Differentiable Pattern Set Mining

Seminário 1 – Aprendizado Descritivo

2025.1

Historiador

Henrique Rotsen
Alexandre Cassimiro Silva Araújo
Wesley Marques Daniel Chaves
Caíque Bruno Fortunato
Victor Gabriel Moura Oliveira
Arthur Yochio R. Codama

Contextualização

O artigo trabalha o contexto de que a maior parte das abordagens de mineração de padrões não são muito eficazes para bases de dados muito grande em número de features, pois o espaço de busca geralmente é duplamente exponencial, e eles tendem a expandir a árvore completa de busca.

- A mineração de padrões enfrenta limitações em bases com muitos atributos (features).
- O espaço de busca cresce de forma duplamente exponencial com o número de atributos.
- Métodos tradicionais costumam explorar grande parte da árvore de busca, sendo ineficientes em bases extensas.
- BinaPs surge como uma alternativa escalável, baseada em aprendizado de máquina.

Tipo de padrão previamente visto em aula

Métodos Clássicos (Apriori, Eclat, FP-Growth)

- Enumeram todos os padrões frequentes com base em suporte mínimo.
- Geram muitos padrões redundantes e pouco úteis.
- Usam heurísticas em espaço de busca exponencial, sem aprendizado.
- Não utilizam função de perda, redes neurais ou otimização contínua.

BinaPs (Fischer & Vreeken, 2021)

- Aprende padrões por meio de função de perda diferenciável.
- Representa padrões como neurônios binarizados, interpretáveis.
- Ajusta automaticamente o número de padrões relevantes.
- Mais escalável e informativo, especialmente em bases grandes.

Comparação Histórica da Mineração de Padrões

Aspecto	Anos 1990 (Apriori)	Anos 2000 (FP-Growth)	2021 (BinaPs)
Algoritmos	Apriori (1994)	FP-Growth (2000)	BinaPs (2021)
Método	Geração de candidatos e múltiplas varreduras	Estrutura de árvore compacta (FP-Tree)	Autoenconders binários com funções de custo diferenciáveis
Hardware	CPUs Pentium, pouca RAM, armazenamento em discos rígidos de baixa rotação	CPUs mais rápidas, início do uso de GPUs	GPUs avançadas, computação em nuvem, armazenamento SSD
Redes Neurais	Pouco exploradas	Começo da popularização	Aprendizado baseado em gradientes e autoencoders sofisticados

Benchmarks: Asso, Slim e Desc

Métodos de mineração de padrões de última geração, usados no artigo como benchmarks para comparar o desempenho do BinaPs

1. **Asso**:
   - Fatoração de matrizes booleanas: Encontrar um conjunto compacto de padrões que represente bem os dados originais. [Rank Automatizado]
2. **Slim**:
   - Minerador de conjuntos, baseado no princípio Minimum Description Length (MDL), que diz que o melhor modelo para explicar seus dados é aquele que permite a compressão mais eficiente desses dados.
3. **Desc**:
   - Minerador de conjuntos usando **modelagem de máxima entropia** (menor quantidade de informação adicional possível)
   - Encontrar um conjunto de padrões que capture com precisão as propriedades estatísticas dos dados sem overfitting

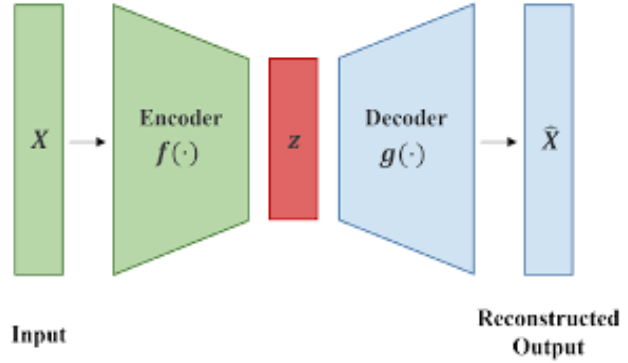
Formalizador

Grupo: ENILDA ALVES COELHO
FÁBIO CÉSAR MARRA FILHO
GABRIEL TONIONI DUARTE
IASMIN CORREA ARAUJO
JOSE VINICIUS DE LIMA MASSARICO
LARISSA DUARTE SANTANA
MARCELO LOMMEZ RODRIGUES DE JESUS

