Uma Comparação de Classificadores para Contorno de Imagens de Folhas

Augusto A. B. Branquinho, Carlos E. S. Sabino, Eduardo C. Campos, Leandro N. Couto
Departamento de Ciência da Computação, Universidade Federal de Uberlândia, Uberlândia, MG

***Resumo*—Este artigo apresenta uma comparação entre diferentes algoritmos de classificação utilizando o *Waikato Environment for Knowledge* (WEKA), um software de código aberto com uma coleção de algoritmos de aprendizagem de máquina para tarefas de mineração de dados. O objetivo deste artigo é de investigar a taxa de acertos de diferentes algoritmos de classificação. No processo de classificação foram utilizados os algoritmos *k-Nearest Neighbors* (k-NN), *Naive Bayes* (NB), *Support Vector Machine* (SVM), Árvore de Decisão J4.8 (J4.8), *Random Forest* (RF), *Multilayer perceptron* (MLP) e *Linear Discriminant Analysis* (LDA). Na tentativa de melhorar os resultados foram realizados experimentos com algoritmos genéticos (AG) e de seleção de atributos na tentativa de descobrir quais atributos são mais importantes para o de reconhecimento de padrões. Um conjunto de amostras de folhas foi usada para os experimentos. Este conjunto de dados contém informações extraídas dos contornos de 600 imagens de folhas, agrupadas em 30 classes, contendo 20 amostras cada. Cada amostra foi caracterizada por 76 atributos. Desses, os 38 primeiros são baseados em um tipo de medida de imagem, enquanto os 38 seguintes são outro tipo de medição obtida.**

I. Introdução

O objetivo do nosso trabalho é investigar a taxa de acerto de diferentes algoritmos de classificação. Para isso, os experimentos foram divididos em três partes: a primeira parte utilizou o *software* WEKA para a análise de informações extraídas dos contornos de imagens de folhas. A segunda parte utilizou algoritmos genéticos para seleção dos melhores atributos e um ambiente distribuído para execução dos experimentos. Já a terceira parte utilizou o *software* MATLAB para realização de experimentos com o método de classificação LDA, cuja implementação não está disponível no WEKA.

Com o intuito de reduzir a dimensionalidade dos dados e melhorar a eficácia dos classificadores, foram realizados experimentos que fazem a seleção ou composição de atributos mais adequados a partir dos originalmente disponíveis.

Aprendizagem de máquina é um ramo da Inteligência Artificial que lida com a construção e estudo de sistemas que podem aprender a partir dos dados. A extração de informação importante a partir de uma grande pilha de dados e suas correlações é uma grande vantagem dessa área de estudo [1].

A mineração de dados é uma etapa do processo de *Knowledge Discovery in Databases* (KDD) que consiste na aplicação de análise de dados e algoritmos de descoberta que, sob as limitações de eficiência computacional aceitáveis, produzem uma enumeração particular de padrões (ou modelos) sobre os dados [12].

O primeiro experimento realizou uma Transformação do Espaço de Atributos utilizando *Principal Component Analysis* (PCA) para gerar um novo conjunto de atributos não correlacionados a partir da combinação de projeções dos atributos originais.

O segundo experimento envolveu a utilização de algoritmos genéticos para selecionar quais atributos classificam melhor o conjunto de amostras para cada algoritmo de classificação.

A Seção 2 descreve a caracterização do problema. Na Seção 3 é feita a fundamentação teórica dos conceitos utilizados no trabalho. A Seção 4 apresenta as ferramentas utilizadas. A seção 5 engloba os experimentos realizados e seus resultados. Na Seção 6 são apresentadas as conclusões do trabalho e alguns possíveis trabalhos futuros.

II. Caracterização do Problema

O conjunto de dados contém informações extraídas dos contornos de 600 imagens de folhas, agrupadas em 30 classes contendo 20 amostras cada. Cada amostra é identificada por um total de 76 atributos. Desses, os 38 primeiros são baseados em um tipo de medida de imagem, enquanto que os 38 seguintes são outro tipo de medição obtida.

A Figura 1 ilustra parte da base de folhas contendo 10 tipos de folhas distintos. Cada tipo de folha representa uma classe. No estudo, foram consideradas 30 classes de folhas.

Figura 1. Representação de parte da base de folhas

III. Fundamentação Teórica

A. k-Nearest Neighbors

O algoritmo *k-Nearest Neighbors* (k-NN) é considerado um algoritmo de aprendizagem estatística. Além de ser extremamente simples de implementar, ele é aberto a uma grande variedade de variações. Geralmente é definido em termos da distância Euclidiana [1].

