Implementação do método de Aprendizagem Relacional no paradigma de Sistemas Neuro-Simbólicos

Fabrício C. Guimarães * **Laura A. Martinho** * **Lorena Bastos** *
Mattheus S. C. Fernandes *

* *Faculdade de Tecnologia, Engenharia da Computação, Universidade Federal do Amazonas, AM. E-mail: {fdcg, lam2, lba, mscf}@icomp.ufam.edu.br.*

06 de Maio de 2022

1. INTRODUÇÃO

Um dos interesses da comunidade acadêmica está relacionado à aprendizagem de máquinas no quesito de representação de conhecimento simbólico em forma de rede neural, este é o objetivo principal desse trabalho, a implementação do algoritmo de conversão neuro-simbólica do sistema híbrido Knowledge-Based Artificial Neural Networks (KBANN). Esse sistema possui capacidade de mapear um domínio teórico pré-existente de regras em uma rede neural artificial, que por sua vez tem capacidade de refinamento utilizando técnicas de aprendizado. No exemplo desse trabalho, será utilizada uma base de conhecimento de parentescos, que servirá para a identificação de quais parentes tem filha ou não.

2. OBJETIVO

O objetivo deste trabalho é aplicar o conceito de redes neurais no problema do reconhecimento dos parentes que tem filha, visando aplicar conceitos de aprendizagem neuro-simbólico apresentando regras extraídas da rede e apresentando métricas e resultados gerados.

3. REFERENCIAL TEÓRICO

3.1 Redes Neurais

A ideia de Redes Neurais Artificiais (RNA's) é uma técnica que tenta imitar a estrutura neural e a capacidade do cérebro humano de reconhecer e generalizar os padrões. Da forma similar em que nós, seres humanos, aprendemos com a experiência em nossas vidas, as redes neurais requerem dados para aprender. Na maioria dos casos, quanto mais tempo e dados nessa rede puderem ser colocados para ajuste, mais eficiente será o aprendizado e menor será o número de erros cometidos.
Uma RNA é composta por camadas de nos contendo Camada de entrada, Camada escondida e Camada de saída. No qual cada nó possui pesos e viés relacionadas entre si. O processamento em cada neurônio se dá pelo que chamamos de função de ativação. A escolha das funções de ativação de uma rede neural são uma consideração importante uma vez que definem como devem ser seus dados de entrada. Algumas delas são: Funções De Ativação Linear, Sigmoid, Tangente Hiperbólica, Softmax, ReLU e ELU.

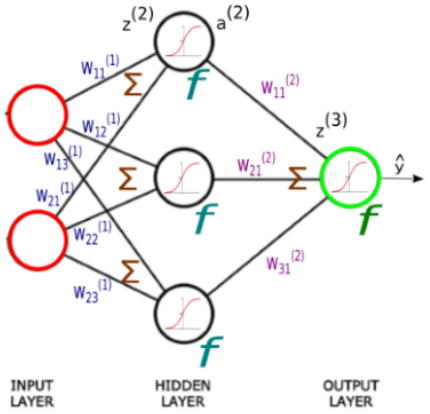

Figura 1. Representação de uma RNA

3.2 Como o aprendizado ocorre

Esse aprendizado ocorre com a rede sendo treinada em várias épocas, onde em cada época os pesos e viés são ajustados para melhor previsão do modelo. Nessa parte que entra o algoritmo de backpropagation, ele é responsável por fazer os cálculos de erro e repassar para os neurônios adjacentes, com a finalidade de diminuir o erro na previsão.

$$E_{rr}(W) = \sum_{k=1}^{n} \frac{1}{2} * (target - output)^2$$

No backpropagation temos duas etapas, a primeira seria a propagação de uma entrada já conhecida na rede(target). É feito todos os cálculos usando a função de ativação e no final temos o resultado dos cálculos(output), usamos a formula acima para obter o erro na precisão. Agora entra a segunda parte, a retro propagação, onde iremos ajustar os pesos, retro propagando o erro através da rede neural, partindo da camada de saída indo até a camada de entrada. Nessa parte é usado o gradiente descente em cima do vetor de pesos W.

