning*. Para um banco de dados com poucas classes, o algoritmo k-vizinhos mais próximos (KNN, do inglês *K-Nearest Neighbors*) pode apresentar um bom desempenho na tarefa de classificação.

O KNN é um algoritmo simples usado comumente em tarefas de classificação e reconhecimento de padrões. Uma das vantagens é que o KNN não necessita de treinamento para realizar a classificação pois ele identifica os k elementos já rotulados com menor distância ao elemento monitorado (Fukunage and Narendra, 1975). O algoritmo KNN foi escolhido para comparação com o modelo *few-shot learning* baseado em redes neurais siamesas.

A Rede Neural Siamesa foi proposta no trabalho de Bromley et al. (1994), sendo aplicada em um problema de análise de imagens para a autenticação de assinaturas. Na Figura 1 (Chaves et al., 2021) é apresentada a arquitetura simplificada de uma rede siamesa.

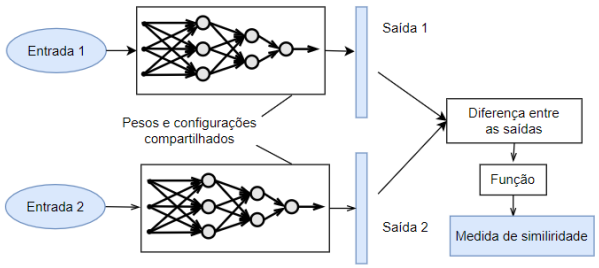

Figura 1. Arquitetura simplificada de uma rede siamesa (Fonte: Chaves et al. (2021))

A rede siamesa é formada por duas sub-redes idênticas. Essas redes, que possuem a mesma estrutura e recebem os mesmos pesos e parâmetros, levam as amostras recebidas em suas entradas para um novo espaço de características. Neste novo espaço, as amostras de mesma classe se encontram mais próximas e as amostras de classes diferentes são distanciadas uma das outras. Em seguida, é feito o cálculo da distância entre as duas amostras no espaço. Esse valor de distância é então processado por outra rede neural (ou um único neurônio), que tem como saída um valor entre 0 e 1. Se as amostras analisadas forem da mesma classe, o valor de similaridade calculado será mais próximo de 1, e se as amostras forem de classes diferentes, então esse valor será próximo de 0. No treinamento, esse valor é comparado com o alvo das entradas, é calculada a perda e os pesos das sub-redes são atualizados via *backpropagation* (Goodfellow et al., 2016).

A função de perda utilizada para o cálculo do erro foi a função de entropia cruzada (*cross entropy*), dada pela Equação 1, em que Y é o valor real da amostra (*label*) e P é a probabilidade da amostra pertencer à classe analisada.

$$Perda = -Y * log(P) \; - \; (1 - Y) * log(1 - P) \qquad (1)$$

Ao concluir o treinamento, as sub-redes aprenderam a determinar a similaridade entre as amostras recebidas e a rede siamesa pode ser usada como um classificador para definir se as amostras pertencem ou não a uma mesma classe.

2.3 Projeto do modelo

Os parâmetros para a rede siamesa foram definidos experimentalmente na etapa de projeto do modelo usando os dados do conjunto de treinamento, em que diferentes valores foram usados para os parâmetros, em uma busca manual, até que os melhores resultados do modelo fossem alcançados. Só foram usadas as classes que continham a partir de 20 amostras para o treinamento e teste do modelo. Além disso, a base de dados foi dividida seguindo a proporção 70/30, ficando 227 amostras para o treino e o conjunto de teste ficou com 98 amostras. Para validar o desempenho do modelo no treino da mesma forma que é analisado durante a escolha de representantes na etapa de testes, foi criado o conjunto de dados de validação, composto por 50% das amostras da base de treino.

O número de épocas foi variado no intervalo de 10 a 1000 épocas, tendo o melhor resultado com 100 épocas. Foram testados todos os otimizadores da biblioteca *Keras* e o que obteve o melhor resultado foi o *Adam* com taxa de aprendizagem de 0,0008 (a taxa foi variada de 0,0001 a 0,001).

Foram testadas diferentes estruturas para a rede interna da rede siamesa, tendo como base uma rede do tipo Perceptron multi-camadas (MLP, do inglês *multilayer Perceptron*) (Rosenblatt, 1958). A estrutura mais simples foi de uma camada oculta, que será chamada estrutura MLP Camada Única (MLPCU). Para a rede interna o número de neurônios foi variado de 5 a 600, tendo o melhor resultado com 20 neurônios na camada.

