Ministério da Educação
Universidade Tecnológica Federal do Paraná
Pró-Reitoria de Pesquisa e Pós-Graduação

RELATÓRIO FINAL DE ATIVIDADES DE INICIAÇÃO CIENTÍFICA

Aplicação de Algoritmos de Aprendizagem de Máquina para a Redução de Dimensionalidade em Bases de Dados

Redução de Dimensionalidade em Base de Dados de Microarranjos

Rafael Felipe Tasaka de Melo
Voluntário
Bacharelado em Ciência da Computação
Data de ingresso no programa: 08/2019
Prof.ª Dra Helyane Bronoski Borges

Área do Conhecimento: 1.03.00.00-7 Ciência da Computação

CAMPUS PONTA GROSSA, 2020

RAFAEL FELIPE TASAKA DE MELO

HELYANE BRONOSKI BORGERS

REDUÇÃO DE DIMENSIONALIDADE EM BASE DE DADOS DE MICROARRANJOS

Relatório de Pesquisa do Programa de Iniciação Científica da Universidade Tecnológica Federal do Paraná.

PONTA GROSSA, 2020

SUMÁRIO

INTRODUÇÃO

Muitos problemas de Aprendizagem de Máquina (AM) utilizam milhares de atributos para o treinamento do algoritmo. Utilizar essa quantidade de atributos faz com que, além de deixar os treinos lentos, fique difícil encontrar um padrão entre os dados para realizar uma possível classificação deles.

Algumas dessas bases possuem, também, a característica de muitos atributos e poucos exemplares. Muitas vezes, isso se deve pela dificuldade em coletar amostras em largas quantidades. Esse problema é conhecido como maldição da dimensionalidade [1]. Por conta dessa característica muitos dos métodos de AM não conseguem criar um modelo de classificação eficiente o suficiente para prever exemplares futuros por conta da dificuldade em generalizar tamanha quantidade de atributos.

Visando diminuir o problema da maldição da dimensionalidade e outros problemas gerados pela alta dimensionalidade, técnicas de redução de dimensionalidade podem ser aplicadas para retirar da base de dados atributos irrelevantes e/ou redundantes. Além disso, representar uma grande quantidade de atributos em um número reduzido de dados pode ajudar a biólogos a identificarem quais genes estão diretamente ligados aos problemas neles identificados.

As técnicas de redução de dimensionalidade podem ser divididas em duas abordagens: extração de atributos e seleção de atributos. Métodos de seleção de atributos selecionam os atributos mais relevantes sem alterar a base de dados, enquanto que os métodos de extração de atributos modificam a base de dados para representar os dados, como por exemplo, o método PCA (Principal Component Analysis) e o LDA (Linear Discriminant Analysis) [1].

Contudo, a projeção feita por esses métodos não se importa com as relações entre as bases originais e as bases reduzidas, fazendo com que uma futura representação não represente bem seus dados originais.

Como alternativa para a tarefa de redução, pode-se utilizar a aprendizagem profunda [1] por meio de uma rede autocodificadora [1] que pode realizar a extração de atributos em suas camadas ocultas.

Utilizar redes autocodificadoras por si só pode apresentar problemas, uma vez em que ela foca em reduzir dados e, depois, reconstrui-los, criando-se assim uma relação mais profunda entre entrada e saída, e não entrada e redução [2].

O método proposto busca reduzir a dimensionalidade da base através da extração de atributos onde, dado um conjunto de atributos X, busca-se a criação de um novo conjunto de atributos Y, que são mais expressivos e melhor representem a variedade dos atributos originais. A estrutura responsável por essa redução se chama codificador.

O codificador é uma estrutura interna da autocodificadora que consiste nas camadas internas entre a camada de entrada, composta pelo número total de atributos a serem reduzidos, e a menor camada da rede (também chamada de *bottleneck*), composta pelo número desejado de redução.

Por conta da importância da estrutura codificadora para a redução da base, criando assim a base reduzida na rede autocodificadora, o método proposto realiza um pré treinamento no codificador utilizando uma rede neural multicamadas (MLP) [2] para classificação e, com os pesos utilizados nesse treinamento, é realizado um novo treinamento com o decodificador, criando assim a autocodificadora. Esse pré-treinamento faz com que o codificador crie uma relação entre base a original e sua classe bem como com o decodificador, para que esse também esteja otimizado para uma possível reconstrução dos atributos.

Logo, esse projeto tem como objetivo propor o método FEA-PTC (*Feature Extraction using Autoencoder: Pre-Training with Classification*) utilizando bases de dados de microarranjo para a realização dos experimentos.

METODOLOGIA

Método Proposto. O Método proposto denominado de FEA-PTC possui uma rede autocodificadora, a qual é composta por três estruturas: codificador, classificador e decodificador.

O codificador possui um mapeamento de entrada X e uma representação reduzida Y definida na equação (1).

$$Y = f(X) = s_f(WX + b_x) \tag{1}$$

em que s_f é uma função de ativação para a rede, W uma matriz de pesos para parametrização e um vetor bias $b \in R^n$

O classificador recebe a representação reduzida Y para a classificação y definida na equação (2).

$$y = g(Y) = s_g(WY + b_y) \tag{2}$$

em que s_g é uma função de ativação para a rede , W uma matriz de pesos para parametrização e um vetor bias $b \in R^n$

O decodificador é responsável por mapear a representação Y para fazer a reconstrução Xi definido na equação (3).

$$\text{Xi} = h(Y) = s_h(W\prime Y + b_y) \tag{3}$$

em que s_h é uma funçao de ativação para a rede , W' uma matriz de pesos para parametrização e um vetor bias $b \in R^n$.