O k-NN é um classificador preguiçoso porque não induz um modelo de categorização de dados de treinamento. O

processo de categorização é conseguido através da comparação da nova instância com todas as instâncias do conjunto de dados. Assim, a categoria para a nova instância é selecionada a partir das categorias dos K exemplos mais similares. Em um problema de categorização as entradas são as características e a saída é uma categoria. O k-NN também é conhecido no WEKA como IBK [2].

B. Naive Bayes

Os classificadores *Naive Bayes* assumem que todos os atributos são independentes e que cada um contribui igualmente para a categorização. A categoria é atribuída a um projeto, combinando a contribuição de cada característica. Esta combinação é atingida estimando as probabilidades *a posteriori* de cada categoria utilizando o Teorema de *Bayes*. As probabilidades *a priori* são estimadas com os dados de treinamento [2].

Este tipo de classificador é capaz de lidar com entradas categóricas e problemas de multi-classe. Portanto, em um problema de categorização, as entradas para o classificador são os atributos e a saída é a distribuição de probabilidade do projeto nas categorias [2].

C. Support Vector Machine

Support Vector Machine divide o espaço do problema em dois conjuntos possíveis por encontrar um hiperplano que maximiza a distância com o ponto mais próximo de cada subconjunto. SVM são classificadores binários, mas podem ser utilizados para classificação multi-classe [2].

A função que divide o hiperplano é conhecida como função *Kernel*. Se os dados são linearmente separáveis, uma função linear *Kernel* é usada com o SVM. Nos outros casos, funções não lineares tais como polinômios, funções de base radial (RBF) e sigmóides devem ser utilizadas [2].

D. Árvore de Decisão J4.8

Árvores de decisão são algoritmos que utilizam a estratégia "*divide and conquer*" para dividir o espaço do problema em subconjuntos. Uma árvore de decisão é modelada de forma que a raiz e os nós são as perguntas, e os arcos entre os nós são possíveis respostas para as perguntas. As folhas da árvore são categorias [2].

As árvores de decisão são capazes de lidar com entradas categóricas e os problemas multi-classe. Assim, em um problema de categorização, as entradas para a árvore são os atributos de uma aplicação e a saída é uma categoria [2].

J4.8 é um algoritmo existente para construção de uma árvore de decisão que permite a manipulação tanto de atributos discretos quanto contínuos. Além disso, permite a manipulação de dados de treinamento com valores de atributos ausentes [7].

E. Random Forests

Random Forests são uma combinação de preditores de árvores de tal modo que cada árvore depende dos valores de um vetor aleatório amostrado de forma independente e com a mesma distribuição para todas as árvores da floresta [8].

Random Forests são treinadas de forma supervisionada. O treinamento envolve a construção de árvores, bem como a atribuição a cada nó-folha da informação sobre as amostras de treinamento atingindo esse nó-folha (e.g. a distribuição da classe no caso de tarefas de classificação). Em tempo de execução, uma amostra de teste é transmitida a todas as árvores da floresta, e a saída é calculada pela média das distribuições registradas nos nós-folha alcançados [9].

F. Multilayer Perceptron

Uma rede *Multilayer Perceptron* (MLP) consiste de um conjunto de unidades sensoriais, que constituem a camada de entrada, uma ou mais camada(s) oculta(s) e uma camada de saída. O sinal de entrada propaga-se através da rede para a frente em uma base de camada-a-camada. Uma MLP representa uma generalização da rede *Single-layer Perceptron* [10].

Multilayer Perceptrons tem sido aplicadas com sucesso para resolver alguns problemas difíceis e diversos, treinando elas de forma supervisionada com um algoritmo muito popular conhecido como algoritmo de retropropagação do erro (*back-propagation*) [10].

G. Algoritmos Genéticos

O algoritmo genético padrão segue o método de reprodução sexuada. Neste algoritmo, a população é composta por um conjunto de indivíduos, de forma que cada indivíduo representa o cromossomo de uma forma de vida [13]. Os indivíduos correspondem a soluções de um dado problema.

Associado a cada cromossomo é atribuída uma função chamada de *fitness* que determina o quão bom é o indivíduo. Outra função é usada para selecionar os indivíduos da população que iram se reproduzir. Após a seleção de dois indivíduos ocorre o cruzamento (*crossover*) dos indivíduos selecionados e eles se dividem novamente. Em seguida, existe uma probabilidade que ocorra a mutação dos novos indivíduos. O processo é repetido durante um certo número de vezes [13]. Cada repetição corresponde a uma geração.

