Relatório do Trabalho Final de Inteligência Artificial

Carlos Andres[1], Davi Carvalho[1], Gabriel Toyoda[1], Pedro Larry[1]

[1]Faculdade de Tecnologia – Universidade Federal do Amazonas (UFAM)
69080-900 – Manaus – AM – Brazil

{carlos.junior, davi.carvalho, gabriel.toyoda,
pedro.lopes}@icomp.ufam.edu.br

Abstract. *This paper explores the application of artificial intelligence to solve the Michalski train problem, aiming to classify trains as going east or west based on their characteristics. To do this, different models are implemented and compared, including neural networks and a neuro-symbolic approach. Neural networks enable hierarchical learning of complex data representations, while the neuro-symbolic approach integrates symbolic techniques and neural networks, providing a more complete solution.*

Resumo. *Este artigo explora a aplicação de inteligência artificial para solucionar o problema do trem de Michalski, visando classificar trens como indo para leste ou oeste a partir de suas características. Para isso, diferentes modelos são implementados e comparados, incluindo redes neurais e uma abordagem neuro-simbólica. As redes neurais possibilitam a aprendizagem hierárquica de representações complexas dos dados, enquanto a abordagem neuro-simbólica integra técnicas simbólicas e redes neurais, proporcionando uma solução mais completa.*

1. Inteligência Artificial: Paradigmas Simbólico e Conexionista

A inteligência artificial (IA) busca replicar a capacidade humana de raciocínio e aprendizado em sistemas computacionais. Ao longo de sua história, a IA se desenvolveu em torno de dois paradigmas principais: o simbólico e o conexionista [Russell and Norvig 2003]. O paradigma simbólico, fundamentado na lógica e em representações explícitas do conhecimento, se destaca pela capacidade de realizar inferências complexas e lidar com raciocínio simbólico, sendo fundamental em áreas como planejamento e resolução de problemas [Haugeland 1989]. Por outro lado, o paradigma conexionista, inspirado no funcionamento do cérebro humano, utiliza redes neurais artificiais para aprender padrões complexos a partir de dados, demonstrando grande eficácia em tarefas como reconhecimento de padrões e processamento de linguagem natural [Haykin 2009].

1.1. Redes Neurais Artificiais: Modelando o Cérebro

Redes neurais artificiais (RNAs) são modelos computacionais inspirados na estrutura e função do cérebro humano [Kohonen 2012]. Elas consistem em unidades básicas de processamento, os neurônios artificiais, organizados em camadas interconectadas. Cada conexão entre neurônios possui um peso associado, que determina a influência da ativação

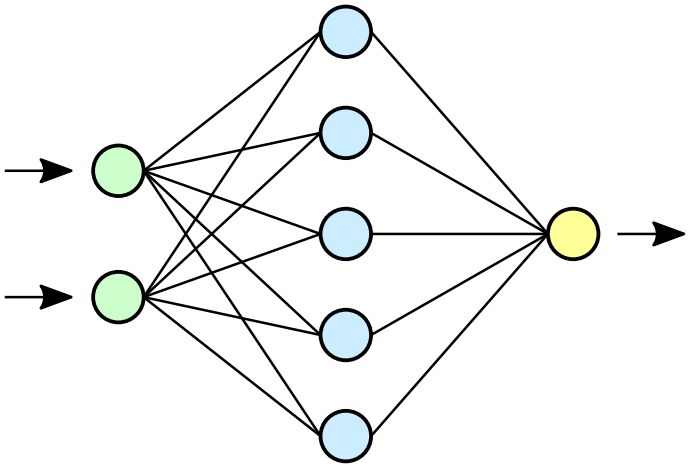

Figure 1. Exemplo de Rede Neural Artificial com múltiplas camadas ocultas.

de um neurônio sobre outro. O aprendizado em uma RNA se dá através do ajuste iterativo desses pesos, permitindo que a rede modifique seu comportamento em resposta aos dados de treinamento [Hinton and Salakhutdinov 2006].

Uma característica crucial das RNAs é a capacidade de aprender representações hierárquicas dos dados de entrada [Bengio 2009]. Através de múltiplas camadas de processamento, a rede aprende a extrair features cada vez mais abstratas e complexas, construindo representações sofisticadas que capturam as relações intrínsecas presentes nos dados. Essa capacidade de aprendizado hierárquico torna as RNAs ferramentas poderosas para lidar com problemas complexos, envolvendo grandes quantidades de dados, como reconhecimento de imagens [Krizhevsky et al. 2012], processamento de linguagem natural [Collobert and Weston 2008] e previsão de séries temporais [Hochreiter and Schmidhuber 1997].

1.2. Aprendizagem Neuro-Simbólica: Em Busca da Sinergia entre Símbolos e Conexões

Apesar do sucesso em diversas áreas, as RNAs tradicionais também apresentam desafios. Um dos principais desafios reside na opacidade da representação do conhecimento. Os pesos ajustados durante o treinamento codificam o conhecimento adquirido pela rede, mas essa codificação geralmente é distribuída e opaca, dificultando a interpretação e explicação do comportamento do modelo por parte dos humanos [Lipton 2018]. Essa falta de transparência pode ser um obstáculo em áreas onde a compreensibilidade do modelo é crucial, como saúde e justiça criminal, onde a explicabilidade das decisões tomadas por sistemas de IA é fundamental [Doshi-Velez and Kim 2017].

Além disso, RNAs tradicionais enfrentam dificuldades em lidar com tarefas que exigem raciocínio simbólico e manipulação de conhecimento estruturado, habilidades inerentes à cognição humana. Sistemas simbólicos, como os baseados em lógica de primeira ordem, excelem nesse tipo de tarefa, mas geralmente têm dificuldades em lidar com a incerteza inerente a muitos problemas do mundo real e com a necessidade de generalizar a partir de grandes conjuntos de dados [Besold et al. 2017].

A aprendizagem neuro-simbólica (ANS) surge como uma área de pesquisa promissora que busca combinar as vantagens das abordagens conexionistas e simbólicas

[Garcez et al. 2002]. Em vez de tratá-las como mutuamente exclusivas, a ANS visa integrá-las, criando sistemas híbridos que combinam a capacidade de aprendizado de máquina das RNAs com a capacidade de raciocínio simbólico dos sistemas baseados em lógica. Essa integração busca criar sistemas de IA mais robustos, flexíveis e capazes de lidar com uma gama maior de tarefas, incluindo aquelas que exigem tanto aprendizado a partir de dados quanto raciocínio simbólico [Bronstein et al. 2017].