A representação da rede autocodificadora é apresentada na figura 1. A rede é totalmente conectada, possuindo uma camada de entrada e duas camadas de saída. Para determinar o número de neurônios em cada camada e a quantidade de camadas ocultas para a estrutura autocodificadora, utilizou-se o critério de Rafael com adaptações para comportar base de dados de microarranjo.

Segundo Siqueira [3], primeiro, é necessário determinar um Fator de Redução (*Fr*) e uma Porcentagem de Redução *(Pr*) definida na equação (4).

$$Fr = Pr/6 \tag{4}$$

em que *Pr* é o percentual de redução desejado.

O número de neurônios em cada camada oculta é apresentado na tabela da Figura 1.

Foram utilizadas três funções de ativações, sendo elas: ReLU (*Rectified Linear Unit*) para as camadas ocultas, Softmax para a camada de saída do classificador e Sigmoid para a camada de saída do autocodificador [1].

Como função de otimização foi utilizada um gradiente estocástico [1]. Para cada estrutura (autocodificadora e classificadora). A tabela 1 mostra a definição de cada parâmetro.

Ainda, conforme proposto por Hinton [4], para evitar *overfiting*, utilizou-se *dropout* em diversas camadas ocultas. Além disso se adicionou erro Gaussiano - fazendo com que a rede "aprenda" somente atributos mais relevantes, e também evitar que a rede crie *outputs* somente fazendo uma cópia dos *inputs* – tornando a rede uma *denoising autoencoder* [5].

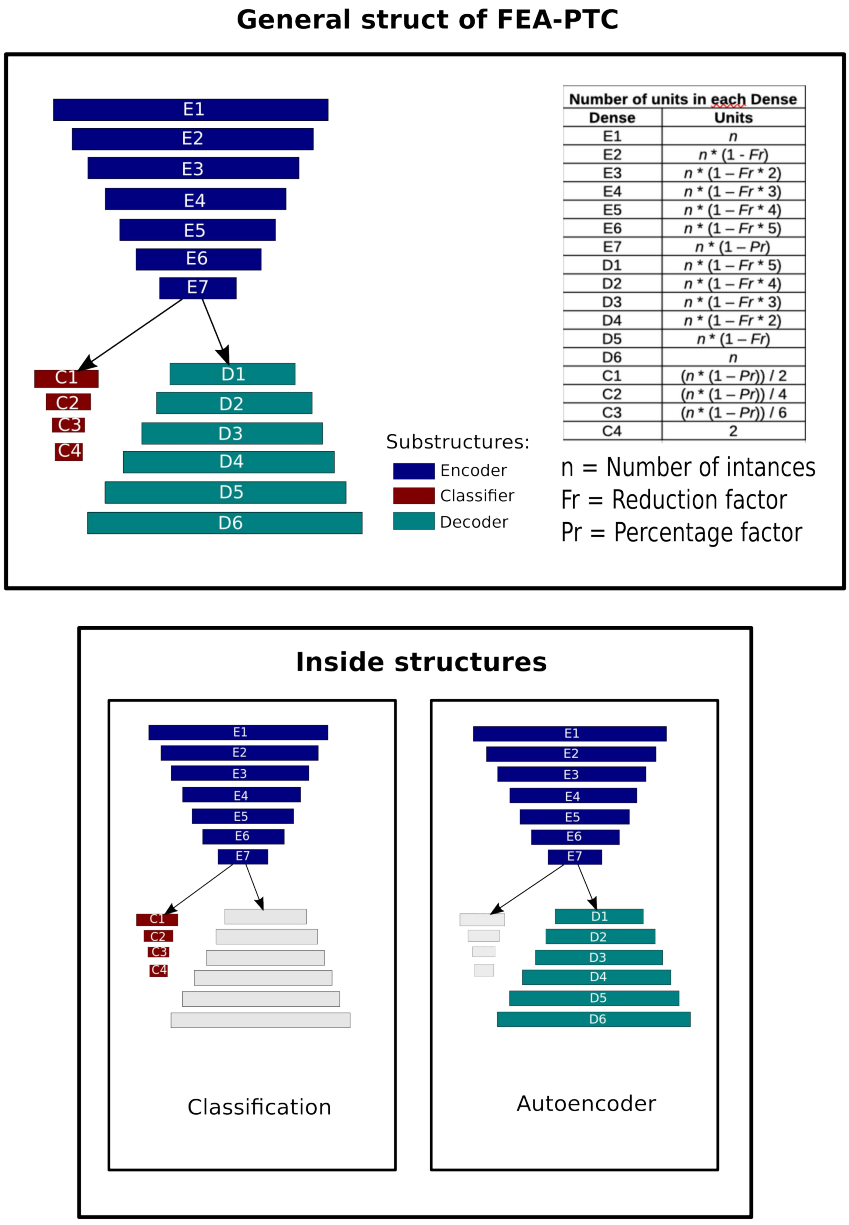

Figura 1. Estrutura geral do modelo FEA-PTC

Tabela 1. Comparação entre as estruturas das redes.

