HubISC: um novo algoritmo baseado em *hubness* para a classificação de fluxo de dados de imagens*

Mateus C. de Lima, Elaine R. Faria, Maria Camila N. Barioni

Faculdade de Computação – Universidade Federal de Uberlândia (UFU)
Uberlândia – MG – Brasil

{mateuscurcino,camila.barioni,elaine}@ufu.br

Abstract. *Image data stream classification presents several challenges. One of the challenges inherent in the domain of image data is the high dimensionality of the data, which can cause the curse of dimensionality. Studies applied in other contexts efficiently employ an aspect, called hubness, inherent in high-dimensional data. This article introduces a new algorithm for image data stream classification, which incorporates the hubness aspect. The experiment results show a good cost-benefit of the algorithm in terms of efficacy and the percentage of labeled instances in relation to commonly used algorithms for image data stream classification.*

Resumo. *A classificação de fluxo de dados de imagens apresenta vários desafios. Um dos desafios inerente ao domínio de dados de imagens é a alta dimensionalidade dos dados, que pode ocasionar a chamada maldição da dimensionalidade. Trabalhos aplicados em outros contextos empregam de forma eficiente um aspecto, denominado hubness, inerente de dados de alta dimensão. Este artigo apresenta um novo algoritmo para a classificação de fluxo de dados de imagens, que incorpora o aspecto hubness. Os resultados dos experimentos mostram uma boa relação de custo e benefício do algoritmo em termos de eficácia e da porcentagem de instâncias rotuladas em relação aos algoritmos comumente usados para a classificação de fluxos de dados de imagens.*

1. Introdução

Atualmente, com a alta disponibilidade de dispositivos de aquisição de imagens, surgiu a necessidade do desenvolvimento de estratégias para lidar com fluxos de dados de imagens. Fluxos de dados consistem em massivas quantidades de dados, geradas em curto espaço de tempo, de maneira contínua e potencialmente infinita [Silva et al. 2013]. Dentre os exemplos de aplicações de fluxos de dados de imagens estão presentes a classificação de objetos em tempo real, reconhecimento de faces e detecção de movimentos. Embora muitos esforços de pesquisa tenham sido feitos, ainda existem questões em aberto relacionadas a tarefas que lidam com a manipulação de fluxos de dados de imagens.

O surgimento de novas classes (*concept-evolution* ou *open-set*) [Wang et al. 2019] e a mudança na distribuição dos dados (*concept-drift*) [Nguyen et al. 2015] são desafios comuns em ambientes de fluxos de dados. Para a classificação de fluxos de dados de

*O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001

imagens, além dos desafios já mencionados, os métodos utilizados para a extração de características também merecem atenção. Nesse contexto, um desafio é a alta dimensionalidade dos dados. Os vetores gerados para a representação das imagens podem conter um alto número de atributos, causando a maldição da dimensionalidade [Samet 2005], que pode ter um impacto negativo em algoritmos baseados em distâncias.

Tradicionalmente, a questão da alta dimensionalidade tem sido tratada na literatura científica da área, incluindo a classificação de imagens, com a utilização de estratégias que procuram atenuar os efeitos da maldição da dimensionalidade, realizando a redução de dimensionalidade por meio de extração e de seleção de atributos [Wang et al. 2021]. Outra maneira explorada para minimizar esse desafio, pode ser a utilização de descritores considerados estado da arte para imagens, por exemplo, as CNNs *(Convolutional Neural Networks)* [Alzubaidi et al. 2021, Wang et al. 2019]. Esse descritor de características permite selecionar diferentes camadas de rede para gerar os vetores de características. Os vetores de camadas mais externas, geralmente, possuem menor dimensionalidade. Enquanto os vetores gerados por camadas mais internas têm maior dimensão. De uma maneira oposta, outras estratégias [Romaszewski et al. 2018, Wu et al. 2020] têm empregado de maneira eficaz um aspecto inerente aos dados de alta dimensão, denominado *hubness*, para o desenvolvimento de técnicas que permitem a classificação em bancos de dados de alta dimensão. O aspecto *hubness* consiste na tendência de algumas instâncias de dados, chamadas *hubs*, ocorrerem com maior frequência nas listas dos K-vizinhos mais próximos de outras instâncias [Mani et al. 2019]. Até onde é de conhecimento dos autores, apesar do potencial demonstrado para a classificação de outros contextos, o aspecto *hubness* não foi explorado para a classificação de fluxos de dados de imagens.

A fim de tratar a presença de *concept-drift* e *open-set* na classificação de fluxos de dados de imagens, o modelo de decisão precisa ser atualizado constantemente. No geral, os trabalhos consideram processos supervisionados, com a necessidade de um usuário especialista capaz de fornecer rótulos de instâncias de dados, sendo um processo custoso e que demanda tempo. Alguns recentes trabalhos [Parreira and Prati 2019, Pham et al. 2021] optaram por usar técnicas de aprendizado ativo a fim de escolher um conjunto interessante de instâncias a serem rotuladas pelo especialista para atualizar o modelo de classificação. Dessa forma, o aspecto *hubness* também pode demonstrar uma importante contribuição para o processo de seleção de instâncias a serem rotuladas. Geralmente, os *hubs* estão localizados em regiões densas, com outras instâncias, o que pode facilitar a propagação de rótulos para as demais instâncias a partir do rótulo *hub*. Então, o aspecto *hubness* além de tratar a questão da alta dimensionalidade, pode também representar uma estrutura de sumarização das instâncias.

O trabalho discutido aqui apresenta um novo algoritmo para a classificação de fluxos de dados de imagens que considera o aspecto *hubness*. Para tanto, o método experimental do trabalho foi definido para responder a seguinte questão de pesquisa: *como a utilização do aspecto hubness na classificação de fluxos de dados de imagens impacta na eficácia e na escolha das instâncias a serem rotuladas quando comparado com outros trabalhos correlatos?*.

