Solução do problema do trem de Michalski utilizando *Logic Tensor Network*

Andrew M. Santos, Daniel M. S. Saldanha, Letícia B. Silva
Emails:
{andrewsantos,dmss,leticia.balbi}@icomp.ufam.edu.br

I. Introdução

O relatório descreve uma solução para o problema exposto na especificação do trabalho de Inteligência Artificial, abordando o conteúdo dos itens descritos no documento disponibilizado pelo professor. O relatório em questão conta com uma descrição do problema; um referencial teórico; A solução do problema utilizando a abordagem especificada no documento do trabalho; A análise dos resultados obtidos na solução alcançada, e por fim, a conclusão.

O objetivo do trabalho é solucionar o problema para a base de trens de **Michalski**, utilizando um framework que realiza uma abordagem *Neuro-Simbólica de Aprendizado*, a ***Logic Tensor Networks (LTN)***. Para alcançarmos este objetivo, foi necessária a consulta do livro ***Neural-Symbolic Cognitive Reasoning*** [1]; a consulta aos artigos sobre LTN [2], [3] e o repositório do *framework* [4].

II. Referencial Teórico

A. Redes Neurais Artificiais

Redes Neurais Artificiais (RNA) podem ser definidas como uma estrutura complexa interligada por neurônios os quais possuem a habilidade de realizar operações para processamento de dados e representação de conhecimento. Uma RNA é um grafo direcionado, os vértices desse grafo são os neurônios. Cada neurônio possui as seguintes características: Um tempo, um vetor de entrada, um potencial de entrada, um estado de ativação e uma saída. Os neurônios são interconectados, e essas conexões são associadas a um determinado peso.

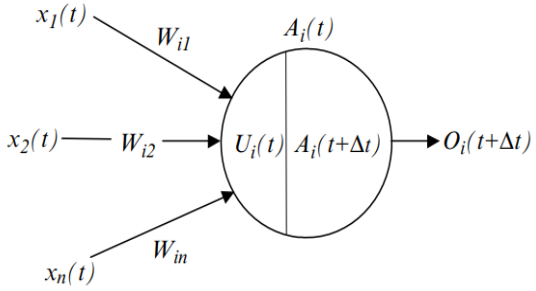

Fig. 1. O processo de um Neurônio

A imagem acima demonstra o processo de um neurônio em uma RNA. Eles recebem uma entrada [x1,x2,...,xn] em um tempo *t*; A conexão de cada uma dessas entradas com o neurônio tem um peso *W*; No neurônio existe um potencial de entrada *Ui(t)* e um estado de ativação *Ai(t + Δt)* que gera uma saída *Oi(t + Δt)*.

As redes neurais possuem diferentes tipos de aprendizado:

1) **Aprendizado Supervisionado**: Quando é utilizado um agente externo que indica à rede a resposta desejada para o padrão de entrada;
2) **Aprendizado Não Supervisionado**: Quando não existe uma agente externo indicando a resposta desejada para os padrões de entrada;
3) **Reforço**: Quando um crítico externo avalia a resposta fornecida pela rede.

O aprendizado em uma RNA ocorre a partir de um processo iterativo de ajustes aplicado aos pesos das conexões entre os neurônios, chegando a uma resposta generalizada.

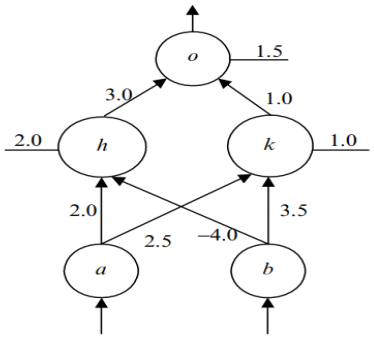

Fig. 2. Exemplo de uma rede neural com suas respectivas conexões e pesos associados

B. Sistemas Neuro-simbólicos de Aprendizado

Um **Sistema neuro-simbólico** realiza a integração de redes lógicas e neurais, oferecendo uma caracterização lógica de um sistema conexionista, um conexionismo a implementação de uma lógica, ou um sistema híbrido de aprendizado que junta os recursos do conexionismo e da inteligência artificial simbólica. A integração dessas duas áreas começou a receber uma atenção considerável no final dos anos noventa. Os pesquisadores Towell e Shavlik apresentaram o modelo **KBANN** (Knowledge-Based Artificial Neural Network) [5], um sistema com regras de inserção refinamento e extração de redes neurais. Os pesquisadores mostraram que as redes neurais baseadas no conhecimento (Knowledge-Based), treinadas usando o algoritmo de aprendizagem *background knowledge*, forneceram uma maneira muito eficiente de aprender a partir de exemplos e conhecimentos básicos. O algoritmo da rede de KBANN constrói arvores de operações lógicas AND/OR, definindo os limites e os pesos da rede. A regra define uma hierarquia onde as disjunções com mais de um termo são eliminadas.