Rede classificadora X Rede autocodificadora		
Parâmetros	**Classificadora**	**Autocodificadora**
Learning Rate	0.05	0.07
Avaliação de desempenho	Acurácia	MSE (*Mean Squared Error*)
Número de épocas	50	100
Função de ativação na camada de saída	Softmax	Sigmoid

Metodologia. A metodologia utilizada para a realização dos experimentos com o método proposto é ilustrada na figura 2.

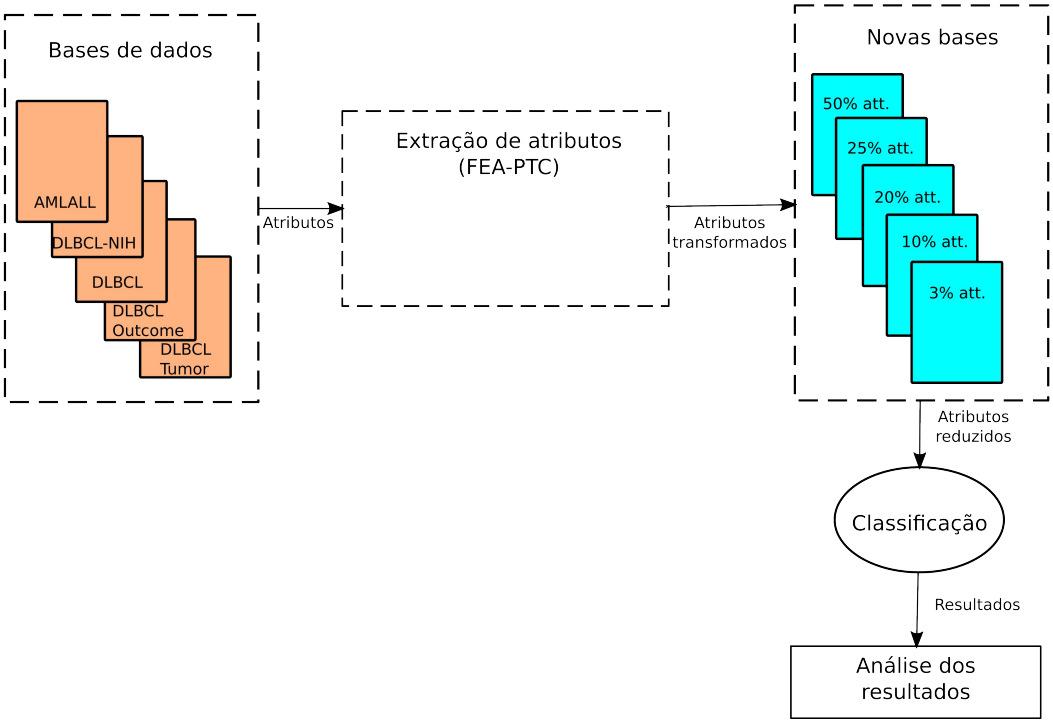

Figura 2. Representação geral da metodologia implementada

Base de dados. Para a realização de experimentos com o método proposto foram utilizadas 5 (cinco) base de dados de microarranjos [6]. Umas das características desses tipos de dados é que são formados por uma grande quantidade de atributos, na ordem de milhares, e relativamente poucas amostras. As características dessas bases de dados mostradas na tabela 2.

Para a realização do treinamento da rede utilizou-se o método *holdout* [7]. Como método de imputação dos valores faltantes [8], foi utilizado a imputação pela média aritmética, ou seja, o valor a ser atribuído é a média aritmética dos valores de todos os exemplares para determinado atributo.

Tabela 2. Quantidade de atributos por base de dados.

Bases de Dados	Quantidade de atributos	Quantidade de exemplos
AMLALL	7129	72
DLBCL-NIH	7399	240
DLBCL	4026	47
DLBCLOutcome	7129	58
DLBCLTumor	7129	77

Extração de Atributos. Essa etapa foi descrita no Método Proposto. Ressalta-se que ao final dessa etapa têm-se os conjuntos de dados reduzidos pelo método FEA-PTC.

Subconjuntos de Atributos. Para a redução dos atributos fez-se experimentos reduzindo a quantidade de atributos em 3%, 10%, 20%, 25% e 50%. Esses valores foram selecionados para posterior comparação com os resultados obtidos em Borges [9].

Classificação. Para a avaliação das bases reduzidas, as bases originais são divididas em treinamento e teste e ambos são reduzidos no modelo gerado pela rede. Em seguida são aplicados os algoritmos de classificação K-NN (*K-Nearest Neighbor*) (com K igual a 1, 3, 5 e 7), SVM (*Support Vector Machine*), Naive Bayes e Árvore de Decisão [1]. Esse processo de repete vinte vezes para cada base, obtendo-se assim uma média de acurácias bem como seus desvios padrão.

RESULTADOS E DISCUSSÕES

Esta seção mostra os resultados do treinamento de cada rede para cada base, bem como a avaliação das bases reduzidas.

A tabela 3 apresenta os resultados obtidos na base de dados AML/ALL conforme os algoritmos de classificação utilizado e o percentual de redução atributos. Essa base de dados obteve as melhores acurácias dentre todas as outras.