Autoencoders e BinaPs (Binary Pattern Network)

Autoencoders:

- Objetivo: Tentam extrair características principais dos dados para reconstrução dos mesmos.
- Se ajustam através do erro da saída em comparação com o dado inicial.
- Quando aplicados em serviços embarcados aprendem dados discretos através de otimização de variáveis contínuas, com redes neurais binárias com valores entre –1 e 1.
- Não tem uma relação clara e direta entre seus padrões, precisando do conjunto como um todo para formular a saída.



BinaPS:

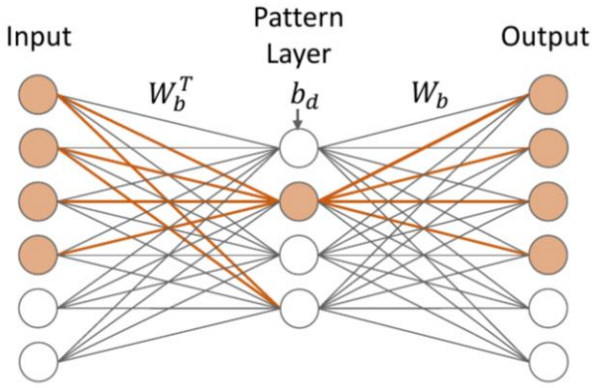
- Cada neurônio da rede neural é responsável por um padrão e seus pesos indicam esse padrão encontrado
- Sua binarização do peso é feita em 0 ou 1 fazendo com que seja bem claro se determinada feature está no padrão ou não

Database D

Items I

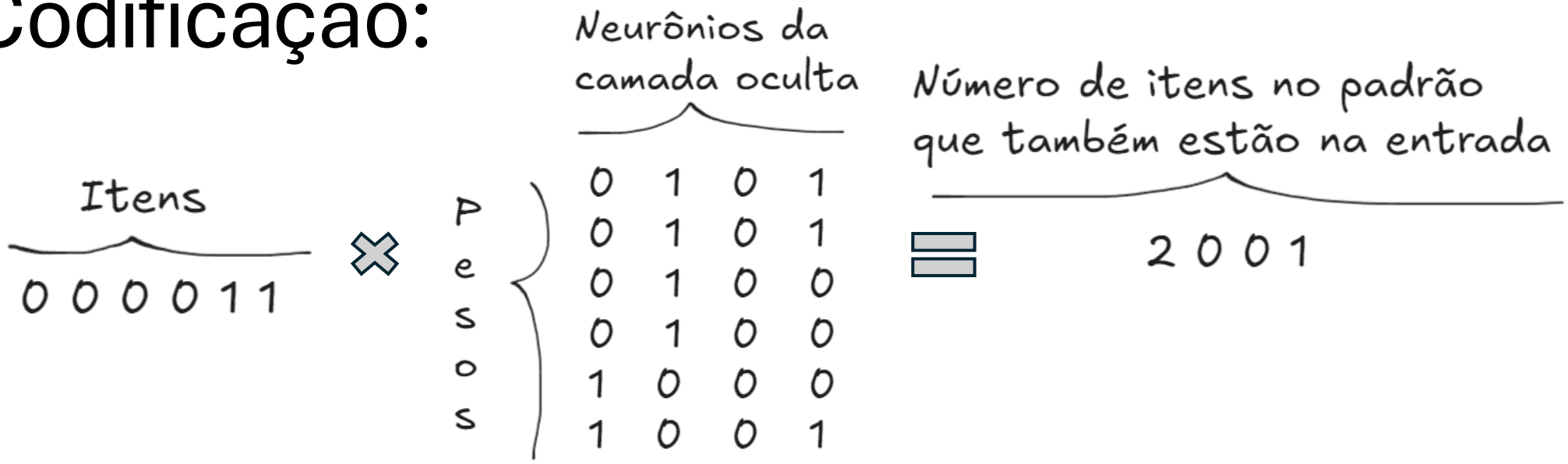
Samples S

0	0	0	0	1	1
1	1	1	1	0	0
1	1	1	1	1	1
1	1	0	0	0	1
1	1	0	0	0	1
0	0	0	0	1	1
1	1	0	0	0	1



