

# Uma abordagem Multi-objetiva Híbrida para Seleção e Atribuição de Pesos a Características para Classificadores k-NN

Salomão S. Madeiro, Joás E. Souza, Carmelo J. A. Bastos Filho e Adriano L. I. Oliveira

Departamento de Sistemas e Computação – Universidade de Pernambuco (UPE)  
CEP 50.750-470 – Recife – PE – Brasil

{ssm, jes, cjabf, adriano}@dsc.upe.br

**Abstract.** *Selection and assignment of weights to the input features directly influence performance of a k-Nearest Neighbor (k-NN) classifier. The manual selection may become infeasible for databases with a large quantity of input features (e.g. 100 features). In this work, a new approach is proposed to select features using a multi-objective PSO algorithm combined with guided mutation. We used a Cross-Searching Strategy Multi-Objective Particle Swarm Optimization (CSS-MOPSO) technique to assign the weights. Simulation results show that the approach proposed in this work is able to improve the precision of a k-NN classifier in comparison with other approaches proposed previously.*

**Resumo.** *Seleção e atribuição de pesos às características de um classificador como o k-Nearest Neighbor (k-NN) influenciam diretamente o desempenho do classificador. A seleção manual pode se tornar inviável para bases de dados com uma grande quantidade de características (e.g. 100 características). Neste trabalho, é proposta uma nova abordagem para seleção de características por meio de um processo baseado em um algoritmo de inteligência de enxames multi-objetivo e mutação guiada. Para a atribuição de pesos é usada a técnica Cross-Searching Strategy Multi-Objective Particle Swarm Optimization (CSS-MOPSO). Os resultados obtidos nas simulações demonstram que a abordagem proposta neste trabalho é capaz de melhorar a precisão de um classificador k-NN em relação a outras abordagens propostas anteriormente.*

## 1. Introdução

Seleção e atribuição de pesos a características de entradas são técnicas que podem ser usadas para melhorar a precisão de um classificador k-NN [Tahir *et al.* 2007]. Algumas características não contribuem e outras chegam até a prejudicar significativamente a precisão na classificação. Já a atribuição de pesos a determinadas características pode contribuir para a melhoria da capacidade de generalização do classificador. Tahir *et al.* [Tahir *et al.* 2007] propõem uma solução híbrida que contempla as duas propostas (ou seja, seleção e atribuição de pesos) para melhoria na capacidade de generalização do k-NN. A idéia do método proposto por eles é usar a técnica de busca *Tabu Search* [Glover *et al.* 1989] com o objetivo de encontrar o vetor de pesos a serem aplicados aos padrões de entrada e aos padrões já conhecidos (fase de treinamento) no intuito de realçar

algumas características e eliminar outras, com o objetivo de melhorar a capacidade de generalização de um classificador  $k$ -NN.

Para classificações com uma quantidade relativamente grande de características (*e.g.* 100 características), o método exaustivo para a busca do vetor de pesos ótimo seria impraticável. Com isso, surge a necessidade de heurísticas que orientem a busca desse vetor em menor tempo e com menor esforço computacional. A definição dos pesos e a indicação da eliminação ou permanência da característica são feitas pela atribuição aleatória de valores reais e binários respectivamente. Na proposta deste trabalho, a atribuição de valores reais é feita pela técnica *Cross-Searching Strategy Multi-Objective Particle Swarm Optimization* (CSS-PSO) [Chiu *et al.* 2007]. Por outro lado, a seleção de características é feita pelo operador genético mutação com algumas adaptações para lidar com o problema em questão.

Esse trabalho está dividido em cinco partes: a seção 2 descreve as técnicas de Inteligência Computacional utilizadas neste trabalho. A seção 3 propõe um classificador híbrido CSS-PSO /  $k$ -NN. A seção 4 apresenta os resultados obtidos nos experimentos e a seção 5 conclui o trabalho, apresentando perspectivas para trabalhos futuros.

## 2. Background

Nesta seção, são apresentados conceitos relativos a Algoritmos Genéticos, classificador  $k$ -NN, PSO e CSS-PSO utilizados na abordagem proposta neste trabalho.