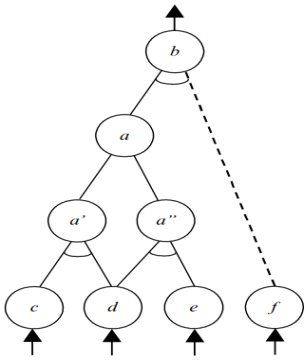

Fig. 3. Exemplo de KBANN

O modelo KBANN serviu de inspiração para a construção do **CILP** (*Connectionist Inductive Learning and Logic Programming*) system [6]. O CILP fornece uma base teórica sólida para o aprendizado indutivo e o raciocínio em redes neurais artificiais por meio de teoremas que mostram como a programação lógica pode ser uma linguagem de representação do conhecimento.

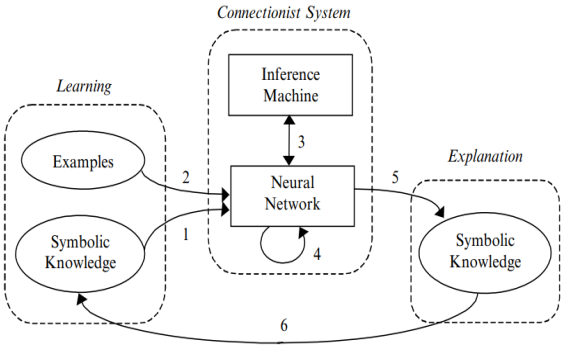

Fig. 4. Ciclo de aprendizagem Neuro-Simbólico do CILP

O CILP dividiu a aprendizagem neuro-simbólica em três partes, cada parte está associada a determinada fase numerada no diagrama acima:

1) Refinação do conhecimento de uma rede neural (fases 1, 2 e 3 do CILP);
2) Extração do conhecimento de uma rede treinada (fase 5 do CILP);
3) Revisão da teoria de uma rede neural (fase 4 do CILP).

Com isso, o ciclo de aprendizagem é fechado, como mostrado na figura 6.

C. Logic Tensor Networks

Logic Tensor Networks (LTN) é um *framework* o qual provê uma boa forma de combinar redes neurais com as regras simbólicas utilizando uma lógica ***fuzzy***. Ele utiliza uma sintaxe de Lógica de Primeira Ordem (LPO) interpretada em números reais, chamada *Real Logic*, a qual é implementada como uma *deep tensor network*.

Em uma LTN, a entrada recebe um vetor de características e realiza operações reais, resultando em uma saída com um valor entre zero e um. A LTN converte formas de lógica real em um grafo computacional ***TensorFlow***. Essas fórmulas expressam consultas de dados complexas, apredizado prévio e declarações. No LTN pode-se representar e computar as tarefas mais importantes de *deep-learning*.

III. Descrição do problema de Michalski

O problema de Michalski [7] descreve um conjunto de trens indo para a direção leste ou oeste.

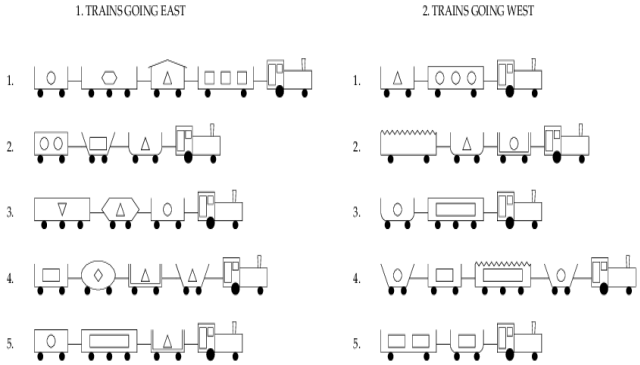

Fig. 5. Problema do trem de Michalski

Cada trem é composto pelos mesmos atributos, e se diferem em seus respectivos valores atribuidos. Os atributos são:

- Quantidade de vagões (valores entre 3 a 5);
- Quantidade de cargas diferentes que se pode levar (valor entre 1 a 4);
- Quantidade de eixo com rodas em um vagão (Valor entre 2 e 3);
- O comprimento do vagão (Curto ou Longo;
- O formato da carroceria do vagão (são nove tipos de carroceria);
- Quantidade de cargas no vagão (valores entre 0 a 3);
- Formato da carga (Quatro formatos de carga: círculo, hexágono, retângulo ou triângulo);

Cada trem conta com dez varíaveis booleanas as quais descrevem se qualquer par de tipos de carga estão ou não em vagões adjacentes do trem, como descrito abaixo:

1) Existe um retângulo próximo de um retângulo;
2) Existe um retângulo próximo de um triângulo;
3) Existe um retângulo próximo de um hexágono;
4) Existe um retângulo próximo de um círculo;
5) Existe um triângulo próximo de um triângulo;
6) Existe um triângulo próximo de um hexágono;
7) Existe um triângulo próximo de um círculo;
8) Existe um hexágono próximo de um hexágono;
9) Existe um hexágono próximo de um círculo;
10) Existe um círculo próximo de um círculo.