Fitness é uma medida de "bondade"de um cromossomo, isto é, o quão bem um cromossomo se encaixa no espaço de busca, ou resolve o problema em questão [13].

Seleção é o processo de escolha do par de indivíduos para reproduzir [13].

Crossover é um processo de troca de genes entre os dois indivíduos que estão se reproduzindo [13].

Mutação é o processo de alteração aleatória dos cromossomos [13].

H. Principal Component Analysis

A Análise dos Componentes Principais (PCA) é uma das técnicas mais bem sucedidas que têm sido utilizada em reconhecimento de imagem e compressão. PCA é um método estatístico cujo propósito é reduzir a grande dimensionalidade do espaço de dados (variáveis observadas) para a menor dimensionalidade instrínseca do espaço de características (variáveis independentes). Este é o caso quando existe uma forte correlação entre as variáveis observadas [11].

Os trabalhos que PCA pode fazer são previsão, remoção de redundância, extração de características, compressão de dados, entre outros. [11]

I. *Linear Discriminant Analysis*

Linear Discriminant Analysis (LDA) [3] [4] é um método popular para redução de dimensionalidade linear, que maximiza a dispersão entre as classes e minimiza a dispersão dentro da classe. O LDA tende a dar resultados indesejados se as amostras de alguma classe formam vários agrupamentos distintos, isto é, multimodal [4].

Geralmente, o LDA pode se referir a um método de classificação em que primeiramente as amostras de dados são projetadas em uma espaço unidimensional e depois classificadas considerando um *thresholding*. A incorporação de um espaço unidimensional utilizada acima é dado como o maximizador do chamado critério de *Fisher*. Este critério é frequentemente utilizado para redução de dimensionalidade de um subespaço com dimensão maior que um [4].

J. *Standard Score*

Standard Score ou *Z-Score*, em estatística, é o número de desvios padrão de um dado acima ou abaixo da média. Um *Z-Score* positivo representa um dado acima da média, enquanto que um negativo representa um dado abaixo da média. É uma quantidade desprovida de dimensões obtida subtraindo a média da população de um dado bruto e dividindo o resultado pelo desvio padrão [14].

$$z = \frac{(x - \mu)}{\sigma} \quad (1)$$

Essa ferramenta é utilizada para diminuir a diferença que escalas podem fazer na classificação de determinados algoritmos.

IV. Ferramentas Utilizadas

A. *WEKA*

O WEKA é um *software* de mineração de dados que foi desenvolvido utilizando a linguagem JAVA pela Universidade de *Waikato* na Nova Zelândia. Possui uma coleção de algoritmos de aprendizagem de máquina para tarefas de mineração de dados. Ele implementa também algoritmos para pré-processamento de dados, classificação, regressão, *clustering* e regras de associação, além de incluir ferramentas de visualização [1].

O *software* encontra-se licenciado ao abrigo da *General Public License* (GPL) sendo portanto possível estudar e alterar o respectivo código fonte [1].

O arquivo de dados normalmente utilizado pelo WEKA é o formato de arquivo ARFF (*Attribute-Relation File Format*), que consiste de *tags* especiais para indicar diferentes elementos no arquivo de dados (e.g. nomes de atributos, tipos de atributos, valores de atributos e os dados) [1].

B. *MATLAB*

O *Matrix Laboratory* (MATLAB) é ao mesmo tempo, um ambiente e linguagem de programação para cálculos numéricos com vetores e matrizes. Ele é um produto da empresa *The Math Works Inc.* [5].

	KNN	*Naive Bayes*	SVM	J4.8	*Random Forest*	MLP
Todos os atributos	78,00%	71,50%	79,50%	71,33%	81,50%	84,50%
Com PCA	71,16%	74,83%	60,16%	68,50%	74,66%	79,66%
Com Z-Score	78,00%	71,66%	78,66%	74,00%	81,16%	85,16%