### 2.1. Algoritmos Genéticos

Algoritmos Genéticos baseiam-se nos processos naturais de seleção, cruzamento e mutação [Goldberg 1986]. O problema (ambiente de sobrevivência) pode ser descrito sob a forma de uma função matemática, em que as “estruturas” (indivíduos) mais fortes obterão valores mais adequados para uma determinada função. Assim cada indivíduo corresponde a uma possível solução. Então, trabalhando com um grupo de indivíduos simultaneamente, verifica-se a potencialidade de cada um em relação ao grupo, tentando selecionar os mais aptos para o cruzamento. Depois de se efetuar o cruzamento, cada gene de cada indivíduo estará sujeito a uma eventual ação da mutação.

Um indivíduo ou estrutura corresponde a uma concatenação de variáveis ou cadeias de caracteres (cromossomos), onde em cada caractere (gene), encontra-se numa dada posição (locus) e com seu valor determinado (alelo). Um sinônimo de indivíduo em genética natural é o genótipo e a sua estrutura decodificada é o fenótipo. Em outras palavras o fenótipo, em sistemas artificiais, significa um conjunto de parâmetros, ou um ponto solução no espaço de procura. A partir do fenótipo, o potencial de sobrevivência pode ser obtido através da avaliação da função desempenho. Termos como reprodução, cruzamento, mutação, população, estão diretamente ligados a indivíduos.

### 2.2. $k$ -Nearest Neighbors ( $k$ -NN)

O treinamento de um classificador  $k$ -NN consiste unicamente em armazenar os padrões apresentados durante essa fase. Para cada padrão a ser classificado por um  $k$ -NN, é necessário o cálculo da distância deste padrão para todos os padrões de treinamento. Então, o  $k$ -NN classifica um determinado padrão de entrada como a classe que corresponde à maioria dos  $k$  padrões de treinamento mais próximos daquele padrão de

entrada, onde  $k$  é um valor natural normalmente pequeno (*e.g.* menor que 10) [Duda *et al.* 2000]. No caso de classificações binárias, ou seja, onde existem apenas duas classes, é recomendável escolher um valor ímpar para  $k$ , a fim de evitar empates. Como contribuição à precisão da classificação, pode ser utilizado um peso multiplicador das características, dessa forma é possível uma melhor distinção de vizinhança, recurso utilizado nesse trabalho. Normalmente, para o cálculo da distância entre dois padrões, é utilizada a distância Euclidiana, mas outras métricas de distâncias, como a distância de Manhattan, também podem ser utilizadas.

### 2.3. Particle Swarm Optimization (PSO)

Kennedy e Eberhart [Kennedy *et al.* 1995], inspirados em modelos já existentes para simulação do comportamento de um bando de pássaros, propuseram uma nova abordagem de busca baseada no comportamento emergente de um enxame de entidades, chamadas de partículas. Cada partícula no espaço de busca (*i.e.* conjunto de todas as possíveis soluções para um problema) representa uma possível solução para um problema de otimização. Logo, um enxame é um subconjunto das possíveis soluções para um problema de otimização. O processo de busca no PSO é guiado pelas equações (1) e (2).

$$\vec{v}_i(t+1) = \omega \vec{v}_i(t) + c_1 r_1 (\vec{p}_{Best_i}(t) - \vec{x}_i(t)) + c_2 r_2 (\vec{g}_{Best_i}(t) - \vec{x}_i(t)),$$

(1)

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1).$$

(2)

$r_1$  e  $r_2$  são valores randômicos uniformemente distribuídos no intervalo  $[0,1]$ .  $\vec{p}_{best_i}(t)$  é um vetor que aponta para a melhor posição (*i.e.* de acordo com a função objetivo) que a  $i$ -ésima partícula já visitou. Por outro lado,  $\vec{g}_{best}(t)$  aponta para a melhor posição já visitada por todas as partículas no enxame ( $g_{best}$ ) ou em uma vizinhança ( $l_{best}$ ).  $c_1$  e  $c_2$  são normalmente usados como constantes para determinar o valor da influência dos termos cognitivos e sociais da equação de velocidade. Por fim,  $\omega$  é normalmente usado como uma constante para determinar a influência da direção da velocidade na iteração anterior sobre a direção da velocidade na próxima iteração.