$$\Delta\omega = -n \cdot \nabla_{\omega} * E_{rr}(W)$$

3.3 Aprendizagem Neuro-Simbólica

O Aprendizado neuro simbólico diz respeito à combinação de redes neurais artificiais com métodos simbólicos, a partir da representação do conhecimento baseado em lógica e do raciocínio em inteligência artificial. Esse aprendizado é traduzido para o sistema.
Um algoritmo de tradução mapeia um programa de logica genérico para uma rede neural N de uma camada oculta, que computa o programa mapeado. Além disso N pode ser treinada a partir de exemplos utilizando backpropagation, utilizando o programa P como background knowledge. Por fim, o conhecimento gerado pode ser extraído, fechando o ciclo de aprendizagem.
Com isso, a aprendizagem neuro-simbólica pode ser separada em 3 partes: Refinamento de conhecimento na rede neural, extração de conhecimento de uma rede neural treinada e revisão da teoria da rede neural.
O conhecimento adquirido pelo treinamento pode então ser extraído, fechando o ciclo de aprendizagem, mostrado na imagem abaixo.

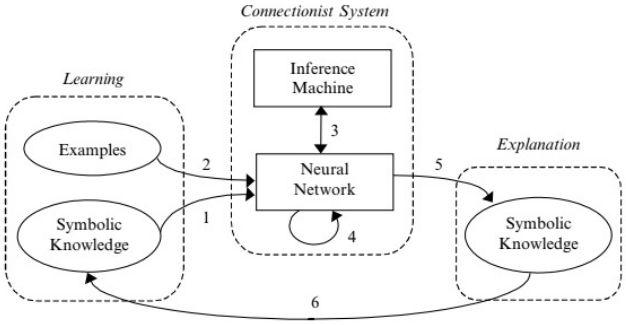

Figura 2. Aprendizagem Neuro-Simbólica

O passo 1 utiliza a técnica de CILP para transformar um conjunto de cláusulas em uma rede neural. A camada de entrada possui um neurônio para cada literal da cláusula. A camada oculta é composta pela quantidade de regras. A conexões entre as camadas é feita segundo as conjunções das cláusulas, como elas interagem para formar a regra. O passo 2 representa a alimentação da rede com os dados de entrada para treino. No passo 3 representa o treino em si, onde usaremos a rede neural e os métodos de aprendizagem. No passo 4 a extração de regras refinadas pela rede. E o último, o passo 5, seria a retro propagação, onde faremos os ajustes dos pesos e viés.

4. PROBLEMA PROPOSTO

Considere o seguinte conhecimento prévio (ou BK - Background Knowledge). E o conjunto de fórmulas F com candidatas a hipóteses. Implemente o método de Fast Relational Learning no paradigma de sistemas NeSy (Neural-Symbolic).

Base de conhecimento prévio: parents

(1) parent(pam,bob).
(2) parent(tom,bob).
(3) parent(tom,liz).
(4) parent(bob,ann).
(5) parent(bob,pat).
(6) parent(pat,jim).
(7) parent(ann,eve).
(8) male(tom).
(9) male(bob).
(10) male(jim).
(11) female(pam).
(12) female(liz).
(13) female(ann).
(14) female(pat).
(15) female(eve).

Fórmulas F candidatas a hipóteses de h(x): has daughter

(1) hd(X) ← male(X), parent(Y,X).
(2) hd(X) ← male(X), parent(X,Y).
(3) hd(X) ← female(X), parent(Y,X).
(4) hd(X) ← male(X), parent(Y,X).
(5) hd(X) ← female(X), parent(X,Y), female(Y).
(6) hd(X) ← male(X), parent(Y,X), female(Y).
(7) hd(X) ← male(X), parent(X,Y), female(Y).
(8) hd(X) ← female(X), parent(Y,X), male(Y).
(9) hd(X) ← female(X), parent(X,Y), female(Y).
(10) hd(X) ← male(X), parent(Y,X), parent(Y,Z).
(11) hd(X) ← male(X), parent(X,Y), female(Y).
(12) hd(X) ← male(X), parent(X,Y), female(Y), parent(Y,Z), male(Z).
(13) hd(X) ← female(X), parent(Y,X), male(Y), parent(Z,Y), female(Z).
(14) hd(X) ← female(X), parent(Y,X), male(Y), parent(Y,Z), female(Z).
(15) hd(X) ← female(X), parent(Y,X), male(Y), parent(Y,Z), female(Z), parent(Z,W), male(W).