Figura 2. Taxa de acerto considerando todos os 76 atributos

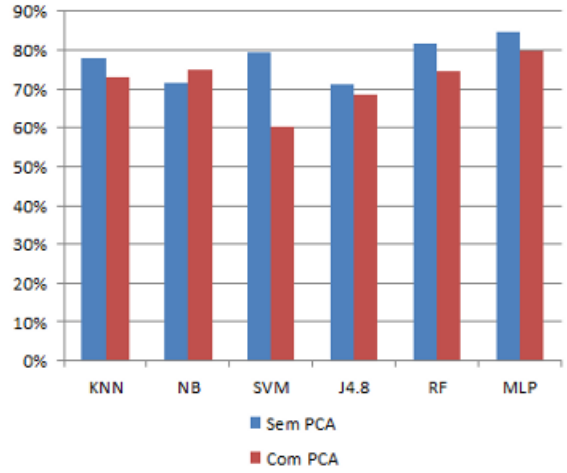

Figura 3. Comparação utilizando ou não o PCA nos 76 atributos

O MATLAB permite executar tarefas computacionalmente intensivas mais rápido do que com linguagens de programação tradicionais. Ele tem uma ampla variedade de funções úteis para o praticante de algoritmos genéticos [6].

C. *ECJ*

Evolutionary Computation Java (ECJ) consiste em uma biblioteca Java para a execução de algoritmos evolutivos. Ele possui um conjunto de algoritmos já prontos e de fácil extensão. Além disso ele permite o processo de criação e avaliação dos indivíduos de forma distribuída [16].

Neste trabalho o ECJ foi usado para encontrar melhores atributos usando um AG distribuído.

V. Experimentos

Os experimentos foram divididos em três partes. Na primeira parte foram realizados experimentos através da interface gráfica do WEKA considerando k-fold igual a 10. A segunda parte utiliza algoritmos genéticos para a seleção dos melhores atributos e um ambiente distribuído para a execução dos experimentos. Por último, foi usado o MATLAB para a execução de um novo conjunto de testes utilizando o algoritmo LDA.

A. *Parte 1*

Inicialmente, foram utilizados todos os atributos como entrada para os algoritmos de classificação. Além disso, foram feitas duas modificações nos atributos na tentativa de melhorar a taxa de classificação correta: normalização dos dados com o algoritmo *Z-Score* e combinação de atributos utilizando o

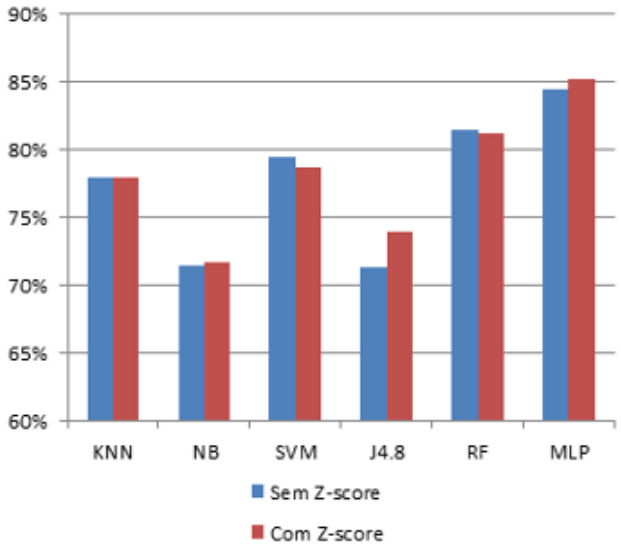

Figura 4. Comparação utilizando ou não o Z-Score nos 76 atributos

PCA. Estas modificações foram feitas considerando todos os 76 atributos.

A utilização do PCA resultou na geração de seis componentes principais. Todos esses componentes foram considerados no cálculo da taxa de acerto dos algoritmos de classificação, uma vez que ao utilizar menos componentes principais prejudicou o desempenho dos classificadores.

Como podemos observar na Figura 2, entre os seis algoritmos utilizados, o que melhor se destacou foi o MLP com o *Z-Score* aplicado, com uma taxa de acerto de 85,16% das amostras. Ainda na Figura 2 podemos notar que na utilização do PCA apenas o algoritmo *Naive Bayes* se beneficiou, enquanto que todos os outros apresentaram melhores resultados sem nenhuma modificação nos atributos. A taxa de acerto do algoritmo *Naive Bayes* utilizando PCA nos 76 atributos foi de 74,83%.

A Figura 3 ilustra um gráfico comparando a taxa de acerto em % dos algoritmos de classificação com a utilização do PCA e sem a utilização do PCA. A Fig. 3 ilustra um gráfico comparando a taxa de acerto em % dos algoritmos de classificação com a utilização do *Z-Score* nos 76 atributos e sem a utilização da normalização.

Numa segunda abordagem, a base foi dividida em duas partes: 38 primeiros e 38 últimos atributos. Assim como na abordagem inicial, foram utilizados os seis algoritmos definidos anteriormente. Foram feitos dois testes: o primeiro teste aplicou o PCA apenas nos 38 primeiros atributos e o segundo teste aplicou o PCA apenas nos últimos 38 atributos.

