Aprendizagem de Máquina Algébrica

Caio A. de C. Costa[1]

[1]Instituto Tecnológico de Aeronáutica (ITA) – São José dos Campos – SP – Brasil

caio.costa@ita.br

Abstract. *This article discusses a Machine Learning paradigm published in 2018. An algebraic formalism that facilitates demonstrations of convergence theorems is introduced. In the attached file, there is an implementation in Python of an engine containing the six routines used by the method.*

Resumo. *Este artigo discorre sobre um paradigma de Aprendizagem de Máquina publicado em 2018. É introduzido um formalismo algébrico que facilita demonstrações de teoremas de convergência. Em arquivo anexo, há uma implementação em* Python *de uma* engine *contendo as seis rotinas utilizadas pelo método.*

1. Introdução

A Aprendizagem de Máquina Algébrica (AMA) é um paradigma de Aprendizagem de Máquina (AM) bastante recente e, portanto, pouco explorado. Trata-se de um método matematicamente formal de classificação supervisionada. O seu objetivo final é produzir um classificador a partir dos exemplos fornecidos. Para tanto, é preciso separar os exemplos (que mais tarde chamaremos de termos) em suas unidades fundamentais (constantes) e criar unidades de aprendizagem (átomos), executando repetidamente e de maneira sistemática um ciclo de seis algoritmos, mais um algoritmo opcional.

Em relação aos métodos tradicionais de AM, a AMA apresenta pelo menos três vantagens, a saber:

(i) **A AMA é independente de parâmetros**
De acordo com [2], parâmetros estão presentes em vários métodos de AM, como Regressão Logística, Análise Discriminante Linear, *Perceptron*, *Naive Bayes*, e Redes Neurais Simples. Enquanto métodos paramétricos são mais simples, mais rápidos e utilizam menos dados, eles são mais limitados e restritos que métodos não-paramétricos, como k-Vizinhos mais próximos, Árvores de Decisão, e *Support Vector Machines*. Assim sendo, por ser independente de parâmetros, o método da AMA é poderoso, flexível e de alta performance.

(ii) **A AMA é totalmente discreta**
Dessa forma, não existem erros por operações aritméticas de ponto flutuante.

(iii) **A AMA não utiliza minimização de função de custo.**
Portanto, não é preciso contornar o problema recorrente em AM de convergir para extremos locais.

Como aplicação da AMA, os autores de [1] sugerem a classificação de dígitos manuscritos e a solução do problema das N-rainhas. Nesta última, o método se destaca por fornecer uma solução *human-friendly*, em contraste com Redes Neurais, por exemplo, que codifica o conhecimento por uma matriz de números reais que funciona como uma "caixa

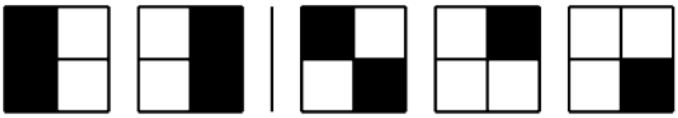


Figura 1. Exemplos positivos (à esquerda) e negativos (à direita).

preta". Embora haja considerável interesse por ver o método funcionando na prática, este projeto, de apenas três meses de duração, focou na implementação de uma *engine* de AMA, não havendo tempo hábil para reproduzir os resultados de [1] nas referidas aplicações.

2. Materiais e Métodos

2.1. Materiais

Neste projeto foram utilizadas três ferramentas de desenvolvimento e três ferramentas de base.

No desenvolvimento, foram empregados o *Python*, o *Pycharm* e o *git*. O *Python*, por sua versatilidade e tradição na área de Inteligência Artificial, foi a linguagem de programação escolhida. O ambiente de desenvolvimento integrado *PyCharm*, da *Jetbrains*, foi utilizado para programar em *Python*. O sistema de versionamento *git* foi utilizado para e gerenciar e organizar as edições no código.

Como ferramentas de base, valeu-se de [1], sobre o qual todo o projeto foi construído; do LaTeX, para escrever este artigo; e do *software OBS*, para gravar o vídeo que o acompanha.

2.2. Métodos

Foi implementada uma *engine* capaz de executar as seis rotinas da AMA citadas na Revisão Bibliográfica. Para atingir tal objetivo, foi necessário interpretar corretamente o texto de [1] e transformar o pseudocódigo de seus algoritmos em código de fato. Foi necessário entender a fundo cada algoritmo para depurar os *bugs* que apareceram na fase de desenvolvimento.