O restante do artigo está organizado da seguinte maneira: a Seção 2 apresenta os conceitos fundamentais; a Seção 3 discute os trabalhos correlatos; o algoritmo desenvolvido para a classificação de fluxos de dados de imagens é descrito na Seção 4; a discussão

sobre os resultados obtidos é descrita na Seção 5. Finalmente, as conclusões e os trabalhos futuros são apresentados na Seção 6.

2. Conceitos Fundamentais

Fluxos de dados consistem em massivas quantidades de dados, geradas em curto espaço de tempo, de maneira contínua e potencialmente infinita [Silva et al. 2013]. Formalmente, um fluxo de dados de imagens S, é uma sequência massiva de imagens i_1, i_2, ..., i_n, potencialmente infinita, que chega nos *timestamps* t_1, t_2, ... ,t_n, sendo cada imagem i_z descrita por um vetor de características $\vec{\boldsymbol{x}}$.

Formalmente, o problema de classificação de fluxos de dados de imagens pode ser definido da seguinte forma: dado um conjunto inicial de imagens da forma $(\vec{\boldsymbol{x}}, y)$, onde $\vec{\boldsymbol{x}} = x_1, ..., x_d$ é um vetor de características, obtido por um método de extração de características. O vetor $\vec{\boldsymbol{x}}$ descreve uma imagem i e y representa o rótulo de uma classe discreta de um conjunto C com n_C diferentes classes. O classificador cria um modelo $y = f(\vec{\boldsymbol{x}})$ para prever um rótulo y para cada futuro exemplo de imagem pertencente a um fluxo de dados de imagens (S). Entretanto, em S podem surgir w novas classes de imagens (*open-set*). Dessa forma, a quantidade de classes do problema se torna $n_C + w$. Além disso, pode ocorrer a mudança de conceito das classes já conhecidas (*concept-drift*).

Para lidar com os fenômenos *open-set* e *concept-drift* os algoritmos de classificação de fluxo de dados de imagens devem ser capazes de detectar novas classes e também as mudanças de conceitos das classes já conhecidas. Dessa forma, o modelo de decisão f não pode ser treinado uma única vez, já que no teste do modelo de decisão podem aparecer $n_C + w$ classes, além de mudanças nas n_C classes já conhecidas. Portanto, o modelo de decisão f deve ser treinado incrementalmente com o fluxo de imagens S, para considerar também as alterações de classes.

Dentre os descritores considerados estado da arte para a extração de características das imagens e geração do vetor $\vec{\boldsymbol{x}}$ estão as CNNs *(Convolutional Neural Networks)*. Uma questão importante sobre o uso desses descritores está relacionada com a escolha da camada da CNN para a geração do vetor $\vec{\boldsymbol{x}}$ [Alzubaidi et al. 2021]. As camadas mais internas da CNN geram vetores que representam mais detalhes das imagens, por exemplo: texturas, bordas e formas. Entretanto, esses vetores apresentam alta dimensionalidade. As camadas mais externas da CNN possibilitam a geração de vetores com menor dimensionalidade, porém os vetores apresentam níveis de detalhes inferiores, ou seja, algumas informações importantes das imagens podem ser perdidas.

Uma das abordagens comumente usadas para classificação de imagens é a utilização de algoritmos de classificação baseados em protótipos [Nguyen et al. 2015]. Esses algoritmos são interessantes para a aplicação em fluxos de dados de imagens, pois, no geral, possuem características incrementais, ou seja, a possibilidade de adicionar ou remover classes de imagens sem a necessidade de reconstruir o modelo, o que pode facilitar o tratamento dos fenômenos *concept-drift* e *open-set*. Geralmente, esses algoritmos sumarizam as instâncias por meio da utilização de microgrupos, centróides e medóides.

O algoritmo *k-Nearest Neighbor (kNN)* é um dos algoritmos baseados em protótipos mais utilizados para problemas de classificação. Outra estratégia baseada em protótipos, comumente usada para a classificação de fluxos de dados de imagens, é conhecida como NCM (*Neareast Class Mean*) [Rebuffi et al. 2017]. Essa estratégia considera

a utilização de um centróide (μ_j) para representar cada classe de imagens (C_j), conforme a Equação 1.

$$\mu_j = \frac{1}{n_{C_j}} \sum_{x^d \in C_j} x^d \quad (1) \qquad\qquad c^* = \underset{c \in C}{\operatorname{argmin}}\, m(x, \mu_c) \qquad (2)$$

Tradicionalmente, os algoritmos baseados em protótipos, utilizam funções de distância para a atribuição de uma imagem a sua respectiva classe, por exemplo, no algoritmo NCM. Nesse algoritmo, a imagem i, representada por um vetor de características $\vec{\boldsymbol{x}}$, é classificada de acordo com a distância m de $\vec{\boldsymbol{x}}$ em relação ao centróide μ de cada classe de imagem, conforme a Equação 2. Dessa forma, observa-se que esses algoritmos podem sofrer os efeitos da maldição da dimensionalidade. De maneira a minimizar esses impactos, uma estratégia pode ser considerar o aspecto *hubness*.

Hubness é um aspecto da *maldição da dimensionalidade*. *Hubs* são instâncias de dados que aparecem na lista de vizinhos mais próximos de um grande número de outras instâncias [Romaszewski et al. 2018]. Por outro lado, outras instâncias de dados tornam-se *anti-hubs*, quando raramente ou nunca são os vizinhos mais próximos de outras instâncias. Para tanto, é necessário calcular a pontuação *hubness* ($h_K((i))$) de cada instância: seja $X = \{i_1; ...; i_n\}$ um conjunto de instâncias de dados, $h_K(i)$ representa o número de K-ocorrências de instâncias $i \in X$, isto é, o número de vezes que i ocorre na listagem dos K-vizinhos mais próximos de outras instâncias de dados pertencentes a X.