Tabela 3. Resultado na base de dados AML/ALL

	Percentual de Redução de Atributos				
Algoritmo	**3%**	**10%**	**20%**	**25%**	**50%**
k-NN (k=1)	0.9520 +- 0.0438	0.9708 +- 0.0465	0.9583 +- 0.0355	0.9666 +- 0.0386	0.9583 +- 0.0383
k-NN (k=3)	0.9458 +- 0.0438	0.9604 +- 0.0465	0.9437 +- 0.0355	0.9666 +- 0.0386	0.9604 +- 0.0383
k-NN (k=5)	0.9354 +- 0.0465	0.9666 +- 0.0386	0.9458 +- 0.0418	0.9583 +- 0.0510	0.9520 +- 0.0355
k-NN (k=7)	0.9270 +- 0.0454	0.9583 +- 0.0475	0.9354 +- 0.0383	0.9541 +- 0.0572	0.9333 +- 0.0482
SVM	0.95 +- 0.0408	0.9770 +- 0.0207	0.9583 +- 0.0348	0.9708 +- 0.0438	0.9312 +- 0.0422
NAIVE BAYES	0.95 +- 0.0338	0.9645 +- 0.0422	0.9625 +- 0.0454	0.9708 +- 0.0325	0.9625 +- 0.0346
DECISION TREE	0.9208 +- 0.0572	0.95 +- 0.0485	0.9437 +- 0.0514	0.9583 +- 0.0527	0.9458 +- 0.0698

A tabela 4 apresenta os resultados obtidos na base de dados DLBCL-NIH conforme os algoritmos de classificação utilizado e o percentual de redução atributos. Dentre todas as reduções, reduzir a 50% apresentou as piores acurácias dentre todas as outras na mesma base.

Tabela 4. Resultado na base de dados DLBCL-NIH

DLBCL-NIH					
Algoritmo	**Número de atributos**				
	3,00%	10,00%	20,00%	25,00%	50,00%
k-NN (k=1)	0.8387 +- 0.0306	0.8256 +- 0.0424	0.8193 +- 0.0375	0.8018 +- 0.0321	0.7293 +- 0.0409
k-NN (k=3)	0.8662 +- 0.0306	0.8456 +- 0.0424	0.8493 +- 0.0375	0.8268 +- 0.0321	0.7443 +- 0.0409
k-NN (k=5)	0.8768 +- 0.0283	0.8631 +- 0.0441	0.8487 +- 0.0416	0.8243 +- 0.0371	0.7106 +- 0.0413
k-NN (k=7)	0.8737 +- 0.0298	0.8706 +- 0.0407	0.8468 +- 0.0328	0.8237 +- 0.0305	0.6868 +- 0.0336
SVM	0.8618 +- 0.0294	0.8693 +- 0.0441	0.8631 +- 0.0257	0.8425 +- 0.0319	0.6662 +- 0.0524
NAIVE BAYES	0.8656 +- 0.0279	0.825 +- 0.0472	0.8225 +- 0.0475	0.8056 +- 0.0411	0.6581 +- 0.0553
DECISION TREE	0.84 +- 0.0367	0.8175 +- 0.0398	0.8337 +- 0.0431	0.8181 +- 0.0400	0.7331 +- 0.0437

A tabela 5 apresenta os resultados obtidos na base de dados DLBCL conforme os algoritmos de classificação utilizado e o percentual de redução atributos. A base se apresentou com boas acurácias - sendo muito idêntica à base AML/ALL, porém apresentou desvios padrões maiores.

Tabela 5. Resultado na base de dados DLBCL

DLBCL					
Algoritmo	**Número de atributos**				
	3,00%	10,00%	20,00%	25,00%	50,00%
k-NN (k=1)	0.9375 +- 0.0612	0.9437 +- 0.0453	0.9093 +- 0.0633	0.925 +- 0.0681	0.9312 +- 0.0559
k-NN (k=3)	0.95 +- 0.0612	0.9468 +- 0.0453	0.8968 +- 0.0633	0.9218 +- 0.0681	0.9375 +- 0.0559
k-NN (k=5)	0.9437 +- 0.0652	0.9312 +- 0.0589	0.8812 +- 0.0589	0.9312 +- 0.0621	0.9375 +- 0.0441
k-NN (k=7)	0.9437 +- 0.0652	0.9218 +- 0.0651	0.9 +- 0.05	0.9437 +- 0.0621	0.9406 +- 0.0418
SVM	0.9406 +- 0.0669	0.9406 +- 0.0540	0.9093 +- 0.0462	0.9343 +- 0.0639	0.9562 +- 0.0286
NAIVE BAYES	0.9625 +- 0.0414	0.9406 +- 0.0608	0.9031 +- 0.0502	0.9437 +- 0.0652	0.9281 +- 0.0453
DECISION TREE	0.95 +- 0.0612	0.9156 +- 0.0663	0.8718 +- 0.0826	0.9062 +- 0.0640	0.9281 +- 0.0495

A tabela 6 apresenta os resultados obtidos na base de dados DLBCL-Outcome conforme os algoritmos de classificação utilizado e o percentual de redução atributos. Esta base de dados foi a que apresentou os piores resultados se comparada com as demais, ela também apresenta a pior acurácia obtida dentre todos os resultados do estudo (somente 42% de acurácia para 50% dos atributos para o Algoritmo 7NN)