Diversas abordagens têm sido exploradas dentro da ANS, incluindo:

- **Injeção de Conhecimento:** Nesta abordagem, o conhecimento simbólico, na forma de regras lógicas, restrições ou conhecimento prévio do domínio, é incorporado na estrutura da rede neural ou no seu processo de treinamento [Hu et al. 2018]. Essa integração pode se dar através da inicialização dos pesos da rede com base em regras lógicas, da adição de termos regulares à função de perda que incentivam a conformidade com o conhecimento simbólico, ou mesmo da modificação da própria arquitetura da rede para refletir as relações simbólicas presentes no domínio do problema.
- **Extração de Regras:** Após o treinamento de uma rede neural, técnicas de extração de regras são aplicadas para traduzir o conhecimento aprendido pela rede, expresso implicitamente nos pesos das conexões, em regras lógicas ou outras formas de representação simbólica, tornando o conhecimento adquirido pela rede mais compreensível para humanos [Haykin 2009]. Essas regras podem ser utilizadas para diversas finalidades, como a validação do modelo, a identificação de novas relações nos dados ou a criação de sistemas especialistas baseados em regras.
- **Arquiteturas Híbridas:** Sistemas híbridos combinam módulos neurais e simbólicos em uma arquitetura única e coesa [Serafini and Garcez 2016]. Esses módulos interagem entre si, permitindo a troca de informações e a resolução conjunta de problemas. Um exemplo clássico é a arquitetura Connectionist Modal Logic (CML) proposta por Garcez et al. (2007), que integra redes neurais com lógica modal para representar e raciocinar sobre conhecimento relacional.

A escolha da abordagem mais adequada para a integração neuro-simbólica depende da natureza do problema, dos dados disponíveis, dos objetivos da aplicação e dos recursos computacionais disponíveis.

2. Fundamentos Teóricos da Aprendizagem Neuro-Simbólica

A aprendizagem neuro-simbólica (ANS) representa uma área de pesquisa vibrante e promissora dentro da inteligência artificial. Seu objetivo principal é a criação de sistemas que combinam as capacidades de aprendizado de máquina de sistemas conexionistas, como as redes neurais, com as capacidades de representação do conhecimento e raciocínio lógico de sistemas simbólicos [**?**]. Essa integração visa superar as limitações inerentes a cada paradigma, criando sistemas mais robustos, interpretáveis e eficientes [Besold et al. 2017].

2.1. Motivações para a Hibridização Neuro-Simbólica

A busca pela integração neuro-simbólica é motivada pelas limitações dos paradigmas simbólico e conexionista quando considerados isoladamente.

- **Opacidade das Redes Neurais:** Redes neurais, apesar de eficazes em diversas tarefas, geralmente atuam como ”caixas pretas”, dificultando a compreensão humana do processo de tomada de decisão [Castelvecchi 2016]. A falta de interpretabilidade limita a confiança e a aplicabilidade dessas redes em domínios críticos, como saúde e finanças, onde a explicabilidade e justificativa das decisões são cruciais [Guidotti et al. 2018].
- **Limitações dos Sistemas Simbólicos:** Sistemas simbólicos, por outro lado, oferecem transparência e explicabilidade, mas enfrentam desafios em lidar com dados ruidosos, incerteza e aprendizado a partir de grandes conjuntos de dados, cenários onde as redes neurais se destacam [Smolensky 1988]. A manipulação simbólica também pode ser computacionalmente cara em cenários complexos, limitando a escalabilidade desses sistemas.

A hibridização neuro-simbólica busca aproveitar os pontos fortes de cada paradigma, criando sistemas que combinam a capacidade de aprendizado robusto e generalização a partir de dados das redes neurais com o raciocínio lógico, representação transparente do conhecimento e capacidade de lidar com conhecimento estruturado dos sistemas simbólicos.

2.2. Técnicas de Integração Neuro-Simbólica

Existem diversas abordagens para construir sistemas neuro-simbólicos, cada uma com suas vantagens e desvantagens:

- **Injeção de Conhecimento:** Nesta abordagem, o conhecimento simbólico, na forma de regras lógicas, restrições ou conhecimento prévio do domínio, é inserido na rede neural antes ou durante o treinamento [Hu et al. 2018]. Essa integração pode se dar de diversas maneiras, como pré-configurando os pesos da rede com base em regras lógicas, incorporando restrições simbólicas na função de perda utilizada para treinar a rede, ou ainda modificando a própria arquitetura da rede para refletir as relações simbólicas presentes no domínio do problema.
- **Extração de Regras:** Após o treinamento de uma rede neural, técnicas de extração de regras são aplicadas para obter representações simbólicas do conhecimento aprendido pela rede, expresso implicitamente nos pesos das conexões [Andrews et al. 1995]. Essas regras podem ser expressas na forma de árvores de decisão, lógica proposicional, programas lógicos ou outros formalismos simbólicos. A extração de regras torna o conhecimento adquirido pela rede mais compreensível para humanos, facilitando a depuração, validação e refinamento do modelo.
- **Arquiteturas Híbridas:** Sistemas híbridos combinam módulos neurais e simbólicos em uma arquitetura única, permitindo que os módulos interajam entre si, trocando informações e complementando as capacidades um do outro [Badreddine et al. 2020]. Um exemplo clássico é a arquitetura Connectionist Modal Logic (CML) proposta por Garcez et al. (2007) [Garcez et al. 2007], que integra redes neurais com lógica modal para representar e raciocinar sobre conhecimento relacional. Arquiteturas híbridas oferecem grande flexibilidade, permitindo que diferentes tipos de módulos, com diferentes níveis de granularidade, sejam combinados para construir sistemas complexos.

A escolha da abordagem mais adequada depende da natureza do problema, dos dados disponíveis, dos objetivos específicos da aplicação, dos recursos computacionais disponíveis e do nível de interpretabilidade desejado.

2.3. Vantagens e Desafios da Aprendizagem Neuro-Simbólica

Sistemas neuro-simbólicos de aprendizagem oferecem diversas vantagens potenciais:

- **Interpretabilidade e Explicabilidade:** Permitem entender o processo de tomada de decisão do modelo, aumentando a confiança e a aceitabilidade em domínios críticos, onde a explicabilidade é fundamental [Adadi and Berrada 2018].
- **Robustez a Dados Incompletos ou Ruidosos:** Combinam a tolerância a falhas das redes neurais com a capacidade de generalização dos sistemas simbólicos, criando sistemas mais robustos a dados imperfeitos [Hu et al. 2018].
- **Raciocínio e Inferência:** Permitem realizar inferências lógicas a partir do conhecimento aprendido, indo além do aprendizado puramente baseado em padrões e possibilitando a integração de diferentes fontes de conhecimento [Manhaeve et al. 2014].