Para mudança de topologia na rede interna, foram adicionadas mais duas camadas ocultas com função de ativação *relu*, sendo que a primeira camada ficou com 100 neurônios, a segunda camada com 50 neurônios e a última camada com 25 neurônios. Essa abordagem foi identificada como estrutura MLP Três Camadas (MLPTC). Tais números foram definidos experimentalmente visando o melhor desempenho do modelo e também que a última camada tivesse dimensão semelhante à utilizada na MLPTC.

Para entender o impacto do aumento do número de camadas no desempenho do modelo, foi adicionada mais uma camada à estrutura MLPTC, com 25 neurônios, sendo esta estrutura chamada de MLP Quatro Camadas (MLPQC).

Além disso, foi implementada uma camada de *Dropout* que elimina aleatoriamente a contribuição de alguns neurônios ocultos na rede (Sreepada and Patra, 2020). Essa camada foi colocada antes da primeira camada da es-

trutura MLPTC e foi identificada como MLPDCV. Em outra abordagem identificada como MLPDCO, a camada de *DropOut* foi colocada entre a primeira e a segunda camada oculta da estrutura MLPTC. Em ambos os casos, a taxa de aprendizagem foi aumentada para 0,001.

As siglas e descrições das arquiteturas descritas nos parágrafos anteriores foram colocadas na Tabela 2 para referência e organização das informações apresentadas.

Tabela 2. Siglas e descrições das arquiteturas utilizadas

Siglas	Descrição
MLPCU	MLP Camada Única
MLPTC	MLP Três Camadas
MLPDCV	MLP *DropOut* Camada Visível
MLPDCO	MLP *DropOut* Camada Oculta
MLPQC	MLP Quatro Camadas

2.4 Escolha de representantes

O desempenho de um modelo utilizando a rede neural siamesa está diretamente relacionado com a qualidade do elemento utilizado como representante de cada classe na primeira entrada da rede, uma vez que a rede usa tal elemento para calcular a distância com o elemento desconhecido recebido em sua segunda entrada. Nem todas as amostras de uma classe são bons representantes, já que podem haver *outliers* ou amostras que se localizam nas bordas do *cluster*.

O modelo foi previamente treinado e os representantes foram escolhidos dentre as amostras utilizadas para treinamento da rede. Os pares de amostras enviados para a rede siamesa foram montados de forma que, para cada representante usado como elemento de referência, cada amostra da base de teste foi colocada como elemento monitorado.

O primeiro método de definição de representantes foi com a escolha feita de modo aleatório dentre as amostras disponíveis. Para analisar a influência do representante aleatório, foram escolhidos e testados 20 vezes. O próximo método para definir o representante foi pelo cálculo do centroide de cada classe usando os dados de treinamento do modelo.

Outro método foi o uso de uma estrutura de *ensembles*, na qual é possível escolher n representantes para cada classe (González et al., 2020). Assim sendo, existe a possibilidade de que sejam selecionados melhores representantes, considerando que uma classe pode não ser homogênea e ter as amostras distribuídas em diferentes locais no espaço de características. Foram montados *ensembles* com 3 e 5 representantes para cada classe, sendo os elementos escolhidos aleatoriamente. Assim como na abordagem com elemento aleatório, aqui também os elementos são escolhidos e testados 20 vezes.

Para utilizar uma métrica na escolha dos elementos para o *ensemble* foi escolhido aplicar o algoritmo *k-Means* (MacQueen et al., 1967). Tal algoritmo divide as classes em n *clusters* e calcula o centroide para cada uma. Porém a posição dos pontos finais pode variar de acordo com a divisão inicial feita pelo algoritmo. Os centroides calculados pelo *k-Means* foram utilizados como representantes para o *ensemble*, sendo testados os cenários com 3 e 5 representantes.

Para cada *ensemble* de representantes, a decisão da similaridade entre a amostra não rotulada e a classe do *ensemble* é feita por voto majoritário. Ou seja, depois de definir a similaridade entre cada representante, são necessários a metade de votos mais um, para que seja atribuído o valor 0 ou 1 como similaridade daquele *ensemble*.

3. ANÁLISE E DISCUSSÃO DOS RESULTADOS

Para o teste usando o algoritmo KNN, na base de teste estavam os elementos a serem rotulados e a base de treino continha os elementos que foram usados como vizinhos disponíveis para escolha do algoritmo. Foi alcançada uma acurácia de 92,85%.