5. SOLUÇÃO

5.1 Identificando Cláusulas Negativas

Para que uma das hipóteses de h(x) seja positiva, é necessário que X seja pai de Y (*parent(X,Y)*) e ao mesmo tempo Y seja feminino (*female(Y)*). Sabendo disso, podemos destacar os exemplos de cláusulas negativas, assim adicionando o sinal de ˜ para identificá-las.

Exemplos:

- ˜ hd(X) ← male(X), parent(Y,X).
- ˜ hd(X) ← female(X), parent(Y,X).
- ˜ hd(X) ← female(X), parent(X,Y), male(Y).
- ˜ hd(X) ← male(X), parent(Y,X), parent(Y,Z).

5.2 Outra maneira de representar as cláusulas

Trocando os predicados parent por father ou mother, teremos outras hipóteses, primeiro vamos definir o que seriam esses novos predicados.

- father(X,Y) ← male(X), parent(X,Y).
- mother(X,Y) ← female(X), parent(X,Y).

Com isso é possível alterar as hipóteses que tem os predicados necessários para troca.

Antigo:

- ˜ hd(X) ← male(X), parent(X,Y).
- ˜ hd(X) ← female(X), parent(X,Y).
- hd(X) ← male(X), parent(X,Y), female(Y).
- hd(X) ← female(X), parent(X,Y), female(Y).
- ˜ hd(X) ← female(X), parent(Y,X), male(Y), parent(Z,Y), female(Z).

Novo:

- ˜ hd(X) ← father(X,Y).
- ˜ hd(X) ← mother(X,Y).
- hd(X) ← father(X,Y), female(Y).
- hd(X) ← mother(X,Y), female(Y).
- ˜ hd(X) ← female(X), father(Y,X), mother(Z,Y).

Indefinido:

- ˜ hd(X) ← female(X), parent(Y,X).

Não pode-se gerar uma nova hipótese devido não está presente o sexo do parent Y, então não é definido se é mother ou father de X.
Com isso, é possível observar que para transformarmos totalmente todas as hipóteses originais, seriam necessárias mais informações que não estão presentes, o que faz com que se perca algumas cláusulas. Porém ainda tem outro aspecto, caso todas as cláusulas tivessem informações suficiente para a mudança de parent para mother/father, teríamos menos predicados por cláusulas, o que diminuiria a quantidade de neurônios da camada de entrada.

5.3 Base de Dados

Com todas as hipóteses devidamente rotuladas (positivas ou negativas), são utilizadas entradas do tipo one-hot, onde todos os termos que aparecem nas hipóteses recebem 1 caso estejam presentes na cláusula ou 0 caso contrário. É o que mostra a base de dados criada na Figura 3:

Cláusula N°	male(X)	female(X)	parent(Y,X)	parent(X,Y)	female(Y)	male(Y)	parent(Y,Z)	male(Z)	female(Z)	parent(Z,Y)	parent(Z,W)	male(W)	hd(X)
1	1	0	1	0	0	0	0	0	0	0	0	0	0
2	1	0	0	1	0	0	0	0	0	0	0	0	0
3	0	1	1	0	0	0	0	0	0	0	0	0	0
4	1	0	1	0	0	0	0	0	0	0	0	0	0
5	0	1	0	1	1	0	0	0	0	0	0	0	1
6	1	0	1	0	1	0	0	0	0	0	0	0	0
7	1	0	0	1	1	0	0	0	0	0	0	0	1
8	0	1	1	0	0	1	0	0	0	0	0	0	0
9	0	1	0	1	1	0	0	0	0	0	0	0	1
10	0	0	1	0	0	0	1	0	0	0	0	0	0
11	1	0	0	1	1	0	0	0	0	0	0	0	1
12	1	0	0	1	1	0	1	1	0	0	0	0	1
13	0	1	1	0	0	1	0	0	1	1	0	0	0
14	0	1	1	0	0	1	1	0	1	0	0	0	0
15	0	1	1	0	0	1	1	0	1	0	1	1	0

Figura 3. Base de dados inicial: has daughter

Para fins de aprendizado da rede neural, existem poucos casos positivos e muitos negativos, então é necessário balancear essa base inicial adicionando diversos outros casos positivos e negativos, assim gerando uma entrada de treino mais eficiente para a RNA.