Weight W						W_b						P	Bias b	b_d
0.02	0.01	0.00	0.01	0.95	0.91	0	0	0	0	1	1	{ {5, 6},	-1.1	-1
0.78	0.81	0.92	0.82	0.03	0.00	1	1	1	1	0	0	{1, 2, 3, 4},	-2.9	-2
0.03	0.04	0.07	0.05	0.01	0.00	0	0	0	0	0	0		-1.0	-1
0.93	0.90	0.03	0.01	0.07	0.87	1	1	0	0	0	1	{1, 2, 6} }	-2.1	-2

Codificação:



Número de itens no padrão que também estão na entrada

2 0 0 1

Viés negativo discretizado

-1 -2 -1 -2

1 -2 -1 -1

CLAMP(<1 -2 -1 -1>, 0, 1) = 1 0 0 0

ROUND(1 0 0 0) = 1 0 0 0

O padrão associado ao primeiro neurônio da camada oculta é suportado pelos dados da entrada

Funcionamento do Modelo

Algorithm 1: BINAPs training

input : dataset D, initial BINAPs (W^1, b^1), number of epochs $e_{\max}$, batch size l
output : pattern set P

$t \leftarrow 0$; // Step
for $e = 1 \ldots e_{\max}$ **do** // Epochs
  for $i = 1 \ldots n$ **do** // For each sample
    // **Forward pass**
    $W_b^t \leftarrow \mathcal{B}(W^t)$; // Binarize weights
    $b_d^t \leftarrow \lceil b^t \rceil$; // Discretize bias
    $y_i \leftarrow \text{round}(\text{clamp}(D[i](W_b^t)^\intercal + b_d^t, 0, 1))$; // f_E
    $z_i \leftarrow \text{round}(\text{clamp}(y_i W_b^t, 0, 1))$; // f_D
    $L(D[i]; W, b)$; // Compute loss
    // **Backward pass**
    $g_W \leftarrow \frac{dL}{dW}$; // Weight gradient
    $g_b \leftarrow \frac{dL}{db}$; // Bias gradient
    $W^{t+1} \leftarrow \text{update}(W^t, g_W)$; // Optimizer step
    $b^{t+1} \leftarrow \text{update}(b^t, g_b)$;
    $t \leftarrow t + 1$;
$P = \{\{j \mid \text{round}(W^t[i, j]) = 1\} \mid i \in 1 \ldots k\}$; // Pattern set
return P

Funções Base:

$$W_b[i,j] = \mathcal{B}(W[i,j]) \qquad \text{clamp}(x, a, b) = \begin{cases} a & \text{if } x < a\,, \\ x & \text{if } a \leq x \leq b \\ b & \text{if } b < x\,. \end{cases}$$

Funções de ativação:

Camada de Encoding: $\lambda_E(x) = \text{round}(\text{clamp}(x + b_d, 0, 1))$

Onde o bias é: $b_d \in \{-\infty, \ldots, -2, -1\}^k$

O Bias e sua atualização garante que a ativação do neurônio será somente quando um número mínimo de features do padrão está ativo, assim o neurônio aprende padrões e não features.

Camada de Decoding: $\lambda_D(\bar{x}) = \text{round}(\text{clamp}(x, 0, 1))$,

Atualização dos pesos e bias

Função de Perda: $$L_\alpha(D[i];W,b) = \sum_{j\in[1,m]} \big((1-D[i,j])\alpha + D[i,j](1-\alpha)\big)|z_{ij} - D[i,j]|$$

É normalizada para que o cálculo funcione em dados esparsos ou densos $\alpha = \frac{\#1s}{\#1s+\#0s}$

