Active Learning in Data Stream with Intermediate Latency

Pedro H. Parreira[1], **Ronaldo C. Prati**[1]

[1]Centro de Matemática, Computação e Cognição
Universidade Federal do ABC (UFABC)
Av. dos Estados, 5001
Santo André, SP, Brazil

pedro.parreira@ufabc.edu.br, ronaldo.prati@ufabc.edu.br

Abstract. *A data stream is characterized by a massive production of examples due to their continuous arrival. In classification problems in data streams, the actual sample label is usually required for performance evaluation or concept drift detection. However, in many applications, obtaining all true labels is impractical due to the high cost associated with the labeling process. Although there are several papers that use the active learning approach in data stream to obtain a set of labeled examples, they usually consider that the true labels are made available immediately upon request, which is not always possible because of the time required for the example analysis. This article aims to investigate a new scenario of data streams with intermediate latency and restriction of labeling. Furthermore, we propose some active learning strategies for this scenario, as well as a theoretical support framework for them.*

Resumo. *Um fluxo de dados é caracterizado por uma produção massiva de exemplos devido à chegada contínua desses exemplos. Em problemas de classificação em fluxo de dados, normalmente o rótulo real do exemplo é necessário para a avaliação do desempenho ou a detecção de mudança de conceito. No entanto, em muitas aplicações, a obtenção de todos os rótulos verdadeiros é impraticável devido ao alto custo associado. Apesar de existirem diversos trabalhos que utilizam a aprendizagem ativa em fluxo de dados para obter uma porção de exemplos rotulados, normalmente eles consideram que o rótulo verdadeiro é disponibilizado de forma imediata na requisição, o que nem sempre é possível devido ao tempo necessário de análise do exemplo. Este artigo tem por objetivo a investigação de um novo cenário de fluxo de dados com latência intermediária e restrição de rotulagem, além de propor algumas estratégias de aprendizagem ativa para esse cenário e um framework de suporte teórico para essas estratégias.*

1. Introdução

A popularização de diversos dispositivos móveis, tais como os celulares, e a possibilidade de objetos com o advento da internet das coisas, tais como eletrodomésticos conectados, coletarem e transmitirem informações pela Internet, por exemplo, provocaram um grande incremento na quantidade de fluxos de dados gerados diariamente por diversas aplicações. Os fluxos de dados tem como características a contínua chegada de dados de maneira sequencial, o grande volume de dados e a alta taxa de chegada desses dados [Golab and Özsu 2003, Aggarwal 2007].

Tais características dos fluxos de dados são um verdadeiro desafio para os tradicionais algoritmos de aprendizado de máquina que assumem que os dados de treinamento estão em lotes acessíveis a qualquer momento [Gama 2010]. Para dificultar ainda mais esse desafio tais algoritmos, em sua maioria, também supõem que os dados estejam em uma ambiente estacionário, ou seja, no qual a distribuição dos dados não se altera com o tempo.

Essa suposição é extremamente otimista quando aplicada nos problemas do mundo real, nos quais ambientes dinâmicos podem influenciar a natureza dos dados com o tempo [Ditzler et al. 2015]. Por exemplo, os hábitos de compra de um determinado usuário podem se alterar ao longo do tempo. Portanto é mais realista assumir que tais fluxos de dados estejam em um ambiente não estacionário, ou seja, que ao longo do tempo a distribuição dos dados se altera, o que é definido como *mudança de conceito*. Outra característica presente nos fluxos de dados é a mudança na distribuição de classes ao longo do tempo, o que se denomina evolução de conceito [Gama et al. 2014, Webb et al. 2016]. Ao longo do fluxo, novas classes podem emergir ou classes existentes podem desaparecer, e a distribuição dos dados nessas classes pode se alterar ao longo do tempo, ou padrões recorrentes podem se alternar, o que torna o ambiente ainda mais desafiador para os tradicionais algoritmos de aprendizagem de máquina.

Existe ainda um outro desafio em fluxo de dados pouco explorado na literatura que é chamado de *latência de verificação*. O problema é caracterizado pela existência de um intervalo de tempo entre a classificação do exemplo e a disponibilidade do seu rótulo verdadeiro para verificação e adaptação do modelo de classificação, em que tal intervalo é denominado *latência* [Marrs et al. 2010].

Pode-se classificar os cenários de latência em três tipos, de acordo com a *latência*, isto é, de acordo com o seu tempo de atraso: (1) *latência nula*, no qual ao ser requisitado o rótulo verdadeiro do exemplo X_t no instante t, o rótulo é disponibilizado em $t + \Delta_t$, em que $\Delta_t \to 0$; (2) *latência extrema*, no qual o rótulo nunca estará disponível (ou equivalentemente, ele é disponibilizado em $t + \Delta_t$, onde $\Delta_t \to \infty$); e (3) *latência intermediária*, no qual o rótulo verdadeiro do exemplo é disponibilizado em $t + \Delta_t$, em que Δ_t pertence ao intervalo $(0, l), l < \infty$.