### 2.3. Cross-Searching Strategy para PSO Multi-Objetivo

Diversas abordagens foram propostas para adaptação do PSO para problemas com vários objetivos [Coello *et al.* 2002], [Parsopoulos *et al.* 2002]. Chiu *et al.* [Chiu *et al.* 2007] propuseram estratégias para o MOPSO relativas à escolha de dois novos guias para as partículas em seu processo de busca, em substituição ao  $p_{best}$  e ao  $g_{best}$ . Os novos guias locais para as partículas são designados como  $g_{best1}$  e  $g_{best2}$ . Como existe a idéia de busca cruzada dos dois líderes, esta técnica foi denominada *Cross Search Strategy Multi-Objective Particle Swarm Optimization* (CSS-MOPSO). A nova equação de atualização da velocidade das partículas é dada por:

$$\vec{v}_i(t+1) = \omega \vec{v}_i(t) + c_1 r_1 (\vec{g}_{Best_{1i}}(t) - \vec{x}_i(t)) + c_2 r_2 (\vec{g}_{Best_{2i}}(t) - \vec{x}_i(t)).$$

A estratégia para escolha de  $g_{best1}$  para cada partícula se dá inicialmente com determinação de um ponto, ponto *Datum c*, como sendo a intersecção de duas retas

perpendiculares que passam pelas soluções mais extremas do conjunto solução. Em seguida, o ângulo  $\theta$  é calculado como sendo o ângulo entre a reta *Datum* - definida como a reta que liga um membro do arquivo externo e o ponto *c* - e a linha que conecta a solução a um membro do arquivo. Logo, o membro do arquivo que tem o menor ângulo  $\theta$  com a *i*-ésima partícula será eleito como o  $g_{best1}$  daquela partícula. A Figura 1(a) ilustra o processo de escolha do  $g_{best1}$  descrito anteriormente.

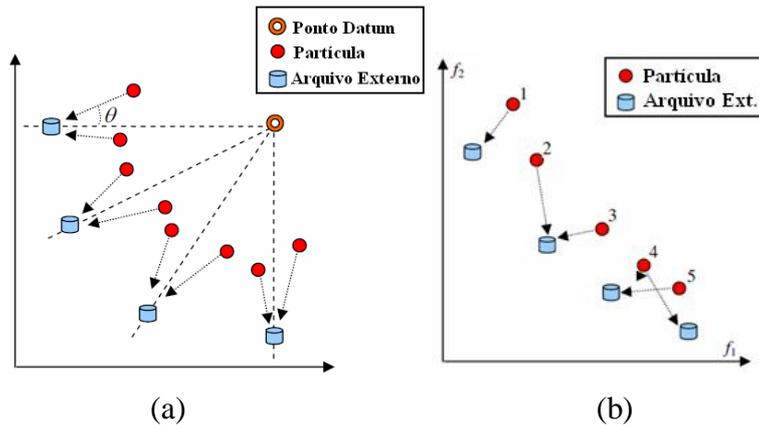


Figura 1. Estratégia de escolha do (a)  $g_{best1}$ , e (b)  $g_{best2}$ . Figura adaptada de [Chiu *et al.* 2007].

A estratégia de escolha do  $g_{best1}$  é baseada no valor da aptidão para um objetivo  $f_i$  selecionado randomicamente. Primeiramente, as partículas no conjunto solução são ordenadas de acordo com o objetivo  $f_i$ . Às partículas no conjunto ordenado é atribuído um valor serial que varia entre 1 e  $s$ , onde  $s$  é a quantidade de partículas no conjunto solução, seguindo a sua ordem no conjunto ordenado. Para as partículas com índice par, o membro do arquivo mais próximo com maior valor de aptidão para o objetivo  $f_i$  é escolhido como  $g_{best2}$  para estas partículas. Por outro lado, para as partículas com índice ímpar no conjunto ordenado, o membro mais próximo do arquivo com menor valor de aptidão para o objetivo  $f_i$  é escolhido como  $g_{best2}$  para estas partículas. A Figura 1(b) ilustra a estratégia de escolha do  $g_{best2}$  para o objetivo  $f_1$ .