Cálculo de Gradiente: Gated Straight-Through Estimator, baseado em Bengio et al. [3]

$$\frac{d\lambda_E}{db} = \begin{cases} g_o & \text{if } \lambda_E(x) = 1 \\ 0 & \text{if } \lambda_E(x) = 0 \end{cases}, \quad \frac{d\lambda_E}{dx} = \begin{cases} g_o & \text{if } \lambda_E(x) = 1 \\ \max(0, g_o) & \text{if } \lambda_E(x) = 0 \end{cases}.$$

Baseado na proposta de um dos referenciais teóricos para a determinação de gradientes quando a variada das funções não é definida. Com uma pequena alteração para não punir quando o neurônio não estava ativo e a saída foi muito diferente da entrada

Função de atualização dos Pesos: $$W^{t+1} = \text{clamp}(W^{t+1}, 1/m, 1)$$

O mínimo de 1/m garante que o neurônio não morra precocemente e que se consiga fugir de mínimos locais

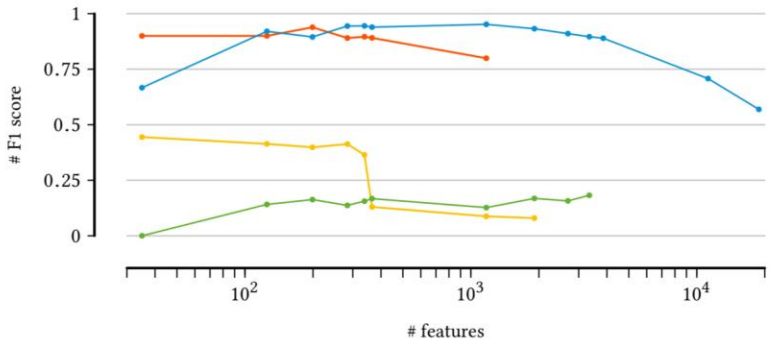
Metodologista

Grupo: DANIEL SCHLICKMANN BASTOS
GABRIEL CASTELO BRANCO ROCHA
GUILHERME BUXBAUM MARINHO GUERRA
JOSE EDUARDO DUARTE MASSUCATO
LEONARDO CAETANO GOMIDE
LUCAS MESQUITA ANDRADE
VINICIUS LEITE CENSI FARIA

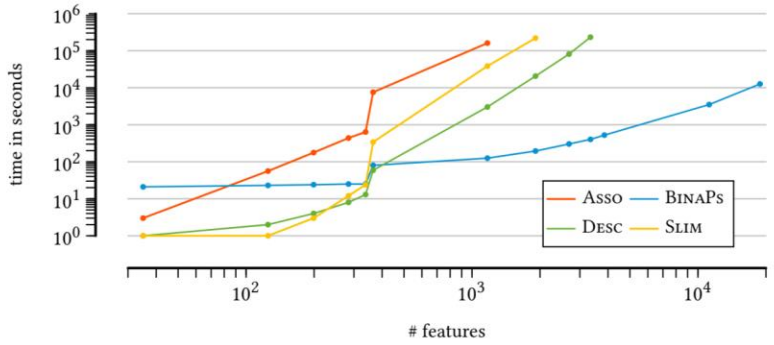
Experimentos

- O modelo proposto (BinaPs) foi comparado a três outros algoritmos:
  - Asso – Baseado em fatorização de matrizes
  - Slim – Baseado no princípio de *Minimum Description Length*
  - Desc – Baseado em *maximum entropy modeling*
- O modelo proposto e *Asso* utilizaram o parâmetro de máximo de padrões igual ao número de features
  - Ambos demonstram ser robustos a essa escolha
- Dois Tipos de Experimentos:
  - Sintéticos – Padrões conhecidos
  - Reais – Dados mais complexos

Dados Sintéticos



(a) F1 score on scale data (higher is better).



(b) Runtime on scale data (lower is better).

Figure 2: *BinaPs is scalable.* For synthetic data with known ground truth of varying number of features and planted patterns, we show the F1 score (a) and the runtime (b) for all methods. Experiments were aborted when exceeding 3 days of runtime.