Tais atrasos são bastante comuns em aplicações reais, e podem ser ocasionados por diversos motivos, como em uma falha de transmissão dos dados; a necessidade de um maior tempo de análise para tornar o rótulo do exemplo acessível; ou a indisponibilidade de recursos (por exemplo, a falta de especialistas humanos para analisar os exemplos). Além disso, diferentes classes podem ter tempos de latência diferentes. Em um problema de transações de cartões de crédito, por exemplo, a verificação de uma transação classificada como fraudulenta pode ser feita de maneira relativamente rápida com uma verificação com o usuário. Já uma contestação de uma transação classificada como legítima pode envolver uma análise documental por um analista de crédito.

A disponibilidade de rótulos reais é muito importante em fluxos de dados que envolvem mudança de conceito. Ao longo do fluxo de dados, o desempenho de um modelo de classificação é afetado negativamente pelos fenômenos de mudança e evolução de conceito. Por exemplo, no surgimento de um novo conceito no fluxo de dados o modelo de classificação não irá predizer corretamente os exemplos correspondentes a este novo conceito por seu atual conhecimento não contemplá-lo, o que torna necessário a sua

adaptação com exemplos do fluxo de dados correspondentes a este novo conceito.

Esse processo de adaptação do modelo de classificação, afim de retratar de modo mais fidedigno o estado atual do fluxo de dados, se torna mais desafiador na presença da *latência intermediária*. Nesse caso, o rótulo verdadeiro de um determinado exemplo do fluxo de dados é disponibilizado apenas após um intervalo de tempo da sua solicitação. Portanto, o ganho de conhecimento obtido pelo modelo de classificação com tal adaptação será obtido de forma tardia, podendo até mesmo tal conhecimento não retratar mais o estado atual do fluxo de dados. Como no exemplo anterior, com o surgimento de um novo conceito, em que o modelo de classificação será adaptado para contemplar este novo conceito apenas após um intervalo de tempo a frente e, portanto, o modelo de classificação não será capaz de predizer corretamente os exemplos correspondentes a este conceito enquanto o rótulo real do exemplo não estiver disponibilizado.

Somado aos desafios descritos anteriormente, em aplicações tradicionais de mineração de dados o processo de rotulagem é bastante custoso. Em fluxo de dados, no qual existe uma produção massiva de dados, é pouco realista assumir que todos os exemplos serão disponibilizados para a adaptação do modelo de classificação. Nesse contexto, a utilização da abordagem de *aprendizagem ativa*, que busca selecionar uma pequena porção de exemplos para serem rotulados e utilizados na adaptação de um modelo de classificação, em fluxo de dados se mostra bastante adequada.

Desta maneira, este artigo visa a investigação e o desenvolvimento de algumas estratégias de aprendizagem ativa para o cenário de fluxo de dados com *latência intermediária*. Para esse fim, estendemos o trabalho descrito em [Žliobaitė et al. 2014]. Entretanto, enquanto esse trabalho considera a existência de *latência nula*, nossa contribuição é generalizar o problema de latência, incorporando a *latência intermediária*. Ressalta-se que não encontramos na literatura nenhum artigo que considera o problema de latência intermediária juntamente com aprendizagem ativa em fluxo de dados.

O restante deste artigo está estruturado da seguinte maneira: na Seção 2 são apresentados os principais trabalhos existentes na literatura de *latência intermediária* e aprendizagem ativa em fluxo de dados; na Seção 3 é proposto um *framework* e um conjunto de estratégias de aprendizagem ativa para lidar em a *latência intermediária* em fluxo de dados; na Seção 4 é apresentado os resultados ao utilizar tais estratégias em um conjunto de dados real; e na Seção 5 são apresentadas as considerações finais sobre o estudo.

2. Trabalhos Relacionados

Apesar de bastante presente em aplicações reais, a *latência intermediária* em fluxo de dados recebeu pouca atenção na literatura. Em [Kuncheva and Sánchez 2008], os autores abordam três diferentes estratégias, utilizando o algoritmo *online k-Nearest Neighbors(kNN)* com as extensões IB2 e IB3, para lidar com o atraso de rotulagem em fluxo de dados. Já os autores em [Plasse and Adams 2016] fornecem um *framework* que utiliza uma versão do algoritmo *Linear Discriminant Analysis* (LDA) em fluxo de dados que possui a capacidade de incorporar rótulos atrasados e, além disso, os autores também fornecem uma taxonomia para a *latência intermediária*. O *framework* SRALD (*Sliding Reservoir Approach for Delayed Labeling*) proposto em [Hanqing Hu 2017], lida com a latência intermediária em fluxo de dados por meio de um algoritmo de aprendizagem semi-supervisionada com uma nova abordagem para gerenciar os exemplos rotulados.

