AutoML Optimized by Genetic Algorithm and Item Response Theory

Lucas F. F. Cardoso[1,2], **Vitor C A Santos**[1,2], **Regiane S. Kawasaki Francês**[1], **Ronnie C. O. Alves**[2]

[1]Faculdade de Computação – Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

[2]Instituto Tecnológico Vale (ITV)
Belém – PA – Brasil.

lucas.cardoso@icen.ufpa.br, kawasaki@ufpa.br

(vitor.cirilo.santos, ronnie.alves)@itv.org

Abstract. *Generating machine learning models automatically remains an area of much recent and challenging research. And there is still no definitive way capable of generating satisfactory results and that presents a low complexity. Therefore, in this work, the results of a proposed methodology are presented that uses the concepts of Item Response Theory and Genetic Algorithm to create an AutoML algorithm of the NAS type that is capable of generating a Neural Network competitive with the already known models. in literature. In the results obtained, it was possible to generate a competitive model compared to AutoKeras, but with a complexity lower than 5.5% of the total complexity of the model of AutoKeras itself.*

Resumo. *Gerar modelos de aprendizado de máquina automaticamente segue sendo uma área de muita pesquisa recente e desafiadora. E ainda não há uma forma definitiva capaz de gerar resultados satisfatórios e que apresente uma complexidade baixa. Com isso, neste trabalho, são apresentados os resultados de uma metodologia proposta que utiliza os conceitos da Teoria de Resposta ao Item e Algoritmo Genético para criar um algoritmo de AutoML do tipo NAS que seja capaz de gerar uma Rede Neural competitiva com os modelos já conhecidos na literatura. Nos resultados obtidos, foi possível gerar um modelo competitivo se comparado ao AutoKeras, porém com complexidade inferior a 5.5% da complexidade total do modelo do próprio AutoKeras.*

1. Introdução

Devido à grande variedade de aplicações, sistemas que utilizam aprendizado de máquina são cada vez mais comuns, sendo estes aplicados em diversos problemas do mundo real. Porém, à medida que os problemas a serem resolvidos se tornam mais complexos, também se torna mais difícil desenvolver um modelo de aprendizado de máquina capaz de resolver suficientemente bem uma dada tarefa. Comumente, grandes modelos de alto desempenho só são alcançados após muito tempo de tentativa e erro investido. Essas condições faz com que seja necessário que se possua um alto nível de conhecimento sobre aprendizado de máquina e o problema em questão para poder desenvolver um modelo apto, elevando ainda mais os custos de produção [He et al. 2021].

Diante disso, foram desenvolvidas soluções chamadas de AutoML que visam automatizar o processo de produção de um modelo e que diminua os custos envolvidos no desenvolvimento. De forma simplificada, o AutoML pode ser entendido como sendo um sistema capaz de gerar modelos de aprendizado de alto desempenho sem a necessidade de interação humana, comumente utilizando técnicas de computação evolucionária como Algoritmos Genéticos (AG). Dentre os diferentes tipos de AutoML existentes, um dos mais estudados é a busca por arquitetura neural (NAS) que é voltado principalmente para o aprendizado profundo. O NAS visa criar uma arquitetura neural que alcance o melhor resultado possível sobre um determinado problema [Ren et al. 2021].

Um estudo recente [D'Amour et al. 2022] aponta que é comum que modelos de aprendizado de máquina tenham apresentem desempenhos satisfatórios no treinamento e teste, porém não conseguem manter a mesma performance quando postos no mundo real. Os autores chamam essa condição de subespecificação do modelo. Apesar de este ser um termo novo e com poucos trabalhos que o utilizem, outros artigos já demonstraram que esse é um problema real. No estudo [Ribeiro et al. 2016] os autores demonstram que um modelo treinado para diferenciar lobos e Huskys apenas acertava as olhando para o fundo da imagem e não para os animais, de forma que se um Husky estivesse na neve este era classificado como lobo e vice-versa. O modelo apresentava acurácia de 0.8, e apesar da sua alta acurácia o mesmo foi considerado inapropriado para o mundo real, pois errava justamente os exemplos na qual os animais apareciam em ambientes trocados.