Tem sido demonstrado que o conceito *hubness* consiste em uma propriedade inerente de grande parte dos conjuntos de dados de alta dimensão, não sendo peculiar de conjuntos de dados específicos. O aspecto *hubness* pode indicar uma boa referência de pontos de centralidade [Mani et al. 2019]. Os principais *hubs* podem ser usados efetivamente como representantes para guiar o processo de atribuição de rótulos de imagens, de maneira similar a como ocorre nas técnicas que consideram outros tipos de representantes, por exemplo, centróides.

3. Trabalhos relacionados

Vários trabalhos correlatos têm considerado a utilização de estratégias baseadas em protótipos para a classificação de fluxos de dados de imagens [Hu et al. 2017, Ristin et al. 2014, Rebuffi et al. 2017]. O trabalho descrito em [Hu et al. 2017] utilizou o algoritmo NCM para a classificação de fluxos de imagens pessoais, já o trabalho [Ristin et al. 2014] definiu uma abordagem que integrou os algoritmos NCM e *Random Forest* para a classificação de grandes conjuntos de dados. O foco desses trabalhos foi avaliar avanços nos classificadores para lidar com os fenômenos *concept-drift* e *open-set*. O estudo descrito em [Rebuffi et al. 2017] desenvolveu uma variação do algoritmo NCM, chamada iCaRL (*Incremental Classifier and Representation Learning*). Essa variação considera múltiplos centróides para cada classe de imagens.

O trabalho apresentando em [Castro et al. 2018] propõe uma única arquitetura de CNN para a representação das imagens e a classificação, isto é, a mesma CNN foi utilizada tanto para a representação e atualização dos descritores de características quanto para a classificação das imagens. Apesar das importantes contribuições dos trabalhos

realizados em [Rebuffi et al. 2017, Castro et al. 2018], a questão da alta dimensionalidade dos dados não foi discutida. Outra estratégia baseada em protótipos foi descrita em [Wang et al. 2019]. Esse estratégia descreve um framework, chamado CPE (*CNN-based Prototype Ensemble*), para a classificação de fluxos de dados de imagens. O problema da alta dimensionalidade dos dados foi abordado, por meio da utilização de uma CNN capaz de gerar vetores de características com um número reduzido de atributos. Nessa CNN, a representação das imagens é continuamente atualizada, isto é, após a descrição gerada pela rede, o processo de redução da dimensionalidade continua durante todo o processamento do fluxo de dados de imagem. Além disso, o algoritmo CPE considera múltiplos protótipos na fase de classificação para representar as classes de imagens. Dessa forma, a utilização do aspecto *hubness* se mostra interessante, pois os *hubs* podem ser considerados tanto para o problema da maldição da dimensionalidade, quanto para indicar possíveis instâncias para a atribuição de rótulos e serem consideradas representantes das classes.

Uma importante questão sobre os trabalhos [Hu et al. 2017, Ristin et al. 2014, Rebuffi et al. 2017, Castro et al. 2018] é que os rótulos de todas as imagens estão disponíveis imediatamente, o que pode ser incompatível com cenários de aplicações reais. Para fluxos de dados de imagens, alguns métodos de aprendizado ativo para a escolha das melhores instâncias a serem rotuladas foram definidos em [Žliobaitė et al. 2014, Parreira and Prati 2019]. As técnicas de [Žliobaitė et al. 2014] e [Parreira and Prati 2019] que utilizam funções de distância podem ter o desempenho degradado em contextos de alta dimensão. Portanto, é necessário investigar se essas estratégias podem ser aplicadas a imagens e analisar se o aspecto *hubness* pode ser capaz de auxiliar nessa extensão.

A utilização do conceito *hubness* tem sido observada em muitas tarefas, tais como: agrupamento de dados, recuperação da informação, classificação de imagens e detecção de ruídos. O trabalho descrito em [Mani et al. 2019] realizou um estudo do potencial do aspecto *hubness* para a sua aplicação nessas tarefas. Os trabalhos [Romaszewski et al. 2018, Wu et al. 2020] consideraram o aspecto *hubness* para a classificação de conjuntos de dados de outros contextos. Além disso, em [Tomašev et al. 2014] o aspecto *hubness* foi utilizado para a classificação de imagens em cenários estáticos.

4. HubISC

O algoritmo de classificação de fluxos de dados de imagens apresentado neste artigo tem como objetivo incorporar o aspecto *hubness* para a realização dessa tarefa. Além de tratar a alta dimensionalidade dos dados, esse algoritmo também tem o objetivo de tratar outros aspectos importantes nesse contexto, como: *I)* surgimento de novas classes (*open-set*), isto é, na etapa de testes surgem novas classes diferentes das disponíveis na fase de treinamento; *II)* evolução dos conceitos de classes já existentes (*concept-drift*); *III)* utilização de diferentes quantidades de imagens rotuladas para a atualização do modelo de decisão. Para tanto, o algoritmo HubISC *(Hubness for Image data Stream Classification)* possui duas fases: fase *offline* para a construção inicial do modelo de decisão; fase *online* para o tratamento do fluxo de dados de imagens. O Algoritmo 1 apresenta uma visão geral do algoritmo HubISC.