Para a rede siamesa, cada estrutura da rede interna foi testada considerando o mesmo conjunto de dados. Para que uma amostra fosse considerada similar a outra, ou seja, pertencente à uma mesma classe, o valor de similaridade calculado pela rede siamesa precisava ser maior que o limiar de arredondamento de 0,5. Com os valores arredondados para 0 ou 1, o desempenho do modelo foi calculado, considerando que o modelo acertou a classificação quando o valor arredondado era igual ao alvo das entradas.

A primeira abordagem foi com o elemento escolhido de forma aleatório dentre as amostras utilizadas no treinamento, sendo que o teste foi repetido 20 vezes para observar o desempenho do modelo com diferentes representantes. A abordagem seguinte foi com o uso do centroide calculado com os dados de treinamento de cada classe. Uma vez que a base de treinamento não sofreu alterações, o centroide foi calculado apenas uma vez, não havendo desvio padrão a ser contabilizado. Os resultados de acurácia do modelo com representante aleatório e com o centroide são mostrados na Tabela 3, sendo a acurácia mostrada em porcentagem nas colunas Aleatório (%) e Centroide (%).

Tabela 3. Resultados de acurácia com representante aleatório e centroide calculado com a base de treinamento

Estrutura	Aleatório (%)	Centroide (%)
MLPCU	93,88 +- 1,98	98,29
MLPTC	96,35 +- 0,95	98,46
MLPDCV	97,70 +- 0,41	98,29
MLPDCO	97,07 +- 0,41	98,12
MLPQC	96,32 +- 0,63	97,78

Quando comparados ao resultado obtido com o KNN, os resultados com a escolha aleatória são melhores do que o KNN em todas as estruturas. Porém, quando considerado o desvio padrão da escolha aleatória, o resultado da estrutura MLPCU pode ficar inferior ao KNN com alguns representantes.

Por isso, vê-se a importância da escolha de representante já que alguns escolhidos aleatoriamente podem não representar a classe, por ser elementos que estão localizados na borda do *cluster*. Além disso, na escolha aleatória não há controle sobre a repetitividade das amostras utilizadas, logo, a mesma amostra pode ter sido escolhida como representante mais de uma vez. Isto certamento acontece nas classes com menos de 20 amostras na base de treinamento.

Caso esta amostra esteja representando pobremente sua classe, o desempenho do modelo não será tão bom quanto usando outras amostras.

No caso do representante feito com o cálculo do centroide das amostras de treinamento, ele será um bom representante caso a classe tenha uma distribuição gaussiana. Todas as estruturas superaram o uso do KNN, superando 98% de acurácia em todas as estruturas, exceto na MLPQC. Isto pode indicar que aumentar o número de camadas não garante que o modelo vá ter um aprendizado melhor e, consequentemente, um melhor desempenho. Quando comparado ao desempenho do representante aleatório, o centroide ainda é a melhor opção, sendo que com a estrutura MLPDCV, considerando o desvio padrão, foi a estrutura em que o representante aleatório se aproximou mais do desempenho do centroide.

Para a abordagem com *ensemble* de representantes, foram escolhidos aleatoriamente 3 representantes na base de treinamento. Em seguida, foi montado o *ensemble* de cada classe. As amostras recebidas na entrada da rede siamesa foram comparadas com todos os representantes. Para cada *ensemble*, a decisão do valor de similaridade final foi feito pelo voto majoritário após o arredondamento do valor de similaridade obtido com cada representante e a amostra recebida pela rede. O valor final foi então comparado ao alvo definido entre a classe da amostra desconhecida e a classe do *ensemble* para o cálculo de similaridade.

A escolha e teste dos representantes foram repetidos 20 vezes, já que os representantes são aleatórios. O mesmo procedimento foi adotado para o caso em que foram usados 5 representantes. Os resultados de média e desvio padrão da execução de ambos experimentos foram colocados na Tabela 4.