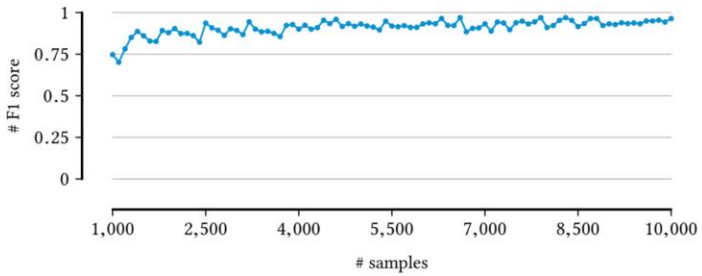
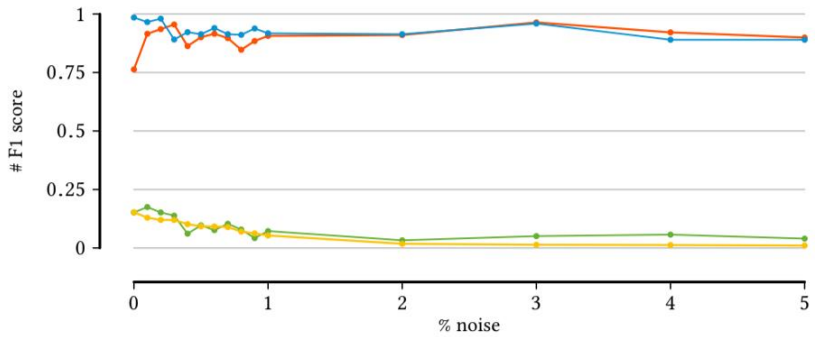
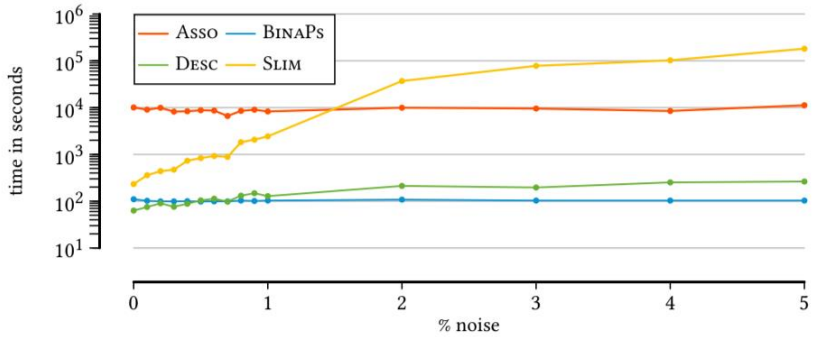


Figure 3: *BinaPs is applicable on data with few samples.* F1 score for pattern sets discovered by BinaPs on synthetic data with varying number of samples.



(a) F1 score on noise data (higher is better).



(b) Runtime on noise data (lower is better).

Figure 4: *BinaPs is robust to noise.* For synthetic data with known ground truth and varying the level of noise, we show the F1 score (a) and the runtime (b) for all methods. With 5% noise, we have a 1:1 ratio of signal-to-noise.

Dados Reais

- Cinco bases de dados reais foram utilizadas para analisar o método:
  - DNA – Dados de amplificação de DNA
  - Accidents – Dados de acidentes belgas
  - Instacart – Dados de compras de supermercados online
  - Korsarak – Dados de cliques de um site de noticias hungaro
  - Genomes – Dados de indivíduos do projeto 1000 Genomes

			# Patterns				Runtime			
Dataset	# rows	# cols	Asso	BinaPs	Desc	Slim	Asso	BinaPs	Desc	Slim
DNA	2458	391	134	131	345	281	4m	26s	20s	2s
Accidents	340183	468	133	78	215	12261	12h	6m	14m	21h
Instacart	2704831	1235	*n/a*	328	712	8119	∞	44m	25m	8h
Kosarak	990002	41270	*n/a*	302	*n/a*	*n/a*	∞	5h	∞	∞
Genomes	2504	226623	*n/a*	42	*n/a*	*n/a*	∞	9m	∞	∞

Table 1: *Results on Real World Data.* For five real world datasets, we give the number of rows, number of columns, and for Asso, BinaPs, Desc, and Slim we report the number of discovered patterns and runtime required to do so rounded to the most significant order of magnitude. Individual experiments were aborted after 3 days, resp. when exceeding the available 256GB of RAM, corresponding entries are marked with *n/a* and ∞.