No entanto, a construção de sistemas neuro-simbólicos eficazes também apresenta desafios:

- **Complexidade de Integração:** Integrar paradigmas distintos de forma eficaz e eficiente exige o desenvolvimento de novas técnicas e arquiteturas, além de um profundo entendimento de ambos os paradigmas [Besold et al. 2017].
- **Escalabilidade:** Garantir a escalabilidade dos modelos para lidar com problemas reais, envolvendo grandes conjuntos de dados e representações complexas do conhecimento, é um desafio constante [Du et al. 2021].
- **Interpretabilidade Completa:** Mesmo em sistemas neuro-simbólicos, a interpretabilidade completa do modelo pode ser desafiadora, especialmente em arquiteturas complexas com múltiplos módulos e interações. A busca por interpretabilidade deve ser balanceada com a capacidade de aprendizado e generalização do modelo [Rudin 2019].

3. Descrição do Problema

Em meados da década de 1980, Ryszard Michalski concebeu o problema do trem, também conhecido como problema do trem de Michalski. O objetivo deste problema é simples: classificar trens como indo para leste ou oeste. Cada trem possui características específicas que auxiliam nessa classificação, as quais estão ilustradas na Figura 2 e serão detalhadas a seguir:

1. Quantidade de vagões (valor: entre 3 e 5)
2. Quantidade de cargas diferentes que pode leva (valor: entre 1 e 4)
3. Para cada vagão do trem:
    - Quantidade de eixo como rodas (valor: entre 2 e 3)
    - Comprimento (valor: curto ou longo)
    - Formato da carroceria do vagão, o qual pode ser:
        - locomotiva
        - retângulo-fechado

  - retângulo-aberto
  - duplo retângulo-fechado
  - duplo retângulo-aberto
  - elipse
  - hexágono
  - topo dentado
  - trapézio fechado
  - trapézio aberto
  - topo inclinado-retângulo
  - topo inclinado-trapézio
  - em forma de U-fechado
  - em forma de U-aberto
  - duplo topo inclinado-retângulo
- Quantidade de cargas no vagão (valor: entre 0 a 3)
- Formato da carga (círculo, hexágono, retângulo ou triângulo)

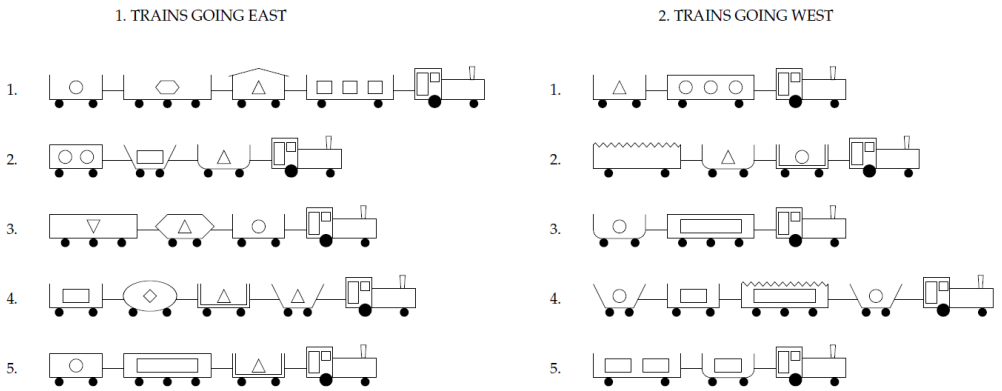

Figure 2. Trens de Michalski

Em seguida, temos 10 variáveis booleanas (proposicionais) que indicam se cada par de tipos de carga está ou não em vagões adjacentes do trem. Além disso, também contamos com as seguintes relações entre os vagões de um trem, cujos valores lógicos variam entre 0 (Falso) e 1 (Verdadeiro):

1. Existe um retângulo próximo a um retângulo (V ou F);
2. Existe um retângulo próximo a um triângulo (V ou F);
3. Existe um retângulo próximo a um hexágono (V ou F);
4. Existe um retângulo próximo a um círculo (V ou F);
5. Existe um triângulo próximo a um triângulo (V ou F);
6. Existe um triângulo próximo a um hexágono (V ou F);
7. Existe um triângulo próximo a um círculo (V ou F);
8. Existe um círculo próximo a um círculo (V ou F).

Existe um atributo de classe que define a direção do trem: leste ou oeste. Observe que para atributos com múltiplos valores, você deve atribuir valores na ordem em que eles aparecem. Por exemplo, o tipo de carga deve ser 1 para círculo, 2 para hexágono, 3 para retângulo e assim por diante.

O neurônio correspondente deve usar uma função linear, ou seja, $h(x) = x$.

Este problema tornou-se um padrão em testes e demonstrações de diversas técnicas de aprendizado de máquina, especialmente em implementações de indução lógica.

4. Modelos utilizados

Os modelos utilizados para a realização dos experimentos foram baseados no livro *Neural-symbolic Cognitive Reasoning* de [Garcez et al. 2002]. Esses modelos podem ser encontrados no link `https://github.com/CarlosARL/IAFINAL`, juntamente com as instruções para execução do experimento.

Primeiramente, realizamos um pre-processamento da base de dados rotulada manualmente

4.1. Modelo da questão 1

A rede neural de camadas densas utilizada segue as instruções do enunciado e é representada na figura fornecida. A arquitetura do modelo consiste em uma camada densa com 37 neurônios de entrada (um pra cada coluna da base da dados *trains-100-mod.csv*, disponível no GitHub) e 9 neurônios de saída com ativação linear, seguida por uma segunda camada densa com 9 neurônios de entrada e um neurônio de saída, com ativação sigmoide. O neurônio de saída é responsável por classificar uma instância como leste (eastbound) ou oeste (westbound). A representação gráfica do modelo é mostrada na Figura 3.