A latência intermediária é um desafio bastante presente na detecção de fraude em transações de cartões de crédito pela própria natureza do problema. Em [Pozzolo et al. 2015] os autores abordam tal problema em um contexto de fluxo de dados, ou seja, é levado em consideração o fenômeno de mudança de conceito. Dos mesmos autores, em [Pozzolo et al. 2018] é proposto uma estratégia de aprendizagem para o processo de autorização de transações, em que é considerado uma amostra de exemplos que foram rotulados por especialistas e outra de exemplos supervisionados com atraso. Em [Alves de Souza et al. 2018] é proposta uma estatística de avaliação, denominada de *Kappa-Latency* (k_{lat}), que compara diferentes modelos de classificação com um valor máximo de atraso que pode ocorrer na aplicação em um contexto de fluxo de dados.

Existem diversos trabalhos na literatura que fazem a abordagem de *aprendizagem ativa* em fluxo de dados, porém eles consideram apenas o cenário de *latência nula*. É descrito em [Žliobaitė et al. 2014] um *framework* de suporte teórico para a aprendizagem ativa em fluxo de dados, além de algumas estratégias de aprendizagem ativa que são capazes de lidar com a mudança de conceito. Em [Mohamad et al. 2018] é descrito um algoritmo de aprendizagem ativa chamado de *novel bi-criteria AL* (BAL), que aborda os desafios de mudança de conceito e de *viés de amostragem* em fluxo de dados utilizando os critérios de incerteza do rótulo e de densidade. Dos mesmos autores, em [Mohamad et al. 2017] é proposto outro algoritmo chamado de *stream-based active learning algorithm* (SAL), que além de ser capaz de lidar com a mudança, tal como o BAL, também é capaz de lidar com a evolução de conceito, além de levar em consideração também o problema de viés de amostragem por meio de uma técnica denominada de *importance weighted*.

Existe ainda outra gama de algoritmos de aprendizagem ativa em fluxo de dados que, além de considerar o grau de informatividade do exemplo, leva também em consideração o esforço necessário para adquirir o rótulo do exemplo, ou seja, o seu custo. Em [Attenberg and Provost 2011] os autores propõem um *framework* para uma estratégia de *inferência ativa* em fluxo de dados, denominada *Active Inference with Utility Distribution Estimation* (AI-UDE), que visa minimizar o custo de classificação total esperado. Em [Zhao and Hoi 2013] é abordado o problema de detecção de URL maliciosas em fluxo de dados, sendo que tal problema é formulado como um problema de classificação binária cujo objetivo é otimizar a sua acurácia. Os autores propõem um *framework* denominado *Cost-Sensitive Online Active Learning* (CSOAL), que tem a capacidade de selecionar uma pequena porção de exemplos para serem rotulados e considera duas medidas sensíveis aos custos para lidar com o desequilíbrio de classes. Em [Hao et al. 2018] é proposto um *framework* denominado *Cost-sensitive Second-order Online Active Learning algorithm* SOAL_{cs}, que visa minimizar o custo de erros ponderados pelas diferentes classes.

3. Estratégias

Nesta seção é apresentada a formulação do cenário proposto de fluxo de dados com latência intermediária. Além disso, é definido um *framework* e algumas estratégias de aprendizagem ativa que são capazes de lidar com tal cenário.

3.1. Formulação do Cenário

Para a formulação do cenário, considere uma estratégia de aprendizagem ativa $\mathcal{S}$, que tem como objetivo a seleção de uma porção de exemplos de um fluxo de dados para ter seus

rótulos requisitados para um Oráculo e que podem, posteriormente, serem utilizados na adaptação de um modelo de classificação $\mathcal{F}$.

Seja X_t um exemplo oriundo de tal fluxo de dados, em que $X_t \in \mathbb{R}^d$. Ao receber o exemplo X_t no instante t, a estratégia de aprendizagem ativa $\mathcal{S}$ estima, com base em alguma medida de informatividade, o ganho de informação da utilização de X_t na adaptação de $\mathcal{F}$ como $\delta_t(X_t)$. Posteriormente, com base em algum critério de decisão e no ganho de informação $\delta_t(X_t)$, a estratégia $\mathcal{S}$ pode selecionar o exemplo X_t e fazer a requisição de seu rótulo verdadeiro para o Oráculo. Ao receber o rótulo real Y_t, ele é utilizado em conjunto com X_t para a adaptação de $\mathcal{F}$.

Na maioria das abordagens de aprendizagem ativa em fluxo de dados existentes, nas quais é considerada a *latência nula*, o rótulo Y_t é disponibilizado de forma imediata, isto é, no instante t, e em conjunto com o exemplo X_t, são utilizados na adaptação de $\mathcal{F}$. Desta maneira, o Oráculo se torna disponível já no instante $t+1$, ou seja, ao receber o exemplo X_{t+1} no instante $t+1$, a estratégia de aprendizagem ativa $\mathcal{S}$ poderá requisitar o rótulo de tal exemplo, de acordo com seu critério de decisão, no mesmo instante $t+1$.