5.4 Parâmetros da Rede Neural

Para a estrutura, treino e teste da rede neural, é necessário a extração de informações da base de dados inicial, com isso definindo itens como:

N° neurônios da camada de entrada: N° de literais possíveis nas hipóteses (12).
N° neurônios da camada escondida: N° de cláusulas do problema (15).
N° neurônios da camada de saída: Neurônios Verdadeiro/Falso para h(X) (2).
Função de Ativação: ReLU (do inglês Rectified Linear Unit, Unidade Retificada Linear)

5.5 Estrutura da Rede Neural

A partir dos parâmetros estabelecidos, foi estruturado a arquitetura da rede neural na Figura 4, onde na camada de entrada existem todos os literais possíveis como neurônios. Cada seta entre a camada de entrada e a camada escondida mostra onde cada predicado está presente dentre as cláusulas. Já entre a camada escondida e a camada de saída, observa-se apenas duas cores de setas, onde as vermelhas representam as hipóteses negativas e as verdes as positivas.

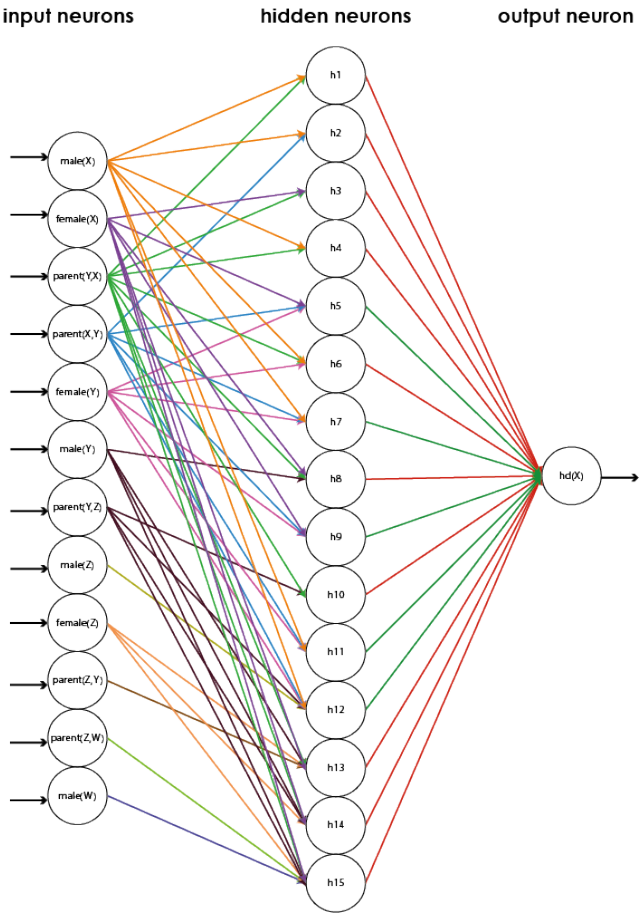

Figura 4. Estrutura da Rede Neural

6. EXPERIMENTO

6.1 Ambiente de Desenvolvimento

Para treino e teste da rede neural, se utilizou da plataforma Google Colaboratory, que possibilita a implementação colaborativa de Python, linguagem que se desenvolveu essa solução.

6.2 Implementação

Com o uso da biblioteca TensorFlow v1.x, foi possível montar a estrutura da rede neural artificial, treinar e testar a mesma. Como foi citado no item 5.3 (Base de Dados), foi necessário adicionar novas hipóteses respeitando a

configuração das originais, assim obtendo 48 cláusulas na base final. Cerca de 90% das cláusulas são para o treino, que tem 15 épocas.
A implementação completa referenciada nesse trabalho tem um HOW TO anexado, no repositório do GitHub, assim como o link para o Google Colaboratory com todos os códigos.

6.3 Resultados

Para todos os dados de teste, a rede neural conseguiu prever pontualmente todos os dados apresentados. Com um teste de 5 cláusulas (aproximadamente 10% de 48) a rede conseguiu uma acurácia de 100% de acerto na previsão de saídas. É o que mostra a Figura 5.

