Conjunto de Dados para Envenenamento de Modelos Baseados em Self-Training em Fluxo de Dados

David Lucas Pereira Gomes
Departamento de Informática
Universidade Federal do Paraná
Curitiba, Paraná, Brasil
david.gomes@ufpr.br

André Ricardo Abed Grégio
Departamento de Informática
Universidade Federal do Paraná
Curitiba, Paraná, Brasil
gregio@ufpr.br

Paulo Lisboa de Almeida
Departamento de Informática
Universidade Federal do Paraná
Curitiba, Paraná, Brasil
paulorla@ufpr.br

Abstract

In problems with a large volume of unlabeled data, semi-supervised learning techniques, such as self-training, are attractive because they make full use of the data and do not require extensive labeling of the data, since it is an expensive process. However, using pseudo-labels to train a model indiscriminately can lead to undue changes in the model's decision boundary, which can happen unintentionally or intentionally, such as in malware classification, where attackers want to classify malicious software as benign. In this paper, we propose a dataset for poisoning models based on self-training that simulates a data stream, intending to evaluate the robustness of these models against intentional or unintentional poisoning by unlabeled instances. Our experiments use models from the *MOA-SS framework*, and show that models that use incremental training and prediction confidence as a criterion for using the unlabeled instance in training are more susceptible to poisoning.

Keywords

Model Poisoning, Data Streams, Self-training, Pseudo-labels, Semi-supervised Learning.

1 Introdução

A quantidade de dados não rotulados geralmente é muito maior que a de rotulados em diversos domínios [1–3], uma vez que o custo de rotulação pode ser significativo ao considerar grandes massas de dados brutos. Além disso, domínios com geração contínua de dados, como transações bancárias, tornam a rotulação integral dos dados inviável, visto que em um curto intervalo de tempo diversas instâncias são geradas, as quais chegam ao modelo de aprendizado de máquina sob a forma de um fluxo de dados.

Métodos de aprendizado de máquina que combinam dados rotulados e não rotulados são denominados aprendizado semi-supervisionado, o qual possui técnicas que visam extrair conhecimento do grande volume de instâncias não rotuladas, sem superestimar sua importância [4]. *Self-training* é uma categoria de aprendizado semi-supervisionado, caracterizada pelo uso de um modelo supervisionado como base cujas predições de instâncias não rotuladas são utilizadas para seu próprio treinamento como *ground-truth* (denominadas pseudo-rótulos), caso o *score* de confiança da predição seja maior que um determinado limiar [5, 6].

Desta forma, técnicas como *self-training* tornam-se relevantes para problemas cuja proporção de dados não rotulados é consideravelmente maior, devido ao custo para rotulação dos dados. Contudo, o uso indiscriminado de pseudo-rótulos para treinamento de modelos pode levar a uma mudança indesejada na fronteira de decisão, deteriorando o desempenho do modelo. Esta alteração pode ser involuntária ou intencional, no caso de um ataque por um agente mal-intencionado.

Em problemas de classificação de dados de segurança, como *malware* e transações bancárias fraudulentas, que geralmente se comportam como um fluxo de dados, há um grande volume de dados não rotulados sendo gerados constante e rapidamente. A utilização de modelos baseados em *self-training* nestes casos poderia abrir um espaço de ataque para agentes mal-intencionados, cujo ataque poderia se basear na introdução de instâncias especialmente selecionadas e não rotuladas ao modelo, de forma a alterar a fronteira de decisão. Com isso, o adversário pode forçar uma classificação errônea de um software malicioso para benigno, por exemplo. Esse tipo de ataque é conhecido como envenenamento de modelos.

Neste trabalho, é proposto um conjunto de dados bidimensional e binário de envenenamento de modelos que simula um fluxo de dados, com o objetivo de avaliar a robustez de modelos baseados em *self-training* contra envenenamento (intencional ou não). Para tanto, mede-se o desempenho do modelo em um conjunto de dados de avaliação ao decorrer de seu treinamento no conjunto de dados de envenenamento. Os experimentos mostram que modelos que utilizam treinamento incremental em *batch* e *score* de confiança baseado em distância são mais resistentes contra o envenenamento, enquanto modelos baseados em treinamento incremental são mais suscetíveis ao ataque.

2 Contexto e Trabalhos Relacionados

Nesta seção, descrevemos conceitos necessários para compreender nosso conjunto de dados proposto, como fluxo de dados, aprendizado semi-supervisionado e *self-training*, e também apresentamos trabalhos do estado da arte.