O resultado do primeiro teste foi a geração de seis componentes principais. A melhor taxa de acerto para os algoritmos foi utilizando todos os seis componentes principais, com exceção do SVM que aumentou sua taxa de acerto de 45,83% para 47,66%.

Já no segundo teste, foram gerados apenas três componentes principais. Da mesma forma, a melhor taxa de acerto para os algoritmos foi utilizando todos os três componentes

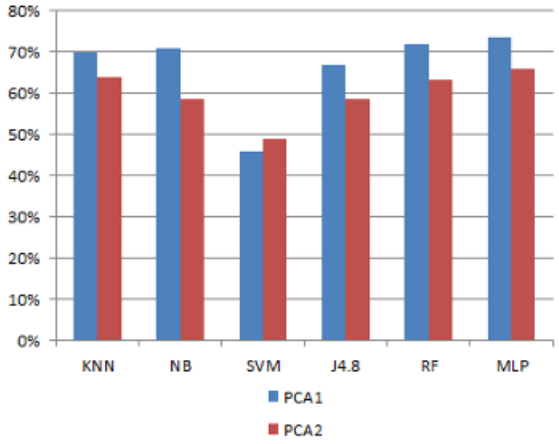

Figura 5. Utilizando PCA nos primeiros 38 atributos e nos últimos 38 atributos

A_1	A_2	A_3	...	A_{75}	A_{76}
true	false	false	...	true	false

Figura 7. Exemplo de um indivíduo usado no AG.

principais. A Fig. 4 representa a comparação entre estes dois testes realizados. O PCA1 refere-se à aplicação do PCA nos primeiros 38 atributos, enquanto que o PCA2 refere-se à aplicação do PCA nos últimos 38 atributos. Logo, podemos concluir que o PCA1 obteve melhores resultados do que o PCA2.

B. Parte 2

Na segunda parte dos experimentos foi usado um AG para a seleção dos melhores atributos. Para esta tarefa foi usada a ECJ [16]. Esta biblioteca permitiu a execução do algoritmo genético de forma distribuída e paralela. [1]

O indivíduo foi representado por um cromossomo de 76 genes. Cada gene guarda um valor booleano que determina a presença ou ausência do atributo usado no cálculo do *fitness*. O primeiro gene está associado ao primeiro atributo, o segundo gene ao segundo atributo e assim por diante até o último gene/atributo. A Figura 7 mostra um exemplo resumido de um indivíduo.

Foram criadas 30 populações com 50 indivíduos cada. Estas populações foram separadas em dois grupos de 15 populações. No primeiro grupo foi usada a base de dados sem normalização e no outro os dados foram normalizados com o *Z-Score*. Para cada população está associado um algoritmo de reconhecimento de padrões e um conjunto de parâmetros. A Tabela I mostra os algoritmos e parâmetros relacionados com cada grupo para suas 15 populações.

Para realizar o cálculo do *fitness* de um indivíduo é executado o algoritmo de reconhecimento de padrões de acordo com a população. A ordem que estão dispostas as amostras

[1]O projeto e os *logs* da parte 2 dos experimentos estão disponíveis em: https://code.google.com/p/pgc204/ e http://wpattern.com/blog/post/2013/07/08/ECJ-e-WEKA-(Reconhecimento-de-Padroes).aspx