No intuito de permitir a exploração de soluções em áreas inalcançáveis pelo processo de busca guiada, Chiu *et al.* propuseram o uso de um operador de mutação em conjunto com CSS-PSO. Para este operador, é aplicado um ruído Gaussiano em  $k$  dimensões do vetor velocidade de  $m$  partículas, sendo as dimensões e as partículas escolhidas aleatoriamente.

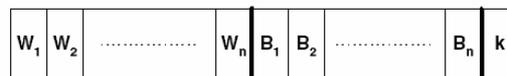
Uma vez que todos os membros do arquivo foram atualizados pelas partículas da geração atual, algumas destas soluções não-dominadas podem possuir características similares. Esta semelhança relativa entre as soluções presentes no arquivo pode gerar baixa diversidade no *Pareto Front*. Para solucionar este problema, Chiu *et al.* propuseram o uso de um procedimento de *cluster* no intuito de garantir diversidade no arquivo externo. Primeiramente, a cada partícula é atribuído o valor de um raio  $k$  dado por  $r = \sum_{i=1}^{n-1} d((f_i^{\min}, f_{i+1}^{\max}), (f_i^{\max}, f_{i+1}^{\min})) / 2s$ , onde  $s$  é o número de soluções no arquivo,  $n$  é o número de objetivos,  $d$  é a distância Euclidiana e  $f_i^{\max}$  e  $f_i^{\min}$  são os valores máximo e

mínimo do *i-ésimo* objetivo. Em seguida, para cada solução escolhida, as demais soluções que estão dentro do raio daquela solução escolhida são eliminadas do arquivo.

### 3. PSO / *k*-NN Híbrido com Operadores Genéticos

Nesta seção, descrevemos a abordagem proposta neste artigo para buscar a melhor combinação de valores para o vetor de soluções para seleção e atribuição de pesos às características de entrada do *k*-NN.

As soluções para o problema de seleção e atribuição de pesos são representadas como um vetor composto de três partes básicas. A primeira parte  $\vec{w} = \{w_1, \dots, w_n\}$  é composta de números reais conforme a quantidade de características dos padrões que serão apresentados. A segunda parte  $\vec{b} = \{b_1, \dots, b_n\}$  é composta de valores binários 0/1 que, multiplicados às características, determinarão sua existência ou não no classificador. Por fim, a terceira parte consta do valor *k* do classificador *k*-NN. A Figura 2 ilustra a forma geral de uma solução para o problema.



**Figura 2. Forma geral de uma solução para o problema de seleção de características e atribuição de pesos a essas características. Figura extraída de [Tahir *et al.* 2007].**

Tendo em vista que os pesos a serem determinados para as características podem assumir valores contínuos dentro de um intervalo de valores reais, PSO foi adotado com o objetivo de (1) melhorar a precisão de um classificador *k*-NN e (2) diminuir a quantidade de características a serem utilizadas pelo processo de classificação. Desta maneira, em nossa abordagem, cada partícula representa um conjunto de pesos a serem aplicados às características presentes em um problema de classificação. Para medir a aptidão das partículas, foram adotados os seguintes objetivos: (1) percentual de acerto obtido pela aplicação de um classificador *k*-NN com as informações codificadas em uma solução e (2) a quantidade de características a serem utilizadas no processo de classificação (*i.e.* quantidade total de características com valor  $b_i = 1$ ).

O teste de um classificador *k*-NN segue a metodologia proposta por Tahir *et al.* [Tahir *et al.* 2007]: para bases com mais de 300 padrões, 70% dos padrões são usados para treinamento e o restante para teste; para avaliar o classificador ao longo das iterações do algoritmo é usada validação cruzada com cinco blocos; a precisão do classificador é dada pela média dos resultados obtidos na validação cruzada; ao final do ciclo de busca do algoritmo, o melhor classificador *k*-NN encontrado é testado usando os 30% restantes do conjunto de dados. A precisão final de classificação é dada pelo resultado obtido sobre a base de dados de teste. Para bases de dados com menos de 300 padrões, foi usado *leave-one-out cross validation* para avaliar o classificador *k*-NN ao longo das iterações do algoritmo. Para cada base de dados os experimentos são repetidos cinco vezes, sendo apresentados média e desvio padrão da precisão final obtidos para cada base.