2.1 Fluxo de Dados

Uma forma de categorizar a aprendizagem de máquina é em relação à forma como os dados são processados e aprendidos. A forma mais clássica é em *batch*, também chamada de treinamento *offline*, em que o modelo é treinado utilizando um conjunto de dados fixo, disponível completamente ao início do treinamento. Esta forma é utilizada em problemas como reconhecimento facial e detecção de objetos, em que os conjuntos de dados são criados integralmente anterior ao treinamento dos modelos e os dados não mudam frequentemente.

Na outra forma, chamada de fluxo de dados ou treinamento *online*, o modelo passa por um fluxo de dados contínuo, sendo considerado infinito, e é treinado incrementalmente a cada instância ou a cada pequeno conjunto de instâncias, que são descartadas após seu processamento [7]. Um exemplo de problema que se categoriza como fluxo de dados é a detecção de fraude de transações bancárias,

no qual a cada pequeno intervalo de tempo várias transações são geradas, e o modelo deve se adaptar rapidamente, visto que os padrões dos dados podem mudar ao longo do tempo.

Em [8], Gomes et al. apresentam uma pesquisa abrangente sobre fluxo de dados em diversas áreas do aprendizado de máquina, como pré-processamento de dados, aprendizado semi-supervisionado, aprendizado desbalanceado e mudança de conceito (*concept drift*).

Como *frameworks* de fluxo de dados, há a biblioteca *River*, proposto por Montiel et al. [9], um módulo Python para aprendizado de máquina *online* que disponibiliza algoritmos e técnicas clássicos adaptados para fluxo de dados, como árvores de decisão, *clustering* e processamento de dados.

Contudo, neste trabalho foi utilizada uma *framework* para fluxo de dados baseada na biblioteca *MOA* (*Massive Online Analysis*, apresentada por Bifet et al. [10], que consiste em um ambiente para experimentos de aprendizado de máquina *online* na linguagem Java, com algoritmos e modelos como a árvore de decisão *Hoeffding Tree*.

2.2 Aprendizado Semi-Supervisionado e *Self-Training*

Classificação em aprendizado de máquina pode ser dividida em duas categorias principais: aprendizado supervisionado e aprendizado não supervisionado [11]. Na primeira, tem-se acesso a dados rotulados, i.e., dados com suas verdadeiras classificações, os quais podem ser utilizados para treinamento de modelos como árvores de decisão e SVMs (*Support Vector Machines*). Na segunda, há dados não rotulados, geralmente em uma maior quantidade, que são utilizados em algoritmos de clusterização, por exemplo. Dados rotulados geralmente possuem um custo maior de criação, visto que o processo de rotulação é grande para um grande volume de dados. Assim, dados não rotulados, na maior parte das vezes, aparecem em maior quantidade.

O aprendizado semi-supervisionado, por sua vez, combina tanto os dados rotulados como os dados não rotulados, utilizando algoritmos de ambas as técnicas para extração de conhecimento do grande volume de dados não rotulados [4, 12]. *Self-training* é uma técnica de aprendizado semi-supervisionado em que o modelo base aprende utilizando suas próprias predições, em um processo iterativo, em que estas predições podem ser utilizadas como pseudo-rótulos, caso a confiança da predição do modelo seja maior que um limiar fixo ou dinâmico [5, 6].

Assim, a instância não rotulada é utilizada para o treinamento do modelo base, em que seu rótulo é a predição dada pelo modelo previamente, i.e., seu pseudo-rótulo. Comumente, esse tipo de modelo é treinado utilizando os dados rotulados disponíveis e, posteriormente, são feitas as predições com as instâncias não rotuladas, em que as que possuem o maior *score* de confiança são utilizadas para o treinamento do modelo, utilizando suas predições, ou seja, seus pseudo-rótulos, como os *ground truths*.

Em [13], Nguyen et al. apresentam as técnicas *cluster-and-label* e *self-training* de aprendizado de máquina semi-supervisionado para fluxo de dados, implementando os modelos utilizando a API do *MOA* e propondo a *framework MOA-SS*. Visto a diversidade dos modelos baseados em *self-training* presentes nesta *framework*, utilizamos-a em nossos experimentos.

Além dos modelos de *self-training* presentes no *MOA-SS*, há outros algoritmos para aprendizado semi-supervisionado que podem variar a escolha de modelos, o estilo de treinamento ou o critério para uso de dados não rotulados no treinamento com pseudo-rótulo, alguns exemplos são citados a seguir.

