Meta-Aprendizado de Algoritmos de Treinamento para Redes Multi-Layer Perceptron

Silvio Guerra, Ricardo Prudêncio, Teresa Ludermir

[1]Centro de Informática, Universidade Federal de Pernambuco
Caixa Postal 7851 - CEP 50732-970 - Recife (PE) - Brasil

silvio.guerra@gmail.com, rbcp@cin.ufpe.br, tbl@cin.ufpe.br

Abstract. *Meta-Learning aims to associate the performance of learning algorithms to features of the problems being tackled. In this work, we investigate the use of Meta-Learning to predict the performance of the Backpropagation (BP) and Levenberg-Marquardt (LM) algorithms, used to train MLP networks. The meta-examples were generated from the application of BP and LM algorithms in 50 regression problems. Each meta-example stored 10 features describing a given problem, and a class attribute that indicated the specific pattern of performance obtained by the algorithms in the problem. Three different classifiers were evaluated to predict the performance class, yielding promissing results.*

Resumo. *Meta-Aprendizado tem como objetivo associar o desempenho de algoritmos de aprendizado com as características dos problemas em que são aplicados. Nesse trabalho, investigamos o uso de Meta-Aprendizado para predizer o desempenho dos algoritmos Backpropagation (BP) e Levenberg-Marquardt (LM), usados no treinamento de redes MLP. Os meta-exemplos foram gerados a partir da aplicação dos algoritmos BP e LM em 50 problemas de regressão. Cada meta-exemplo armazenou 10 características descrevendo um problema específico, e um atributo classe indicando o padrão específico de desempenho obtido pelos algoritmos no problema. Três classificadores foram avaliados para predizer a classe de desempenho dos algoritmos, com resultados promissores.*

1. Introdução

Um dos principais desafios em Aprendizado de Máquina é determinar as propriedades de um problema que tornam um algoritmo de aprendizado mais apropriado que outro [Kalousis et al. 2004]. As abordagens mais tradicionais para a escolha de algoritmos envolvem, em geral, procedimentos de tentativa e erro, ou requerem conhecimento especialista, nem sempre simples de adquirir [Giraud-Carrier et al. 2004]. Meta-Aprendizado surge, nesse contexto, como uma solução mais efetiva, capaz de fornecer ao usuário um auxílio automático e sistemático para a seleção de algoritmos [Vilalta and Drissi 2002, Giraud-Carrier et al. 2004].

Em Meta-Aprendizado, cada exemplo de treinamento (ou *meta-exemplo*) armazena a experiência obtida com a aplicação de um ou mais algoritmos de apredizado em um problema resolvido no passado. Mais especificamente, cada meta-exemplo armazena: (1) os *meta-atributos* do problema, i.e. características descritivas do problema, como número de exemplos e número de atributos; e (2) as informações sobre o desempenho obtido pelos algoritmos para o problema, como as taxas de erro obtidas. Um modelo de

aprendizado (o *meta-aprendiz*) é então aplicado sobre um conjunto de meta-exemplos, e adquire conhecimento de forma automática capaz de predizer o desempenho dos algoritmos de aprendizado a partir das características dos problemas.

Nesse trabalho, aplicamos Meta-Aprendizado para predizer o desempenho de algoritmos de treinamento de redes Multi-Layer Perceptron (MLP) [Rumelhart et al. 1986]. A rede MLP é um dos modelos mais conhecidos de Redes Neurais Artificias (RNAs), e já foi aplicada com sucesso em diferentes problemas. O algoritmo usado para ajuste dos pesos é um aspecto relevante para o bom desempenho dessas redes [Prechelt 1994]. Desta forma, é importante a investigação de estratégias que auxiliem a escolha desses algoritmos. Ressaltamos ainda que poucos trabalhos na literatura sobre Meta-Aprendizado têm focado atenção para algoritmos relacionados ao paradigma de RNAs [Kalousis et al. 2004, Prudêncio and Ludermir 2004].