Percebe-se que o resultado de uma avaliação pelas métricas clássicas pode ser enganoso justamente por não se considerar a complexidade do dado. No geral, as métricas de avaliação são quantitativas, i.e., visam selecionar os modelos que mais acertam e não os que melhor aprendem, uma vez que a complexidade das instâncias não é levada em consideração. Todavia, a busca pelo modelo que melhor aprendeu ou generalizou ainda é uma tarefa complexa dentro do universo aprendizado de máquina. Diante de questões como essas, estudos recentes buscaram conhecimento em outras áreas para preencher essa lacuna. Uma nova abordagem é a aplicação de conceitos psicométricos que são comumente utilizadas para avaliar o aprendizado de indivíduos, entre eles está a Teoria de Resposta ao Item (TRI) [Martínez-Plumed et al. 2019, Cardoso et al. 2020].

A TRI é um conjunto de modelos matemáticos que visa calcular a probabilidade de um individuo acertar corretamente um item de um teste, considerando a dificuldade do item e a habilidade do indivíduo. Com uma analogia simples, é possível aplicar a TRI em aprendizado de máquina ao considerar os modelos como sendo os indivíduos, o dataset de teste como sendo o próprio teste e as instâncias como sendo os itens. Dessa forma, é possível avaliar qual foi o desempenho de um modelo considerando a própria complexidade das instâncias para mensurar com maior precisão qual a real habilidade do modelo diante a da variação de dificuldade das instâncias.

Segundo [D'Amour et al. 2022] para mitigar situações de subespecificação é preciso conhecimento profundo sobre as características do conjunto de dados utilizado e sobre a própria engenharia do algoritmo de aprendizado utilizado. Isso ocorre justamente durante a etapa de treinamento e ajuste fino do modelo almejado, sendo a interação humana decisiva para o sucesso. Entretanto, quando considerado o AutoML, cujo o foco está na automação e desempenho, os modelos gerados estão sujeitos a estarem subespecificados também. No trabalho de [Ribeiro et al. 2016] a condição de subespecificação só

pode ser observada após realização de uma análise visual da explicação das imagens que o modelo errou. Entretanto, não é trivial realizar tais análises de explicação se o dado for tabular ao invés de uma imagem. A TRI também pode ser usada para esse propósito, um estudo recente [Cardoso et al. 2022] utilizou a TRI para realizar a explicação de um modelo e apontar quais *features* podem aumentar a dificuldade de classificação e diminuir a confiança do modelo.

Diante do exposto, neste artigo, são apresentado os resultados do desenvolvimento de uma metodologia de AutoML do tipo NAS baseada em AG e nos resultados obtidos pela TRI, também chamado de ENAS (*Evolutionary Neural Architecture Search*) [Liu et al. 2021] por ser baseada em algoritmo do tipo evolucionário. A seção 2 explana sobre os conceitos da TRI e trabalhos relacionados; a seção 3 demonstra a metodologia utilizada para integrar AG e TRI; a seção 4 trás os resultados e discussões e a seção 5 conclui o trabalho.

2. Contexto

2.1. Teoria de Resposta ao Item

Considerando que um teste é composto por itens ou questões, para se avaliar o desempenho de um indivíduo sobre um teste tradicionalmente atribui-se uma pontuação igual a 1 para cada item respondido corretamente e atribui-se 0 para cada item respondido incorretamente. A soma das pontuações é considerada a nota final que normalmente varia de 0 até 10. No entanto, essa abordagem possui limitações para mensurar a real habilidade do indivíduo e isso se deve principalmente por essa ser uma abordagem quantitativa por avaliar o teste como um todo que apenas soma erros e acertos sem considerar a complexidade do item, semelhantemente de como é feito com as métricas clássicas de aprendizado de máquina. A TRI por sua vez, tem como elemento central o item e tem como principal foco ser capaz de estimar a probabilidade de um indivíduo responder corretamente cada item do teste, segundo a habilidade que o indivíduo j possui e os parâmetros que descrevem item i [Baker 2001, de Andrade et al. 2000].