Ao considerar a existência de *latência intermediária* no fluxo de dados o desafio se torna maior. Diferentemente do cenário em que é considerada a *latência nula*, no cenário de *latência intermediária* o rótulo verdadeiro Y_t não é disponibilizado de forma imediata ao ser requisitado, pois ele só será disponibilizado após algumas rodadas de aprendizagem.

Dessa maneira, quando a estratégia de aprendizagem ativa $\mathcal{S}$ faz a requisição do rótulo Y_t para o Oráculo, ele será apenas disponibilizado para a adaptação de $\mathcal{F}$ após Δ_t rodadas de aprendizagem. Neste intervalo de tempo, entre os instantes $t+1$ e $\Delta_t + t$, o Oráculo estará atuando no processo de rotulagem do exemplo X_t e, portanto, estará indisponível para receber novos exemplos, ou seja, a estratégia de aprendizagem ativa $\mathcal{S}$ não será capaz de requisitar o rótulo de qualquer um dos exemplos do conjunto $X_{t+1}, ..., X_{t+\Delta_t}$.

Além disso, somada a existência da latência intermediária no fluxo de dados com a mudança de conceito, que é caracterizada por mudanças na distribuição dos dados ao longo do fluxo, a estimativa de ganho de informação feita por $\mathcal{S}$ para o exemplo X_t no instante t, pode não se sustentar, isto é, $\delta_t(X_t) \neq \delta_{t+\Delta_t}(X_t)$. Desse modo, o impacto na atualização do modelo de classificação $\mathcal{F}$ no instante $t + \Delta_t$ se torna incerta com o exemplo X_t e seu rótulo, podendo sofrer uma forte alteração com relação a estimativa feita no instante t, ou até mesmo se tornar irrelevante a depender do intervalo de latência e do grau de mudança de conceito no momento.

Pode-se concluir, portanto, que no cenário em que existe a presença de latência intermediária e mudança de conceito, é um problema em aberto para as atuais abordagens de aprendizagem ativa em fluxo de dados. Para lidar com tal cenário, é apresentado em seguida um *framework* que servirá de suporte teórico para as estratégias de aprendizagem.

3.2. Framework

O objetivo do *framework* é o de servir de suporte teórico tanto para as estratégias de aprendizagem ativa quanto para o processo de aprendizagem. O *framework* proposto é descrito no algoritmo 1.

Para lidar com a *latência intermediária* em fluxo de dados é definido um *buffer* $\mathcal{B}$. Diferentemente do cenário de *latência nula*, em que os rótulos são disponibilizados de forma imediata após a sua requisição, no cenário de *latência intermediária* o Oráculo poderá estar indisponível e, neste caso, a estratégia de aprendizagem ativa não será capaz de requisitar o rótulo de um determinado exemplo no mesmo instante. Portanto, enquanto o Oráculo estiver indisponível (ocupado rotulando um exemplo anterior), os exemplos que chegarem através do fluxo de dados no *framework* dentro deste intervalo de tempo serão armazenados temporariamente no *buffer* $\mathcal{B}$.

Considere $\mathcal{B}$, $\mathcal{S}$ e $\mathcal{F}$, respectivamente, um *buffer*, a estratégia de aprendizagem ativa e o modelo de classificação definidos para o *framework*.

Algoritmo 1: *Framework*

Entrada: classificador $\mathcal{F}$, Estratégia $\mathcal{S}$(*parâmetros*), Buffer $\mathcal{B}$
Saída: predição do classificador $\mathcal{F}$ a cada rodada de aprendizado

```
repita
    recebe X_t através do fluxo;
    B ← B ∪ X_t;
    se oráculo disponível então
        recebe (X_{t−Δ_t}, Y_{t−Δ_t}) do oráculo ;
        atualiza o classificador F com (X_{t−Δ_t}, Y_{t−Δ_t}) ;
        Seleciona o exemplo S(B, parâmetros) para requisitar o rótulo ;
        B ← ∅
    fim
até sempre;
```

Ao chegar no *framework* através do fluxo de dados, o exemplo X_t tem seu rótulo predito pelo modelo de classificação $\mathcal{F}$. Em seguida, ele é armazenado no *buffer* $\mathcal{B}$. O *framework* então verifica se o Oráculo está disponível para receber um novo exemplo e, caso esteja, o *framework* recebe o exemplo $X_{t-\Delta_t}$ do oráculo, que foi selecionado Δ_t rodadas de aprendizagem anteriores, juntamente com seu rótulo $Y_{t-\Delta_t}$. Ambos são utilizados na adaptação do modelo de classificação $\mathcal{F}$.

Posteriormente, a estratégia de aprendizagem ativa $\mathcal{S}$ definida para o *framework*, seleciona algum dos exemplos que estão armazenados em $\mathcal{B}$ para ser rotulado e, em seguida, o *buffer* $\mathcal{B}$ é esvaziado para armazenar os exemplos do próximo ciclo de rotulação.