Algoritmo 1: HubISC

Entrada: X conjunto de imagens rotulado, K vizinhança para cálculo *hubness*, h_K limiar *hubness*, S fluxo de imagens, L limiar de distância, T tamanho do *buffer*
Saída: m do modelo de classificação f

1 **início**
  `/* Fase offline */`
2 $X' \leftarrow$ separa as instâncias de X de acordo com a classe;
3 $H \leftarrow$ identifica os hubs de X' a partir de em K e h_K;
4 $f \leftarrow$ constroi o modelo de decisão inicial a partir de H;
  `/* Fase online */`
5 $B \leftarrow \emptyset$;
6 **repita**
7 recebe $\vec{\boldsymbol{x}}$ por meio de S;
  `/*` $\vec{\boldsymbol{x}}$ `é avaliado por` f `*/`
8 **se** *a distância de* $\vec{\boldsymbol{x}}$ *em relação ao hub* $\vec{\boldsymbol{h}}$ *mais próximo* $\leq L$ **então**
9 y de $\vec{\boldsymbol{x}} \leftarrow y$ de $\vec{\boldsymbol{h}}$;
10 **senão**
11 $B \leftarrow B \cup \vec{\boldsymbol{x}}$;
12 **se** $|B|$ == T **então**
13 $H^{'} \leftarrow$ identifica os hubs $h^{'} \in B$ a partir de em K e h_K;
14 solicita os rótulos y de $H^{'}$ e propaga para $(B - H^{'})$;
  `/* atualiza incrementalmente o modelo de decisão a partir de` H `*/`
15 $H \leftarrow H \cup H^{'}$;
16 $B \leftarrow \emptyset$;
17 **até** *o fim de* S ;
18 **fim**

4.1. Fase *offline*

Na fase *offline*, o algoritmo HubISC recebe como parâmetros: um conjunto inicial de imagens X rotulado, com c classes e q imagens para cada classe; um valor K para o cálculo dos vizinhos mais próximos; um limiar h_K que determina a pontuação *hubness* mínima para uma instância ser considera *hub*. Cada imagem $i \in X$ deve ser representada por um vetor $\vec{\boldsymbol{x}}$, após passar por um método de extração de características.

Observa-se na linha 2 do Algoritmo 1, que o conjunto de imagens é organizado por classes. Na linha 3 cada imagem $i \in X^{'}$, representada pelo vetor $\vec{\boldsymbol{x}}$, terá a sua respectiva pontuação *hubness* calculada a partir do valor K. A partir desse cálculo, ocorre a identificação dos *hubs*. Caso a pontuação *hubness* da respectiva imagem seja superior ao parâmetro h_K, a imagem i é considerada um *hub*. Cada classe de imagem pode ter mais de um *hub* associado. Na linha 4 do Algoritmo 1, o modelo de decisão inicial f do algoritmo HubISC é composto por um conjunto H de *hubs* identificados. Somente os *hubs* são armazenados para a utilização nas demais etapas da fase *online*, as demais instâncias processadas na fase *offline* são descartadas pelo algoritmo.

4.2. Fase *online*

Na fase *online*, ilustrada pela Figura 1, o algoritmo HubISC também considera os valores K e h_K da fase *offline*. Além disso, recebe como parâmetros: um fluxo de imagens S; um limiar L para determinar a distância d máxima permitida de uma instância i em relação aos *hubs*; um tamanho T para o *buffer* de instâncias. Cada imagem $i \in S$ também passará por um método de extração de características e será representada por um vetor $\vec{\boldsymbol{x}}$.

Na Etapa (1) da fase *online* do algoritmo HubISC, recebe-se cada imagem $i \in S$,

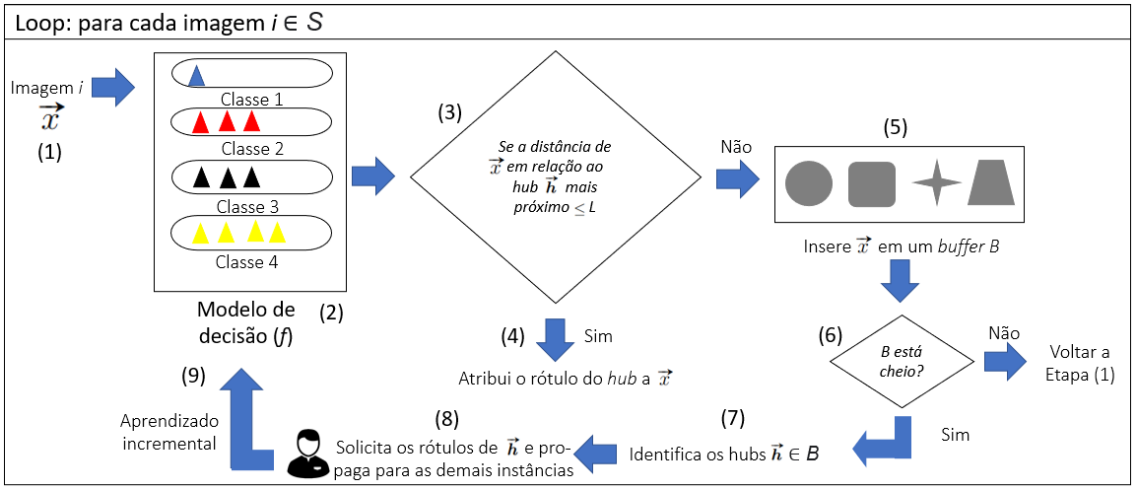

Figura 1. Fase *online* do algoritmo para a classificação de fluxos de dados de imagens.

representada por um vetor de características $\vec{\boldsymbol{x}}$. Na sequência, na Etapa (2), $\vec{\boldsymbol{x}}$ é avaliado pelo modelo de decisão f (linha 8 do Algoritmo 1). Na Etapa (3), é calculada a distância d de $\vec{\boldsymbol{x}}$ para cada *hub* $\vec{\boldsymbol{h}} \in H$. Então, o menor valor de d é analisado em relação ao limiar L. Caso $d \leq L$, o rótulo y do respectivo *hub* é atribuído a $\vec{\boldsymbol{x}}$ na Etapa (4) (linha 9 do Algoritmo 1). Senão, $\vec{\boldsymbol{x}}$ é inserida no *buffer* B na Etapa (5) (linha 11 do Algoritmo 1).