Resultados Qualitativos

DNA

- Dados Densos
- *BinaPs* e *Asso* encontraram padrões de blocos que representam áreas onde os genes são amplificados
- *Slim* também aprende parte desses padrões, mas sofre overfitting para padrões muito grandes
- *Desc* encontra somente pequenos padrões

Resultados Qualitativos

Instacart

- BinaPs encontra padrões densos e ruidosos – ex.: Frutas compradas em combinações arbitrárias
  - Slim encontra os padrões mas quebra eles em padrões separados
  - Desc só consegue encontrar padrões de tamanho 2
- Outros padrões interessantes encontrados foram comidas preparadas; ou comidas de um estilo específico

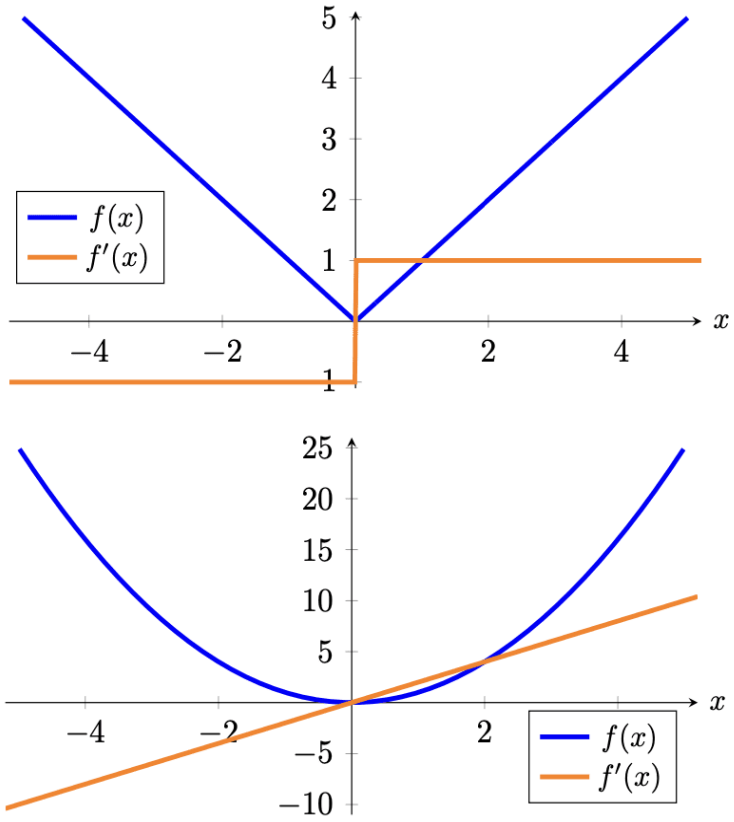
Resultados Qualitativos

Genomes

- São dados com muitas features (> 200.000)
- BinaPs foi capaz de encontrar variantes que são herdadas em conjunto
  - Dentro desses padrões algumas variantes também foram encontradas. Algumas que são conhecidas , e um outro padrão, que os autores disseram não terem capacidade de avaliar o significado
- O método foi capaz de identificar vários padrões que já foram identificados por especialistas, por exemplo:
  - Genes que são associados com diabetes na população japonesa
  - Genes associados com produção de proteínas juntos de genes não caracterizado

Análise Crítica

- Função de perda de reconstrução é MAE – Aderivada não é contínua
- Escolha de c – Experimentalmente, escolher um c grande teve bons resultados, mas experimentos com diferentes valores seria interessante
- Utilizar a estratégia de pesos transpostos pode ser utilizada para outros tipos de padrões – ex.: {Fralda e (cerveja ou vinho)}
- O uso de GPUs para treinar esse tipo de modelo possibilita escalar a abordagem