3. Revisão Bibliográfica

Por se tratar de um método recente, a única fonte de informação disponível é um *paper* de 2018 por Fernando Martin-Maroto e Gonzalo G. de Polavieja [1]. Os autores foram contatados pelo *Linkedin* e, segundo Martin-Maroto, ainda não há trabalhos públicos que tenham utilizado Aprendizagem de Máquina Algébrica. Assim sendo, toda a discussão que se segue é baseada nesse paper.

O objetivo do AMA é construir um classificador binário a partir de exemplos positivos e negativos da classe que se quer distinguir. Tomou-se como problema amostra o mesmo que [1], classificar malhas binárias (preto e branco) 2×2 que contenham uma barra vertical de pixeis pretos. Foram fornecidos como entrada os 5 exemplos da Figura 1.

O primeiro passo é identificar os componentes fundamentais desses exemplos, os quais chamaremos de constantes. No problema amostra, utilizamos como constantes

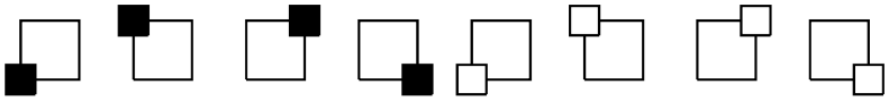


Figura 2. As constantes são as componentes fundamentais dos exemplos.

os pixeis brancos e pretos em cada uma das 4 posições possíveis, totalizando as oito constantes da Figura 2.

3.1. A Álgebra Principal

A partir dos exemplos e das constantes, é construída uma álgebra para representar o nosso aprendizado. Na prática, a álgebra é apenas um formalismo matemático que facilita demonstrações de teoremas de convergência no apêndice de [1] (que estão fora do escopo deste artigo), e a sua representação na máquina se dá por um grafo.

A álgebra, que chamaremos de M, é composta de três categorias de elementos: termos, constantes e átomos. Tal fato se representa pela união disjunta

$$M = \mathbf{T}(M) \dot{\cup} \mathbf{C}(M) \dot{\cup} \mathbf{A}(M). \tag{1}$$

Os termos $\mathbf{T}(M)$ são os exemplos anteriormente introduzidos, mais os termos de fixação criados pelo algortimo 6, de treinamento em lote. As constantes $\mathbf{C}(M)$ são as oito da figura 2, mais uma constante adicional v que representa a classe que queremos distinguir. No problema amostra, v é uma abstração da barra vertical de pixeis pretos. Os átomos $\mathbf{A}(M)$ são unidades de aprendizado que o método cria ao longo de sua execução, e são mais fundamentais que as constantes: da mesma forma que os termos são formados "unindo-se" constantes, as constantes são formadas "unindo-se" os átomos.

Essa "união" é uma das operações de M, a operação combinar, representada por um círculo com um ponto ($\odot$). Combinar é uma operação binária, idempotente, comutativa e associativa. Sejam m, m', m'' elementos quaisquer de M. A operação combinar é binária, pois leva dois elementos de M em um elemento de M; idempotente, pois $m \odot m = m$; comutativa, pois $m \odot m' = m' \odot m$; e associativa, pois $m \odot (m' \odot m'') = (m \odot m') \odot m''$. M também é equipada com uma relação de ordem $<$, definida em função de seu grafo $G(M)$.

$G(M)$ é um grafo acíclico direcionado cujos nós são os elementos de M. Inicialmente, há arestas $c \to T$ ligando cada termo T às constantes que o compõe. Também, introduzimos o primeiro átomo de M, o átomo 0, e o ligamos a todas as constantes (inclusive v). Finalmente, completamos o grafo introduzindo arestas por fechamento de transitividade (arestas azuis). Na figura 3, que mostra o estado inicial de $G(M)$, as arestas são direcionadas de baixo para cima, e não representamos suas pontas para evitar poluição visual. A figura 5a mostra a imagem real de saída do código, feita com os pacotes de *Python* `networkx` e `matplotlib.pyplot`, para este mesmo estado inicial do grafo.