A TRI possui três parâmetros principais que descrevem o item, são eles: b_i, que mensura a Dificuldade do item, de forma que quanto maior o seu valor mais difícil de ser respondido corretamente o item é; a_i é a Discriminação, mensura o quão bem o item serve para diferenciar respondentes de alta e baixa habilidade, de forma que indivíduos com níveis diferentes de habilidade podem apresentar chances de acerto muito próximas caso a discriminação do item seja baixa; c_i é a Adivinhação, mensura a menor probabilidade de acerto um item pode ter independente do indivíduo avaliado, é vista também como a chance de acerto casual [Baker 2001, de Andrade et al. 2000].

Enquanto os parâmetros de item permitem avaliar a complexidade do item segundo seus conceitos, a TRI dispõe da habilidade estimada θ_j como a variável que avalia o individuo. Para calcular a probabilidade de acerto, a TRI dispõe de seus modelos logísticos, sendo o mais comum o modelo de 3 parâmetros (3PL). Neste trabalho, é utilizado o modelo logístico de 1 parâmetro, também conhecido como modelo Rasch [Rasch 1960]. Este modelo logístico pode ser obtido a partir da simplificação do 3PL ao atribuir valores fixos para Discriminação ($a_i = 1$) e Adivinhação ($c_i = 0$). O modelo Rasch é dado pela seguinte fórmula:

$$P(U_{ij} = 1|\theta_j) = \frac{1}{1 + e^{-(\theta_j - b_i)}} \quad (1)$$

Por se tratar de uma forma mais robusta de avaliação psicométrica, a TRI é aplicada em diferentes testes realizados em todo mundo. Por exemplo, no Brasil a TRI é utilizada como metodologia base para avaliação da prova do ENEM que visa mensurar o nível de conhecimento dos, recém formados, alunos do ensino médio em todo o país [INEP 2012]. Além disso, segundo [INEP 2012] a TRI permite a comparação de resultados obtidos em diferentes provas e anos, o que permite medir com maior precisão o crescimento do desempenho obtido pelos estudantes. Neste trabalho, têm-se como objetivo utilizar essas características da TRI para medir o desempenho das Redes Neurais geradas pela metodologia proposta.

2.2. Trabalho Relacionado

De forma simplificada, apesar das diferentes estratégias existentes, o NAS pode ser entendido como sendo um clássico problema de otimização, onde o objetivo é otimizar uma estrutura de hyperparâmetros já definida, com foco na otimização dos pesos da rede, ou otimizar a própria estrutura de hyperparâmetros que será utilizada a fim de encontrar a melhor rede.

No momento da escrita deste artigo foi encontrado apenas um outro trabalho que utiliza da TRI para propor uma metodologia de AutoML e NAS [Neto et al. 2020]. No trabalho, os autores utilizam as características *instance-centric* da TRI para encontrar a melhor estrutura de hyperparâmetros de uma CNN (*Convolutional Neural Networks*) com foco na classificação de dados espectrais, ao combinar 18 valores de hiperparâmetros diferentes por meio da estratégia de *grid-search*. Apesar de utilizarem o modelo logístico 3PL, o trabalho foca nos parâmetros de dificuldade e discriminação em ordem de encontrar a CNN que apresenta o melhor desempenho sobre o conjunto de instâncias mais difíceis e mais discriminativas, dessa forma o resultado final será a CNN considerada mais habilidosa pela TRI.

Em comparação com o artigo citado, este trabalho difere principalmente na estratégia utilizada (AG ao invés de *grid-search*), no modelo logístico da TRI utilizado (modelo Rasch ao invés do 3PL), no tipo de NAS explorado (otimização dos pesos da rede ao invés dos hyperparâmetros) e no tipo de dados utilizado (dados tabulares ao invés de espectrais).

3. Materiais e Métodos

Conforme mencionado em seções anteriores, este trabalho tem como objetivo utilizar a capacidade evolucionária dos algoritmos genéticos para criar um algoritmo de AutoML do tipo NAS, onde os indivíduos são modelos de Redes Neurais e os genes os pesos da rede. A avaliação dos indivíduos será feita por meio do cálculo e conceitos da TRI, além disso os resultados da TRI também são utilizados para realizar a mutação adaptativa no AG. A Tabela 1 resume como é o funcionando do AG proposto para AutoML. Todo o AG foi desenvolvido utilizando a linguagem de programação Python, para implementar as Redes Neurais utilizou-se as bibliotecas Keras e TensorFlow e para calcular os estimadores da TRI foram usadas as bibliotecas Rpy2 e Ltm da linguagem R.