Tabela 4. Resultados de acurácia com *ensemble* de representantes escolhidos aleatoriamente

Estrutura	3 representantes (%)	5 representantes (%)
MLPCU	97,12 +- 1,05	97,62 +- 0,51
MLPTC	97,87 +- 0,47	97,93 +- 0,40
MLPDCV	98,06 +- 0,52	98,43 +- 0,40
MLPDCO	97,96 +- 0,24	98,04 +- 0,27
MLPQC	97,53 +- 0,44	97,72 +- 0,39

Os resultados com *ensemble* de representantes superaram os resultados obtidos com KNN e com o representante aleatório. O desvio padrão para a estrutura mais simples (MLPCU) foi mais alto do que nas outras estruturas, assim como aconteceu com o representante aleatório, o que pode indicar que as estruturas mais complexas levam as amostras para um espaço de características onde as amostras de mesma classe estão mais próximas umas das outras e a escolha aleatória tem menor impacto na acurácia do modelo.

Todas as estruturas obtiveram mais de 97% de acurácia, o que é um resultado bem próximo ao centroide (vide Tabela 3), em que todas exceto a MLPQC obtiveram resultado maior do que 98%. Em relação ao número de representantes, a acurácia do modelo aumentou com o aumento do número de representantes em todas as estruturas. Porém em algumas delas foi um aumento pequeno, como na MLPDCO que foi de menos de 0,10%. Assim sendo, é um ponto de atenção a escolha do número ideal de representantes pois um representante pode não retratar todas as informações pertinentes à classe e, o uso de muitos representantes pode trazer uma complexidade desnecessária ao modelo e não refletir um ganho na acurácia.

Para melhorar o desempenho do algoritmo foi estudada o uso de uma métrica para definição dos representantes. Foi usado o algoritmo *k-Means*, que dividiu cada classe na base de treinamento em n *clusters* e os representantes escolhidos foram seus centroides. A formação de pares e contabilização do resultado foram feitas da mesma maneira que relatado para o caso do *ensemble* com elementos aleatórios. O primeiro teste foi feito com 3 representantes e, em seguida foram feitos os testes com 5 representantes.

É importante ressaltar que, inicialmente, foram feitas 20 execuções do teste, de maneira semelhante aos testes com representantes aleatórios. Porém não houve um desvio padrão significativo ao comparar o resultado das diversas iterações. Assim sendo, foi optado por não colocar o desvio padrão junto aos resultados da Tabela 5.

Tabela 5. Resultados de acurácia com *ensemble* de representantes escolhidos via algoritmo *k-Means*

Estrutura	3 representantes (%)	5 representantes (%)
MLPCU	95,74	96,25
MLPTC	97,44	97,95
MLPDCV	98,46	98,63
MLPDCO	98,29	98,29
MLPQC	97,61	97,95

Os resultados com a utilização do algoritmo *k-Means* para escolha de representantes resultou em uma acurácia maior do que com a escolha aleatória e superou o *ensemble* aleatório com as estruturas MLPDCV e MLPDCO. O *ensemble* com representante escolhido via *k-Means* só superou o resultado do centroide na estrutura MLPDCV.

Deve ser lembrado que ao aumentar o número de representantes n, são excluídas algumas classes do teste, uma vez que são necessárias pelo menos n amostras de cada classe na base de dados de treinamento para que sejam escolhidos os representantes para o *ensemble*. Assim sendo, é fundamental estabelecer o limite até onde é vantajoso aumentar o número de representantes em detrimento da exclusão de algumas classes.

Em relação à estrutura utilizada na rede interna da rede siamesa, houve aumento na acurácia do modelo em todos os casos com a estrutura MLPDCV. Em todos os cenários utilizando a rede siamesa o desempenho foi de pelo menos 93%, o que é um bom resultado considerando que algumas classes continham apenas 5 amostras e o modelo aprendeu a diferenciar se uma amostra pertence ou não a uma classe.

A estrutura MLPQC apresentou um desempenho menor que as estruturas de três camadas com *DropOut* (MLPDCV e MLPDCO) quando usada com *ensembles*, o que indica que aumentar o número de camadas da rede não necessariamente resulta em melhores resultados. Uma rede mais simples é desejada, uma vez que com menos pesos e parâmetros para calcular o algoritmo terá uma execução

mais rápida, resultando em menos tempo gasto com uso de servidores em nuvem e tarefas de processamento de dados.

4. CONCLUSÃO

O uso de inteligência artificial em aplicações de comércio eletrônico traz benefícios para as empresas envolvidas nas vendas de seus produtos que melhoram seu alcance e também para os consumidores que podem contar com serviços de melhor qualidade. Para o desenvolvimento de soluções aplicadas ao *E-commerce* são necessárias grandes bases de dados, com um elevado número de amostras. Entretanto, nem sempre essas informações estão disponíveis, seja por existirem produtos muito específicos de algum nicho ou por alguma questão de confidencialidade das informações. Neste caso, aplicações com aprendizado *few shot learning* são interessantes por permitirem o aprendizado de modelos com poucas amostras de cada classe.