3.2. A Álgebra Dual

A fim de facilitar as manipulações algébricas a serem realizadas nas rotinas do método, é introduzida uma álgebra dual $M^\star$. Esta nova álgebra é construída a partir de M aplicando

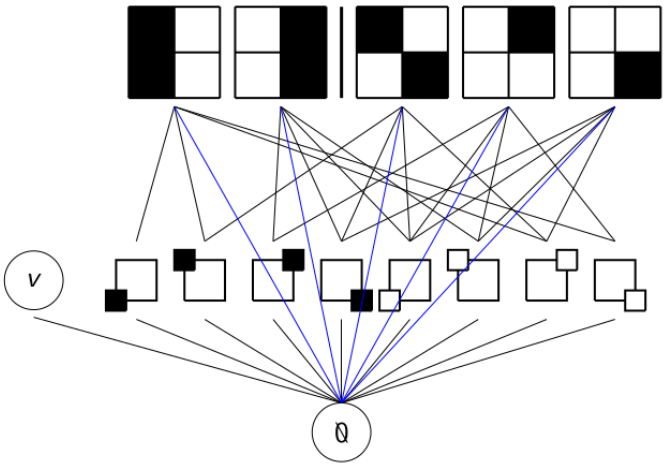


Figura 3. Estado inicial do grafo da álgebra principal.

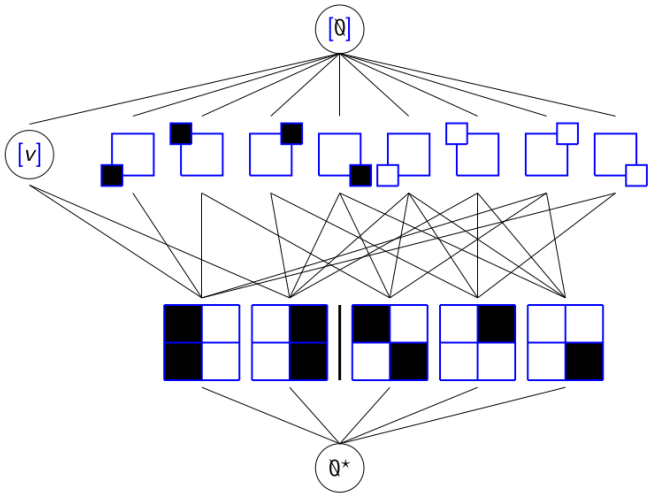


Figura 4. Estado inicial do grafo da álgebra dual. A operação dual está aqui representada ora pelos colchetes, ora pela cor azul.

a operação dual ($[\,\cdot\,]$) a todos os seus elementos e invertendo a direção de todas as arestas de seu grafo. Adicionalmente, um novo elemento $0^\star$ é adicionado aos átomos de $M^\star$.

A operação dual $[\,\cdot\,]\colon M \to M^\star$ é unária. Os duais de constantes e termos de M são constantes de $M^\star$, e os duais de átomos de M são um novo tipo de elemento em $M^\star$, chamados de duais-de-átomos. $M^\star$ não tem termos e, os seus átomos, dos quais $0^\star$ é o primeiro, não são duais de nenhum elemento de M. Completa-se o grafo, novamente, por fechamento de transitividade, o qual não representamos, novamente para evitar poluição visual. Não há ambiguidade, pois o grafo $G(M^\star)$ de $M^\star$ é direcionado e acíclico ($G(M)$ e $G(M^\star)$ são *DAG*s). O grafo está representado na figura 4, com a mesma convenção da figura 3. A figura 5b contém a saída real do código para o estado inicial da álgebra dual.

3.3. Notações

Nesta subseção são introduzidas algumas notações e definições selecionadas do Capítulo 2 e do Apêndice A de [1], úteis nos algoritmos que se seguem.

Estabelecemos em $G(M)$ a notação $\mathbf{GL}_M(x) = \{y \in M \mid y \to x \ \vee y = x\}$

para significar todos os elementos da respectiva álgebra que estão "abaixo"de x, isto é, que estão ligados por uma aresta ao elemento x, além do próprio elemento x, e $\mathbf{GU}_M(x) = \{y \in M \mid x \rightarrow y\}$ para os elementos que estão "acima"de x no grafo. Analogamente, definimos $\mathbf{GL}_{M^\star}(x)$ e $\mathbf{GU}_{M^\star}(x)$, trocando M por $M^\star$. Quando não houver ambiguidade, omite-se o subscrito, escrevendo-se apenas $\mathbf{GL}(x)$ e $\mathbf{GU}(x)$. Essas notações vêm do inglês *graph lower* e *graph upper*, respectivamente. Escrevemos também $\mathbf{GL}^{\mathbf{a}}(x)$ e $\mathbf{GL}^{\mathbf{c}}(x)$ para denotar a interseção de $\mathbf{GL}(x)$ com $\mathbf{C}(M)$ ou $\mathbf{C}(M^\star)$, de acordo com o contexto.