Para testar a viabilidade da nossa proposta, realizamos um estudo de caso com dois algoritmos de treinamento para redes MLP: o algoritmo Backpropagation [Rumelhart et al. 1986] e o algoritmo Levenberg-Marquardt [Levenberg 1944]. Para gerar um conjunto de meta-exemplos, inicialmente os dois algoritmos foram avaliados em 50 conjuntos de dados relacionados a problemas de regressão, e as taxas de erro obtidas foram coletadas. Em seguida, as taxas de erro obtidas pelos algoritmos nos 50 problemas foram analisadas por um algoritmo de clustering com o objetivo de encontrar grupos de problemas em que os algoritmos obtiveram padrões específicos de desempenho. Cada meta-exemplo foi então formado por 10 meta-atributos e o valor de um atributo de classe, correspondendo a um grupo identificado no processo de clustering. Na geração de meta-exemplos, o algoritmo de clustering foi executado duas vezes para identificar 2 e 3 grupos de problemas, respectivamente. Assim, dois conjuntos de 50 meta-exemplos foram gerados, com respectivamente 2 e 3 possíveis valores para o atributo classe.

O meta-aprendiz, no nosso estudo, corresponde a um algoritmo de classificação, usado para predizer a classe dos problemas a partir dos valores de seus meta-atributos. Nos nossos experimentos, usamos três classificadores implementados no ambiente WEKA [Witten and Frank 2003]: (1) o IBk (uma versão do algoritmo k-NN); (2) o J48 (um algoritmo de árvores de decisão); e (3) o algoritmo Naive Bayes. Esses algoritmos foram avaliados com validação cruzada nos dois conjuntos de meta-exemplos construídos. A taxa de acerto obtida pelos algoritmos avaliados variou de 76% a 92%, o que representou, no nosso estudo de caso, um desempenho consistente na classificação.

O restante desse trabalho está organizado como se segue. A seção 2 apresenta uma breve revisão da literatura sobre o tema de Meta-Aprendizado. A seção 3 apresenta os detalhes do trabalho desenvolvido e resultados obtidos nos experimentos. Finalmente, a seção 4 tece algumas considerações finais e apresenta os trabalhos futuros.

2. Meta-Aprendizado

De acordo com [Vilalta and Drissi 2002], existem diferentes interpretações para o termo *Meta-Aprendizado*. No nosso trabalho, focamos na definição de Meta-Aprendizado como o processo de predizer o desempenho de algoritmos de aprendizado a partir de características dos problemas em que são aplicados [Giraud-Carrier et al. 2004]. Nesse contexto, cada exemplo de treinamento, ou *meta-exemplo*, é relacionado a um problema específico, e armazena: (1) as características descritivas do problema, os chamados *meta-*

atributos; e (2) informações sobre o desempenho de um ou mais algoritmos de aprendizado quando aplicados ao problema. A partir de um conjunto de tais exemplos, um *meta-aprendiz* adquire conhecimento de forma automática, relacionando características dos problemas e o desempenho dos algoritmos.

Os meta-atributos são, em geral, estatísticas calculadas sobre os conjuntos de treinamento, como número de exemplos, número de atributos, correlações entre atributos, entropia, dentre outros [Brazdil et al. 2003, Kalousis et al. 2004]. Cada meta-exemplo pode armazenar como informação de desempenho um atributo classe que indica o melhor algoritmo para o problema, dentre um conjunto de candidatos [Prudêncio et al. 2004, Leite and Brazdil 2005]. A etiquetagem (i.e. a definição da classe) de cada meta-exemplo de treinamento é feita através de um experimento de validação cruzada usando os dados disponíveis do problema. O meta-aprendiz, nesse caso, é um classificador que prediz o melhor algoritmo para cada novo problema abordado.

Em [Michie et al. 1994], os autores etiquetaram os meta-exemplos de uma maneira alternativa. Inicialmente, 20 algoritmos foram avaliados em 22 problemas de classificação, através de validação cruzada. Para cada algoritmo, os autores geraram um conjunto de 22 meta-exemplos, etiquetados com uma das classes *aplicável* ou *não-aplicável*. Um meta-exemplo era etiquetado com a classe *aplicável* quando o erro de classificação obtido pelo algoritmo caía dentro de um intervalo de confiança pré-definido, e era etiquetado com a classe *não-aplicável* caso contrário. O algoritmo C4.5 foi usado como meta-aprendiz para predizer a aplicabilidade dos algoritmos candidatos.

Em [Kalousis et al. 2004], os autores realizam a etiquetagem dos meta-exemplos usando um algoritmo de clustering. Inicialmente, os autores estimaram a taxa de erro de 10 algoritmos em 80 problemas de classificação, e definiram uma matriz 80 X 10, onde cada linha armazenava as posições obtidas pelos algoritmos em um problema. Essa matriz foi fornecida como entrada para um algoritmo de clustering, com o objetivo de descobrir grupos de problemas com padrões específicos para o desempenho dos algoritmos (e.g. um grupo em que um conjunto específico de algoritmos se sobressaiu em relação aos demais). Os meta-exemplos foram então etiquetados com valores de classe associados aos grupos identificados. Assim, em vez de prever o melhor algoritmo, ou ainda a aplicabilidade de algoritmos, o meta-aprendiz pode prever padrões mais complexos de desempenho.