Assessor Social

Bernnardo Seraphim, Diná Xavier, Gabriel Fadoul,

Kênia Carolina, Oluwatoyin Joy e Samuel Kfuri

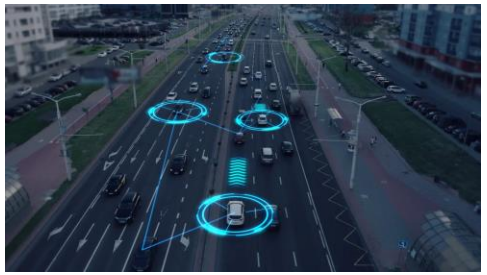
Impacto social

- **Promove a transparência da IA:** a mineração de conjuntos de padrões diferenciáveis gera padrões legíveis por humanos em vez de previsões de caixa-preta, o que pode ajudar médicos, analistas e formuladores de políticas a compreender e confiar nos resultados;

- **Apoia a justiça desde o design**: sua estrutura personalizável permite que os desenvolvedores excluam atributos tendenciosos (cor, gênero e raça), incentivando resultados equitativos em áreas sensíveis como saúde e recrutamento;

- **Risco - Falsa objetividade:** Mesmo padrões interpretáveis podem induzir a erros quando baseados em dados tendenciosos. Por exemplo, a ferramenta de triagem de currículos da Amazon, que aprendeu e reforçou o preconceito de gênero a partir de decisões históricas, desfavorecendo o gênero feminino;

Cenários



Fonte: PUC-RS

Fonte: Revista Crescer

Saúde
- Descoberta de biomarcadores relevantes para diagnósticos;
- Detecção de padrões comuns a doenças auxiliando no diagnóstico precoce;

Cidades Inteligentes
- Padrões de tráfego para otimização de transporte público e funcionamento do trânsito (temporização de semáforos, priorização de obras de vias, alteração de sentidos de fluxo);
- Padrões frequentes de uso de serviços públicos para melhorar atendimento

Educação
- Padrões de trajetória acadêmica definindo políticas de incentivo;
- Identificação de padrões de consumo de conteúdo para personalização de trilhas;

Comércio
- Padrões de compras de usuário para sistemas de recomendação;
- Perfis de consumidores para campanhas de marketing personalizadas;

Governos e Organizações
- Detecção de padrões de fraude em operações bancárias para proteção do usuário;
- Utilização de padrões comportamentais para prevenção de corrupção e crimes;

Problemas com privacidade, equidade, discriminação e igualdade

Exemplo: Utilização do algoritmo para concessão de crédito baseado em localização geográfica, renda ou perfil de consumo, como fatores determinantes;

Risco: Discriminação, Desinformação; Privacidade; Over Marketing;

Formas de mitigar: Garantir governança, auditoria e transparência, promovendo decisões algorítmicas justas, éticas e sem discriminação.

Exemplo: Precificação de seguro e planos de saúde baseados em prontuários médicos de pacientes para descoberta de padrões frequentes em determinados tipos de diagnóstico.

Risco: Diferenciação de grupos pela propensão de doenças de determinados grupos (etarismo, grupos de risco, etc.); Questões de privacidade relacionado à LGPR, caso a anonimização se perca por meio da identificação de determinado padrão (Um remédio específico pode ser vinculado à uma doença);

Formas de mitigar: Adoção de diretrizes e políticas aderentes à LGPD e demais legislações vigentes, incluindo anonimização e informando o usuário sobre quais dados estão sendo utilizados, para qual finalidade e por quanto tempo.

Outros possíveis danos

- O benefício da interpretabilidade do modelo também acarreta num conjunto de brechas possíveis: Através do método de Poisoning e a compreensão de quais atributos o algoritmo utiliza para tomada de decisões, é possível:

    - Fraude em participação de licitação de projetos públicos;
    - Fraude em seleção de candidatos em processos de admissão;

- Além disso temos também o aspecto ético:

Ao se utilizar de dados genéticos, por exemplo, podem ocorrer discriminação baseados em Genes, que não necessáriamente apresentem características esperadas, exemplo: Edição genética (sexo do bebê);

Recomendação de tratamentos específicos para predisposição de doenças em determinados grupos, que podem acarretar em outros problemas de saúde;