Em seguida são definidas as estratégias de aprendizagem ativa. Para tais definições, considere o *buffer* $\mathcal{B} = (X_t, X_{t-1}, X_{t-2}, ..., X_{t-j})$ no instante t.

Mais Recente

Esta estratégia seleciona o exemplo mais recente armazenado no *buffer*. Considerando o *buffer* $\mathcal{B}$, a estratégia selecionaria o exemplo X_t para ter o rótulo requisitado.

Menos Recente

Ao contrário da estratégia *Mais Recente*, a estratégia *Menos Recente* seleciona o exemplo menos recente armazenado no *buffer*. Considerando o *buffer* $\mathcal{B}$, a estratégia selecionaria o exemplo X_{t-j} para ter o rótulo requisitado.

Aleatória

Esta estratégia seleciona de maneira aleatória algum exemplo armazenado no *buffer*. Para isso, considerando o *buffer* $\mathcal{B}$, é gerada uma variável aleatória i que segue uma distribuição uniforme em $[t, t-j]$ e, então é retornado o exemplo X_i para ter o rótulo requisitado.

Incerteza

Esta estratégia tem por objetivo selecionar o exemplo com maior incerteza para o modelo de classificação $\mathcal{F}$ no instante t.

Para isto, considerando o *buffer* $\mathcal{B}$, é gerado o vetor $\mathcal{B}_I^t = (F_I^t(X_t), F_I^t(X_{t-1}), F_I^t(X_{t-2}), ..., F_I^t(X_{t-j}))$, em que é F_I^t definido como: $\underset{Y}{\operatorname{argmax}} P_{\mathcal{F}_t}(Y|X) - \underset{Y}{\operatorname{argmin}} P_{\mathcal{F}_t}(Y|X)$, sendo $\mathcal{F}_t$ o modelo de classificação no instante t.

Dessa maneira, a estratégia baseada em *Incerteza* irá selecionar o exemplo $X_i (X_i \leftarrow \underset{X}{\operatorname{argmin}} \mathcal{B}_I^t)$ para ter o seu rótulo requisitado.

Incerteza com Custo

Da mesma forma que na estratégia *Incerteza*, a estratégia *Incerteza com Custo* considera o grau de incerteza dos exemplos em sua decisão, porém também leva em consideração o seu custo de obtenção, isto é, o intervalo de latência na obtenção do rótulo.

O custo de obtenção dos exemplos é estimado utilizando o rótulo predito pelo modelo de classificação $\mathcal{F}$. Para isso, considerando o *buffer* $\mathcal{B}$, é gerado o vetor $\mathcal{B}_{I_C}^t = \left(\frac{F_I^t(X_t)}{F_C^t(X_t)}, \frac{F_I^t(X_{t-1})}{F_C^t(X_{t-1})}, \frac{F_I^t(X_{t-2})}{F_C^t(X_{t-2})}, ..., \frac{F_I^t(X_{t-j})}{F_C^t(X_{t-j})}\right)$.

A função $F_C^t(X_t)$ retorna uma estimativa do intervalo de latência para a obtenção do rótulo do exemplo X_t, considerando que seu rótulo verdadeiro é o predito pelo modelo de classificação $\mathcal{F}$ no instante t. Para fazer tal estimativa é gerada uma variável aleatória T que segue uma distribuição *Gama*, sendo $T \sim \Gamma(n_{\hat{Y}_t}, T_{\hat{Y}_t})$, na qual seus parâmetros $n_{\hat{Y}_t}$ e $T_{\hat{Y}_t}$ representam, respectivamente, o número de exemplos pertencentes à classe $\hat{Y}_t$ que foram rotulados até o presente momento e o tempo total gasto no processo de rotulagem destes exemplos. Deste modo, o custo estimado de obtenção do rótulo verdadeiro do exemplo X_t é dado por T^{-1}.

Neste contexto, as distribuições *Gama* são utilizadas para estimar o tempo de rotulagem dos exemplos de acordo com as suas características, pois cada classe em particular terá uma distribuição *Gama* diferente com seus respectivos parâmetros. Essa distinção é importante pois classes diferentes podem ter tempos diferentes de rotulação pelo Oráculo.

É adotada a abordagem *Bayesiana* para o processo de atualização das distribuições. Considerando o instante $t + \Delta_t$, o rótulo Y_t do exemplo X_t é disponibilizado após um intervalo latência Δ_t. Deste modo, para atualizar a distribuição *Gama* associada à classe Y_t, o *framework* incrementa em uma unidade o número de exemplos que foram rotulados e pertencentes à classe Y_t, ou seja, $n_{Y_t} \leftarrow n_{Y_t} + 1$, e também atualiza o tempo total gasto no processo de rotulagem de tais exemplos, ou seja, $T_{Y_t} \leftarrow T_{Y_t} + \Delta_t$. Além disso, caso Y_t seja uma nova classe no fluxo de dados, a distribuição *Gama* referente a esta nova classe será iniciada com $n_{Y_t} \leftarrow 1$ e $T_{Y_t} \leftarrow \Delta_t$.