A abordagem de Meta-Regressão [Bensusan and Alexandros 2001] tenta prever diretamente a precisão dos algoritmos, em vez de simplesmente fornecer um valor de classe associado ao desempenho dos algoritmos. Aqui, cada meta-exemplo armazena o valor numérico da precisão obtido por cada algoritmo candidato. Em [Bensusan and Alexandros 2001], os autores usaram modelos de regressão linear para predizer a taxa de acerto de 8 algoritmos de classificação. O meta-aprendiz pode ser usado para selecionar o algoritmo candidato com melhor precisão prevista, ou ainda para fornecer um ranking de algoritmos conforme a ordem das precisões previstas.

Em [Kalousis and Theoharis 1999, Prudêncio and Ludermir 2004], os autores usam uma combinação de meta-aprendizes para fornecer uma ordenação (ou ranking) para os algoritmos candidatos. Nesses trabalhos, para cada par de algoritmos candidatos (X,Y) é construído um meta-aprendiz responsável por prever o melhor algoritmo dentre o par considerado. Dado um novo problema de aprendizado, as respostas dos meta-

aprendizes são coletadas, e pontos são creditados aos algoritmos conforme as respostas. Por exemplo, se 'X' é a resposta do meta-aprendiz (X,Y), então um ponto é creditado ao algoritmo X. A ordem final dos algoritmos candidatos é definida diretamente a partir do número total de pontos que cada algoritmo recebeu.

Em [Brazdil et al. 2003, dos Santos et al. 2004], os autores usaram aprendizado baseado em instâncias (no nível meta) para produzir ranking de algoritmos candidatos levando em consideração precisão e tempo de execução dos algoritmos. Cada meta-exemplo, nesse caso, armazena como informações de desempenho a precisão e o tempo de execução obtidos pelos algoritmos candidatos em problemas no passado. Dado um novo problema, os meta-exemplos mais similares são recuperados baseado na similaridade entre meta-atributos. O ranking de algoritmos é então gerado, agregando as informações de desempenho obtidas pelos algoritmos para os problemas recuperados. Isso é feito através do uso de uma medida de avaliação multi-critério que considerada precisão e tempo, com a importância relativa definida pelo usuário.

3. Trabalho Desenvolvido

Nesse trabalho, propomos o uso de Meta-Aprendizado para algoritmos de treinamento de redes MLP [Rumelhart et al. 1986]. O algoritmo usado no ajuste dos pesos da rede MLP é um fator relevante para o seu desempenho e assim, é importante a investigação de técnicas que auxiliem uma escolha adequada. Destacamos que poucos trabalhos em Meta-Aprendizado têm focado esforços no paradigma de Redes Neurais Artificias (RNAs). Em [Kalousis et al. 2004], os autores aplicaram Meta-Aprendizado para 10 algoritmos, que incluiam uma rede MLP e uma rede Radial Basis Function. Em [Prudêncio and Ludermir 2004], os autores usaram Meta-Aprendizado para seleção de modelos de previsão, incluindo uma rede Time-Delay Neural Network. A diferença da nossa proposta para os trabalhos anteriores é que investigamos um aspecto específico das redes MLP, que é o algoritmo usado para ajuste dos pesos. Assim, acreditamos que nosso trabalho traz contribuições para a pesquisa em Meta-Aprendizado no contexto das RNAs.

Para testar a viabilidade da nossa proposta, realizamos um estudo de caso com dois algoritmos de treinamento: o Backpropagation (BP) [Rumelhart et al. 1986] e o Levenberg-Marquardt (LM) [Levenberg 1944]. Inicialmente, o desempenho dos dois algoritmos foi estimado para 50 problemas de regressão. Cada problema foi descrito através de um conjunto de 10 meta-atributos. Os meta-exemplos foram etiquetados com valores de classe definidos em uma análise de clustering aplicada sobre o desempenho dos algoritmos. Finalmente, três classificadores foram usados como meta-aprendizes, visando prever as classes associadas aos problemas a partir dos valores dos meta-atributos.

Nas próximas subseções, apresentamos detalhes dos pontos mais importantes do trabalho desenvolvido, assim como os experimentos que avaliaram o Meta-Aprendizado.