Tabela 6. Resultado na base de dados DLBCL-Outcome

DLBCLOutcome					
Algoritmo	**Número de atributos**				
	3,00%	10,00%	20,00%	25,00%	50,00%
k-NN (k=1)	0.8175 +- 0.0634	0.8075 +- 0.0749	0.815 +- 0.0657	0.7775 +- 0.0739	0.5325 +- 0.1145
k-NN (k=3)	0.8150 +- 0.0634	0.8475 +- 0.0749	0.8425 +- 0.0657	0.8375 +- 0.0739	0.4750 +- 0.1145
k-NN (k=5)	0.8375 +- 0.0567	0.845 +- 0.0522	0.8300 +- 0.0640	0.855 +- 0.0589	0.4475 +- 0.1089
k-NN (k=7)	0.84 +- 0.0624	0.845 +- 0.0567	0.8150 +- 0.0549	0.855 +- 0.0630	0.4275 +- 0.1089
SVM	0.83 +- 0.0599	0.85 +- 0.0632	0.825 +- 0.0622	0.8475 +- 0.0621	0.455 +- 0.1105
NAIVE BAYES	0.8075 +- 0.0675	0.8575 +- 0.0637	0.8225 +- 0.0558	0.8375 +- 0.0668	0.5275 +- 0.1166
DECISION TREE	0.785 +- 0.0988	0.84 +- 0.0699	0.8 +- 0.0836	0.8150 +- 0.0807	0.5 +- 0.0880

A tabela 7 apresenta os resultados obtidos na base de dados DLBCL-Tumor conforme os algoritmos de classificação utilizado e o percentual de redução atributos. A base se apresentou satisfatória para as acurácias, porém em diversas situações apresentou desvios padrões maiores que as outras bases.

Tabela 7. Resultado na base de dados DLBCL-Tumor

DLBCLTumor					
Algoritmo	**Número de atributos**				
	3,00%	10,00%	20,00%	25,00%	50,00%
k-NN (k=1)	0.9096 +- 0.0445	0.9115 +- 0.0403	0.875 +- 0.0638	0.9326 +- 0.0514	0.9173 +- 0.0547
k-NN (k=3)	0.9076 +- 0.0445	0.9230 +- 0.0403	0.8769 +- 0.0638	0.9134 +- 0.0514	0.8903 +- 0.0547
k-NN (k=5)	0.8961 +- 0.0456	0.8826 +- 0.0576	0.8692 +- 0.0681	0.9 +- 0.0428	0.8557 +- 0.0568
k-NN (k=7)	0.8903 +- 0.0443	0.8634 +- 0.0693	0.8346 +- 0.0852	0.8980 +- 0.0574	0.8365 +- 0.0728
SVM	0.9057 +- 0.0462	0.9076 +- 0.0636	0.8788 +- 0.0712	0.9307 +- 0.0510	0.825 +- 0.0937
NAIVE BAYES	0.9346 +- 0.0472	0.9134 +- 0.0542	0.8961 +- 0.0456	0.9461 +- 0.0392	0.9384 +- 0.0372
DECISION TREE	0.9076 +- 0.0713	0.9057 +- 0.0724	0.8826 +- 0.0478	0.9596 +- 0.0462	0.9230 +- 0.0530

A figura 3 mostra a evolução da acurácia durante o treino da estrutura interna "Classificador" para 3% dos atributos. A figura 4 apresenta uma relação entre a quantidade de épocas usadas pela rede neural e o MSE para a reconstrução da base para 3% dos atributos. Nota-se que há uma relação entre a acurácia e a redução: quanto melhor a acurácia, menor o erro quadrático na reconstrução. Esta relação irá se repetir para as outras reduções.