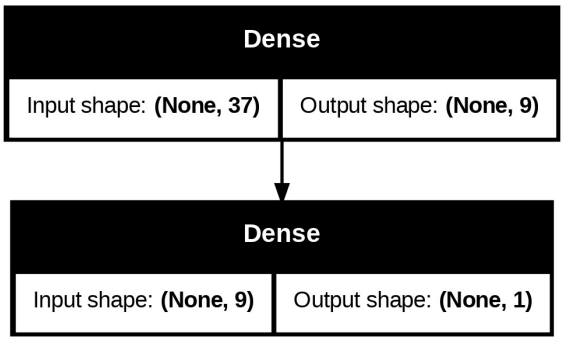

Figure 3. Modelo de camadas densas

4.2. Modelo da Questão 2 e Data Augmentation

Para a segunda questão, optamos por utilizar uma meta-rede, composta por duas redes neurais que representam proposições. A camada final da meta-rede é responsável pela classificação final do problema. No cerne da meta-rede, implementamos 11 regras, cada uma equivalente a uma rede neural individual. As regras foram definidas da seguinte forma:

- **num_cars (t,nc):** Determina se o trem 't' $\in$ [1..10] possui 'nc' vagões, onde 'nc' varia entre 3 e 5.
- **loads(t,nl):** Verifica se o trem 't' $\in$ [1..10] possui 'nl' tipos de carga distintos, com 'nl' $\in$ [1..4].

- **wheels(t,c,w):** Indica se o vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] possui 'w' rodas, onde 'w' pode ser 2 ou 3.
- **length(t,c,l):** Representa o comprimento do vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] como 'l', um vetor em $[-1, 1]^6$.
- **shape(t,c,s):** Codifica a forma do vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] como 's', um vetor em $[1..10]^7$.
- **num_car_loads(t,c,ncl):** Indica a quantidade de cargas 'ncl' $\in$ [0..3] no vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10].
- **load_shape(t,c,ls):** Representa a forma da carga no vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] como 'ls' $\in$ [1..4].
- **next_crc(t,c,x):** Verifica se o vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] está adjacente a um vagão com carga circular. 'x' é um vetor booleano de tamanho 8.
- **next_hex(t,c,x):** Verifica se o vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] está adjacente a um vagão com carga hexagonal. 'x' é um vetor booleano de tamanho 8.
- **next_rec(t,c,x):** Verifica se o vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] está adjacente a um vagão com carga retangular. 'x' é um valor booleano.
- **next_tri(t,c,x):** Verifica se o vagão 'c' $\in$ [1..4] do trem 't' $\in$ [1..10] está adjacente a um vagão com carga triangular. 'x' é um vetor booleano de tamanho 8.

O treinamento da meta-rede foi realizado utilizando o método leave-one-out, onde um trem é selecionado para validação e os demais para treinamento. Para melhorar a capacidade de generalização do modelo, especialmente considerando o conjunto de dados limitado (10 trens), aplicamos uma técnica de aumento de dados (Data Augmentation).

A técnica de Data Augmentation utilizada consiste em tratar cada vagão de um trem como uma instância independente durante o treinamento. Em outras palavras, se um trem possui 4 vagões, ele será considerado como 4 instâncias distintas no processo de treinamento, cada uma com suas características específicas. Com isso, o conjunto de dados original, composto por 10 trens, é expandido para 40 instâncias, aumentando a diversidade e representatividade dos dados para o treinamento da meta-rede.

A aplicação do Data Augmentation visa permitir que a meta-rede aprenda padrões não apenas em nível de trem completo, mas também em nível de vagão individual. Essa granularidade no aprendizado pode levar a um modelo mais robusto e capaz de generalizar melhor para novos dados, mesmo com um conjunto de dados inicial limitado.

Com o aumento de dados, a meta-rede foi treinada com 36 exemplos e validada com 4. A validação cruzada foi utilizada em relação a cada trem, garantindo que o modelo fosse avaliado em diferentes cenários e evitando um viés na avaliação do desempenho.

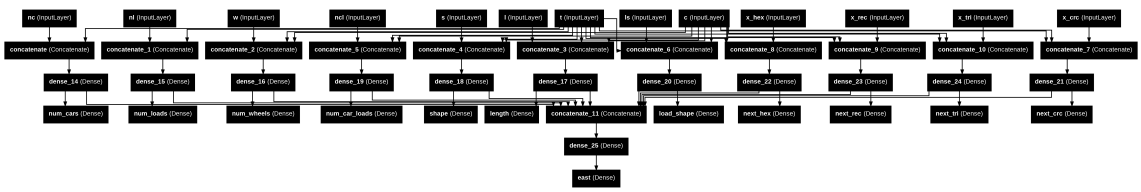

Figure 4. Metanet 2.

5. Resultados

Nesta seção, discutiremos os resultados dos modelos descritos na seção anterior.

5.1. Resultados do modelo da questão 1A

Os gráficos apresentados na Figura 5 mostram as curvas de perda e acurácia durante o treinamento e teste do modelo, utilizando uma divisão de dados de 70/30 para treino e teste, respectivamente.

O gráfico à esquerda na Figura 5 exibe a perda do modelo ao longo das épocas. Observa-se uma diminuição contínua da perda tanto para o conjunto de treinamento quanto para o conjunto de teste. No início do treinamento, a perda é elevada para ambos os conjuntos, mas diminui rapidamente nas primeiras 50 épocas. A partir daí, a redução na perda continua, mas em um ritmo mais lento, até se estabilizar nas últimas épocas, parando o treinamento em torno de 250 épocas por meio de um *EarlyStopping* definido no treinamento.

O gráfico à direita da Figura 5 mostra a acurácia do modelo ao longo das épocas. A acurácia do conjunto de treinamento aumenta de forma constante, atingindo um valor próximo de 70% nas últimas épocas. A acurácia do conjunto de teste também aumenta, mas apresenta mais flutuações ao longo do treinamento, estabilizando-se em torno de 67% nas últimas épocas, como mostrado também pela matriz de confusão da Figura 6.