Em [14], Khezri et al. propõem o algoritmo STDS, baseado em *self-training* para fluxo de dados, levando em consideração mudança de conceitos (*concept drift*) e utilizando como modelo base o Naive Bayes Incremental.

Monteiro et al., em [15], apresentam o treinamento Co-op, um algoritmo de aprendizado semi-supervisionado inspirado em *self-training* que utiliza dois modelos, um modelo base principal e um sub-modelo, que é utilizado para estimar a confiança da instância não rotulada como critério para seu uso no treinamento do modelo principal.

Por fim, *COMPOSE* [16], proposto por Dyer et al., consiste em uma *framework* baseada em geometria computacional, recomendada para ambientes com mudança de conceito incremental ou gradual.

Para o escopo dos experimentos deste trabalho, foram escolhidos apenas os modelos baseados em *self-training* presentes na *framework MOA-SS* [13].

3 Conjunto de Dados Proposto

Nesta seção, propomos um conjunto de dados para fluxo de dados que exemplifica o envenenamento de modelos baseados em *self-training*, ou seja, seu objetivo é avaliar a robustez de um modelo contra envenenamento por instâncias não rotuladas, visando uma possível alteração de sua fronteira de decisão, o que leva a predições incorretas em relação à fronteira de decisão real pela qual o conjunto de dados foi criado. Desta forma, demonstra casos em que as instâncias não rotuladas são dadas ao modelo de forma intencional ou involuntária.

Um exemplo real é a de aprendizado de máquina aplicado à detecção de *malware*, no contexto de segurança computacional, no qual há poucos *softwares* rotulados, tornando o aprendizado semi-supervisionado atraente. Considere que um modelo foi treinado para classificar instâncias de software nas classes *goodware* (benigno) ou *malware* (malicioso). Um atacante, na intenção de levar este modelo a classificar um programa malicioso como *goodware*, pode alimentá-lo com instâncias próximas de um software não malicioso, alterando determinadas características para valores próximos aos associados a *malware*. Assim, o modelo, que utiliza dados não rotulados para seu treinamento, associaria instâncias cada vez mais próximas de software malicioso como *goodware*, tornando mais fácil classificar um *malware* como um software não malicioso, pois a fronteira de decisão se aproximaria dos dados com rótulos da classe maliciosa.

Para sintetizar este cenário em um experimento, propomos um conjunto de dados bidimensional, i.e., com duas características para facilitar sua visualização, que simula um fluxo de dados, e com duas classes, positiva e negativa, em que os primeiros 25% dos dados são totalmente rotulados, gerados considerando uma distribuição uniforme. O rótulo de cada instância é dado de acordo com a seguinte função:

$$ground\ truth = \begin{cases} 0, & \text{se } x_1 - x_2 < 0 \\ 1, & \text{caso contrário} \end{cases}$$

onde x_1 e x_2 são os valores das características de uma determinada instância e $x_1 \in [0, 50]$ e $x_2 \in [0, 50]$. Desta forma, temos a fronteira de decisão real do problema, separando os dados no *cluster* negativo na parte superior do espaço e o *cluster* positivo na porção inferior, mostrada na Figura 1.

Também, é considerada uma margem para a geração das instâncias a partir da fronteira de decisão, ou seja, a distância mínima em relação à fronteira de decisão real em que não há geração de dados, para que possa ocorrer a adaptação da fronteira de decisão do modelo com base nos dados não rotulados. Após a geração dos primeiros 25% dos dados, em que todos possuem rótulos, as instâncias seguintes são distribuídas em 10% rotuladas e 90% não rotuladas. Dentre as instâncias não rotuladas, 80% são geradas no *cluster* negativo (superior) e, por conseguinte, 20% geradas no *cluster* positivo (inferior).