Para as partículas no PSO, adotamos a topologia  $g_{best}$ , para a qual todas as partículas têm conhecimento acerca das melhores soluções encontradas por todas as outras partículas no enxame [Kennedy e Eberhart 1995]. Essa topologia garante rápida

convergência para possíveis soluções para o problema uma vez que a cada iteração, para todas as partículas existe uma componente da velocidade que aponta para a melhor solução comum a todas as partículas. Logo, para a topologia  $g_{best}$ , em geral, é necessário um número relativamente pequeno de iterações para que o enxame de partículas se concentre em uma região do espaço de busca.

Essa topologia é mais adequada para o problema tratado em nossa abordagem, já que o número de iterações a serem realizadas pelo PSO para este problema é relativamente pequeno (*e.g.* 300 iterações). Logo, é necessário dar prioridade à convergência do algoritmo em detrimento da necessidade de tentar encontrar a melhor solução para o problema.

Essa decisão se deve principalmente ao fato de que o PSO em geral necessita de uma população relativamente grande (*e.g.* 30 partículas) para que, da colaboração entre as partículas, possa efetivamente emergir um comportamento inteligente de busca. Caso contrário, uma população com poucas partículas pode gerar baixa diversidade, o que em geral resulta rápida estagnação do enxame que, diferentemente da convergência do enxame, é ruim para o processo de busca.

Tendo em vista que o número de partículas de um enxame para o PSO é relativamente grande e para o problema em questão a avaliação de uma partícula é um processo bastante custoso, resta apenas diminuir o número de iterações a serem executadas para o PSO. Com isso, espera-se que o processo de otimização como um todo seja equiparado em termos de custo computacional à solução proposta por Tahir *et al.*.

Como descrito anteriormente, em uma solução para o problema em questão constam os seguintes valores: (i) vetor de pesos a serem aplicados às características presentes no problema, (ii) vetor unário que informa a presença ou não de uma característica no cálculo da distância entre dois padrões, e (iii) o número  $k$  de vizinho a serem usados pelo  $k$ -NN em seu processo de classificação.

Devido à natureza discreta dos valores presentes em (ii) e (iii), foi adotada uma abordagem baseada em mutação guiada por dois líderes com o objetivo de encontrar a melhor combinação de valores para as partes (ii) e (iii) em cada solução.

Inicialmente, um conjunto de  $S - 1$  soluções é criado aleatoriamente, estando os valores de  $\vec{w} = \{w_1, \dots, w_n\}$  distribuídos uniformemente em  $[0,10]$ . Os valores de  $\vec{b} = \{b_1, \dots, b_n\}$  podem assumir os valores unários 0 ou 1. O valor de  $k$  é um valor ímpar escolhido aleatoriamente em  $[1,9]$ . Sendo  $\vec{w} = \{1, \dots, 1\}$ ,  $\vec{b} = \{1, \dots, 1\}$  e  $k = 1$  a solução mais simples para o problema, esta solução é inserida na população inicial, perfazendo um total  $S$  de soluções inicialmente criadas para o problema.

Um conjunto  $S_0$  é gerado aleatoriamente, como descrito anteriormente, contendo 100 soluções para o classificador. Em seguida, o conjunto  $S_0$  é avaliado segundo a metodologia anteriormente descrita. Por fim, a técnica CSS-MOPSO é aplicada ao conjunto  $S_0$ , gerando um novo conjunto  $S_1$ . Para cada solução em  $S_1$  é aplicado o mecanismo de mutação guiada pelos líderes escolhidos durante a iteração do CSS-MOPSO.

Após esse processo, tem-se um novo conjunto  $S_2$  de soluções. Sobre o conjunto  $S' = S_0 \cup S_2$  é aplicado o mecanismo de agrupamento de soluções em *fronts* quanto à dominância, como descrito em [Deb *et al.* 2002]. As soluções presentes nos primeiros *fronts* são adicionadas à nova população  $S_3$  até que o tamanho máximo da população seja atingido. Caso o tamanho máximo da população seja ultrapassado, o mecanismo de ranqueamento *crowding-distance* [Deb *et al.* 2002] é aplicado às soluções no primeiro *front* não adicionado à população. Serão admitidas na nova população apenas as soluções presentes naquele *front* com melhor *crowding-distance*. Esse processo inicia novamente para  $S_3$  e se repete até que o critério de parada seja satisfeito.