A estratégia *Incerteza com Custo* irá selecionar o exemplo X_i ($\leftarrow \underset{X}{\operatorname{argmin}}\ \mathcal{B}^t_{IC}$) com maior incerteza com relação ao seu custo de obtenção para ter o rótulo requisitado.

Incerteza com Custo e Aleatório

Na estratégia anterior, foi considerado o grau de informatividade do exemplo além do custo de obtenção do seu rótulo para a requisição. Deste modo, a estratégia seleciona o exemplo que irá trazer maior ganho de informação com o menor custo possível.

Porém, caso alguma das classes possua um custo muito menor de obtenção do seu rótulo com relação as demais, a estratégia de *Incerteza com Custo* pode selecionar os exemplos de maneira enviesada, priorizando a classe que possui o menor custo. Assim, a estratégia irá explorar pouco o espaço de entrada dos exemplos, não lidando com a mudança de conceito em parte deste espaço.

Portanto, na estratégia *Incerteza com Custo e Aleatório*, descrita no algoritmo 2, além de ser levado em consideração o grau de informatividade do exemplo e o custo de obtenção do seu rótulo, é adotado a aleatoriedade na seleção dos exemplos de tempos em tempos com o objetivo de explorar melhor o espaço de entrada dos exemplos.

Algoritmo 2: Estratégia *Incerteza com Custo e Aleatório*

Entrada: classificador $\mathcal{F}$, buffer $\mathcal{B} = (X_t, X_{t-1}, X_{t-2}, ..., X_{t-j})$
Saída: retorno exemplo $X_i \in \mathcal{B}$ para ser rotulado

p = 0.3, s = 0.05 ;
$J \sim Bernoulli(p)$;
se $J = 0$ **então**
  $p \leftarrow p \times (1 + s)$;
  $X_i \leftarrow$ *Aleatória* $(\mathcal{B})$;
fim
senão faça
  $p \leftarrow p \times (1 - s)$;
  $X_i \leftarrow$ *Incerteza com Custo* $(\mathcal{F}, \mathcal{B})$
fim
retorna X_i ;

Para isto, é gerada uma variável aleatória J que segue uma distribuição de $Bernoulli$ com parâmetro p e, caso o valor gerado de J for zero, a estratégia utilizada neste caso é a *Aleatória* e o parâmetro p é incrementado, aumentando assim a probabilidade de ser utilizado a estratégia de *Incerteza com Custo* na próxima seleção. Caso o valor gerado de J for um, a estratégia utiliza será a *Incerteza com Custo* e o valor de p é decrementado.

Na próxima seção é apresentado o resultado de alguns experimentos utilizando tais estratégias, juntamente com o *framework*, em diferentes graus de latência.

4. Resultados

As estratégias definidas na seção anterior, juntamente com o *framework*, foram submetidos a um conjunto de experimentos. Em tais experimentos, foi utilizada a base de dados *Electricity* [Harries et al. 1999] que contém 45.312 exemplos pertencentes as classes *UP*

e *DOWN*. Foi escolhido o algoritmo *Naïve Bayes* sem detector de mudanças de conceito do ambiente MOA [Bifet et al. 2010] como modelo de classificação.

Foram considerados 8 cenários distintos de latência apresentados na tabela 1. Em cada cenário, as classes *UP* e *DOWN* possuem um intervalo de latência diferente. Por exemplo, no cenário 5 um exemplo da classe *UP* demandará 600 rodadas de aprendizagem para ter o rótulo disponibilizado pelo Oráculo, já a classe *DOWN* demandará 1.200 rodadas de aprendizagem neste mesmo cenário.

	UP	DOWN
Cenário 1	50	100
Cenário 2	100	200
Cenário 3	150	300
Cenário 4	300	600
Cenário 5	600	1.200
Cenário 6	900	1.800
Cenário 7	1.200	2.400
Cenário 8	1.500	3.000

Tabela 1. Intervalos de latência

Para cada cenário descrito na Tabela 1 foram realizados cinco diferentes execuções, e a média das acurácias globais obtidas nas execuções por estratégia e cenário são apresentadas na Tabela 2.