O *buffer* B possui um tamanho T predefinido. Na Etapa (6) da Figura 1, é verificado se o *buffer* está cheio, isto é, $|B| == T$. Caso estiver, é calculada a pontuação *hubness* de cada vetor $\vec{\boldsymbol{x}} \in B$ na Etapa (7) (linha 13 do Algoritmo 1). Dessa forma, se a pontuação *hubness* do vetor for superior a h_K, o respectivo vetor $\vec{\boldsymbol{x}}$ é considerado um *hub*. Os *hubs* identificados são armazenados em $H^{'}$.

Na Etapa (8) ocorre a fase de aprendizado ativo do algoritmo. Nessa etapa, solicita-se o rótulo de cada *hub*, identificado na etapa anterior. Em um cenário real de aplicações poderia considerar a interação com um usuário especialista nessa etapa. Então, o rótulo informado de cada *hub* é propagado para as demais instâncias. Para tanto, avalia-se a distância de cada vetor $\vec{\boldsymbol{x}} \in (B - H^{'})$ para os *hubs* $\vec{\boldsymbol{h}}^{'} \in H^{'}$. Cada vetor $\vec{\boldsymbol{x}}$ recebe o mesmo rótulo do *hub* $\vec{\boldsymbol{h}}^{'}$ mais próximo (linha 14 do Algoritmo 1).

Observa-se que na Etapa (8) do algoritmo HubISC, com as instâncias rotuladas, pode ocorrer a identificação de novas classes e também a mudança de conceito das classes já conhecidas, sendo possível tratar os fenômenos *open-set* e *concept-drift*. Por fim, na Etapa (9) do algoritmo, ocorre a atualização incremental do modelo de decisão f, para a atualização do conjunto H de *hubs*, com os novos *hubs* identificados na etapa anterior (linha 15 do Algoritmo 1). Para o tratamento do fenômeno *concept-drift*, os novos *hubs* de classes já conhecidas pelo modelo de decisão são incorporados ao conjunto de *hubs* já existente da respectiva classe.

4.3. Complexidade Computacional

No algoritmo HubISC a complexidade computacional pode ser representada como: $O((e * b^2) + |B|^2 + (|S| * |H|))$, onde $(e * b^2)$ representa a etapa de descrição de ca-

racterísticas das imagens. Considerando o uso de CNN, e é o número de épocas e b representa o número de *batches* de imagens. Os *batches* b representam a organização de todas as imagens disponíveis ($X \cup S$). O cálculo da pontuação é representado $|B|^2$. Ressalta-se que a pontuação *hubness* é calculada em *batches*, com base no *buffer* B, e não é necessário calcular para todo o fluxo de dados de imagem S. Por fim, $(|S| * |H|)$ se refere a comparação de cada vetor $\vec{\boldsymbol{x}} \in S$ para os representantes (*hubs*) de cada classe, sendo $|H|$ o número de *hubs* no modelo de decisão.

No algoritmo NCM o principal fator para a complexidade computacional é o cálculo da distância de cada instância de dados do fluxo para os representantes do modelo de decisão. Dessa forma, a complexidade computacional pode ser descrita como: $O((e * b^2) + (|S| * p))$, onde p é o número de representantes no modelo de decisão. No algoritmo CPE, a utilização da CNN para a descrição de características e para o tratamento da alta dimensionalidade dos dados é o principal ponto para a complexidade computacional. Considerando C uma constante para operações de otimização e c a quantidade de classes conhecidas pelo modelo, a complexidade computacional pode ser representada como: $O(C * e * (b^2 + |B| * c) + (|S| * p))$ [Wang et al. 2019]. Nota-se que os algoritmos HubISC e CPE trabalham com o conceito de classificação com atraso, por meio da utilização do *buffer* B. Já no algoritmo NCM isso não ocorre.

5. Avaliação Experimental

Os cenários de experimentos propostos têm como objetivo avaliar o algoritmo HubISC em relação a outros algoritmos para a classificação de fluxos de dados de imagens. Além disso, foram analisadas diferentes parametrizações do algoritmo HubISC. A Seção 5.1 apresenta os conjuntos de dados utilizados nos experimentos, além das configurações consideradas para o descritor de características. A Seção 5.2 realiza uma análise comparativa do algoritmo apresentado na Seção 4 em relação a outros algoritmos para classificação de fluxos de imagens. A Seção 5.3 explora diferentes valores do parâmetro h_K do algoritmo HubISC. A Seção 5.4 analisa diferentes valores do parâmetro L do algoritmo HubISC.

5.1. Conjuntos de dados e Configurações

Para a realização dos experimentos foram selecionados 3 conjuntos de imagens reais, comumente, utilizados em trabalhos de classificação de imagens. Esses conjuntos de dados possuem 10 classes e diferentes quantidades de imagens. O conjunto de imagens *FASHION-MNIST* possui 70.000 imagens. O conjunto *CIFAR-10* possui 60.000 imagens. O maior conjunto de imagens considerado foi o *SVHN* com 100.000 imagens.

O descritor CNN foi considerado para a extração de características das imagens. A arquitetura de rede selecionada para os experimentos foi a DenseNet, com os mesmos parâmetros utilizados em [Wang et al. 2019]. Para a execução no algoritmo HubsISC foram considerados os vetores de características com as mesmas dimensionalidades de [Wang et al. 2019], sem considerar o recurso de redução de dimensionalidade. Utilizou-se 784 dimensões no conjunto de imagens *FASHION-MNIST* e 3.072 dimensões nos conjuntos *CIFAR-10* e *SVHN*. Para a avaliação dos algoritmos foi utilizado o método de avaliação para a classificação de fluxos de dados de imagens definido no trabalho [de Lima et al. 2020]. A eficácia dos algoritmos foi avaliada pelas métricas *acurácia* e *F-Measure*.