Define-se em cada grafo uma relação de ordem ($<$) via função $\mathbf{GL}^{\mathbf{a}}(x)$. Escrevemos que $m < m'$ quando $\mathbf{GL}^{\mathbf{a}}(m') \subset \mathbf{GL}^{\mathbf{a}}(m')$. Observe que, de acordo com essa definição, $m < m$ para todo m em M ou $M^\star$.

3.4. O Objetivo do AMA: Modelos Atomizados

Nota-se que a relação de ordem é uma função do conjunto de átomos $\mathbf{A}$, que evolui com o algoritmo. Logo, as relações de ordem entre os elementos também evoluem. O objetivo da Aprendizagem de Máquina Algébrica é evoluir $\mathbf{A}$ de modo que certas relações de ordem envolvendo o elemento-objetivo v sejam satisfeitas, a saber:

$$v < T_i^+ \tag{2}$$

$$v \not< T_i^- \tag{3}$$

em que T_i^+ percorre todos os termos positivos e T_i^- percorre todos os termos negativos.

3.5. O Traço e as Restrições de Traço

Define-se o traço de um elemento x por

$$\mathbf{Tr}(x) = \cap_{\phi \in \mathbf{GL}^{\mathbf{a}}(x)} \mathbf{GL}^{\mathbf{a}}([\phi]). \tag{4}$$

O traço é útil para atingir nosso objetivo pois apresenta duas propriedades interessantes. Primeiramente, o traço linear em relação à operação combinar, isto é,

$$\mathbf{Tr}(m \odot m') = \mathbf{Tr}(m) \cap \mathbf{Tr}(m'). \tag{5}$$

Ademais, uma desigualdade em M implica em uma continência invertida de traços:

$$m < m' \implies \mathbf{Tr}(m') \subset \mathbf{Tr}(m). \tag{6}$$

A partir das equações 2, 3 e 6 são deduzidas as restrições de traço das equações 7 e 8, que devem ser satisfeitas caso o objetivo seja atingido.

$$\mathbf{Tr}(T_i^+) \subset \mathbf{Tr}(v) \tag{7}$$

$$\mathbf{Tr}(T_i^-) \not\subset \mathbf{Tr}(v) \tag{8}$$

Os algoritmos 1 e 2 servem para tornar verdadeiras as equações 7 e 8, respectivamente.

3.6. Cruzamento Completo e Esparso

Depois de forçar as restrições de traço, todas as relações-objetivo negativas da equação 3 já estão satisfeitas. Com a intenção de satisfazer também as relações positivas da equação 2, aplica-se o algoritmo 3, de cruzamento esparso. O cruzamento esparso é uma versão alternativa ao cruzamento completo, que multiplicaria o número de átomos.

A operação de cruzamento pode ser representada por uma matriz. Suponha que se quer forçar a relação $a < b$, e que as composições atômicas desses elementos sejam $\mathbf{GL^a}(a) = \{\alpha_1, \alpha_2, \xi\}$ e $\mathbf{GL^a}(b) = \{\beta_1, \beta_2, \xi\}$. Então a matriz que representa o cruzamento completo de a em b é dada pela tabela 1. O algoritmo consiste em criar os átomos de dentro da matriz, ligar arestas saindo de cada entrada da matriz para a respectiva linha e coluna – por exemplo, criamos o átomo γ_{13} e o ligamos a β_2, e o átomo γ_{32} e o ligamos a α_2 e a β_1 –, completar por fechamento de transitividade e apagar os átomos antigos de a e b.

	ξ	β_1	β_2
	γ_{11}	γ_{12}	γ_{13}
α_1	γ_{21}	γ_{22}	γ_{23}
α_2	γ_{31}	γ_{32}	γ_{33}

Tabela 1. Matriz de cruzamento completo.

O cruzamento esparso é idêntico ao cruzamento completo, exceto que há elementos faltando na matriz da tabela 1, como na tabela 2.

	ξ	β_1	β_2
	γ_{11}		γ_{13}
α_1		γ_{22}	
α_2		γ_{32}	γ_{33}

Tabela 2. Matriz de cruzamento esparso.

Ambos os algoritmos inserem átomos extras a cada átomo criado, a fim de preservar os traços de todos os elementos e, consequentemente, as restrições de traço já estabelecidas com a execução dos algoritmos 1 e 2.

Ao término da execução desse algoritmo, não apenas as restrições de traço 7 e 8 são obedecidas, mas também todas as relações-objetivo das equações 2 e 3 estão satisfeitas. Portanto, o objetivo do aprendizado foi atingido. Mas é possível melhorar! Há mais átomos que o necessário, o que nos leva ao próximo passo.