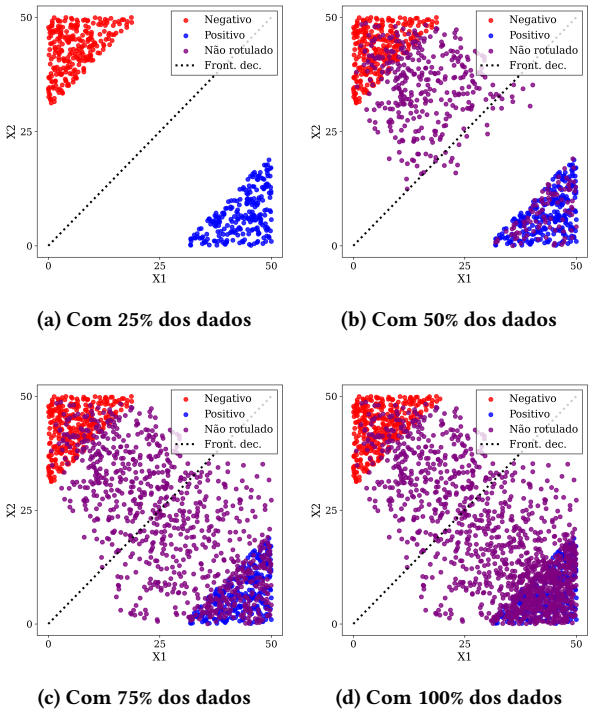
Para simular o envenenamento, é necessário que instâncias não rotuladas sejam geradas, inicialmente, próximas de um *cluster* X e, durante sua geração, aproximem-se do outro *cluster*, Y, levando, possivelmente, a um modelo suscetível a envenenamento modificar sua fronteira de decisão, aproximando-a do *cluster* Y, de forma que as instâncias não rotuladas geradas inicialmente próximas de X sejam classificadas com a classe de X.

Assim, em nosso conjunto de dados, implementa-se um movimento de instâncias não rotuladas partindo da classe negativa (superior) à classe positiva (inferior) utilizando um valor de *shift* que é adicionado à primeira característica x_1 da instância e subtraído de sua segunda característica x_2. Este *shift* é inicializado em zero e incrementado com um valor de passo a cada dado não rotulado gerado no *cluster* negativo.

Para calcular o valor do passo, foi utilizada a seguinte fórmula: $passo = \frac{\text{valor máximo de X} - \text{valor mínimo de X}}{\#\text{instâncias} \times \text{proporção de dados não rotulados}}$, onde X é a característica de uma instância. Ou seja, como mostrado anteriormente, o valor mínimo e máximo de X são, respectivamente, 0 e 50. Esta fórmula para o valor de passo foi escolhida de forma que as instâncias não rotuladas iniciais sejam geradas dentro do *cluster* superior, as seguintes, cada vez mais próximas do *cluster* inferior, e apenas as últimas instâncias não rotulas geradas dentro deste *cluster*.

Assim, dado este movimento gradual de instâncias não rotuladas originando do *cluster* superior (classe negativa) ao inferior (classe positiva), espera-se que um modelo vulnerável a envenenamento tenha sua fronteira de decisão modificada em relação a um modelo que não utiliza *self-training*, movendo-a em direção ao *cluster* inferior.

A Figura 1 apresenta o conjunto de dados proposto, em um plano com duas dimensões, visto que o conjunto possui apenas duas características, a cada 25% dos dados. Ou seja, na Figura 1a, está presente apenas os primeiros 25% dos dados, enquanto a Figura 1b possui os primeiros 50% dos dados etc. Ao todo, foram geradas 2.000 instâncias, cujo valor mínimo de x_1 e x_2 é zero, e máximo, 50. Foi considerada uma margem de valor 30 a partir da fronteira de decisão para a geração de dados rotulados.



(a) Com 25% dos dados **(b) Com 50% dos dados**

(c) Com 75% dos dados **(d) Com 100% dos dados**

Figura 1: Dataset proposto.

4 Experimentos

Nesta seção, descrevemos a metodologia e resultados dos experimentos feitos com diferentes modelos que consideram instâncias não rotuladas, i.e., baseadas em *self-training*. O *dataset* apresentado na Figura 1 foi utilizado para treinar os modelos descritos a seguir, e um conjunto de dados de avaliação, descrito abaixo, para avaliá-los.

4.1 Metodologia

Para testar o conjunto de dados, utilizamos a *framework MOA-SS* [13], que possui modelos de aprendizado semi-supervisionado que utilizam instâncias não rotuladas baseados em *self-training*. Todos utilizam como modelo base a árvore de decisão *Hoeffding Tree*, e variam em três aspectos:

(1) **treinamento**, podendo ser incremental por instância ou por *batch*;
(2) **estimativa do *score* de confiança de uma instância não rotulada**, baseada na distância entre a instância e os outros dados rotulados de mesma classe ou no valor de confiança da predição da instância dado pelo modelo base;
(3) **limiar de confiança**, utilizado para decidir se a instância não rotulada será usada para treinamento com seu pseudo-rótulo, que pode ser fixo ou adaptável.