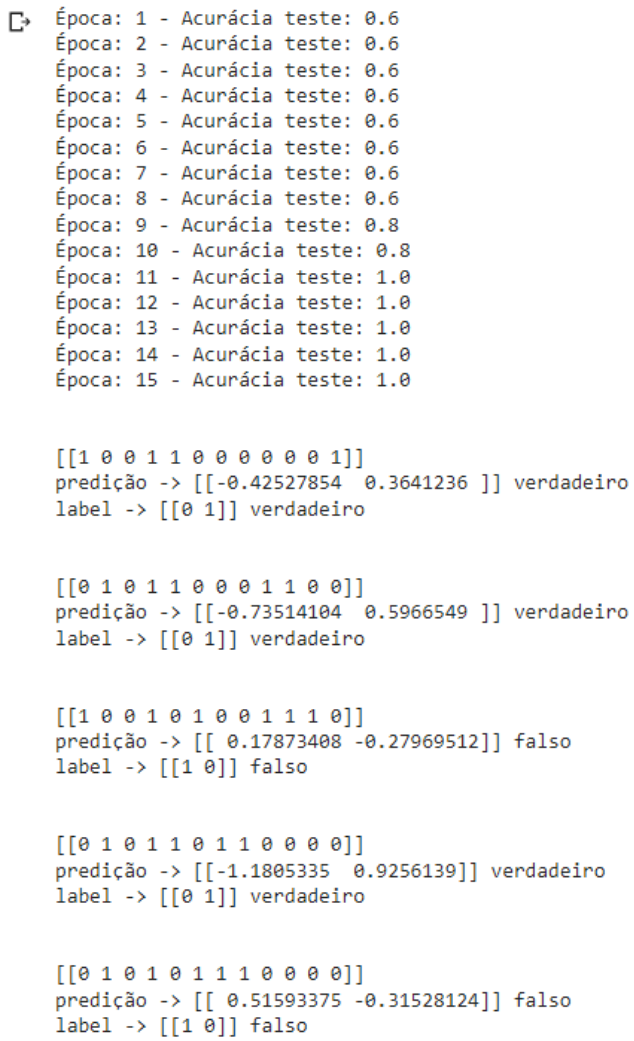

Figura 5. Resultados dos testes

7. CONCLUSÃO

Portanto, tendo em mente toda a fundamentação teórica e um conhecimento prévio de redes neurais foi possível resolver os exercícios propostos e encontrar resultados satisfatórios a partir dessas técnicas apresentadas. Percebe-se também que o uso das técnicas de backpropagation e CILP é algo realmente atual e usando nas maiorias dos modelos de IA, mesmo sendo criadas a bastante tempo. Novamente a IA volta a ser a tecnologia do momento sendo ela técnicas.

REFERÊNCIAS

SILVA, André Quintiliano Bezerra. Implementação e aplicação de algoritmos de aprendizado em um sistema neuro-simbólico. 2017. 89f. Dissertação (Mestrado em Engenharia Elétrica e de Computação) - Centro de Tecnologia, Universidade Federal do Rio Grande do Norte, Natal, 2017.

GARCEZ, A. S. d'Avila; BRODA, K.; GABBAY, D. M. Symbolic knowledge extraction from trained neural networks: A sound approach. Artificial Intelligence, 125:155–207, 2001.

GARCEZ, A. S. d'Avila; ZAVERUCHA, G. The connectionist inductive learning and logic programming system, Applied Intelligence Journal (Special Issue on Neural Networks and Structured Knowledge), 59–77, 1999.

GARCEZ, A. S. d'Avila; LAMB, L. C.; GABBAY, D. M. Neural-symbolic cognitive reasoning. Springer Science & Business Media, 2008.

FRANÇA, Manoel V. M.; ZAVERUCHA, G.; GARCEZ, A. S. d'Avila. Fast Relational Learning using Bottom Clause Propositionalization with Artificial Neural Networks, 2003.

GUIMARÃES, F. C; MARTINHO, L. A.; BASTOS, L.; FERNANDES, M. S. C. Repositório GitHub - Implementação do método de Aprendizagem Relacional no paradigma de Sistemas Neuro-Simbólicos. Disponível em: <https://github.com/abriciof/sistema-neuro-simbolico>. Acesso em: Maio de 2022.