Table 1. Tabela de Parâmetros do AG.

Parâmetro	**Descrição**
Representação	Cada gene do cromossomo é um número tipo *float*
Cruzamento	Mutação Gaussiana
Probabilidade de Mutação	Mutação adaptativa pela TRI com queda exponencial
Seleção de Pais	Torneio
Seleção de Sobreviventes	Ficam só os filhos
Número de gerações máximo	Não foi determinado
Tamanho da população	20, 30 e 50
Inicialização	Aleatória
Critério de Parada	Parada por estagnação ou caso atinja um valor máximo
Fitness	O Fitness é calculado sobre a chance de acerto da TRI

Os indivíduos do AG são representados por um vetor de pesos que compõe a Rede Neural. Por se tratar de um trabalho exploratório e experimental, todas as redes são simples com apenas 3 camadas (entrada, oculta e saída). A quantidade de neurônios de cada rede é definido de acordo com o *dataset* que se deseja classificar. Dentro do AG todos os genes são valores gerados entre 0 e 1. Para utilizar esses valores como pesos de uma Rede Neural, são calculados limites superior e inferior de cada camada da rede utilizando a inicialização de Xavier [Castillo Camacho and Wang 2019]. Antes da execução do AG, o dataset escolhido é dividido em dados de treino e teste de forma que somente os dados de treino são utilizados no AG, os dados de teste são guardados para testes de desempenho posteriores. A Figura 1 apresenta o fluxograma da execução da metodologia proposta.

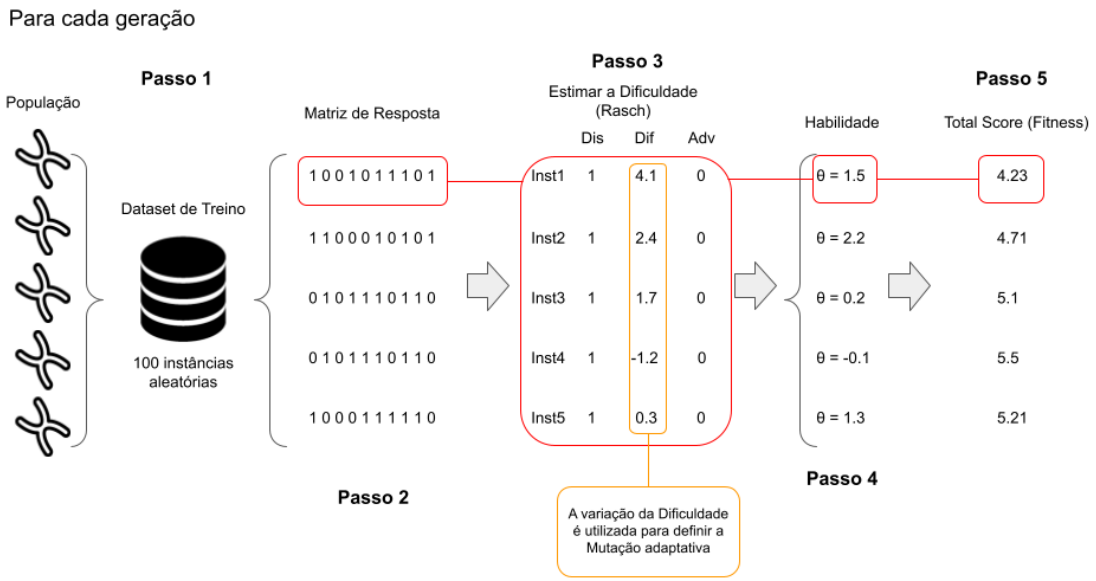

Figure 1. Fluxograma do AG baseado na TRI.