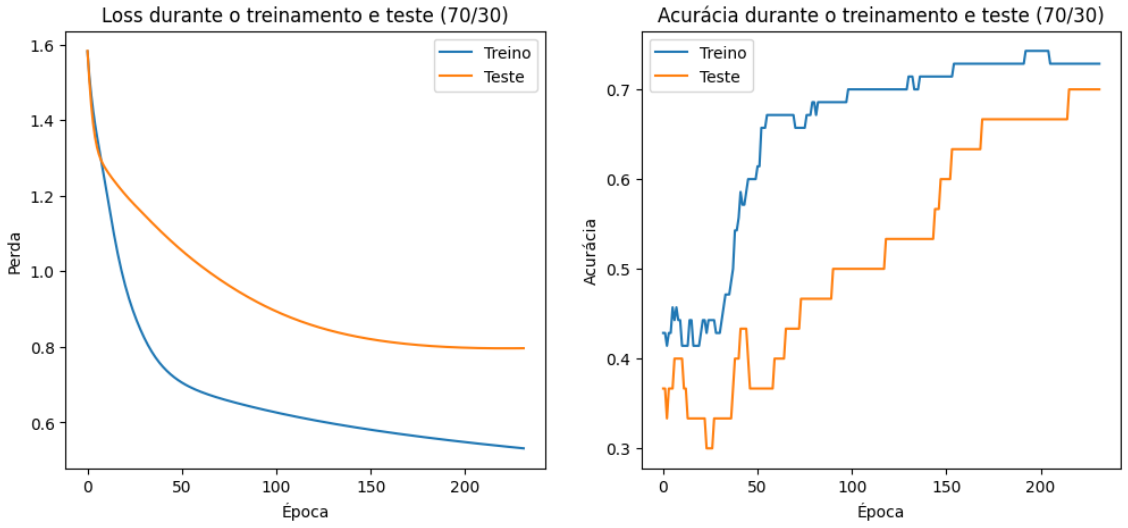

Figure 5. Gráficos de Loss e Acurácia durante o treino do caso A.

5.2. Resultados do modelo da questão 1B

Os gráficos apresentados na Figura 8 mostram as curvas de perda e acurácia durante o processo de validação cruzada *5-fold*, em que 5 *folds*, cada um com divisão 80/20 para treinamento e teste, foram usados a fim de fazer a validação cruzada do modelo.

No primeiro par de gráficos, mostrado na Figura 7, o gráfico à esquerda exibe a perda por fold ao longo dos folds de validação cruzada. A linha azul representa a perda para cada fold, enquanto a linha vermelha pontilhada indica a média da perda ao longo dos folds, e a área sombreada em rosa representa o desvio padrão da perda. Observa-se que a perda varia entre os folds, com a média situando-se em torno de 70%.

O gráfico à direita desse mesmo par mostra a acurácia por fold ao longo dos folds. A linha azul indica a acurácia para cada fold, a linha vermelha pontilhada representa a

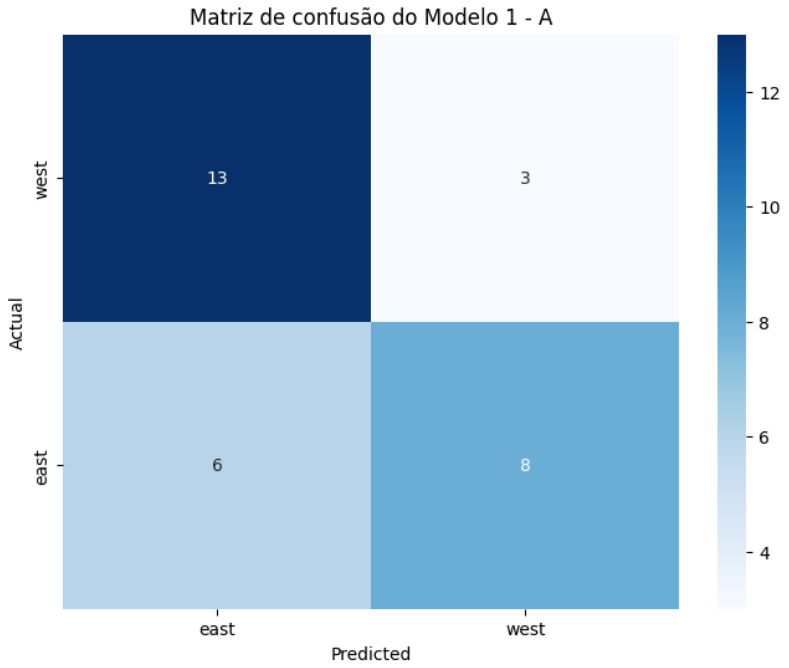

Figure 6. Matriz de confusão do teste do modelo 1 A.

média da acurácia, e a área sombreada em rosa indica o desvio padrão da acurácia. Nota-se que a acurácia flutua entre os folds, com uma média próxima de 57.5%, e o desvio padrão reflete a variabilidade entre as diferentes iterações.

No segundo par de gráficos, mostrados na Figura 8, o gráfico à esquerda mostra a perda durante o treinamento ao longo das épocas para cada fold. Cada cor representa um fold diferente, e observa-se uma diminuição contínua da perda, que é mais acentuada nas primeiras épocas e vai se estabilizando ao longo do tempo, sendo este comportamento comum a todos os folds. O EarlyStopping é aplicado para interromper o treinamento quando a perda se estabiliza, prevenindo overfitting, e observa-se que o Fold 5 foi o que mais treinou, parando apenas depois de centésima época.

O gráfico à direita desse mesmo par exibe a acurácia durante o treinamento ao longo das épocas para cada fold. Assim como no gráfico de perda, cada cor corresponde a um fold diferente. A acurácia aumenta de forma constante durante o treinamento, estabilizando-se ao longo das épocas em torno de 70% para todos os folds.

5.3. Resultados do modelo da questão 2

Os gráficos nas Figuras 9 e 10 mostram as curvas de perda e acurácia durante o treinamento e teste do modelo, utilizando o método LeaveOneOut de validação cruzada para treino e teste do modelo descrito.

O gráfico da Figura 9 exibe a acurácia binária do modelo ao longo das épocas. Observa-se que a acurácia do conjunto de validação (val) é muito alta desde o início, atingindo rapidamente o valor de 1.0. Já a acurácia do conjunto de treinamento (train) apresenta um comportamento mais instável no começo, mas se estabiliza em torno de 0.6 após as primeiras 50 épocas.