3.7. Redução do Modelo

Os algoritmos 4 e 5 têm como objetivo diminuir a cardinalidade do conjunto de átomos da álgebra principal e da álgebra dual, respectivamente. Tais algoritmos não mais operam mantendo as restrições de traço de todos os elementos, mas apenas das constantes. Pela propriedade de linearidade do traço (equação 5), isso é suficiente, visto que todos os termos envolvidos nas restrições e relações são composição de constantes.

Para tal fim, essas rotinas testam, em ordem aleatória, para cada constante do modelo, quais átomos podem ser eliminados sem alterar o traço dessa constante e das

constantes anteriores. Após a última constante ser testada, os átomos selecionados são excluídos.

3.8. Geração de Termos de Fixação

Os algoritmos até aqui mostram como obter um modelo atomizado a partir de um conjunto R de relações positivas e negativas. Em vez de um único conjunto R, é mais eficiente utilizar vários lotes de treinamento $R_0, R_1, R_2, \ldots, R_n$. Para manter os grafos $G(M)$ e $G(M^\star)$ pequenos o suficiente para rápida execução das rotinas, é desejável apagar os nós correspondentes ao conjunto R_0 antes de passar adiante para R_1. Contudo, ao fazê-lo, se perderia informação, pois algumas relações que valem em R_0 deixariam de valer.

O algoritmo 6 contorna esse problema criando termos e relações de fixação, a serem adicionados às álgebras entre a deleção de R_k e a execução de R_{k+1}. Para cada átomo $\phi \in \mathbf{A}(M)$, criamos o termo de fixação

$$T_\phi = \odot_{\phi \not< c}\, c. \tag{9}$$

As relações de fixação R_p são todas negativas e formadas pela interação das outras constantes com o termo T_ϕ.

$$R_p = \{(c \not< T_\phi) \mid \phi < c\}. \tag{10}$$

4. Experimentos e Resultados

4.1. Execução dos Algoritmos

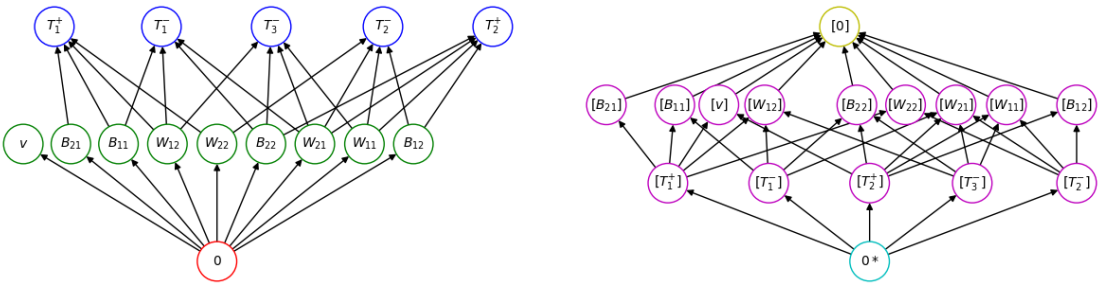
Nesta seção, mostra-se na prática a aplicação dos seis algoritmos apresentados na Revisão Bibliográfica. Nas figuras a seguir, diferentes categorias de elementos das álgebras estão codificados por cor, de acordo com a Tabela 3.

M	$M^\star$
átomos	átomos
constantes	constantes
termos	duais-de-átomos

Tabela 3. Legenda de cores para os elementos da àlgebra principal (à esquerda) e da álgebra dual (à direita).

Durante todo o processo, os grafos $G(M)$ e $G(M^\star)$ são transitivamente fechados, mas a representação nas figuras 5-12 é simplificada para mostrar apenas as arestas do grafo reduzido, como na figura 4.

A figura 5 mostra o estado inicial de construção de M e $M^\star$. No lugar da representação visual dos termos e constantes das figuras 1 e 2, usamos os simbolos T_i^+ e T_j^- para os termos positivos e negativos, respectivamente, com $i \in \{1, 2\}$ e $j \in \{1, 2, 3\}$. Para as constantes, utilizamos a letra B ou a letra W conforme a cor fosse preta (*black*) ou branca (*white*), seguida de um índice para indicar a linha e outro para indicar a coluna. Para a visualização do gráfico foi empregado o algoritmo de posição do tipo `dot` do pacote `graphviz_layout`, de modo que os nós não estão organizados da mesma maneira que na figura 3.



(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 5. Estado inicial dos grafos.