Como pode ser observado na Figura 1, nesse estudo é utilizado o modelo Rasch, onde apenas considera-se a dificuldade do item como parâmetro junto com a habilidade estimada do indivíduo. A metodologia proposta pode ser dividida em 5 passos principais:

1. Após a divisão do *dataset* em treino e teste, a cada geração do AG sorteia-se 100 instâncias aleatórias do conjunto de treino para serem classificadas pelos indivíduos. Essa etapa é realizada com o objetivo é avaliar os modelos utilizando conjuntos diferentes de instâncias, que podem ser mais fáceis ou difíceis pela TRI e dessa forma escolher os modelos que melhor aprenderam/generalizaram para seguirem adiante.
2. Cada modelo da população é colocado para classificar as instâncias sorteadas e depois as respostas são dispostas na Matriz de Respostas, onde 1 indica acerto e 0 indica erro.
3. A Matriz de Resposta é então utilizada para calcular a dificuldade das instâncias sorteadas.
4. Com o valor de dificuldade estimado para cada instâncias e os valores de discriminação e adivinhação constantes, por se usar o modelo Rasch. Pode-se então estimar a habilidade de cada modelo da geração.
5. Por conseguinte, calcula-se a probabilidade de acerto para cada instância de treino e então obtém-se o Total Score que é a soma da probabilidade de acerto sempre que o modelo acertar e a subtração da probabilidade de erro sempre que o modelo errar. O valor máximo que pode ser obtido pelo score é de 100, caso o modelo tenha probabilidade de acerto máxima em todas as instâncias.

Dessa forma, busca-se tornar mais evidente a separação de modelos mais habilidosos dos menos habilidosos e também para não classificar o indivíduo somente pela quantidade de acertos, mas também pela qualidade dos acertos. Além disso, pode-se usar os níveis de dificuldade obtidos para entender como está o desempenho total da população. Para isso, a cada geração é calculado a média da dificuldade normalizada entre 0 e 4, então verifica-se se essa média aumentou ou diminuiu. Caso tenha aumentado, então incrementa-se a probabilidade de mutação e o sigma (intensidade da mutação) e potencializa o *exploration*, caso contrário continua o decaimento exponencial das duas variáveis e potencializa o *exploitation*. Os valores de dificuldade são normalizados, pois teoricamente podem variar entre $-\infty$ até $+\infty$, assim mitiga-se a ocorrência de *outliers* de dificuldade.

4. Resultados e Discussão

Nesta seção serão apresentados os resultados obtidos a partir de um experimento como estudo de caso. Para isso, foi escolhido o *dataset Banknote-Authentication* [1], obtido pela plataforma OpenML [Vanschoren et al. 2014]. O dataset mencionado foi escolhido por se tratar de um conjunto de dados simples e por possui apenas com 4 *features* e nenhum dado faltoso. O que o torna uma escolha adequada para testar a metodologia proposta. O *dataset* escolhido foi divido em 70% para treino e 30% para teste. O AG então foi executado 5 vezes para cada população para que pudesse ser feita a avaliação de desempenho médio para cada população. Para execução dos experimentos foi utilizado a plataforma Google Colab na sua condição padrão. Não foi feito nenhum ajuste nos parâmetros iniciais do AG durante as execuções. Manteve-se: $sigma = 0.8$, $mutacao = 0.9$, $decaimento = 0.0005$ e $cruzamento = 0.9$ para cada execução[2].

[1]Acesso ao *dataset Banknote-Authentication*: `https://www.openml.org/search?type=data&sort=runs&id=1462&status=active`

[2]Notebook com código fonte e resultados: `https://github.com/LucasFerraroCardoso/TRI_AG`

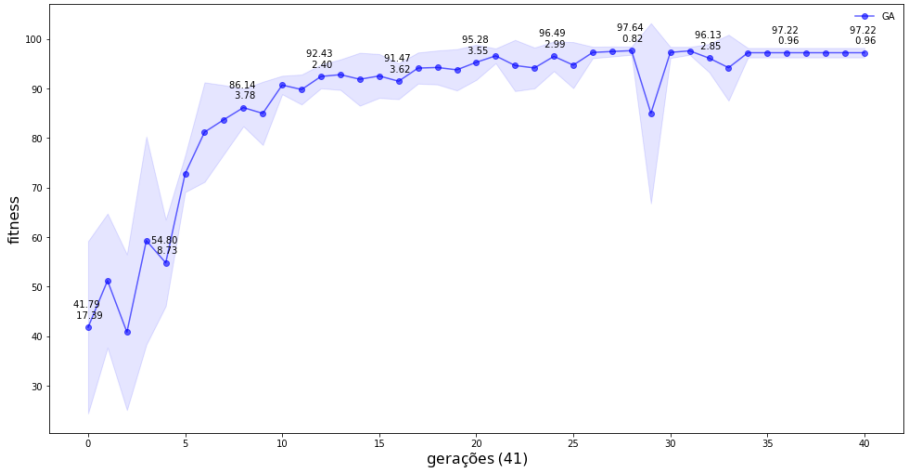

Figure 2. Gráfico de Convergência para população de 50 indivíduos.