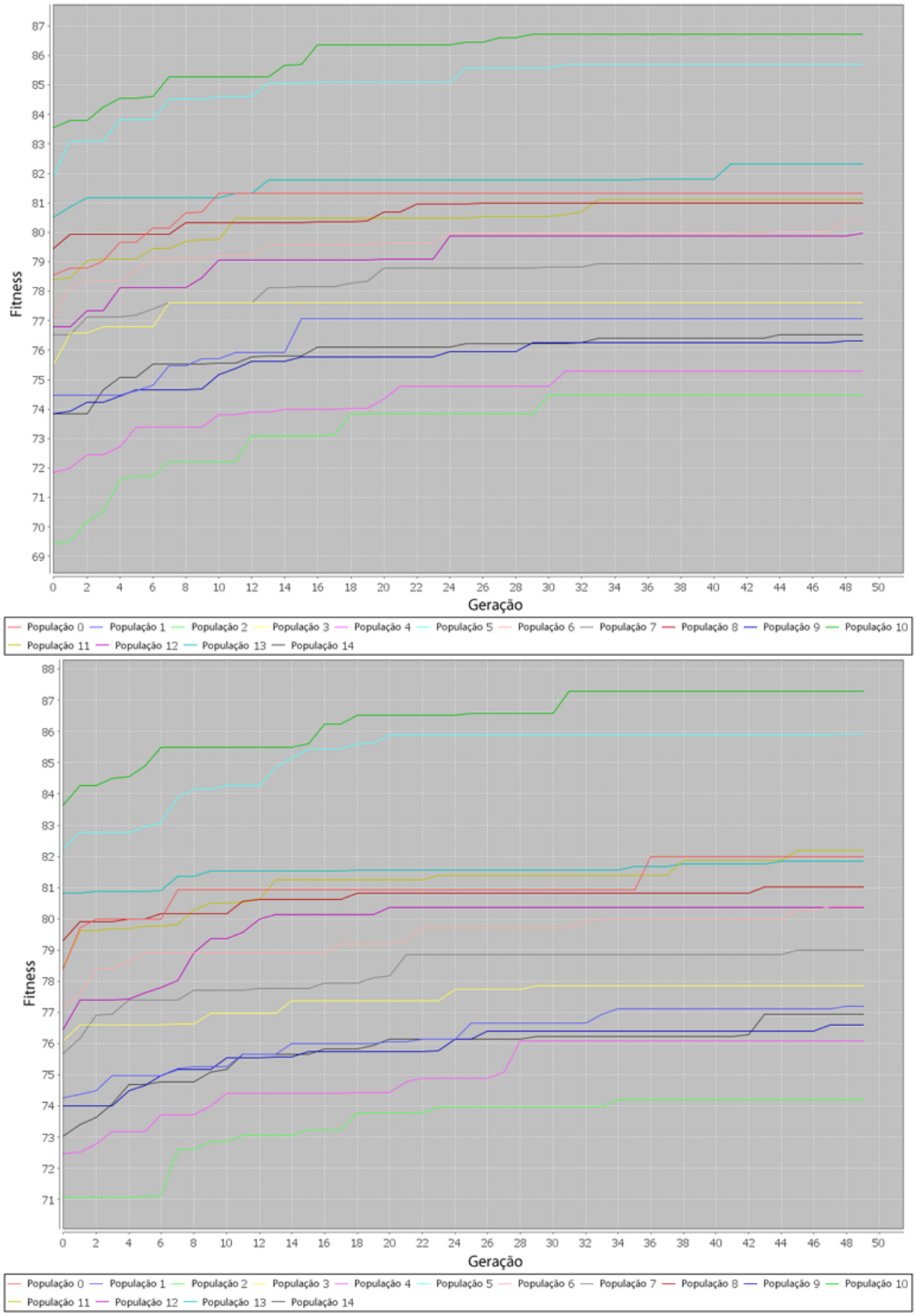

Figura 6. Gráficos com os percentuais de acerto durante as gerações do AG para os experimentos do grupo 1 e grupo 2, respectivamente.

Tabela I. PARÂMETROS USADOS POR CADA POPULAÇÃO DO AG.

População	Algoritmo	Cross-validation (folds)
0	k-NN (1)	2
1	k-NN (5)	
2	k-NN (9)	
3	Random Forest	
4	Naive Bayes	
5	k-NN (1)	5
6	k-NN (5)	
7	k-NN (9)	
8	Random Forest	
9	Naive Bayes	
10	k-NN (1)	10
11	k-NN (5)	
12	k-NN (9)	
13	Random Forest	
14	Naive Bayes	

durante a execução dos algoritmos influência no percentual de acertos. Sendo assim, antes de qualquer execução a base de dados é reorganizada aleatoriamente e o *fitness* é dado pela média percentual de acertos após 5 execuções.

A população inicial é gerada de forma totalmente aleatória. Para cada população é executado o AG com 50 gerações. Ao final de cada geração são criados 50 novos indivíduos a partir da geração anterior. A seleção dos indivíduos é feita pelo processo de torneio simples. Já o *crossover* ocorre através do sorteio de duas posições (1 até 76) e a respectiva troca de genes entre estas posições. Cada novo indivíduo possui uma chance de 3% de sofrer mutação, sendo que a mutação apenas altera o valor de um gene que foi escolhido aleatoriamente. Ao final de cada geração os piores indivíduos são eliminados.

Os melhores *fitness's* obtidos após cada geração para cada população do primeiro e segundo grupos são mostrados na Figura 6.