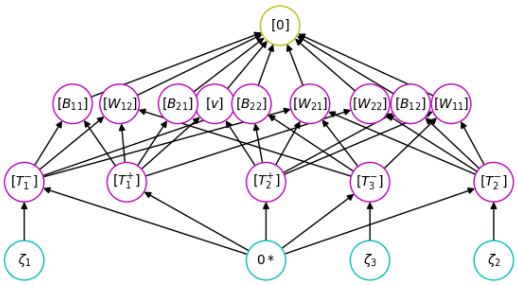
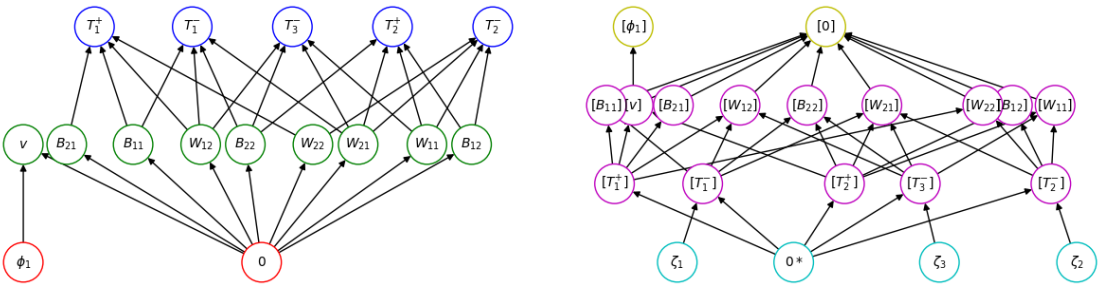


Figura 6. Alg. 0: Satisfazer as relações negativas invertidas.

Para cada lote de treinamento, todos os algoritmos 1-6 devem ser executados em ordem. O algoritmo 0 é opcional e tem como propósito estabelecer as relações negativas invertidas em $M^\star$, encurtando o tempo de execução dos algoritmos 1 e 2 que seguem. Nesse algoritmo, criamos átomos ζ_i em M^* para cada dual de termo negativo T_i^-, lembrando de executar o fechamento de transitividade, chegando ao estado da figura 6.

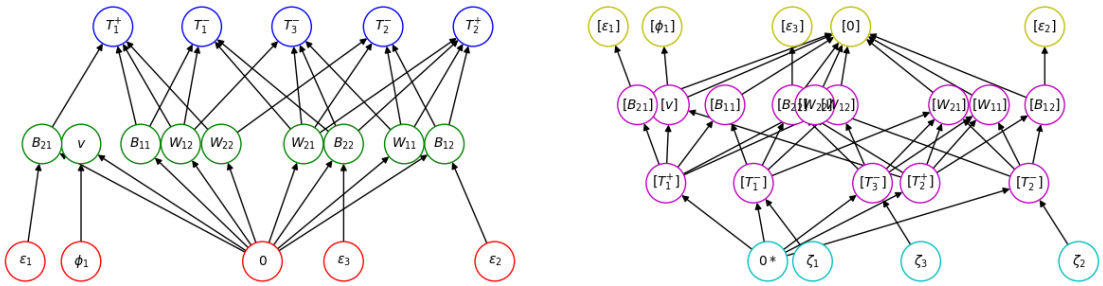
Em seguida, aplicamos o algoritmo 1, com o objetivo de forçar as relações da equação 8. Esse algoritmo promove a criação de um átomo ϕ_1 e de uma aresta $\phi_1 \to v$, bem como o dual-de-átomo e a aresta correspondente em $M^\star$, além das outras criadas por fechamento de transitividade em ambos os grafos, levando ao estado da figura 7.

O algoritmo 2 força as relações da equação 7, por meio da criação dos átomos ε_1, ε_2 e ε_3, além das arestas $\varepsilon_1 \to B_{21}$, $\varepsilon_2 \to B_{12}$ e $\varepsilon_3 \to B_{22}$. Criam-se também os



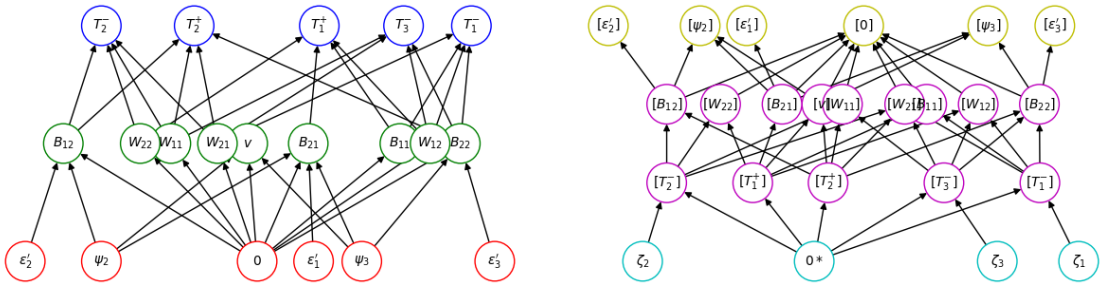
(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 7. Alg. 1: Forçar restrições de traço negativas.