Como exemplo, a Figura 2 apresenta o gráfico de convergência obtido para a população com 50 indivíduos. Como já era esperado, à medida que o número de indivíduos por população aumenta a quantidade de gerações necessárias para o AG atingir a convergência diminui. Entretanto, isso não influenciou grandemente nos modelos de Rede Neural que foram obtidos ao final da execução do AG, foi observado que o desempenho médio final manteve-se muito próximo para todas as populações. A Tabela 2 apresenta os valores finais de média de *fitness* obtidos.

Table 2. Resultados obtidos pelo AG para o *dataset Banknote-Authentication*.

População	Média do Fitness	Desvio Padrão do Fitness
20	97.2	1.6
30	96.8	1.6
50	97.22	0.95

Como pode ser observado na Tabela 2, a diferença entre as médias de *fitness* das 3 populações foi menor que 0.5, o que indica que mesmo com poucos indivíduos ainda é possível alcançar resultados próximos do máximo de 100. É importante destacar que esse desempenho de *fitness* foi obtido com o *dataset* de treino. Para comparar se os modelos gerados de fato são competitivos, foi utilizado o *dataset* de teste.

Para execução do AG, duas redes diferentes são salvas: o melhor modelo da geração final, antes da condição de parada; e o melhor modelo (maior valor de *fitness*) obtido dentre todas as gerações. Essa abordagem foi feita, pois durante os experimentos, notou-se que alguns modelos conseguiam atingir valores de *fitness* mais altos que a geração final. Para verificar qual dos modelos é melhor, todas as redes são testadas utilizando o *dataset* reservado para teste para se obter a acurácia e então é feito o cálculo da média e desvio padrão por população.

Como pode ser observado na Tabela 3 tanto a média quanto o desvio padrão da acurácia obtida pelos modelos finais do AG são melhores que as redes com maior *fitness* calculado, para qualquer população. O que explica esse fato é a natureza do experimento, onde a cada geração são sorteadas 100 instâncias diferentes do *dataset* de treino para

Table 3. Resultados dos melhores modelos com o *dataset* de teste.

	Modelo de maior *fitness*		Melhor modelo Final	
População	Média da Acc	D. Padrão da Acc	Média da Acc	D. Padrão da Acc
20	0.972	0.024	0.979	0.006
30	0.983	0.008	0.987	0.005
50	0.971	0.011	0.978	0.009

calcular o *fitness*, dessa forma é possível que um modelo tenha obtido um valor de *fitness* mais alto, pois estava diante de um conjunto de instâncias mais fáceis. Como pode ser visto na Figura 3, que compara a média de dificuldade obtida para os conjuntos de instâncias que foram sorteados a cada geração para as populações de 20 e 50. Na Figura 3 optou-se por não normalizar a dificuldade, para que possam ser observado a variação real dos valores obtidos pela TRI.