Hacker

Caio Jorge Carvalho Lara
Juan Marcos Braga Faria
Luisa Vasconcelos de Castro Toledo
Luiza Sodré Salgado
Mateus Reis Evangelista
Samuel Henrique Miranda Alves

Créditos: Bing AI

Instalação e Dependências

- Artigo possui link para o repositório: https://eda.rg.cispa.io/prj/binaps/

- Passo 1: Baixar e descompactar a pasta de arquivos

- Passo 2: Instalar as dependências manualmente (`Pytorch, Scipy, Pandas e Numpy e R` -> Não estava listado como necessário, mas é usado)

- Passo 3.0: Alterar genSynth.sh para rodar em uma máquina convencional (8 a 16GB de RAM) e salvar os arquivos em uma pasta (ele gera no diretório raiz)

Hacker

- Passo 3.1: Gerar os dados sintéticos: chmod +x genSynth.sh
- Passo 4: Executar ./genSynth.sh
- Passo 5: Copiar um arquivo .dat para o diretório BinapsCode
- Passo 6: Executar python main.py --input <arquivo.dat> --batch_size <32 ou 64>

Parâmetro	Tipo	Valor padrão	Descrição
--input (-i)	str	Obrigatório	Caminho para o arquivo de entrada (dados usados para treino e teste)
--train_set_size	float	0.9	Proporção dos dados usada para treinamento
--batch_size	int	64	Tamanho do batch para treinamento
--test_batch_size	int	64	Tamanho do batch para teste
--epochs	int	10	Número de épocas de treinamento
--lr	float	0.01	Taxa de aprendizado (learning rate)
--weight_decay	float	0	Fator de penalização L2 (regularização dos pesos)
--gamma	float	0.1	Fator de decaimento do learning rate (usado no agendador de LR)
--seed	int	1	Semente para reprodutibilidade (random seed)
--log_interval	int	10	Intervalo (em batches) para exibir logs de treino
--save_model	bool	False	Se ativado, salva o modelo treinado para disco
--hidden_dim	int	-1	Número de neurônios ocultos (usa #features se -1)
--thread_num	int	16	Número de threads a serem usadas no treinamento

Geração dos Dados de Entrada

O código assume um arquivo de transações como entrada no formato de uma matriz binária esparsa. Assim, cada entrada não-zero da matriz corresponde ao índice da linha com valor 1.

Por exemplo, a matriz:

1	0	0	0
0	1	1	0
1	0	0	1

É representada da seguinte forma no arquivo .dat:

1	
2	3
1	4

Obs: Arquivos gerados possuem 100.000 linhas

Execução

```
Train Epoch: 10 [88960/89739 (99%)]    Loss: 1.261767
Train Epoch: 10 [89600/89739 (100%)]   Loss: 1.493594

Test set: Average loss: 211.604980, Accuracy: 30/9971 (0%)


Patterns:

[45 46 47 48 49] ( 4404 / 4425 )
[578 579 580] ( 4833 / 4847 )
[534 535 536 537 538 539 540 541 542 543] ( 4910 / 4956 )
[218 219 220 221 222 223 224 225 226 227] ( 4462 / 4511 )
```

- `[45 46 47 48 49]`: Padrão reconstruído.
- `(4404 / 4425)`:
- 4404: Amostras que ativaram **todos** esses bits padrão completo encontrado.
- 4425: Amostras que ativaram pelo **menos metade** dos bits do padrão.
- O resultado `[45 46 47 48 49] ( 4404 / 4425 )` indica que o padrão completo de ativação dos bits 45 a 49 apareceu exatamente em 4404 amostras (suporte completo ou **`supp_full`**) e em pelo menos metade dos bits em 4425 amostras (suporte parcial ou **`supp_half`**), o que mostra uma correspondência extremamente alta entre o padrão e os dados.

Conclusão: O padrão `[45–49]` é altamente confiável, pois aparece completo em mais de 99% das vezes em que é parcialmente ativado, indicando forte consistência local nos dados.

Referências

Referências bibliográficas

- Y. Bengio, N. Léonard, and A. C. Courville. 2013. Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation. CoRR abs/1308.3432 (2013). arXiv:1308.3432 http://arxiv.org/abs/1308.3432