A melhor indivíduo obtido para o primeiro grupo possui as seguintes características:

- Parâmetros da população: k-NN (1) com fold de 10.
- Menor *fitness*: 86.33
- *Fitness* médio: 86.73
- Maior *fitness*: 87.16
- Atributos selecionados: A1, A2, A4, A5, A6, A7, A9, A16, A21, A22, A23, A24, A25, A26, A27, A29, A30, A31, A34, A39, A40, A45, A46, A48, A49, A50, A51, A52, A54, A56, A57, A59, A60, A62, A63, A64, A65, A66, A68, A69, A76.

A melhor indivíduo obtido para o segundo grupo possui as seguintes características:

- Parâmetros da população: k-NN (1) com fold de 10.
- Menor *fitness*: 86.5
- *Fitness* médio: 87.3
- Maior *fitness*: 88.0
- Atributos selecionados: A1, A2, A4, A5, A7, A8, A9, A10, A11, A16, A17, A18, A20, A21, A22, A23, A25, A26, A27, A28, A29, A31, A32, A33, A34, A37, A39, A40, A46, A47, A49, A50, A51, A53, A54, A55, A57, A58, A60, A61, A63, A65, A66, A67, A69, A71, A73.

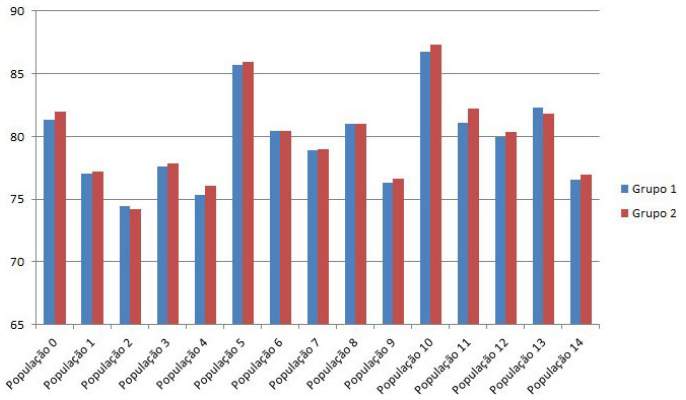

Figura 8. Comparação dos grupos considerando o *fitness* do melhor indivíduo de cada população após o processo de evolução.

Conforme é mostrada na Figura 8, em geral a normalização dos dados permitiu a obtenção de melhores resultados. Além disso, o algoritmo k-NN com $k = 1$ demonstrou melhores *fitness* que as outras configurações.

Apenas neste experimento foram realizados 75000 execuções de classificação. Como os algoritmos SVM e MLP demandam maior tempo de execução e a quantidade de execuções em geral é grande eles não foram testados. Contudo, este é um dos trabalhos futuros.

C. Parte 3

Os experimentos com o LDA foram feitos com a implementação do classificador presente no MATLAB considerando k-fold igual a 10. Após a classificação das amostras, foi gerada a matriz de confusão, a partir da qual foram realizadas as análises do método.

Foram realizados quatro testes independentes utilizando o LDA e o PCA:

1) Classificação utilizando o LDA nos 76 atributos sem a utilização do PCA;
2) Utilização do PCA nos 76 atributos e posterior classificação utilizando o LDA;
3) Utilização do PCA apenas nos 38 primeiros atributos e posterior classificação utilizando o LDA;
4) Utilização do PCA apenas nos últimos 38 atributos e posterior classificação utilizando o LDA.

Os melhores resultados foram obtidos no teste 1, cuja taxa de acerto foi de 88,66%. No teste 2 foram gerados 6 componentes principais e a taxa de acerto foi de 76,00%. Já no teste 3 foi obtida uma taxa de acerto de 71,00%, enquanto que no teste 4, a taxa de acerto foi de 59,67%.

VI. CONCLUSÃO

Na primeira parte do experimento, o algoritmo de classificação que obteve maior taxa de acerto foi o MLP com *Z-score*, que alcançou o valor de 85,16%. Além disso, podemos concluir que a utilização do PCA nos 76 atributos originais diminuiu a taxa de acerto dos algoritmos de classificação considerados, com exceção do classificador *Naive Bayes*, que aumentou sua taxa de acerto atingindo o valor de 74,83%.

A segunda parte do experimento que envolveu AG em dois grupos de populações mostrou que o algoritmo k-NN configurado com k = 1 e 10 *folds* obteve melhores *fitness* em ambos os grupos do que as outras configurações. Inclusive o melhor indivíduo do segundo grupo obteve maior *fitness* médio (87,3%) do que o melhor indivíduo do primeiro grupo (86,73%). O primeiro grupo foi composto por 15 populações de 50 indivíduos cada porém sem a normalização da base de dados, enquanto que o segundo grupo foi composto por 15 populações de 50 indivíduos cada cuja base de dados foi normalizada com *Z-Score*. A utilização do AG foi bastante útil na seleção dos melhores atributos.