5.2. Avaliando a utilização do aspecto *hubness* para classificação de fluxos de imagens

O experimento descrito aqui tem o objetivo de investigar a questão de pesquisa descrita na Seção 1. Para tanto, o algoritmo HubISC, que considera o aspecto *hubness*, foi comparado em relação a dois algoritmos comumente utilizados para a classificação de fluxos de dados de imagens: NCM (versão original, sem otimizações) [Rebuffi et al. 2017] e CPE [Wang et al. 2019]. Para o algoritmo HubISC, os parâmetros considerados foram: $L = 5$, $T = 1.000$, $K = 50$, $h_K = 70\%$ da maior pontuação *hubness* obtida na fase *offline*. Os valores de L e T foram baseados em [Wang et al. 2019]. As demais configurações dos algoritmos CPE e NCM foram definidas com base nos melhores parâmetros informados pelos autores.

Em relação a organização dos conjuntos de dados, o conjunto X, utilizado na fase *offline* dos algoritmos para a construção do modelo de decisão inicial, foi organizado com 10% de instâncias em relação ao total do conjunto de dados e com 50% das classes. Dessa forma, o fluxo de dados S, utilizado na fase *online*, é composto por 90% da instâncias e 100% de classes, o que permite simular os desafios *concept-drift* e *open-set*. As classes de imagens e as instâncias consideradas foram selecionadas aleatoriamente da mesma forma que utilizado em [Wang et al. 2019]. Os resultados dos experimentos com o primeiro cenário estão descritos na Tabela 1.

Tabela 1. Acurácia (A) e F-Measure (F). Avaliação de diferentes algoritmos para a classificação de fluxos de dados de imagens.

Algoritmo	FASHION-MNIST			SVHN			CIFAR-10		
	A	F	% rótulos	A	F	% rótulos	A	F	% rótulos
CPE	91,88	64,19	20,63	80,24	58,16	22,62	70,47	52,96	27,15
NCM	82,43	59,44	15,31	72,84	51,63	16,89	67,72	48,51	18,79
HubISC	89,12	63,23	14,55	78,86	55,45	14,38	69,63	51,39	15,87

Analisando os resultados apresentados na Tabela 1 é possível perceber que o algoritmo CPE apresentou os maiores valores de *F-measure* e acurácia. Comparando os algoritmos CPE e NCM a maior diferença encontrada foi na acurácia do conjunto de dados FASHION-MNIST, com 9,45% de diferença. Em comparação ao algoritmo HubISC, a diferença de acurácia e *F-measure* é menos evidente, a maior diferença encontrada também foi na acurácia do conjunto FASHION-MNIST com o valor de 2,76%.

Apesar do algoritmo CPE ter apresentado os melhores resultados em termos de *F-measure* e acurácia, esse algoritmo necessita da maior quantidade de rótulos informados, em comparação aos algoritmos NCM e HubISC. O algoritmo HubISC requisitou a menor quantidade de rótulos, com uma diferença de até 11,28% de rótulos no conjunto de dados CIFAR-10. Observa-se que nesse conjunto de dados, a diferença de acurácia entre os algoritmos CPE e HubISC foi de apenas 0,84%. Essa diferença na quantidade de rótulos informados se justifica pela utilização do aspecto *hubness* e por sua influência na seleção das instâncias a serem rotuladas. Com base nos resultados apresentados sobre a investigação da questão de pesquisa, o algoritmo HubISC apresentou um potencial interessante para lidar com a classificação de fluxos de dados de imagens, considerando que apresenta valores superiores de acurácia e *F-measure* em relação ao algoritmo NCM, e valores próximos se comparado ao algoritmo CPE. Além disso, permite utilizar uma me-

nor quantidade de rótulos em comparação aos dois algoritmos. Dessa forma, a influência dos parâmetros h_K e L do algoritmo *HubISC* foi avaliada nas Seções 5.3 e 5.4.

5.3. Avaliando diferentes valores do parâmetro h_K

O experimento descrito aqui tem o objetivo de investigar a influência do parâmetro h_K nos resultados do algoritmo HubISC. A investigação desse parâmetro se mostra interessante, pois impacta na quantidade de *hubs* a serem identificados. Consequentemente, a identificação dos *hubs* também influencia na quantidade de rótulos a serem fornecidos (linhas 13, 14 e 15 do Algoritmo 1).

No primeiro cenário de experimentos, descrito na Seção 5.2, o parâmetro h_K foi definido com base na maior pontuação *hubness* (h_P) identificada na fase *offline* do algoritmo HubISC. O valor definido foi de 70% da maior pontuação ($h_K = h_P * 0.7$), isto é, as instâncias que as pontuações *hubness* sejam $\geq h_K$ são consideradas *hubs*. Dessa forma, esse experimento avaliou diferentes porcentagens para a definição do valor de h_K. Os resultados obtidos com essa variação estão apresentados na Tabela 2.

Tabela 2. Acurácia (A) e F-Measure (F). Avaliação de diferentes valores do parâmetro h_K.

h_K	FASHION-MNIST			SVHN			CIFAR-10		
	A	F	% rótulos	A	F	% rótulos	A	F	% rótulos
40%	92,51	64,77	22,56	81,78	58,32	25,98	71,66	53,23	27,36
50%	91,75	63,90	19,72	79,86	57,45	21,84	70,51	52,98	25,39
60%	89,71	63,59	16,28	79,11	56,09	17,35	70,37	51,89	19,44
70%	89,12	63,23	14,55	78,86	55,45	14,38	69,63	51,39	15,87
80%	88,05	62,62	13,12	77,57	54,92	12,95	67,35	50,97	13,91
90%	86,13	61,22	10,98	75,09	52,63	11,12	65,46	48,77	11,98

Como pode ser observado na Tabela 2, o parâmetro h_K possui influência na acurácia e na *F-measure* do algoritmo HubISC. Além disso, esse parâmetro também impacta na quantidade de rótulos solicitados. Os maiores valores de acurácia e *F-measure* foram observados para 40% de h_K. Entretanto, nessa configuração também foram solicitadas as maiores quantidades de rótulos. Já para o cenário de 90% de h_K foram constatados os menores valores de acurácia, *F-measure* e de quantidades de rótulos.