No treinamento incremental por instância, cada dado recebido no fluxo de dados é avaliado isoladamente, i.e., seu pseudo-rótulo é gerado com a predição do modelo e sua estimativa do *score* de confiança é comparada ao limiar de confiança e, se for maior ou igual a este limiar, a instância é usada para treinar o modelo, utilizando seu pseudo-rótulo como *ground truth*.

Por outro lado, no treinamento incremental por *batch*, um pequeno *buffer* para armazenar as instâncias recebidas é utilizado, que, quando cheio, apenas as instâncias com maior *score* de confiança são utilizadas para treinamento, e as outras, descartadas, esvaziando o *buffer*.

Tabela 1: Modelos de *Self-Training*.

Nome	Treinamento	*Score* de conf.	Limiar de conf.
BDF	Batch	Distância	Fixo
BDA	Batch	Distância	Adaptável
BPF	Batch	Predição	Fixo
BPA	Batch	Predição	Adaptável
IPF	Instância	Predição	Fixo
IPA	Instância	Predição	Adaptável
IDW	Instância	Predição	-
IEW	Instância	Predição	-

A Tabela 1 lista os modelos baseados em *self-training* presentes na *MOA-SS* que são utilizados em nossos experimentos. Os modelos IDW e IEW diferem na utilização da confiança, isto é, utilizam todas as instâncias não rotuladas para treinamento, atribuindo um peso a cada instância com base no *score* de confiança da predição dado pelo modelo base. O IDW utiliza um treinamento incremental por instância em que o peso dado às instâncias é o próprio *score* de confiança da predição dado pelo modelo, enquanto o IEW também utiliza treinamento incremental por instância, mas todas as instâncias possuem o mesmo peso (1,0).

Os modelos passam pelo fluxo de dados, com o conjunto de instâncias de envenenamento gerado previamente, em que todo dado é utilizado para treinamento, na forma *test-then-train*. Ou seja, para cada instância do conjunto, o modelo faz sua predição, que é usada para estimar seu desempenho, e utiliza-a para treinamento, levando em consideração se possui rótulo ou não.

Para avaliar a robustez de cada modelo ao envenenamento, foi utilizado um conjunto de pontos de avaliação, totalmente rotulado, gerado aleatoriamente no espaço seguindo uma distribuição uniforme, com seus rótulos atribuídos da mesma forma do *dataset* de envenenamento.

Assim, a cada 10% dos dados de envenenamento utilizados para treinamento, o modelo, treinado com todos as instâncias vistas até o momento no fluxo de dados, é utilizado para classificar os pontos do conjunto de avaliação, e suas predições usadas para calcular sua acurácia.

Desta forma, espera-se que um modelo de *self-training* vulnerável a envenenamento diminua sua acurácia, após seu máximo, no *dataset* de avaliação conforme é treinado com dados não rotulados do *dataset* de envenenamento, visto que sua fronteira de decisão foi alterada por conta de um envenenamento acidental ou intencional.

4.2 Resultados

Nesta seção, são apresentados os resultados do desempenho de cada modelo após passar pelo fluxo de dados, avaliando sua robustez ao envenenamento. As acurácias de cada modelo no conjunto de avaliação a cada 10% dos dados de envenenamento utilizados para treinamento encontram-se na Tabela 2.

Tabela 2: Acurácia dos modelos no conjunto de avaliação a cada 10% dos dados de treinamento.

Modelo	30%	40%	50%	60%	70%	80%	90%	100%
BDF	96%	97%	97%	97%	97%	97%	96%	97%
BDA	96%	97%	97%	97%	97%	97%	96%	97%
BPF	96%	97%	97%	87%	87%	87%	87%	87%
BPA	96%	97%	97%	87%	87%	87%	87%	87%
IPF	98%	85%	75%	69%	69%	71%	72%	72%
IPA	98%	85%	75%	69%	69%	71%	72%	72%
IDW	98%	85%	75%	70%	70%	72%	72%	72%
IEW	98%	85%	75%	70%	70%	72%	72%	72%