(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 8. Alg. 2: Forçar restrições de traço positivas



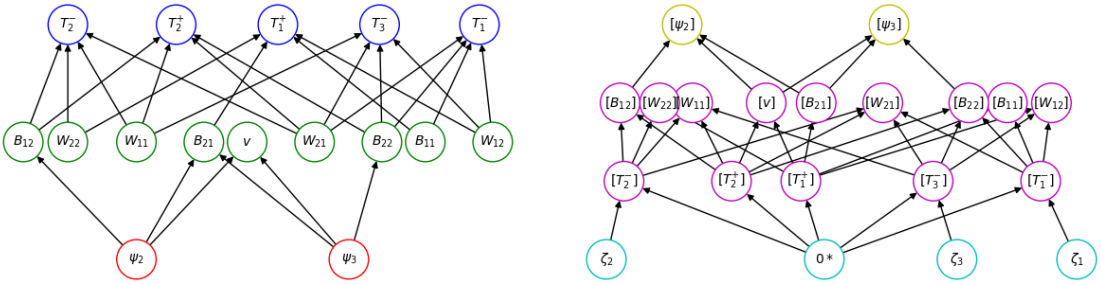
(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 9. Alg. 3: Cruzamento esparso.

elementos e arestas correspondentes em $M^\star$ e realiza-se o fecho de transitividade para chegar aos grafos da figura 8.

Após aplicarmos os algoritmos 1 e 2 de forçação das restrições de traço, temos que as relações-objetivo negativas da equação 3 já são satisfeitas, por contrapositiva da propriedade da equação 6. O terceiro algoritmo, de cruzamento esparso, melhora as restrições positivas de traço da equação 7 para obter as relações-objetivo positivas da equação 2. O cruzamento esparso é uma versão do cruzamento completo que envolve menos átomos, e portanto é mais eficiente. O algoritmo 3 é aplicado a cada par (v, T_i^+) oriundo da relação positiva que se deseja obter $v < T_i^+$. Ao aplicá-lo ao par (v, T_1^+), trocam-se os átomos ϕ_1 e ε_1 por novos átomos ψ_1 e ε'_1. Aplicando novamente, dessa vez ao par (v, T_2^+), trocam-se ψ_1, ε_2 e ε_3 pelos novos ψ_2, ψ_3, ε'_2 e ε'_3. Ao final dessa etapa, todas as relações-objetivo são satisfeitas, porém com 6 átomos na álgebra principal e 4 na álgebra dual. O estado atual após o fechamento de transitividade é o da figura 9.

A próxima etapa consiste em diminuir o tamanho do modelo com o algoritmo 4. Enquanto o algoritmo anterior, de cruzamento, preserva todos os traços de todos os elementos e, consequentemente, as restrições de traço impostas pelos algoritmos 1 e 2, o algoritmo de redução do modelo apenas preserva o traço das constantes. É feita uma checagem em ordem aleatória, para cada constante $c \in \mathbf{C}(M)$, quais átomos de $\mathbf{A}(M)$ podem ser removidos sem alterar $\mathbf{Tr}(c)$. Como resultado, pela linearidade do traço (equação 5), os traços dos termos envolvidos nas restrições de traço permanecem constantes, visto que todo termo é escrito como combinação de constantes, $T_i = \odot_j c_j$. Após a remoção dos átomos irrelevantes para a manutenção dos traços das constantes, chegamos ao estado da



(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 10. Alg. 4: Redução do modelo da álgebra principal.