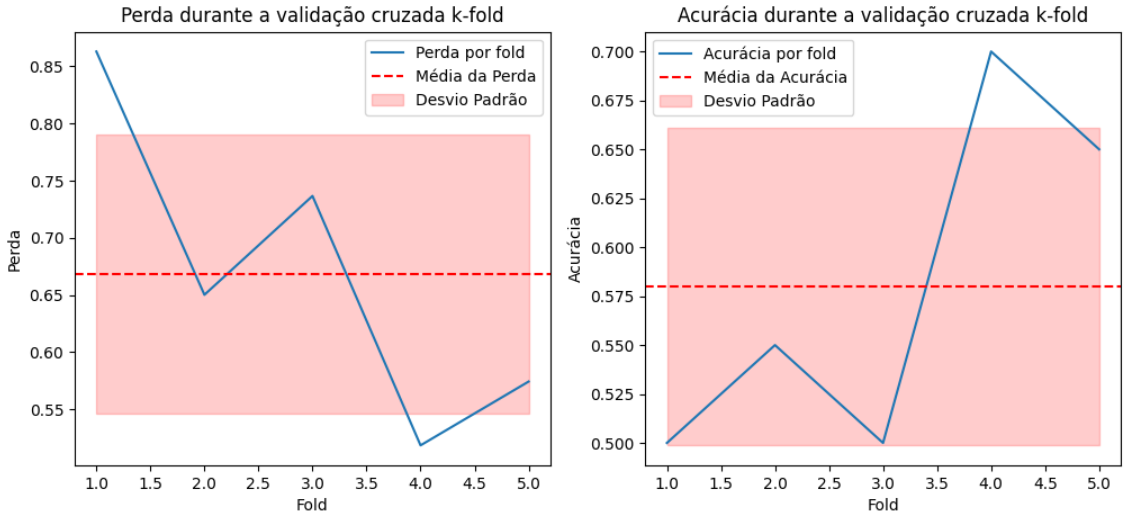

Figure 7. Perda e acurácia por fold no modelo B da questão 1.

O gráfico à direita da Figura 10 mostra o erro quadrático médio (MSE) do modelo ao longo das épocas. Inicialmente, o MSE é alto para ambos os conjuntos de dados, mas diminui rapidamente nas primeiras 50 épocas. A partir daí, o MSE para o conjunto de validação continua a diminuir até quase zero, enquanto o MSE do conjunto de treinamento se estabiliza em torno de 0.2.

5.4. Resultados dos testes para a questão 3

5.4.1. Letra A

Nesta seção, discutiremos os resultados da questão 3. Na letra A, os modelos utilizados nas questões 1 e 2 apresentam diferenças significativas em suas arquiteturas, o que resulta em variações nos resultados numéricos. Na Questão 1, foi utilizado um modelo neuro-simbólico adaptado para classificar trens com base em predicados lógicos. Este modelo combina aprendizagem simbólica com redes neurais, utilizando predicados lógicos e as relações entre estes atributos para inferir a direção dos trens. A abordagem neuro-simbólica tende a ser mais interpretável devido aos componentes simbólicos que permitem uma compreensão mais direta das inferências feitas pelo modelo.

Por outro lado, na Questão 2, foi implementada uma solução utilizando LTNTorch, que integra a lógica simbólica diretamente na arquitetura da rede neural. O LTNTorch usa técnicas de aprendizado profundo para ajustar os pesos da rede neural, otimizando uma função de custo que incorpora restrições lógicas. Essa abordagem combina o poder de representação das redes neurais com as capacidades interpretativas da lógica simbólica. A integração profunda da lógica na rede neural permite que o modelo LTNTorch capture relações mais complexas nos dados, o que pode levar a uma melhor generalização dos resultados.
Nos resultados numéricos, o modelo LTNTorch tende a apresentar uma performance superior, especialmente em cenários que requerem a modelagem de relações complexas. A capacidade das redes neurais profundas de aprender representações complexas dos dados confere ao LTNTorch uma vantagem em termos de precisão e generalização. Por outro

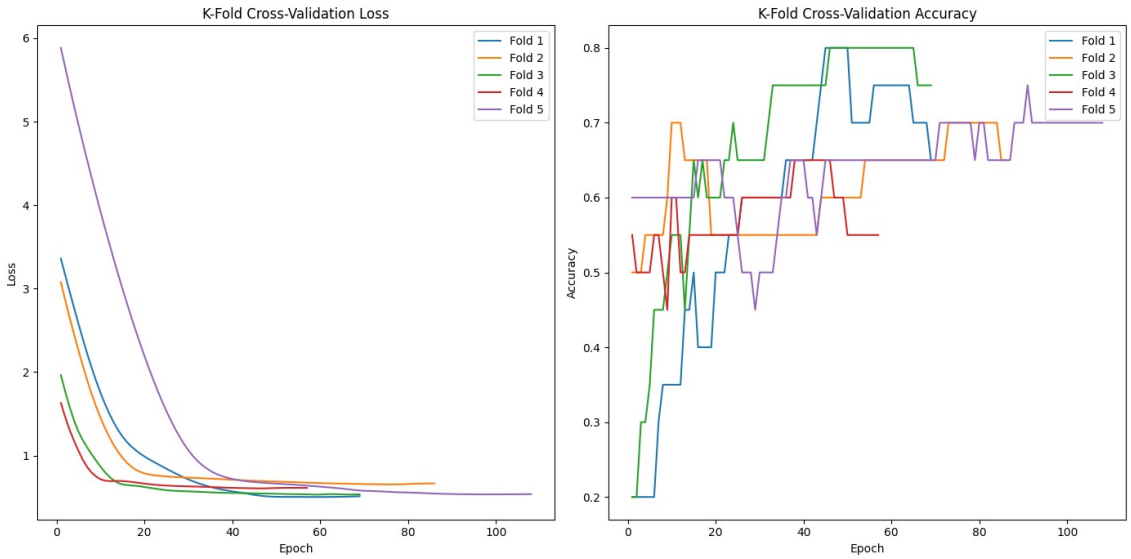

Figure 8. Perda e acurácia por época no modelo B da questão 1.

lado, o modelo neuro-simbólico, embora melhor interpretável, pode enfrentar limitações ao lidar com relações complexas que são mais facilmente capturadas por redes neurais profundas.

5.4.2. Letra B

Para extrair a regra genérica $\text{car}(T, C) \wedge \text{short}(C) \wedge \text{closed\_top}(C) \rightarrow \text{east}(T)$ dos dois modelos, o algoritmo de extração de conhecimento descrito no material de classe é útil para o propósito. A extração de regras de um modelo treinado envolve identificar padrões nas entradas e saídas que permitem derivar regras simbólicas.

Primeiramente, abordamos o processo de extração com base nos princípios gerais discutidos no material *Symbolic Knowledge Extraction from Trained Neural Networks* por [d'Avila Garcez et al. 2001] O processo de extração pode ser dividido em etapas distintas, começando pela identificação dos conceitos. Cada neurônio na camada de entrada representa uma característica dos dados (como car, short, e closed_top), enquanto a camada de saída representa a classe ou rótulo predito, que neste caso é east.