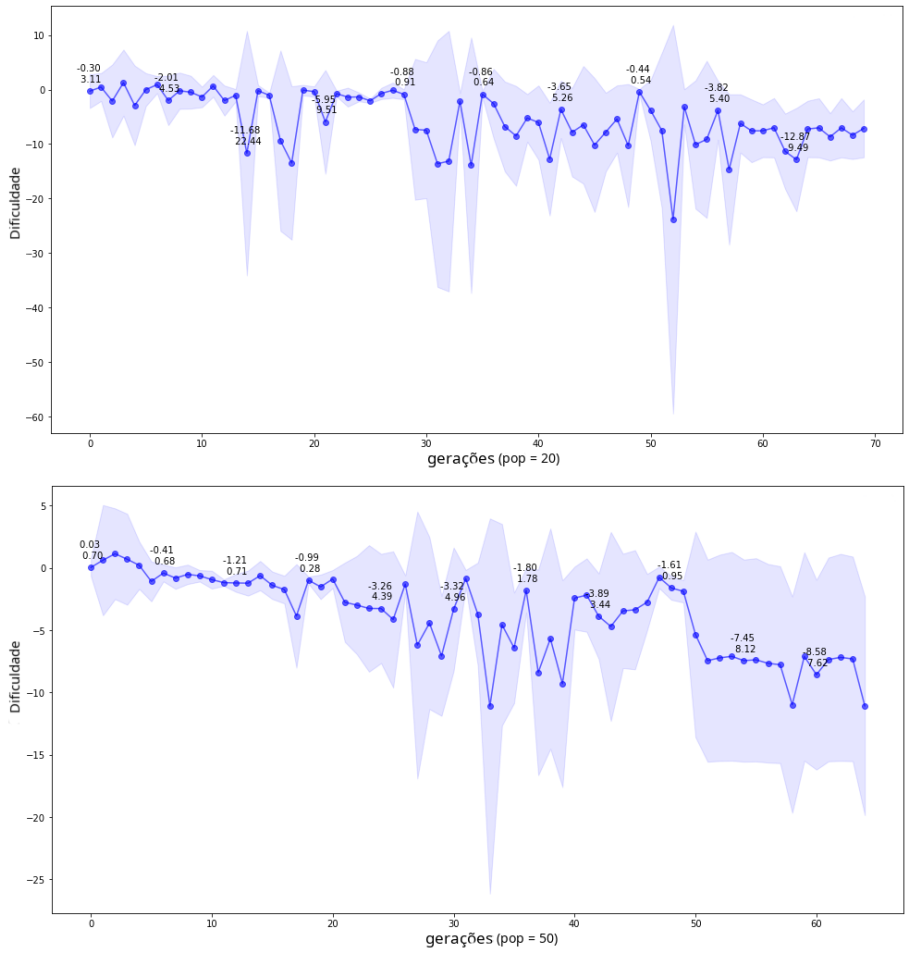

Figure 3. Comparação entre a dificuldade média para entre as gerações da população 20 e 50.

Ambos os gráficos apresentam uma tendência de queda no valor médio da dificuldade das instâncias. Isso já era esperado, pois entende-se que a cada geração os modelos devem ficar mais proficientes, logo as instâncias do *dataset* tende a ficarem mais fáceis. A ocorrência de picos e vales de dificuldade acontece devido a aleatoriedade das instâncias

selecionadas. Os gráfico de dificuldade (ver Figura 3) pode também ser entendido como a curva de aprendizado médio dos modelos gerados pelo AG. Um trabalho futuro, seria utilizar a informação da dificuldade das instâncias para comparar o poder de classificação do modelo gerado com o de outros modelos gerados por outras técnicas de AutoML, utilizando apenas as instâncias mais difíceis por exemplo.

Como este trabalho visa testar uma metodologia de AutoML, foi feita então a comparação das redes obtidas com o AutoKeras [Jin et al. 2019], um dos sistemas mais consolidados de NAS. Para isso, os 5 melhores modelos gerados foram avaliados utilizando o *dataset* de teste e obtida a média das acurácias, o AutoKeras foi executado 5 vezes também em suas configurações padrão. Na comparação direta o AutoKeras foi o vencedor, pois obteve uma média 0.993 de acurácia enquanto a metodologia proposta obteve 0.981. Apesar de não alcançar o mesmo nível de acerto, nota-se que a metodologia proposta mantêm-se competitiva com desempenho muito próximo. Entretanto, é importante destacar que a rede gerada pelo AutoKeras possui o total de 1258 parâmetros, enquanto todos os modelos gerados pelo AG possem apenas 69 parâmetros que corresponde a apenas 5.5% do total de parâmetros do modelo do AutoKeras. Isso se deve pelo fato de se utilizar uma estrutura simples de rede com apenas três camadas e pelo número de nós serem definidos segundo a quantidade de *features* do conjunto de dados utilizados. O AutoKeras por sua vez gerou uma rede com 9 camadas de diferentes tipos, como *Dense*, *ReLu*, *Normalization*, etc. Apesar do desempenho menos preciso, tais condições tornam a rede gerada pelo AG mais fácil de ser explicada e interpretada.