Neste trabalho, para cada solução, é calculado o número de mutações a serem efetuadas em função da aptidão desta solução como  $n_{mut} = (d/2)((f_i - f_{min}) / (f_{max} - f_{min})) + 1$ , onde  $d$  é o número de características presentes no problema. Logo, o número máximo de mutações é  $n_{mut} = (d/2) + 1$  para o caso em que  $f_i = f_{max}$  e o número mínimo de mutações é 1, para o caso em que  $f_i = f_{min}$ . É importante salientar que neste caso, quanto menor o valor de  $f_i$ , menor será o número de mutações aplicadas sobre a respectiva solução. Por isso,  $f_i$  é definido como o valor do erro de classificação obtido pelo teste de um  $k$ -NN com as informações correspondentes àqueles presentes na  $i$ -ésima solução. O número máximo de mutações foi definido desta forma, pois este valor mostrou-se mais apropriado para a maioria dos experimentos realizados.

Em relação a  $\vec{b}$ , as  $n$  mutações são compartilhadas com o  $g_{best1}$  e  $g_{best2}$  para cada iteração. Ou seja,  $n/2$  valores de um indivíduo assumirão os respectivos valores presentes no  $g_{best1}$  enquanto que  $n/2$  valores restantes de  $\vec{b}$  assumirão os respectivos valores presentes em  $g_{best2}$ . O valor de  $k$  é escolhido aleatoriamente em [1,9] para todas as soluções. Desta forma, a cada iteração, o valor de  $k$  para todas as soluções é alterado randomicamente.

Portanto, CSS-PSO e mutação guiada foram usados nesta abordagem com o intuito de buscar o melhor conjunto de parâmetros (*e.g.* pesos para as características, características presentes no cálculo da distância, e quantidade  $k$  de vizinhos) que resulta no melhor classificador para uma dada base de dados. Ao final das  $I$  iterações previstas para um ciclo de busca, as soluções presentes no arquivo externo são apresentadas ao usuário como aquelas que resultam em um  $k$ -NN com maior taxa de acerto para uma base de dados específica e, ao mesmo tempo, que usa uma quantidade relativamente pequena de características para efetuar a classificação.

#### 4. Experimentos

Os experimentos realizados por Tahir *et al.* [Tahir *et al.* 2007] foram repetidos neste trabalho com o objetivo de comparar os resultados obtidos usando  $TS / k$ -NN com os resultados obtidos usando a proposta deste trabalho. As bases de dados utilizadas neste trabalho foram as mesmas utilizadas por Tahir *et al.* em seus experimentos.

Para todos os experimentos, os parâmetros usados para o PSO foram:  $c_1 = c_2 = 2,05$ ; o tamanho da população foi de 100 partículas e  $w = 0,7$ .

Para cada experimento, o cálculo do número de iterações para o PSO em nossa abordagem para cada base de dados foi baseado nos valores presentes na Tabela 4 de [Tahir *et al.* 2007] como  $I = (M * N + P) * 500 / |S|$ , onde  $|S|$  é o número de partículas do enxame.  $(M * N + P)$  representa a quantidade total de vizinhos para a abordagem de Tahir *et al.*. Em todos os experimentos, Tahir *et al.* utilizaram 500 iterações. Logo, a quantidade total de  $k$ -NNs avaliados é dada por:  $(M * N + P) * 500$ . Então, após  $I$  iterações, a abordagem proposta neste trabalho é equivalente à abordagem proposta por Tahir *et al.* em relação à quantidade de  $k$ -NNs avaliados. Esta escolha visa apenas tornar justa a comparação dos resultados obtidos nesse trabalho com os resultados apresentados por Tahir *et al.*.

O primeiro experimento realizado neste trabalho visou comparar a taxa de erro do melhor  $k$ -NN obtido ao final do ciclo de busca da solução proposta com a taxa de erro apresentada na Tabela 2 de [Tahir *et al.* 2007] para todas as bases de dados. Os resultados para o primeiro experimento estão listados na Tabela 1. Para as bases de dados com menos de 300 padrões, os resultados apresentados para a abordagem CSS-PSO /  $k$ -NN correspondem ao classificador com menor erro de classificação dentre os classificadores presentes no arquivo externo obtido ao final do processo de busca. Para todas as bases de dados, os resultados apresentados para a abordagem CSS-PSO /  $k$ -NN correspondem à média e desvio padrão do erro obtido pelo classificador com menor erro de classificação dentre os classificadores presentes no arquivo externo obtido ao final do processo de busca.