Estratégias	Cenário 1	Cenário 2	Cenário 3	Cenário 4	Cenário 5	Cenário 6	Cenário 7	Cenário 8
Aleatório	70.76%	69.16%	66.87%	64.78%	58.08%	**59.47%**	54.21%	54.39%
Mais Recente	70.29%	**70.22%**	**68.93%**	63.32%	51.62%	50.61%	55.89%	55.98%
Menos Recente	70.05%	67.61%	65.6%	**67.14%**	58.71%	48.5%	46.59%	55.24%
Incerteza	**70.80%**	65.85%	68.24%	62.81%	51.2%	53.26%	55.89%	56.79%
Incerteza com Custo	69.31%	68.43%	67.71%	67.14%	**61.35%**	58.62%	**59.34%**	57.03%
Incerteza com Custo e Aleatório	70.50%	68.41%	67.9%	66.07%	58.58%	58.85%	58.81%	**57.58%**

Tabela 2. Resultados obtidos

Para analisar estatisticamente a diferença entre as estratégias, os resultados foram submetidos ao método não paramétrico descrito em [Brunner and Puri 2001]. Utilizando tal método, foi possível obter os *efeitos relativos* das estratégias nos diferentes cenários descritos, conforme é mostrado na Figura 1.

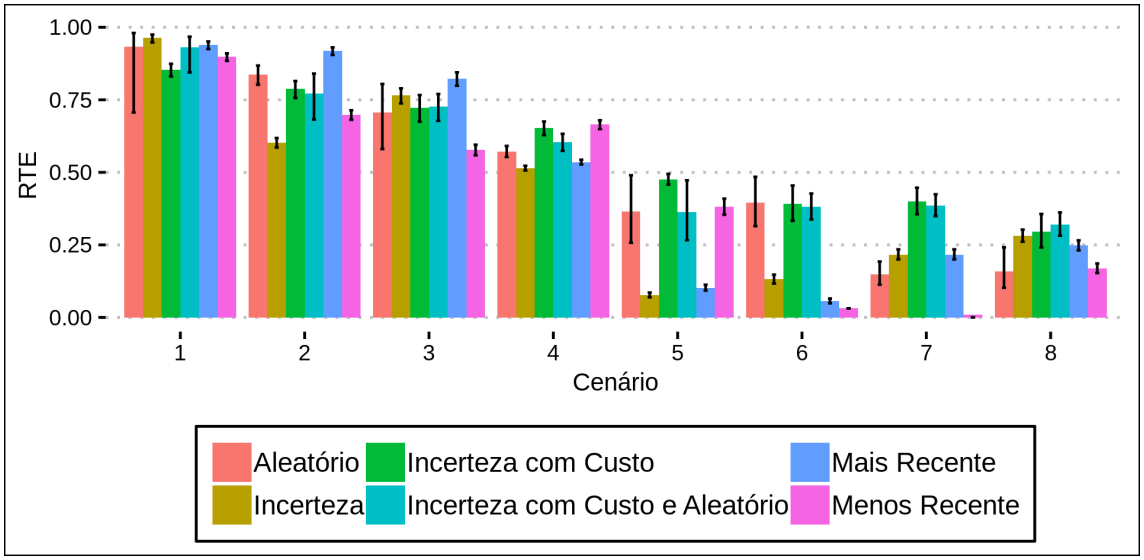

Figura 1. Resultados

O *efeito relativo* indica que, por exemplo, considerando todos os cenários, qualquer resultado escolhido de forma aleatória possui 40.04% de chance de ser inferior a um resultado escolhido de forma aleatória dentro do cenário 7, que foi obtido pela estratégia de *Incerteza com Custo*, ou de 38.63% no caso da estratégia de *Incerteza com Custo e Aleatório*. Desta maneira, é possível observar que conforme o intervalo de latência aumenta, maior se torna o *efeito relativo* das estratégias baseadas em custo em relação as demais, indicando um aumento do grau de importância do custo em relação ao ganho de informação.

Analisando os resultados, é possível observar que a medida que o intervalo de latência aumenta, menos exemplos rotulados serão disponibilizados para a adaptação do modelo de classificação ao longo do fluxo e, consequentemente, maior será a dificuldade em lidar com a mudança de conceito. Dessa maneira, é natural esperar uma acurácia maior com intervalos de latência menor.

Além disso, devido à mudança de conceito, conforme o intervalo de latência aumenta, mais incerto se torna o ganho de informação dos exemplos na adaptação do modelo de classificação, ou seja, menor se torna o grau de importância do ganho de informação na decisão de escolha do exemplo que terá requisitado seu rótulo para a adaptação do modelo de classificação.

Dessa maneira, as estratégias de aprendizagem ativa que consideram o custo de obtenção dos rótulos conseguem obter uma melhor performance nos cenários que apresentam um maior intervalo de latência, pois diminuem o grau de importância do ganho de informação na escolha do exemplo que terá requisitado seu rótulo e, consequentemente, aumenta o número de exemplos rotulados para a adaptação do modelo de classificação ao longo do fluxo.

5. Conclusão

Com o avanço de diversas tecnologias, as aplicações de fluxos de dados tendem a se tornar mais comum. Apesar de existir diversas abordagens na literatura que assumem que

apenas é possível obter uma pequena porção de exemplos rotulados, nenhuma contempla a existência da *latência intermediária*. Tais abordagens assumem um cenário pouco provável em aplicações reais de fluxo de dados, o da disponibilidade imediada dos rótulos, uma vez que é presumível assumir que o processo de rotulação dos exemplos consome um tempo não negligenciável para a obtenção do rótulo.