A aplicação de algoritmos *few shot learning* associados ao processamento de linguagem natural ainda é uma área pouco explorada, pois os estudos com tais algoritmos se concentram em problemas que envolvem a análise de imagens. O desenvolvimento de um algoritmo baseado nas redes siamesas proposto neste trabalho trouxe bons resultados, sendo possível alcançar uma acurácia de 98% utilizando uma base de dados com menos de 400 amostras divididas em 34 classes de forma não balanceada.

Para a escolha do melhor representante foram testadas as abordagens de escolha aleatória e cálculo do centroide para cada classe. Outra abordagem foi a criação de um *ensemble* de representantes aleatórios e também a utilização do *k-Means* para escolha dos representantes. O melhor resultado foi com o uso do centroide, que melhorou o desempenho de 93% no pior cenário da escolha aleatória para mais de 98% de acerto do modelo, que pode ser a melhor opção para classes com distribuição gaussiana, o que não pode ser garantido neste caso. Na abordagem utilizando *ensemble* de 5 representantes escolhidos via *k-Means* com a estrutura MLPDCV, o resultado superou o centroide, porém são analisadas menos classes, uma vez que só podem ser usadas as classes com mais de 5 amostras.

Além disso, foi proposta a alteração da rede interna da rede siamesa. O uso de uma rede variando o número de camadas ocultas trouxe impacto no desempenho do modelo independente do método de escolha de representantes empregado. O melhor resultado foi utilizando uma estrutura de três camadas ocultas e aplicando a técnica de *DropOut* na primeira camada oculta da rede.

Como trabalhos futuros, visando o melhor desempenho da rede siamesa, podem ser estudadas alternativas para escolha de representantes, como o cálculo do representante após o processamento pela rede interna, uma vez que as amostras estarão em um espaço de características onde a distância entre amostras da mesma classe é minimizado. Além disso, são necessários testes em bases de dados maiores para estudar como o modelo proposto se comporta ao lidar com um grande volume de dados.

AGRADECIMENTOS

Os autores agradecem à empresa *Omnilogic* Inteligência S/A e também à CAPES, CNPq e 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.

Chaves, A.G.S., e Silva, F.C., Ferreira, D.D., Barbosa, B.H.G., and Nascimento, S.T. (2021). Extração de entidades de produtos utilizando técnicas de few-shot learning. *Anais do XV Simpósio Brasileiro de Automação Inteligente (SBAI)*, 1. doi:10.20906/sbai.v1i1.2771.

Chen, L. and Wang, F. (2013). Preference-based clustering reviews for augmenting e-commerce recommendation. *Knowledge-Based Systems*, 50, 44–59.

Coelho, L. and Richert, W. (2015). *Building Machine Learning Systems with Python - Second Edition*. Community experience distilled. Packt Publishing, Birmingham, Reino Unido.

de Souza, A., Lacerda, W., Forster, C., Lima, D., and Nazaré, T. (2019). Aplicação de redes neurais siamesas na autenticação de condutores. In *Anais do 14º Simpósio Brasileiro de Automação Inteligente (SBAI)*. SBAI, Ouro Preto, MG, Brasil.

Fukunage, K. and Narendra, P.M. (1975). A branch and bound algorithm for computing k-nearest neighbors. *IEEE Trans. Comput.*, 24(7), 750–753.

González, S., García, S., Ser, J.D., Rokach, L., and Herrera, F. (2020). A practical tutorial on bagging and boosting based ensembles for machine learning: Algorithms, software tools, performance study, practical perspectives and opportunities. *Information Fusion*, 64.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

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

Kim, R.Y. (2020). The impact of covid-19 on consumers: Preparing for digital sales. *IEEE Engineering Management Review*, 48(3), 212–218.

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*.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, 281–297. Oakland, USA, Cambridge University Press, Oakland, CA, USA.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6), 386–408.

Sreepada, R.S. and Patra, B.K. (2020). Mitigating long tail effect in recommendations using few shot learning technique. *Expert Systems with Applications*, 140, 112887.

Velez, I., Rascon, C., and Fuentes-Pineda, G. (2018). One-shot speaker identification for a service robot using a cnn-based generic verifier. *Proceedings of the International Conference on Robotics and Automation (ICRA 2018)*.