**Tabela 1. Valores dos Erros Médios Obtidos para cada uma das Propostas.**

	Aust.	Bal.	Diab.	Glass	Heart	Iono.	Iris	Liver	Son.	Vehi.
TS / $k$ -NN	10,20	10,90	22,30	19,60	37,40	12,20	3,33	26,20	5,80	26,30
CSS-PSO / $k$ -NN	12,27 (0,004)	12,28 (0,007)	22,94 (0,005)	<b>0,00</b> (0,00)	<b>14,29</b> (0,008)	<b>8,29</b> (0,005)	<b>2,40</b> (0,004)	26,76 (0,003)	10,58 (0,01)	<b>25,83</b> (0,007)

Aust. = Australian, Bal. = Balance, Diab. = Diabetes, Iono. = Ionosphere, Son. = Sonar e Vehi. = Vehicle.

Como é possível observar a partir dos dados na Tabela 1, usando a abordagem proposta neste trabalho, foi possível melhorar os resultados em relação àqueles alcançados pela abordagem TS /  $k$ -NN [Tahir *et al.* 2007] para as bases de dados em destaque. Além disso, em quatro bases os métodos obtiveram desempenhos similares. Em apenas uma base (Son.) o resultado para o método de Tahir *et al.* foi relativamente distante do resultado obtido neste trabalho.

A Tabela 2 ilustra a quantidade total de  $k$ -NNs avaliados ao final do ciclo de busca e o tempo total em minutos para avaliar tal quantidade para as abordagens TS /  $k$ -NN e CSS-PSO /  $k$ -NN. A configuração do hardware utilizado foi: Intel Core 2 Quad Q6600 (2400 MHz), 8 GB DDR2 SDRAM e sistema operacional Ubuntu 8.04.

A Tabela 2 mostra que para as bases de dados *Glass*, *Heart*, e *Vehicle* o tempo total para a execução da proposta deste trabalho foi menor do que o valor para a

proposta em TS /  $k$ -NN e adicionalmente, conforme apresentado na Tabela 1, foi possível melhorar a precisão do classificador.

**Tabela 2. Comparativo entre os Tempos de Execução para TS /  $k$ -NN e CSS-PSO /  $k$ -NN.**

	Qt. Total $k$ -NNs Avaliados	Tempo Trein. e Teste (TS / $k$ -NN)	Tempo Trein. e Teste (CSS-PSO / $k$ -NN)
Australian	24*500 = 12000	140,0	68,22
Balance	22*500 = 11000	16,3	33,31
Diabetes	23*500 = 11500	169,3	65,32
Glass	23*500 = 11500	13,1	13,08
Heart	24*500 = 12000	25,5	25,4
Ionosphere	26*500 = 13000	23,34	39,27
Iris	22*500 = 11000	4,5	5,09
Liver	23*500 = 11500	15,24	11,94
Sonar	29*500 = 14500	24,45	68,06
Vehicle	25*500 = 12500	153,38	135,01

Por fim, a Tabela 3 mostra um comparativo entre os melhores resultados obtidos pelas abordagens TS /  $k$ -NN e CSS-PSO /  $k$ -NN para cada uma das bases de dados. A Tabela 3 apresentada neste trabalho é adaptada da Tabela 3 presente no trabalho de Tahir *et al.*. Da Tabela 3 pode-se observar que para as bases de dados *Glass*, *Heart*, *Iris* e *Vehicle*, foi possível reduzir a quantidade de características e ainda obter melhores resultados para o classificador  $k$ -NN resultante em relação à abordagem TS /  $k$ -NN.