O comportamento do parâmetro h_K identificado nos resultados é devido sua influência na identificação dos *hubs*. Quanto maior é o valor h_K, a identificação dos *hubs* se torna mais restritiva. Dessa forma, o número de *hubs* é menor e a quantidade de rótulos solicitada também é inferior. Então, a tendência é que o algoritmo apresente valores menores de acurácia e *F-measure*. Por outro lado, quanto menor é o valor de h_K, mais *hubs* são identificados e, consequentemente, mais rótulos são solicitados. Esse fato auxilia na obtenção de valores mais elevados de acurácia e *F-measure*.

Em comparação aos resultados do algoritmo CPE (Tabela 1), o algoritmo HubISC apresentou valores de acurácia e *F-measure* superiores em todos os conjuntos de imagens, para o caso de 40% de h_k. A maior diferença observada foi de 1,54% na acurácia do conjunto SVHN. Entretanto, nessa configuração, a quantidade de rótulos solicitada pelo algoritmo HubISC também foi superior. No conjunto SVHN, uma quantidade de 3,36% a mais de rótulos foram solicitados. No cenário de 50% de h_k, o algoritmo HubISC foi superior ao algoritmo CPE no conjunto CIFAR-10, com uma diferença de 0,04%

na acurácia e uma quantidade de rótulos 1,76% inferior. Dessa forma, a utilização do algoritmo *hubness* se mostra promissora para a classificação de fluxos de dados de imagens.

5.4. Avaliando diferentes valores do parâmetro L

O experimento descrito aqui tem o objetivo de investigar a influência do parâmetro L nos resultados do algoritmo HubISC. Essa análise foi realizada pois o parâmetro L impacta nas instâncias de dados inseridas no *buffer* B (linha 8 do Algoritmo 1). Dessa forma, esse parâmetro também pode influenciar na identificação dos *hubs* e na quantidade de rótulos a serem fornecidos. O parâmetro L foi definido com o valor 5 no primeiro cenário de experimentos, descrito na Seção 5.2, com base nos valores utilizados em [Wang et al. 2019]. Esse experimento avaliou diferentes valores para o parâmetro L. Os resultados obtidos com essa variação estão apresentados na Tabela 3.

Tabela 3. Acurácia (A) e F-Measure (F). Avaliação de diferentes valores do parâmetro L.

L	FASHION-MNIST			SVHN			CIFAR-10		
	A	F	% rótulos	A	F	% rótulos	A	F	% rótulos
1	94,11	67,96	33,49	85,63	62,43	35,71	73,48	56,39	36,32
3	92,93	65,01	27,94	81,82	59,33	29,23	71,79	53,52	31,89
5	89,12	63,23	14,55	78,86	55,45	14,38	69,63	51,39	15,87
7	86,56	62,09	11,55	74,94	51,13	10,92	65,35	48,31	11,79
9	82,63	58,29	8,43	70,07	48,91	7,66	60,86	45,12	8,23

Observando os resultados apresentados na Tabela 3, o parâmetro L impacta na quantidade de rótulos solicitados pelo algoritmo HubISC. Além disso, influencia na acurácia e na *F-measure*. Os maiores valores de acurácia e *F-measure* foram constatados para $L = 1$. Com essa configuração também foi observado o maior número de rótulos solicitados. No cenário com o valor $L = 9$ foram observadas as menores quantidades de rótulos e também os menores valores de acurácia e *F-measure*.

A influência do parâmetro L é devido ao seu impacto na seleção de instâncias a serem armazenadas no *buffer* B. Observa-se que quanto menor é o valor L, mais instâncias são inseridas em B. Dessa forma, a tendência é que B fique cheio mais vezes durante o processamento do fluxo de dados de imagens. Então, mais rótulos são solicitados e o algoritmo pode apresentar valores maiores de acurácia e *F-measure*. Por outro lado, quanto maior é o valor de L, menos instâncias são inseridas em B. Então, uma quantidade inferior de *hubs* é identificada, menos rótulos são solicitados e a tendência é que a eficácia do algoritmo seja inferior com essa configuração.

Comparando os resultados apresentados na Tabela 3 com o algoritmo CPE (Tabela 1), nas configurações $L = 1$ e $L = 3$, o algoritmo HubISC apresentou valores de acurácia e *F-measure* superiores em todos os conjuntos de imagens. Foi observada uma melhora de até 5,39% na acurácia do conjunto de imagens SVHN. Entretanto, nessa configuração, a quantidade de rótulos solicitada pelo algoritmo HubISC foi até 13,09% superior.

6. Conclusão

A classificação de fluxos de dados de imagens é uma tarefa importante que ainda possui questões em aberto para serem exploradas. Nesse contexto, uma dessas questões é

a alta dimensionalidade dos dados. A maior parte dos estudos da literatura científica da área não aborda esse desafio. O estudo apresentado neste artigo desenvolveu o algoritmo HubISC para a classificação de fluxos de dados de imagem, que incorpora o aspecto *hubness*, inerente de dados de alta dimensão. O algoritmo HubISC apresenta características interessantes para a classificação de fluxos de dados de imagens:

- permite por meio do aspecto *hubness* lidar com a alta dimensionalidade dos dados;
- possui uma estrutura de sumarização das instâncias de dados das classes por meio da utilização dos *hubs*;
- apresenta um recurso natural para a seleção de instâncias no processo de aprendizado ativo, pois os *hubs* podem representar essas instâncias de dados;
- apresenta eficácia similar a outros algoritmos estado da arte para classificação de fluxos de dados de imagens;
- permite o tratamento dos fenômenos *open-set* e *concept-drift* a partir da atualização incremental do modelo de decisão;
- possui parâmetros (h_K e L) que podem influenciar na quantidade de rótulos solicitados, o que pode contribuir para cenários de aplicações reais com disponibilidade limitada de usuários especialistas.