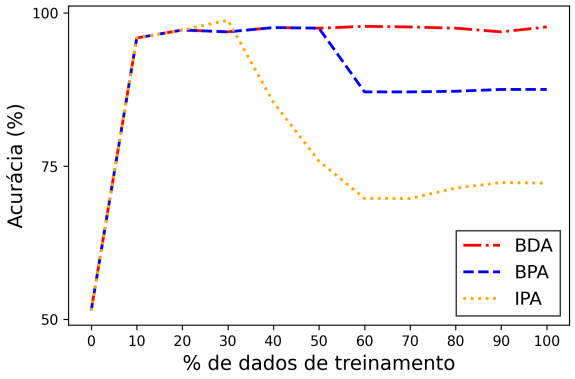


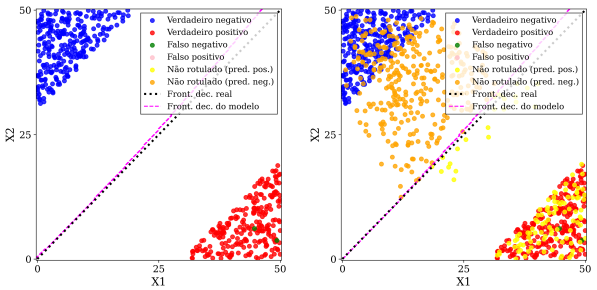
Figura 2: Acurácias dos modelos BDA, BPA e IPA no conjunto de avaliação a cada 10% dos dados de treinamento.

A partir das acurácias presentes na Tabela 2, conclui-se que alguns modelos variam seu desempenho de forma similar. Desta forma, pode-se dividir os modelos com base na diminuição de sua acurácia, após seu máximo, ao longo do treinamento:

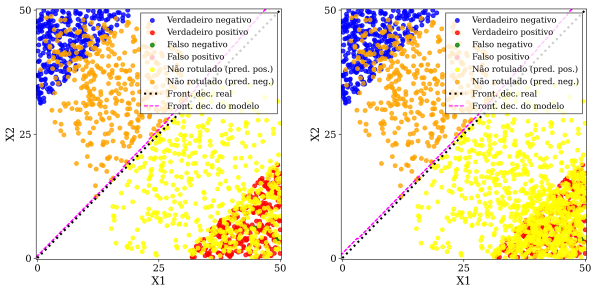
- Diminuição não significativa: BDF e BDA, com acurácia final de 97%;
- Menor diminuição: BPF e BPA, com acurácia final de 87%;
- Maior diminuição: IPF, IPA, IDW e IEW, com acurácia final de 72%.

Para simplificar a visualização dos resultados, foram escolhidos os modelos BDA, BPA e IPA, das categorias diminuição não significativa, menor diminuição e maior diminuição respectivamente. Suas acurácias no conjunto de dados de avaliação durante o treinamento no *dataset* de envenenamento proposto estão representadas na Figura 2.

Os modelos BDA e BDF não apresentaram diminuição significativa de suas acurácias. Eles têm em comum o treinamento incremental em *batches* de instâncias e a utilização do *score* de confiança com base na distância, descrita anteriormente. A Figura 3 apresenta as predições do modelo BDA, similares às do modelo BDF, pela qual é possível verificar a ausência de uma mudança significativa na fronteira de decisão do modelo ao longo de seu treinamento nos dados de envenenamento, ou seja, estes modelos mostraram-se os mais robustos contra envenenamento por instâncias não rotuladas.

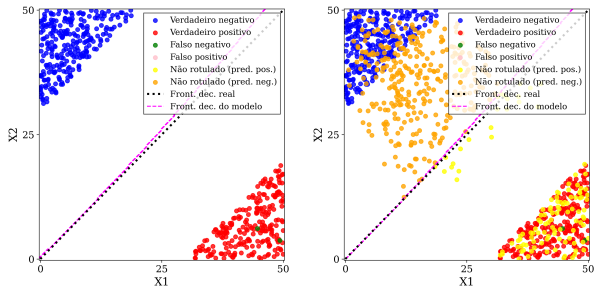


(a) Com 25% dos dados **(b) Com 50% dos dados**

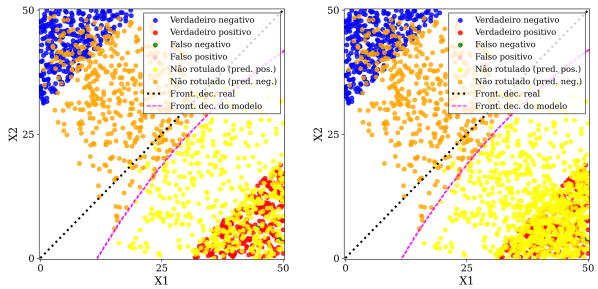


(c) Com 75% dos dados **(d) Com 100% dos dados**

Figura 3: Predições do modelo BDA.