A terceira parte do experimento que trabalhou com LDA obteve uma taxa de acerto de 88,66% considerando os 76 atributos originais, ou seja, sem a aplicação do PCA nos atributos.

Para este tipo de problema, embora o PCA contribuiu para a redução da dimensionalidade do espaço de atributos, sua utilização acarretou uma grande perda de informação nos algoritmos considerados na primeira e terceira partes do experimento. Como exemplo, o SVM foi o mais prejudicado pela utilização do PCA, obtendo uma taxa de acerto de apenas 60,16%.

Com taxas de classificação abaixo de 90%, fica evidente que a classificação desse conjunto de dados em particular não configura um problema trivial. A análise do conjunto de dados através do uso de múltiplos métodos de classificação oferece observações importantes em relação à natureza do conjunto de dados.

Trabalhos Futuros

O AG demonstrou seu poder de encontrar melhores atributos. Contudo, após uma dada quantidade de gerações o *fitness* ficou estagnado. Este cenário ocorre quando o processo de evolução encontrou um máximo local. Para tentar explorar outros máximos locais existe a possibilidade de utilizar uma abordagem de evolução baseado em ilhas [15] ou aumentar a taxa de mutação. Além disso, existe a possibilidade de iniciar populações com diferentes percentuais de presença de atributos, já que isso pode permitir encontrar outros máximos locais. Ainda considerando o AG, existe a necessidade da utilização de outros algoritmos de classificação. Dentre estes algoritmos temos o SVM, LDA e MLP.

REFERÊNCIAS

[1] M. Fauzi, T. Moh, *Comparison of different classification techniques using WEKA for Breast Cancer*, IFMBE Proceedings, vol. 15, pp. 520-523, 2007.

[2] M. Linares-vásquez, C. Mcmillan, D. Poshyvanyk et al, *On Using Machine Learning to Automatically Classify Software Applications into Domain Categories*, pp. 7-8, 2009.

[3] R. A. Fisher, *The use of multiple measurements in taxonomic problems. Annals of Eugenics*, 7, pp. 179-188, 1936.

[4] K. Fukunaga, *Introduction to statistical pattern recognition*, Boston: Academic Press, Inc. Second edition, 1990.

[5] J. Ortega, M. Del, R. Boone et al, *Research issues on K-means Algorithm: An Experimental Trial Using Matlab*, pp. 89-90

[6] J. Furtado, Z. Cai, L. Xiaobo, *Digital Image Processing: Supervised classification using genetic algorithm in MATLAB Toolbox*, vol. 2, pp.54-55, 2010.

[7] L. Sehgal, N. Mohan, P. Sandhu, *Quality Prediction of Function Based Software Using Decision Tree Approach*, International Conference on Computer Engineering and Multimedia Technologies, Bangkok, Thailand, pp. 44-44, 2012.

[8] L. Breiman, *Random forests*, Statistics Department University of California, Berkley, pp.1-2, 2001.

[9] V. Lempitsky, M. Verhoek, A. Noble et al, *Random Forest Classification for Automatic Delineation of Myocardium in Real-Time 3D Echocardiography*, pp. 449-449, 2009.

[10] S. Haykin, *Neural Networks: a comprehensive foundation*, Second Edition, pp. 178-179, 1998.

[11] K. Kim, *Face Recognition using Principle Component Analysis*, Department of Computer Science, University of Maryland, USA, pp. 1-1.

[12] U. Fayyad, G. Piatetsky-shapiro, P. Smyth, *From Data Mining to Knowledge Discovery in Databases*, pp. 41-41, 1996.

[13] E. Prebys, *The Genetic Algorithm in Computer Science*, MIT Undergraduate Journal of Mathematics, pp. 166-166.

[14] Kreyszig, E (fourth edition 1979). *Applied Mathmatics*, Wiley Press.

[15] Martin, W. N. and Lienig, J. and Cohoon, J. P. *Island (migration) models: evolutionary algorithms based on punctuated equilibria.* Handbook of Evolutionary Computation. 1997.

[16] Luke, S. *The ECJ Owner's Manual A User. Manual for the ECJ Evolutionary Computation Library*. Department of Computer Science, George Mason University. May, 2013.