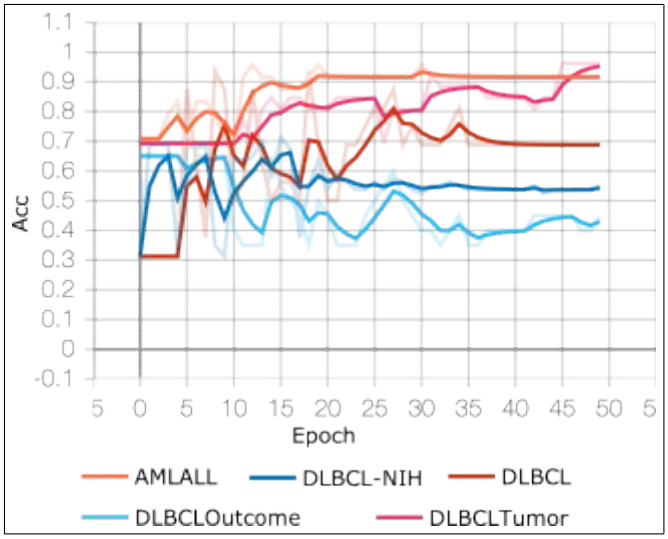

Figura 3. Acurácia da rede classificadora para 3% dos atributos

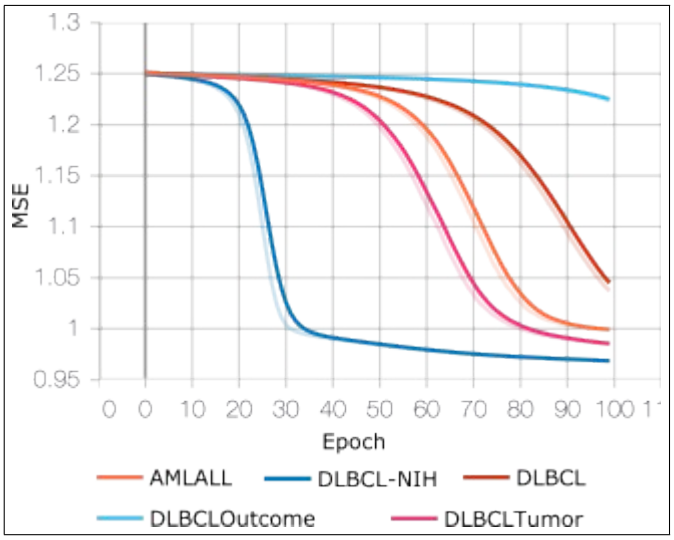

Figura 4. Erro quadrático da rede autocodificadora para 3% dos atributos

A figura 5 mostra a evolução da acurácia durante o treino da estrutura interna "Classificador" para 10% dos atributos. A figura 6 apresenta uma relação entre a quantidade de épocas usadas pela rede neural e o MSE para a reconstrução da base para 10% dos atributos. Assim como nas tabelas, os resultados entre 3% e 10% se mostraram bastante idênticos.

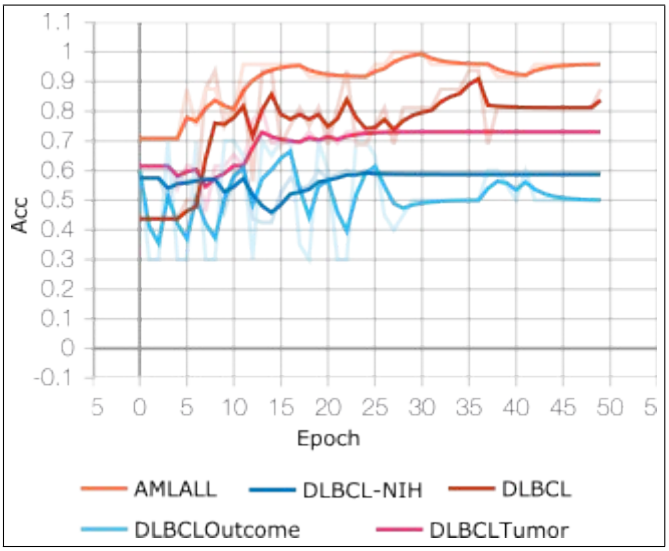

Figura 5. Acurácia da rede classificadora para 10% dos atributos

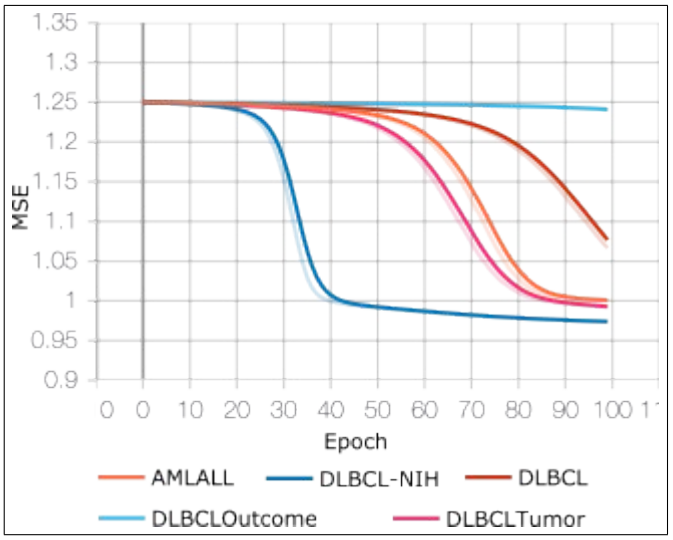

Figura 6. Erro quadrático da rede autocodificadora para 10% dos atributos

A figura 7 mostra a evolução da acurácia durante o treino da estrutura interna "Classificador" para 20% dos atributos. A figura 8 apresenta uma relação entre a quantidade de épocas usadas pela rede neural e o MSE para a reconstrução da base para 20% dos atributos. Se comparado aos resultados anteriores, as acurácias se mostraram piores, contudo os erros quadráticos se mantiveram idênticos.