Neste artigo foram propostas algumas estratégias de aprendizagem ativa e um *framework* que são capazes de lidar com a existência da *latência intermediária*. Os resultados do experimento realizado mostraram que, conforme o intervalo de *latência* aumenta, maior se torna o grau de importância do custo em relação ao ganho de informação. Desta maneira, é esperado que futuros trabalhos de aprendizagem ativa em fluxo de dados possam ser desenvolvidos considerando a existência da *latência intermediária*, no qual se tenha um *trade-off* entre o ganho de informação e o custo de obtenção dos rótulos de acordo o grau da latência. Em trabalhos futuros, pretendemos estender a validação experimental para outras bases de dados, bem como investigar outras estratégias de aprendizagem ativa dentro do contexto de latência intermediária de rotução de exemplos.

Agradecimentos

O presente trabalho foi realizado com apoio parcial da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 1765710.

Referências

Aggarwal, C. C. (2007). *Data streams: models and algorithms*, volume 31. Springer Science & Business Media.

Alves de Souza, V., Pinho da Silva, T., and Batista, G. (2018). Evaluating stream classifiers with delayed labels information. In *7th Brazilian Conference on Intelligent Systems, BRACIS 2018, São Paulo, Brazil*, pages 408–413.

Attenberg, J. and Provost, F. (2011). Online active inference and learning. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 186–194, New York, NY, USA. ACM.

Bifet, A., Holmes, G., Kirkby, R., and Pfahringer, B. (2010). Moa: Massive online analysis. *J. Mach. Learn. Res.*, 11:1601–1604.

Brunner, E. and Puri, M. L. (2001). Nonparametric methods in factorial designs. *Statistical Papers*, 42(1):1–52.

Ditzler, G., Roveri, M., Alippi, C., and Polikar, R. (2015). Learning in nonstationary environments: A survey. *IEEE Computational Intelligence Magazine*, 10(4):12–25.

Gama, J. (2010). *Knowledge discovery from data streams*. CRC Press.

Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., and Bouchachia, A. (2014). A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):44.

Golab, L. and Özsu, M. T. (2003). Issues in data stream management. *ACM Sigmod Record*, 32(2):5–14.

Hanqing Hu, M. K. (2017). Sliding reservoir approach for delayed labeling in streaming data classification. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

Hao, S., Lu, J., Zhao, P., Zhang, C., Hoi, S. C. H., and Miao, C. (2018). Second-order online active learning and its applications. *IEEE Transactions on Knowledge and Data Engineering*, 30(7):1338–1351.

Harries, M., cse tr, U. N., and Wales, N. S. (1999). Splice-2 comparative evaluation: Electricity pricing. Technical report, University of New South Wales.

Kuncheva, L. I. and Sánchez, J. S. (2008). Nearest neighbour classifiers for streaming data with delayed labelling. In *2008 Eighth IEEE International Conference on Data Mining*, pages 869–874.

Marrs, G. R., Hickey, R. J., and Black, M. M. (2010). The impact of latency on online classification learning with concept drift. In Bi, Y. and Williams, M.-A., editors, *Knowledge Science, Engineering and Management*, pages 459–469, Berlin, Heidelberg. Springer Berlin Heidelberg.

Mohamad, S., Bouchachia, A., and Sayed-Mouchaweh, M. (2018). A bi-criteria active learning algorithm for dynamic data streams. *IEEE Transactions on Neural Networks and Learning Systems*, 29(1):74–86.

Mohamad, S., Sayed Mouchaweh, M., and Bouchachia, H. (2017). Active learning for classifying data streams with unknown number of classes. *Neural Networks*, 98.

Plasse, J. and Adams, N. (2016). Handling delayed labels in temporally evolving data streams. In *2016 IEEE International Conference on Big Data (Big Data)*, pages 2416–2424.

Pozzolo, A. D., Boracchi, G., Caelen, O., Alippi, C., and Bontempi, G. (2015). Credit card fraud detection and concept-drift adaptation with delayed supervised information. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Pozzolo, A. D., Boracchi, G., Caelen, O., Alippi, C., and Bontempi, G. (2018). Credit card fraud detection: A realistic modeling and a novel learning strategy. *IEEE Transactions on Neural Networks and Learning Systems*, 29(8):3784–3797.

Webb, G. I., Hyde, R., Cao, H., Nguyen, H. L., and Petitjean, F. (2016). Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994.

Zhao, P. and Hoi, S. C. (2013). Cost-sensitive online active learning with application to malicious url detection. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, pages 919–927, New York, NY, USA. ACM.

Žliobaitė, I., Bifet, A., Pfahringer, B., and Holmes, G. (2014). Active learning with drifting streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):27–39.