A seguir, o conhecimento adquirido pela rede neural durante o treinamento é codificado na arquitetura da rede, na função de ativação associada e nos valores dos pesos. Para extrair o conhecimento, devemos interpretar a combinação desses elementos de forma compreensível. Assim, definimos as regras de extração, que consistem em mapear as entradas diretamente nas saídas da rede. Utilizamos uma abordagem decomposicional onde analisamos as ativações individuais dos neurônios na camada intermediária e como eles contribuem para a ativação do neurônio de saída.

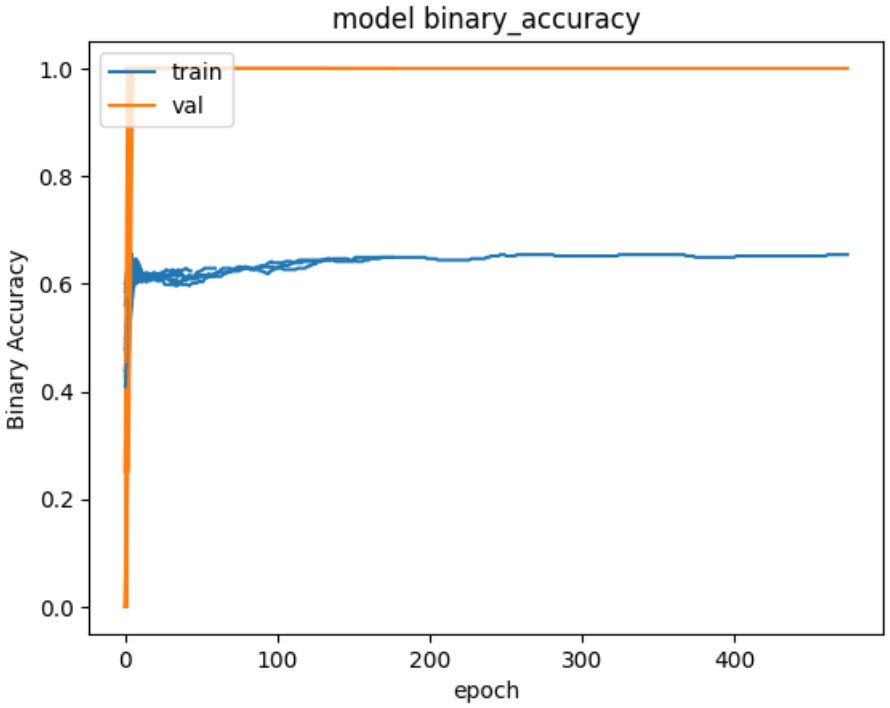

Figure 9. Perda e acurácia por época no modelo B da questão 1.

5.4.3. Letra C

Na letra C, três casos de verificação de aprendizado das redes foram apresentados, sendo eles:

- Se um trem tem um vagão curto e fechado, então ele vai para o leste, caso contrário, vai para o oeste.
- Se um trem tem dois vagões, ou tem um vagão com teto irregular, então ele vai para o oeste, caso contrário, vai para o leste.
- Se um trem tiver mais de dois tipos diferentes de carga, então ele vai para o leste, caso contrário, vai para o oeste

Para averiguar os itens definidos, criamos dados de trens com e sem as características descritas em cada um dos casos, e os resultados podem ser vistos nas Figuras 11, 12 e 13.

Na Figura 11, definimos o caso de um trem com vagão curto e fechado que foi guardado na variável casoA1, e outro trem que define um trem com vagão curto e aberto, guardado na variável casoA2. Neste caso, o modelo definiu que o trem casoA1 está indo para leste, enquanto o trem casoA2 está indo para oeste.

Na Figura 12, definimos o caso de um trem com vagão com teto irregular que foi guardado na variável casoB1, e outro trem que define um trem que possui apenas dois vagões de carga, guardado na variável casoB2. Neste caso, o modelo definiu que tanto o trem casoB1 quanto o trem casoB2 estão indo para leste, confirmando a tese definida na questão.

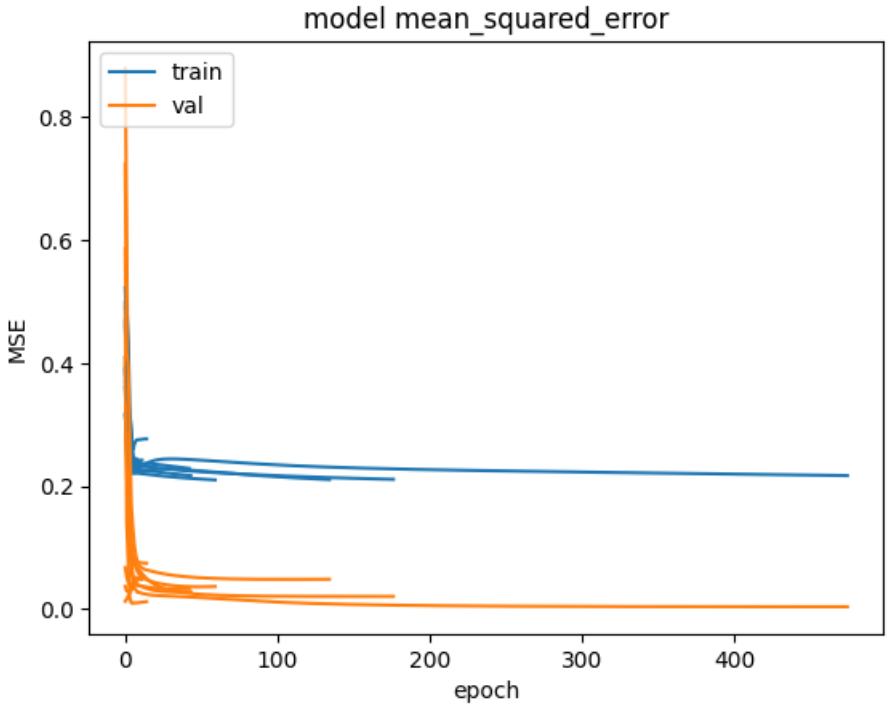

Figure 10. Perda e acurácia por época no modelo B da questão 1.

Na Figura 13, definimos o caso de um trem que carrega mais de dois tipos diferentes de carga (index 1 do array) que foi guardado na variável casoC1, e outro trem que possui exatamente dois tipos de carga, guardado na variável casoC2. Neste caso, o modelo definiu que o trem casoC1 vai para leste, enquanto o trem casoC2 vai para oeste.

References

Adadi, A. and Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE access*, 6:52138–52160.

Andrews, R., Diederich, J., and Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-based systems*, 8(6):373–389.

Badreddine, A. E., Donnat, C., Lou, J.-M., and Medioni, G. (2020). Reinforcement learning with pretrained neuralsymbolic representations for zero-shot generalization. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1671–1680. IEEE.