**Tabela 3. Comparativo entre os Melhores Resultados para TS /  $k$ -NN e CSS-PSO /  $k$ -NN.**

Método	Base de Dados	E	F	K	Base de Dados	E	F	K
TS / $k$ -NN	Balance	10,6	4	9	Iris	3,3	4	9
CSS-PSO / $k$ -NN		11,33	1	3		<b>2,0</b>	<b>1</b>	<b>3</b>
TS / $k$ -NN	Liver	24,0	4	3	Diabetes	20,1	4	5
CSS-PSO / $k$ -NN		26,31	2	7		22,03	2	1
TS / $k$ -NN	Glass	19,6	6	1	Heart	37,4	8	7
CSS-PSO / $k$ -NN		<b>0,0</b>	<b>1</b>	<b>9</b>		<b>13,70</b>	<b>3</b>	<b>1</b>
TS / $k$ -NN	Australian	7,7	9	3	Vehicle	25,4	13	3
CSS-PSO / $k$ -NN		11,83	5	3		<b>24,76</b>	<b>7</b>	<b>1</b>
TS / $k$ -NN	Ionosphere	5,7	15	3	Sonar	5,8	17	1
CSS-PSO / $k$ -NN		7,68	16	3		9,13	19	9

E = taxa de erro, F = número de características, K = valor de  $K$  no classificador  $k$ -NN.

## 5. Conclusão

Neste trabalho, foi proposta uma nova abordagem multi-objetiva híbrida para seleção de características e atribuição de pesos a estas características simultaneamente em um classificador  $k$ -NN. A abordagem proposta tem como objetivos: (a) melhorar a precisão

de um classificador  $k$ -NN e (b) diminuir o número de características utilizadas pelo classificador.

Em relação à proposta de Tahir *et al.* [Tahir *et al.* 2007], a abordagem apresentada neste trabalho foi capaz de melhorar a precisão de um classificador  $k$ -NN para as bases de dados *Glass, Heart, Iris e Vehicle*. Adicionalmente, para aquelas bases de dados, a nossa abordagem ainda foi capaz de reduzir o número de características utilizadas na classificação.

Em apenas uma das dez bases de dados a proposta de Tahir *et al.* obteve precisão consideravelmente superior ao método proposto neste artigo. Entretanto, é necessário um estudo mais detalhado com o objetivo de avaliar a viabilidade da aplicação da proposta em termos de melhoria da precisão e esforço computacional necessário para tal melhoria.

Como trabalhos futuros, pretende-se: (i) investigar o uso de outras abordagens para a seleção de características; (ii) investigar o uso de outras métricas a serem usadas como possíveis objetivos para este problema multi-objetivo em substituição à quantidade de características, e (iii) avaliar o desempenho da abordagem para bases realmente grandes.

## Referências

- Chiu, S.-Y., Sun, T.-Y., Hsieh, S.-T. e Lin, C.-W. (2007) “Cross-Searching Strategy for Multi-objective Particle Swarm Optimization”, IEEE Congress on Evolutionary Computation (CEC 2007), pp. 3135-3141.
- Coello Coello, C. A., e Lechuga, M. S. (2002) “MOPSO: A proposal for multiple objective particle swarm optimization,” in Proc. of IEEE World Congress on Computational Intelligence (CEC’02), pp. 1051-1056.
- Deb, K., Pratap, A., Agarwal, S. e Meyarivan, T. (2002) “A fast and elitist multiobjective genetic algorithm: NSGA-II,” IEEE Transactions on Evolutionary Computation, vol. 6, pp. 182-197.
- Duda, R. O., Hart, P. E., Stork, D. G. (2000) “Pattern Recognition”, Second edition, Wiley-Interscience.
- Glover, F., Laguna, M., 1989. Tabu search I. ORSA J. Comput. 1 (3), 190–206.
- Goldberg, D. E. (1989) “Genetic Algorithms in Search, Optimization and Machine Learning”. Reading MA: Addison Wesley.
- Kennedy, J. e Eberhart, R. C. (1995) “Particle Swarm Optimization”, In Proc. of IEEE ICNN, Perth, Australia, pp. 1942-1948.
- Parsopoulos, K. E. e Vrahatis, M. N. (2002) “Particle swarm optimization method in multiobjective,” in Proc. of the ACM Symposium on Applied Computing (SAC 2002), pp. 603-607.
- Tahir, M. A., Bouridane, A. e Kurugollu, F. (2007) “Simultaneous feature selection and feature weighting using Hybrid Tabu Search/K-nearest neighbor classifier,” Pattern Recognition Letters, Vol. 28, Issue 4, pp. 438-446.