O documento também descreve que existe apenas um único atributo de classe que define a direção do trem.

IV. Modelo em LTN

O método de modelagem da LTN para este trabalho foi o proposto em [8]. Neste experimento, foi criada uma Perceptron Multicamadas com uma camada de entrada com 37 neurônios (atributos), uma camada oculta com 16 neurônios, uma camada de regularização *dropout* com taxa 0.2 e uma camada de saída com 1 neurônio, que denota se o trem pertence à classe leste ou oeste. A arquitetura da rede citada está exposta na Figura 6. figura 5 abaixo.

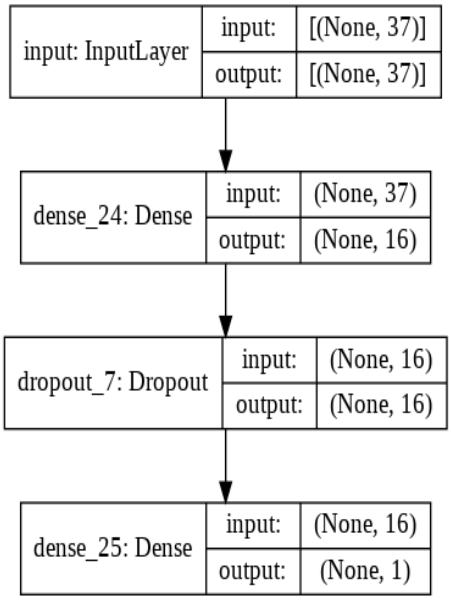

Fig. 6. Arquitetura da rede neural proposta.

V. Resultados

Neste experimento, foram separadas 8 instâncias para base de treino e 2 instâncias para base de teste. Ao longo do treinamento da LTN foram guardados os valores de média e acurácia. Ao fim foi obtida uma tabela de resultados, expressa pela Tabela 1.

Epoch	Train-mean	Test-mean	Train-accuracy	Test-accuracy
0	0.0160	0.0221	0.0000	0.0000
20	0.0336	0.0404	0.0000	0.5000
40	0.0672	0.0637	0.8750	1.0000
60	0.1601	0.1400	1.0000	0.5000
80	0.4253	0.3709	1.0000	0.5000
100	0.7434	0.6312	1.0000	1.0000
120	0.8781	0.7447	1.0000	1.0000
140	0.9518	0.6100	1.0000	0.5000
160	0.9781	0.5978	1.0000	0.5000
180	0.9897	0.6086	1.0000	0.5000
200	0.9919	0.4645	1.0000	0.5000
220	0.9964	0.6152	1.0000	0.5000
240	**0.9981**	**0.8362**	**1.0000**	**1.0000**
260	0.9983	0.7023	1.0000	1.0000
280	0.9986	0.6843	1.0000	1.0000

O experimento realizado pela equipe pode ser visualizado https://github.com/leticiabalbi/AI/tree/main/MichalskiTrain.

VI. Conclusão

Nesse trabalho implementamos uma solução do problema do trem de Michalski utilizando LTN. O método de modelagem da LTN para este trabalho foi o proposto em [8]. Os resultados obtidos demonstram que a aplicação da LTN é forma eficiente para a obtenção do conhecimento e tarefas de predição de dados.

References

[1] A. S. Garcez, L. C. Lamb, and D. M. Gabbay, *Neural-symbolic cognitive reasoning*. Springer Science & Business Media, 2008.

[2] S. Badreddine, A. d'Avila Garcez, L. Serafini, and M. Spranger, "Logic tensor networks," 2021.

[3] A. d. Garcez, M. Gori, L. C. Lamb, L. Serafini, M. Spranger, and S. N. Tran, "Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning," *arXiv preprint arXiv:1905.06088*, 2019.

[4] s. mspranger, "Logic tensor networks," https://github.com/logictensornetworks/logictensornetworks, 2018.

[5] G. G. Towell and J. W. Shavlik, "Knowledge-based artificial neural networks," *Artificial intelligence*, vol. 70, no. 1-2, pp. 119–165, 1994.

[6] A. S. d. Garcez, K. B. Broda, and D. M. Gabbay, *Neural-symbolic learning systems: foundations and applications*. Springer Science & Business Media, 2012.

[7] R. S. Michalski, "Learning strategies and automated knowledge acquisition," in *Computational models of learning*. Springer, 1987, pp. 1–19.

[8] Garcez, "Logic tensor networks," https://nbviewer.jupyter.org/github/logictensornetworks/logictensornetworks/blob/master/examples/multiclass$_c lassification/multiclass - singlelabel.ipynb$, 2018.