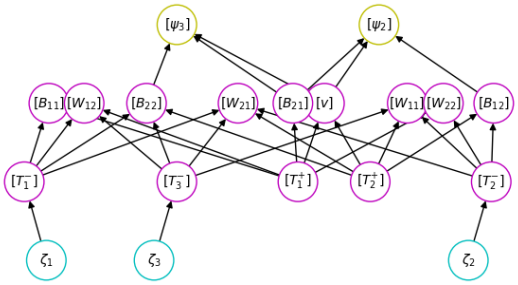
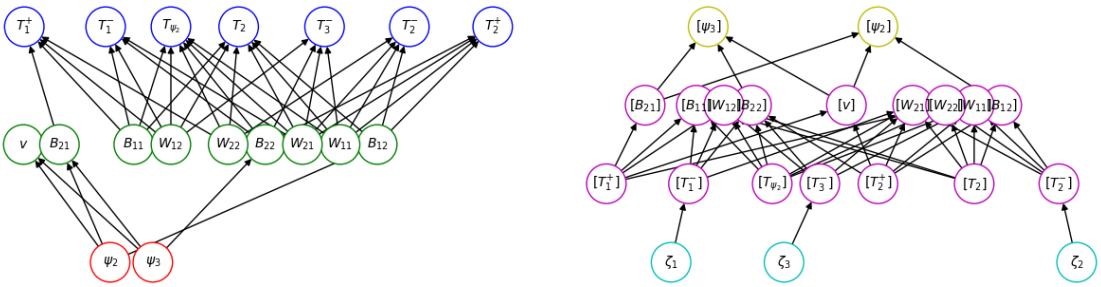


Figura 11. Alg. 5: Redução do modelo da álgebra dual.

figura 10, em que restam na álgebra principal apenas os átomos ψ_2 e ψ_3.

O algoritmo 5 é parecido com o algoritmo 4, exceto que remove átomos de M^* dessa vez, levando M^* ao estado da figura 11. Apenas o átomo $0^\star$ foi removido, o que demonstra como a aplicação prévia do algoritmo 0 torna o processo de criação de átomos nos algoritmos 1 e 2 mais eficiente.

Por último, a execução do algoritmo 6 fornece os termos e relações de fixação que compactam a informação aprendida nesse lote, preparando as álgebras para o aprendizado das informações contidas nos lotes posteriores. Após o fechamento por transitividade, M e M^* se encontram no estado da figura 12.



(a) Álgebra Principal M **(b)** Álgebra Dual $M^\star$

Figura 12. Alg. 6: Geração de termos de fixação.

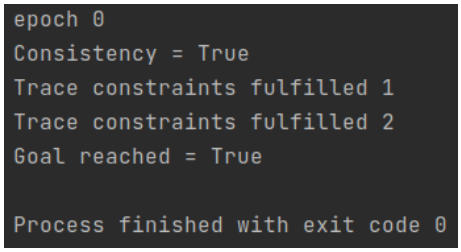


Figura 13. Mensagem de êxito na execução da AMA.

4.2. Resultados

Foi possível realizar o objetivo de satisfazer as relações positivas e negativas satisfatoriamente, obtendo ainda um modelo com poucos átomos (apenas dois na álgebra principal, e três na álgebra dual). A figura 13 mostra a mensagem de êxito imprimida na tela ao se executar o arquivo `main.py` na máquina do autor deste artigo.

5. Discussão e Conclusões

O presente projeto está inserido na disciplina PO-240, Introdução à Inteligência Artificial, do Programa de Pós-Graduação em Pesquisa Operacional (PPG-PO) ITA-UNIFESP. Nesse contexto, a ideia de proporcionar aos estudantes a escolha do próprio projeto era condicionada à aplicação de um método de Inteligência Artificial (IA) a algum projeto de caráter prático. Tal projeto poderia ser selecionado de alguma competição de IA ou AM, ou de sites como o Kaggle, mas não de forma obrigatória.

A escolha deste projeto para esta disciplina foi ousada, dadas a inovação e dificuldade do tema. Também obstou o semestre atípico do ITA, que foi quatro semanas mais curto devido à pandemia da COVID-19. Ademais, o projeto, que foi desenhado para realização em equipe de até três estudantes, foi realizado por apenas um, pois os outros desistiram quando perceberam que haveria muito trabalho.

Em meio a todos esses obstáculos, considera-se que foi concluída com maestria a principal componente do projeto original, que é a *engine* que acompanha este artigo. O tempo nela investido não foi de modo algum desperdiçado. Não obstante, é possível dar prosseguimento à execução do projeto na disciplina eletiva PO-233, Aprendizagem de Máquina, também ministrada pela professora Ana Carolina Lorena.

6. Referências

[1] MARTIN-MAROTO, F.; DE POLAVIEJA, G. G. (2018). **Algebraic Machine Learning.** Disponível em: <arXiv:1803.05252v2 [cs.LG]>. Acesso em: 11 out. 2020.

[2] BROWNLEE, J., 2020. **Parametric And Nonparametric Machine Learning Algorithms.** Disponível em: Machine Learning Mastery. Acesso em: 11 dez. 2020.