(a) Com 25% dos dados **(b) Com 50% dos dados**



(c) Com 75% dos dados **(d) Com 100% dos dados**

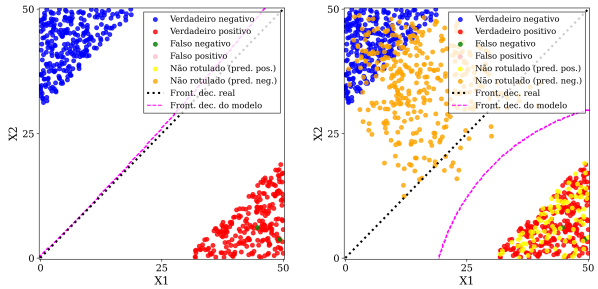
Figura 4: Predições do modelo BPA.

A Figura 4, por sua vez, possui as predições do modelo BPA, similares às predições do modelo BPF. Estes utilizam treinamento incremental em *batch* e o próprio *score* de confiança da predição do modelo para escolher as instâncias não rotuladas que serão usadas para seu treinamento com pseudo-rótulos, apresentando uma leve diminuição em sua acurácia final e alteração na fronteira de decisão em direção à classe positiva (*cluster* inferior).

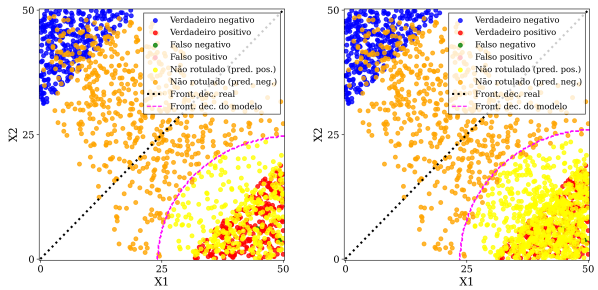
Por fim, a Figura 5 apresenta os as predições do modelo IPA, similares às predições dos modelos IPF, IDW e IEW. Estes têm como característica em comum a utilização do treinamento incremental por instância, na qual toda instância não rotulada pode ser utilizada para treinamento com pseudo-rótulo, dependendo unicamente de seu *score* de confiança da predição, dado pelo modelo, ser maior ou igual ao limiar de confiança escolhido (fixo ou adaptativo). Desta forma, de acordo com a Tabela 2, estes foram os modelos com maior redução na acurácia ao longo do treinamento, i.e., os mais vulneráveis a envenenamento por instâncias não rotuladas.

Assim, mostra-se que os modelos com maior resistência ao envenenamento são os que utilizam o treinamento incremental em *batches* de instâncias, i.e., salvam instâncias consecutivas em um pequeno *buffer*, que, quando cheio, as instâncias não rotuladas com maior confiança são selecionadas para treinamento, e, também, a estimativa do *score* com base na distância da instância em relação às instâncias rotuladas da classe.

Por outro lado, os modelos mais suscetíveis ao envenenamento foram os que utilizam o treinamento incremental por instância, ou seja, que processam um dado de cada vez, no qual seu *score* de confiança é avaliado isoladamente das demais instâncias, e que



(a) Com 25% dos dados **(b) Com 50% dos dados**



(c) Com 75% dos dados **(d) Com 100% dos dados**

Figura 5: Predições do modelo IPA.

utilizam o *score* de confiança de predição do modelo como estimativa de *score* de confiança.

Estes resultados podem-se dar pois o treinamento incremental em *batches* é menos sensível a ruído, i.e., instâncias com valores atípicos, tornando-se mais estável que o treinamento incremental por instância.

Também, o fato dos modelos com estimativa do *score* de confiança baseado na distância da instância não rotulada para as instância rotuladas de uma determinada classe ser mais robusto que o uso do *score* de confiança da predição dado pelo modelo base, pode-se dar por conta da distância não ser afetada pelo envenenamento, i.e., pelo uso de instâncias não rotuladas no treinamento com pseudo-rótulos, visto que utiliza apenas instâncias com *ground truth*, enquanto a confiança da predição do modelo é enviesada pelo uso de pseudo-rótulos como *groud truths*.

5 Conclusão

Neste trabalho, apresentamos um *dataset* capaz de degradar o desempenho de modelos baseados em *self-training* suscetíveis a envenenamento, alterando sua fronteira de decisão e diminuindo sua acurácia no conjunto de avaliação ao longo do treinamento com instâncias não rotuladas.