Bengio, Y. (2009). Learning deep architectures for ai. In *Foundations and trends in Machine Learning*, volume 2, pages 1–127. Now Publishers Inc.

Besold, T. R., Garcez, A. d., Bader, S., Bowman, H., Domingos, P., Hitzler, P., Küchler, K.-U., Lamb, L. C., Lowd, D., et al. (2017). Neural-symbolic learning and reasoning: A survey and interpretation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

```
caseA1 = np.asarray([ 4.,  3.,  2.,  0.,  1.,  1.,  0.,  2.,  0.,  2.,  1.,  2.,  2.,
        0., 13.,  1.,  3.,  2.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,
        0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.]).astype(float)
        # Caso com trem contendo vagão curto e fechado

caseA2 = np.asarray([4., 2., 2., 1., 2., 1., 0., 2., 0., 2., 1., 2., 2., 0., 9., 1., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
       0., 0., 0.]) # Caso de trem com vagões curto e aberto

model.predict(caseA1.reshape(1,37)) ## EAST

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 72ms/step
array([[0.75850886]], dtype=float32)

model.predict(caseA2.reshape(1,37)) ## WEST

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
array([[0.03048049]], dtype=float32)
```

Figure 11. Testes para o caso A.

```
caseB1 = np.asarray([ 5.,  3.,  2.,  0., 13.,  1.,  0.,  2.,  1.,  7.,  1.,  2.,  2.,
        0.,  9.,  1.,  3.,  2.,  0.,  4.,  1.,  3.,  2.,  0.,  0.,  0.,
        0.,  0.,  1.,  0.,  0.,  1.,  0.,  0.,  0.,  0.,  0.]) ## teto irregular

caseB2 = np.asarray([[3., 2., 2., 0., 2., 1., 2., 2., 0., 4., 1., 0., 2., 0., 0., 0., 0.,
       0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 0., 0., 0.,
       0., 0., 0.]]) ## dois vagões

model.predict(caseB1.reshape(1,37)) ## Vai para o LESTE

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 36ms/step
array([[0.93104255]], dtype=float32)

model.predict(caseB2.reshape(1,37)) ## Vai para o LESTE

1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step
array([[0.8356873]], dtype=float32)
```

Figure 12. Testes para o caso B.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Castelvecchi, D. (2016). Can we open the black box of ai? *Nature*, 538(7623):20–23.

Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167.

d'Avila Garcez, A., Broda, K., and Gabbay, D. (2001). Symbolic knowledge extraction from trained neural networks: A sound approach. *Artificial Intelligence*, 125(1):155–207.

Doshi-Velez, F. and Kim, B. (2017). Towards a rigorous science of interpretable machine learning. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1319–1328.

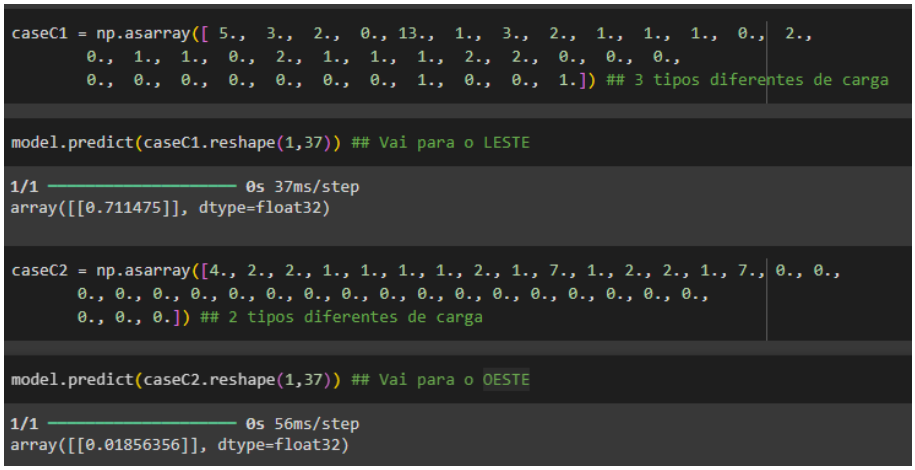

Figure 13. Testes para o caso C.

Du, Z., Huang, S., Zheng, W., Dong, Y., Wang, P., He, G., Zhou, W., Zhao, Z., Xiang, T., Xie, Z., et al. (2021). Glm: General language model pretraining with autoregressive blank infilling. *arXiv preprint arXiv:2112.01074*.

Garcez, A. S. d., Broda, K., and Gabbay, D. M. (2002). Neural-symbolic learning systems: foundations and applications. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence-Volume 1*, pages 488–493.

Garcez, A. S. d., Lamb, L. C., and Gabbay, D. M. (2007). Connectionist modal logic: Representing modalities in neural networks. In *Theoretical Computer Science*, volume 371, pages 34–53. Elsevier.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., and Pedreschi, D. (2018). A survey of methods for explaining black box models. In *Proceedings of the 2018 International Conference on Data Mining Workshops (ICDMW)*, pages 550–557. IEEE.

Haugeland, J. (1989). Artificial intelligence: The very idea. *MIT press*, 10(1):1–35.

Haykin, S. S. (2009). *Neural networks and learning machines*. Pearson Education.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. In *Science*, volume 313, pages 504–507.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hu, Z., Ma, X., Liu, J.-G., Hovy, E., and Xing, E. (2018). Harnessing deep neural networks with logic rules. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2410–2420.

Kohonen, T. (2012). Self-organizing maps. *Springer Series in Information Sciences*, 30:1–683.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, volume 25, pages 1097–1105.

Lipton, Z. C. (2018). The mythos of model interpretability. *Queue*, 16(3):31–57.

Manhaeve, R., Dumancic, S., Kimmig, A., De Raedt, L., and Demeester, H. (2014). Neural-symbolic cognitive reasoning: A step into first-order logic. *arXiv preprint arXiv:1406.1038*.

Rudin, C. (2019). Stop explaining black box models for high stakes decisions and use interpretable models instead. *Nature*, 566(7744):484–488.

Russell, S. J. and Norvig, P. (2003). Artificial intelligence: a modern approach.

Serafini, L. and Garcez, A. d. (2016). Logic tensor networks: Deep learning and logical reasoning from data and knowledge. *arXiv preprint arXiv:1606.04422*.

Smolensky, P. (1988). On the proper treatment of connectionism. *Behavioral and brain sciences*, 11(1):1–74.