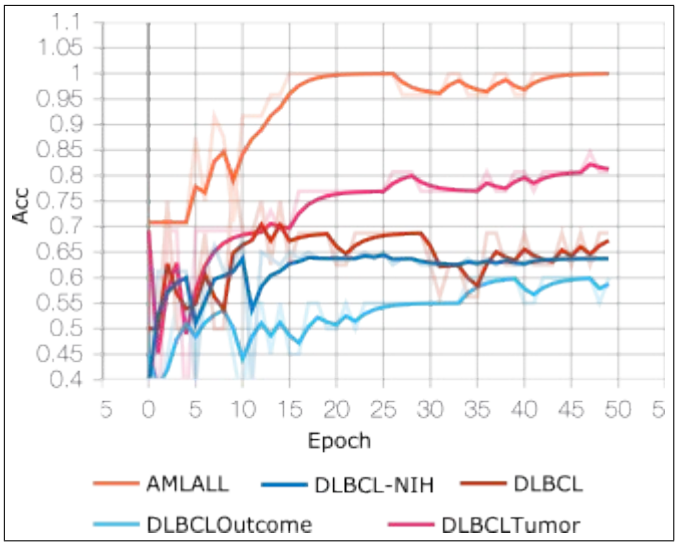

Figura 7. Acurácia da rede classificadora para 20% dos atributos

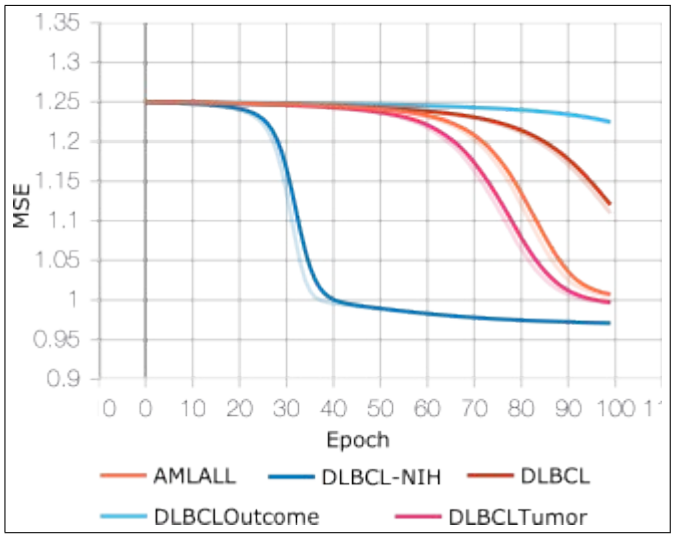

Figura 8. Erro quadrático da rede auto codificadora para 20% dos atributos

A figura 9 mostra a evolução da acurácia durante o treino da estrutura interna "Classificador" para 25% dos atributos. A figura 10 apresenta uma relação entre a quantidade de épocas usadas pela rede neural e o MSE para a reconstrução da base para 25% dos atributos. Assim como as demais, as acurácias não se diferenciaram muito, contudo o erro quadrádico foi mais lento para convergir.

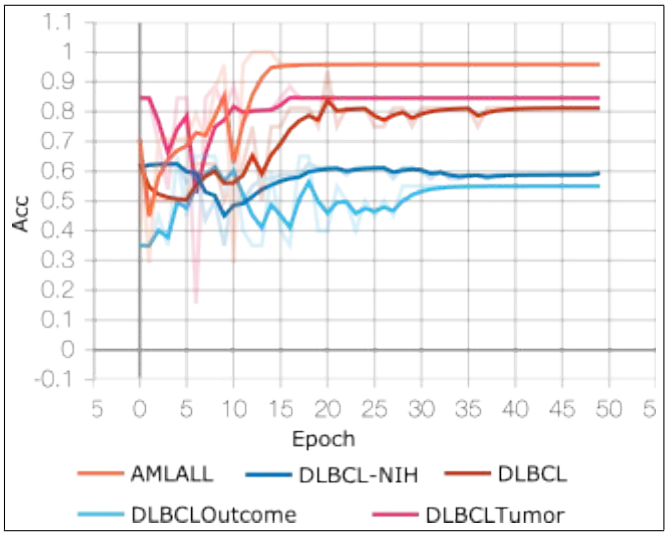

Figura 9. Acurácia da rede classificadora para 25% dos atributos

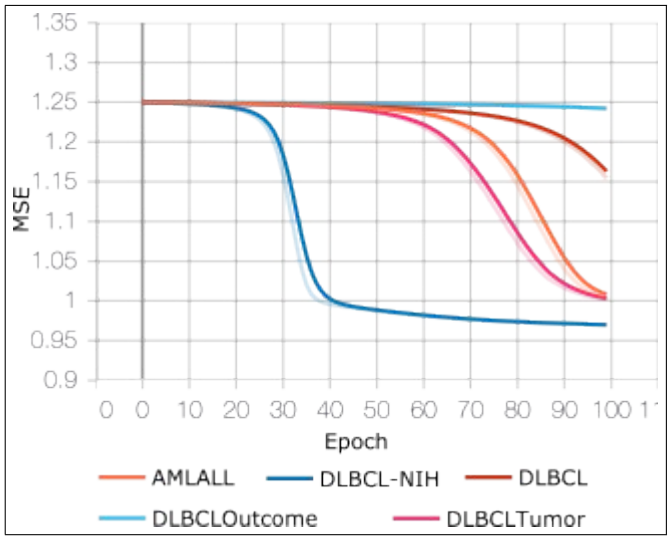

Figura 10. Erro quadrático da redea utocodificadora para 25% dos atributos

A figura 11 mostra a evolução da acurácia durante o treino da estrutura interna "Classificador" para 50% dos atributos. A figura 12 apresenta uma relação entre a quantidade de épocas usadas pela rede neural e o MSE para a reconstrução da base para 50% dos atributos. Dentre todas as reduções, reduzir a 50% foi a que mais apresentou resultados divergentes as demais - especialmente a base de dados DLBCL-Outcome, onde obteve a sua maior acurácia dentre os seus treinos anteriores e seu erro quadrático foi o que mais convergiu rapidamente.