Mostramos que modelos baseados em *self-training* que utilizam treinamento incremental em pequenos *batches* de instâncias e estimativa do *score* de confiança baseado em distância para as instâncias rotuladas são mais resistentes ao envenenamento, enquanto modelos com treinamento incremental por instância e que utilizam o *score* de confiança de predição como estimativa do *score* de confiança são mais suscetíveis.

Como trabalho futuro, planejamos aplicar o *dataset* proposto em outros modelos de aprendizado semi-supervisionado, como os citados na Seção 2, além de avaliar conjuntos de dados para envenenamento com mais classes e dimensões.

Agradecimentos

Este trabalho foi financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) – Bolsa 405511/2022-1.

Referências

[1] Sally Goldman and Yan Zhou. Enhancing supervised learning with unlabeled data. In *ICML*, pages 327–334. Citeseer, 2000.

[2] Tong Zhang and F Oles. The value of unlabeled data for classification problems. In *Proceedings of the Seventeenth International Conference on Machine Learning,(Langley, P., ed.)*, volume 20, page 0. Citeseer, 2000.

[3] Nitesh V Chawla and Grigoris Karakoulas. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *Journal of Artificial Intelligence Research*, 23:331–366, 2005.

[4] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440, 2020.

[5] Massih-Reza Amini, Vasilii Feofanov, Loic Pauletto, Lies Hadjadj, Emilie Devijver, and Yury Maximov. Self-training: A survey. *Neurocomputing*, page 128904, 2024.

[6] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896. Atlanta, 2013.

[7] Jonathan A Silva, Elaine R Faria, Rodrigo C Barros, Eduardo R Hruschka, André CPLF de Carvalho, and João Gama. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*, 46(1):1–31, 2013.

[8] Heitor Murilo Gomes, Jesse Read, Albert Bifet, Jean Paul Barddal, and João Gama. Machine learning for streaming data: state of the art, challenges, and opportunities. *SIGKDD Explor. Newsl.*, 21(2):6–22, November 2019. ISSN 1931-0145. doi: 10.1145/3373464.3373470. URL https://doi.org/10.1145/3373464.3373470.

[9] Jacob Montiel, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine, Heitor Murilo Gomes, Jesse Read, Talel Abdessalem, et al. River: machine learning for streaming data in python. 2021.

[10] Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. Moa: Massive online analysis. *Journal of Machine Learning Research*, 11(52):1601–1604, 2010. URL http://jmlr.org/papers/v11/bifet10a.html.

[11] Mohamed Alloghani, Dhiya Al-Jumeily, Jamila Mustafina, Abir Hussain, and Ahmed J. Aljaaf. *A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science*, pages 3–21. Springer International Publishing, Cham, 2020. ISBN 978-3-030-22475-2. doi: 10.1007/978-3-030-22475-2_1. URL https://doi.org/10.1007/978-3-030-22475-2_1.

[12] Nitin Namdeo Pise and Parag Kulkarni. A survey of semi-supervised learning methods. In *2008 International Conference on Computational Intelligence and Security*, volume 2, pages 30–34, 2008. doi: 10.1109/CIS.2008.204.

[13] Minh Huong Le Nguyen, Heitor Murilo Gomes, and Albert Bifet. Semi-supervised learning over streaming data using moa. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 553–562, 2019. doi: 10.1109/BigData47090.2019.9006217.

[14] Shirin Khezri, Jafar Tanha, Ali Ahmadi, and Arash Sharifi. STDS: self-training data streams for mining limited labeled data in non-stationary environment. *Applied Intelligence*, 50(5):1448–1467, May 2020. ISSN 1573-7497. doi: 10.1007/s10489-019-01585-3. URL https://doi.org/10.1007/s10489-019-01585-3.

[15] Paulo Martins Monteiro, Elvys Soares, and Roberto Souto Maior de Barros. Co-op Training: a Semi-supervised Learning Method for Data Streams. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 933–938, October 2021. doi: 10.1109/SMC52423.2021.9658598. URL https://ieeexplore.ieee.org/abstract/document/9658598. ISSN: 2577-1655.

[16] Karl B Dyer, Robert Capo, and Robi Polikar. Compose: A semisupervised learning framework for initially labeled nonstationary streaming data. *IEEE transactions on neural networks and learning systems*, 25(1):12–26, 2013.