5. Considerações Finais

Este trabalho apresentou uma proposta de metologia de AutoML do tipo NAS baseada em Algoritmo Genético e Teoria de Resposta ao Item. Os resultados mostram-se promissores, não só pelo desempenho obtido, mas também pelas novas informações que podem ser incorporadas na avaliação dos modelos. Além disso, o modelo gerado pela metodologia proposta conseguiu alcançar um desempenho médio quase ótimo e com uma Rede Neural muito mais simples. Visto que hoje há uma necessidade cada vez maior de construir modelos e sistemas inteligentes mais transparentes e explicáveis. Além disso, pelo fato da metodologia proposta utilizar a TRI ainda é possível utilizar os parâmetros de item para entender a complexidade do conjunto de dados utilizado e entender como o modelo se comporta diretamente para cada instância. Conforme mencionado na seção 1 um trabalho recente utilizou os estimadores da TRI para indicar a confiabilidade de um modelo e a correlação entre as *features* do dataset e a dificuldade da instância. Diante disso, entende-se que um modelo gerado pela metodologia proposta também poderia utilizar a TRI para apontar em quais instâncias o modelo é mais confiável. Apesar dos resultados ainda serem preliminares, entende-se que a união de AutoML e TRI pode vir a gerar bons frutos.

References

Baker, F. B. (2001). *The basics of item response theory*. ERIC.

Cardoso, L. F., de S. Ribeiro, J., Santos, V. C. A., Silva, R. L., Mota, M. P., Prudêncio, R. B., and Alves, R. C. (2022). Explanation-by-example based on item response theory. In *Intelligent Systems: 11th Brazilian Conference, BRACIS 2022, Campinas, Brazil, November 28–December 1, 2022, Proceedings, Part I*, pages 283–297. Springer.

Cardoso, L. F., Santos, V. C., Francês, R. S. K., Prudêncio, R. B., and Alves, R. C. (2020). Decoding machine learning benchmarks. In *Brazilian Conference on Intelligent Systems*, pages 412–425. Springer.

Castillo Camacho, I. and Wang, K. (2019). A simple and effective initialization of cnn for forensics of image processing operations. In *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, pages 107–112.

D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., Chen, C., Deaton, J., Eisenstein, J., Hoffman, M. D., et al. (2022). Underspecification presents challenges for credibility in modern machine learning. *The Journal of Machine Learning Research*, 23(1):10237–10297.

de Andrade, D. F., Tavares, H. R., and da Cunha Valle, R. (2000). Teoria da resposta ao item: conceitos e aplicações. *ABE, Sao Paulo*.

He, X., Zhao, K., and Chu, X. (2021). Automl: A survey of the state-of-the-art. *Knowledge-Based Systems*, 212:106622.

INEP (2012). *NOTA TÉCNICA: Teoria de Resposta ao Item*.

Jin, H., Song, Q., and Hu, X. (2019). Auto-keras: An efficient neural architecture search system. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1946–1956.

Liu, Y., Sun, Y., Xue, B., Zhang, M., Yen, G. G., and Tan, K. C. (2021). A survey on evolutionary neural architecture search. *IEEE transactions on neural networks and learning systems*.

Martínez-Plumed, F., Prudêncio, R. B., Martínez-Usó, A., and Hernández-Orallo, J. (2019). Item response theory in ai: Analysing machine learning classifiers at the instance level. *Artificial intelligence*, 271:18–42.

Neto, H. A., Alves, R. C., and Campos, S. V. (2020). Nasirt: Automl based learning with instance-level complexity information. *arXiv preprint arXiv:2008.11846*.

Rasch, G. (1960). Studies in mathematical psychology: I. probabilistic models for some intelligence and attainment tests.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Chen, X., and Wang, X. (2021). A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144.

Vanschoren, J., Van Rijn, J. N., Bischl, B., and Torgo, L. (2014). Openml: networked science in machine learning. *ACM SIGKDD Explorations Newsletter*, 15(2):49–60.