Por meio da análise dos resultados experimentais obtidos com o algoritmo HubISC, nota-se que é possível obter resultados interessantes em termos de eficácia, mesmo com uma menor quantidade de rótulos fornecida. Nesse contexto, como trabalhos futuros, pretende-se: comparar o algoritmo HubISC com outros algoritmos de classificação de imagens, incluindo algoritmos baseados em CNNs; utilizar outras estratégias de aprendizado ativo em conjunto com o aspecto *hubness*; avaliar outros conjuntos de dados de imagens nos experimentos; analisar formas de esquecimento de representantes, o que consequentemente reduz a quantidade de *hubs* no modelo; considerar o aspecto *hubness* para a detecção de ruídos nos conjuntos de dados; otimizar o cálculo da pontuação *hubness* para diminuir a complexidade computacional do algoritmo.

Referências

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-dujaili, A., Duan, Y., Al-Shamma, O., Santamaría, J., Fadhel, M. A., Al-Amidie, M., and Farhan, L. (2021). Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8. DOI: 10.1186/s40537-021-00444-8.

Castro, F. M., Marin-Jimenez, M. J., Guil, N., Schmid, C., and Alahari, K. (2018). End-to-end incremental learning. In *ECCV*, pages 241–257, Munich, Germany. Springer. DOI: 10.1007/978-3-030-01258-8_15.

de Lima, M. C., Barioni, M. C. N., Faria, E. R., and Razente, H. L. (2020). Evisclass: a new evaluation method for image data stream classifiers. In *ICMLA*, pages 399–406. DOI: 10.1109/ICMLA51294.2020.00070.

Hu, J., Sun, Z., Li, B., Yang, K., and Li, D. (2017). Online user modeling for interactive streaming image classification. In *MMM*, pages 293–305, Reykjavik, Iceland. Springer. DOI: 10.1007/978-3-319-51814-5_25.

Mani, P., Vazquez, M., Metcalf-Burton, J., Domeniconi, C., Fairbanks, H., Bal, G., Beer, E., and Tari, S. (2019). The hubness phenomenon in high-dimensional spaces. *AWMS*, pages 15–45. DOI: 10.1007/978-3-030-11566-1_2.

Nguyen, H.-L., Woon, Y.-K., and Ng, W.-K. (2015). A survey on data stream clustering and classification. *Knowl.Inf.Syst.*, 45(3):535–569. DOI:10.1007/s10115-014-0808-1.

Parreira, P. and Prati, R. (2019). Active learning in data stream with intermediate latency. In *ENIAC*, Salvador, Brazil. DOI: 10.5753/eniac.2019.

Pham, T., Kottke, D., Krempl, G., and Sick, B. (2021). Stream-based active learning for sliding windows under the influence of verification latency. *Machine Learning*. DOI:10.1007/s10994-021-06099-z.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. (2017). iCaRL: Incremental classifier and representation learning. In *CVPR*, pages 5533–5542, Honolulu, Hawaii. IEEE. DOI: 10.1109/CVPR.2017.587.

Ristin, M., Guillaumin, M., Gall, J., and Gool, L. V. (2014). Incremental learning of ncm forests for large-scale image classification. In *CVPR*, pages 3654–3661, Columbus, Ohio. IEEE. DOI: 10.1109/CVPR.2014.467.

Romaszewski, M., Głomb, P., and Cholewa, M. (2018). Adaptive, hubness-aware nearest neighbour classifier with application to hyperspectral data. In *ISCIS*, pages 113–120, Poznan, Polônia. DOI: 10.1007/978-3-030-00840-6_13.

Samet, H. (2005). *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann.

Silva, J. A., Faria, E. R., Barros, R. C., Hruschka, E. R., Carvalho, A. C. P. L. F. d., and Gama, J. a. (2013). Data stream clustering: A survey. *ACM Comput. Surv.*, 46(1):13:1–13:31. DOI: 10.1145/2522968.2522981.

Tomašev, N., Radovanović, M., Mladenić, D., and Ivanović, M. (2014). Hubness-based fuzzy measures for high-dimensional k-nearest neighbor classification. *Int. J. Mach. Learn. e Cyber.*, 5:445–458. DOI: 10.1007/s13042-012-0137-1.

Wang, H., Zhou, Z., Wang, Y., and Yan, X. (2021). Feature selection for image classification based on bacterial colony optimization. In *ICSI*, page 430–439, Qingdao, China. Springer. DOI: 10.1007/978-3-030-78811-7_40.

Wang, Z., Kong, Z., Changra, S., Tao, H., and Khan, L. (2019). Robust high dimensional stream classification with novel class detection. In *ICDE*, pages 1418–1429, Macao, Macao. IEEE. DOI: 10.1109/ICDE.2019.0012.

Wu, Q., Lin, Y., Zhu, T., and Zhang, Y. (2020). Hiboost: A hubness-aware ensemble learning algorithm for high-dimensional imbalanced data classification. *J. Intell. Fuzzy Syst.*, 39:1–12. DOI: 10.3233/JIFS-190821.

Žliobaitė, I., Bifet, A., Pfahringer, B., and Holmes, G. (2014). Active learning with drifting streaming data. *TNNLS*, 25(1):27–39. DOI: 10.1109/TNNLS.2012.2236570.