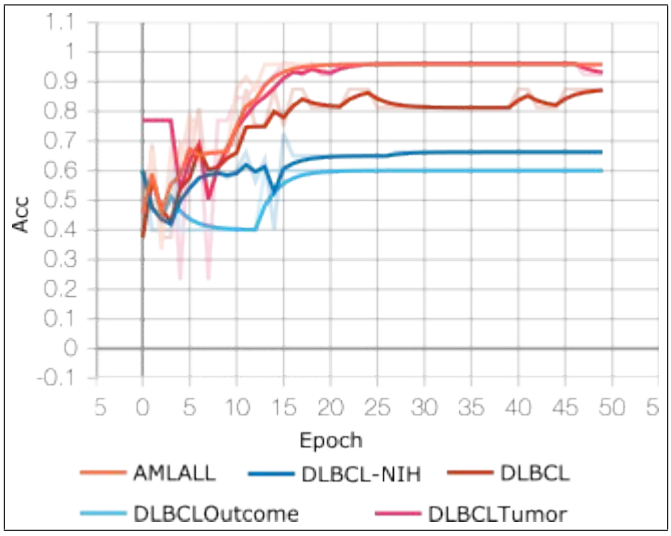

Figura 11. Acurácia da rede classificadora para 50% dos atributos

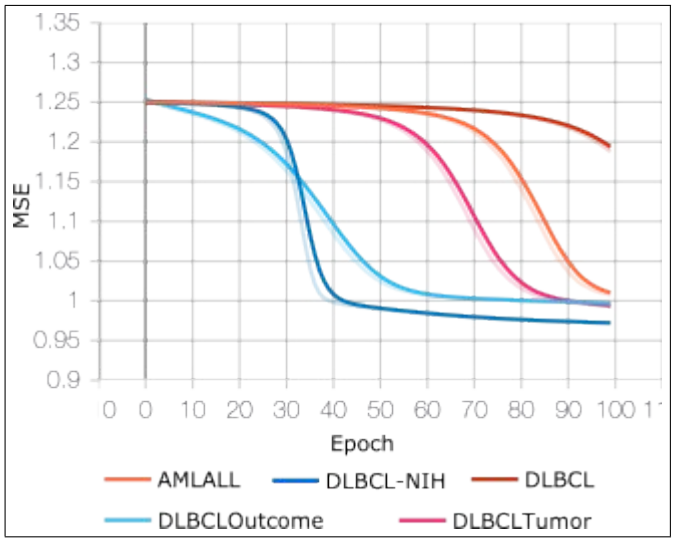

Figura 12. Erro quadrático da rede autocodificadora para 50% dos atributos

Comparando os resultados de Borges [9], o método apresentou uma acurácia maior na maioria dos casos, independemente das bases e do método de classificação. Na base de dados DLBCL Outcome, o método proposto se mostrou muito melhor em todos os casos.

CONCLUSÕES

A redução de dimensionalidade pode melhorar o desempenho na tarefa de classificação, visualização e armazenamento de grandes bases de dados [10]. Este estudo teve como objetivo propor um método para a redução treinando, primeiro, a estrutura codificadora com classificação e, em seguida, utilizar a estrutura codificadora numa rede autocodificadora. A proposta de um novo método não só visa realizar as vantagens de um algoritmo de redução de dimensionalidade como também procura otimizar algoritmos já existentes. O método proposto cumpre sua função de reduzir as bases de dados de microarranjo, contudo seu custo computacional se mostrou elevado.

O método se mostrou bom para redução nas bases de dados testados, contudo, em especial, a base de dados DLBC-Outcome se mostrou problemática ao método, mostrando resultados insatisfatórios em diversos momentos, assim como também se mostrou no método de Borges [9].

REFERÊNCIAS

[1] GÉRON, Aurélien. Handos-On Machine Learning with Scikit-Learn & TensorFlow. 1. ed. Sebastopol: O'Reilly Media Inc. 2017.

[2] MENG, Qinxue et. al. Relational Autoencoder for Feature Extraction. IEEE. 2017.

[3] SIQUEIRA, Rafael Fernandes Siqueira. Redução de dimensionalidade em bases de dados de classificação hierárquica multirrótulo usando autoencoders. 2019. 120 f. Dissertação (Mestrado em Ciência da Computação) - Universidade Tecnológica Federal do Paraná, Ponta Grossa.

[4] HINTON, G.E. et. al. Improving neural networks by preventin co-adaptation of features detectors. Universidade de Toronto. 2012.

[5] VICENT, Pascal et. al. Extracting and Composing Robust Features with Denoising Autoencoders. Universidade de Montreal. 2008.
[6] FILHO, Pierre de Almeida Telles. Asma Brônquica. 2004. Disponível em. http://www.asmabronquica.com.br/PDF/introducao_ChipDNA.pdf. Acesso em 24 ago. 2020
[7]SCHNEIDER, Jeff. Cross Validation. 1997. Disponível em. https://www.cs.cmu.edu/~schneide/tut5/node42.html. Acesso em 24 ago. 2020.
[8] NUNES, Luciana Neves. Métodos de imputação de dados aplicados na área da saúde. 2007. 120 f. Tese (Doutorado em Epidemiologia) - Universidade Federal do Rio Grande do Sul, Porto Alegre.
[9] BORGES, Helyane Bronoski. Redução de dimensionalidade em bases de dados de expressão gênica. 2006. 153 f. Dissertação (Mestrado) - Pontifícia Universidade Católica do Paraná. Curitiba.
[10] HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the Dimensionality of Data with Neural Networks. Washington: Science. 2017.

Helyane Bronoski Borges	Rafael Felipe Tasaka de Melo
	
Nome Orientador e Assinatura	Nome do Estudante e Assinatura