3.1. Conjuntos de Dados

Para gerar o conjunto de meta-exemplos, coletamos 50 conjuntos de dados referentes a problemas de regressão de diversos domínios, disponibilizados no site do projeto Weka [1].

[1]Esses problemas se referem mais especificamente a 50 conjuntos de dados armazenados nas amostras *numeric* e *regression* disponibilizadas para download em http://www.cs.waikato.ac.nz/ml/weka/

Em média, os conjuntos de dados coletados apresentam 4.392 exemplos e 13,92 atributos. Percebemos nessa amostra uma grande variação tanto no número de exemplos quanto no número de atributos dos conjuntos. Essa variação pode se mostrar conveniente para estudos de Meta-Aprendizado uma vez que se espera que os algoritmos tenham comportamentos significativamente diferentes dependendo do problema analisado.

Os atributos dos conjuntos de dados originais foram normalizados para o intervalo [-1; +1], com o objetivo de melhor tratamento por parte das redes MLP. Os conjuntos tiveram ainda a ordem de seus exemplos alterada de maneira aleatória, para minimizar alguma eventual tendência no processo de coleta dos dados do conjunto original.

3.2. Avaliação do Desempenho dos Algoritmos

Um passo importante para gerar o conjunto de meta-exemplos é a avaliação dos algoritmos BP e LM para cada um dos 50 problemas de regressão coletados. Para definir o desempenho dos algoritmos em cada problema, adotamos a metodologia descrita abaixo.

Cada conjunto de dados foi dividido nos subconjuntos de treinamento, validação e teste, na proporção de 50%, 25% e 25%, respectivamente. Como usual, o conjunto de treinamento é usado para ajuste de pesos, o conjunto de validação é usado para estimar a capacidade de generalização da rede durante o treinamento, e o conjunto de teste é usado para avaliar o desempenho final da rede. O melhor número de neurônios da camada oculta[2] foi definido experimentalmente, testando os valores 1, 2, 4, 8, 16 e 32, mas limitado pelo tamanho da camada de entrada da rede. Para cada número de neurônios ocultos, o algoritmo sendo avaliado (BP ou LM) foi executado 10 vezes com inicialização aleatória dos pesos da rede. Destacamos que os algoritmos BP e LM foram implementados com o auxílio do toolbox NNET [Demuth and Beale 1993], e as taxas de aprendizado foram definidas como os valores default do toolbox. Os critérios de parada do treinamento da rede seguiram as regras benchmarking definidas em [Prechelt 1994].

O melhor número de neurônios ocultos foi selecionado como o que obteve a menor média do erro NSSE (Normalized Sum of Squared Errors) no conjunto de validação para as 10 execuções do algoritmo de treinamento. O erro NSSE é definido como:

$$NSSE = \frac{\sum_{i=1}^{n}(y_i - \widetilde{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \quad (1)$$

Na fórmula acima, n é o número de exemplos no conjunto de validação, y_i e $\widetilde{y}_i$ são respectivamente o valor alvo e o valor previsto para o exemplo i, e $\overline{y}$ é a média dos valores alvo. O desempenho final da rede MLP com o algoritmo sendo avaliado é definido como o erro NSSE médio obtido no conjunto de teste pela melhor configuração de rede selecionada no treinamento.

De uma forma geral, o algoritmo LM obteve resultados superiores ao algoritmo BP, considerando os 50 problemas analisados. De fato, o algoritmo LM obteve o menor erro NSSE de teste em 38 problemas (total de 76% dos conjuntos de dados). O bom desempenho do algoritmo LM pode ser visualizado ainda através da figura 1, que mostra um gráfico box-plot para os erros NSSE obtidos pelos algoritmos BP e LM nos 50 problemas. A superioridade do algoritmo LM pode ser vista, considerando o nível das caixas.

[2] A rede MLP foi definida com uma única camada oculta.

O tamanho das caixas indica ainda que a variação do desempenho obtido pelo algoritmo LM foi menor que a do algoritmo BP. Para alguns conjuntos de dados, o algoritmo BP obteve de fato um desempenho muito fraco (incluindo um conjunto de dados com erro NSSE = 3,2 que está representado no gráfico como um ponto espúrio). Ressaltamos aqui que essa análise inicial é apenas uma descrição geral dos resultados obtidos por BP e LM. Como veremos, o processo de Meta-Aprendizado será capaz de discriminar melhor esses resultados, associando características dos problemas ao desempenho dos algoritmos.

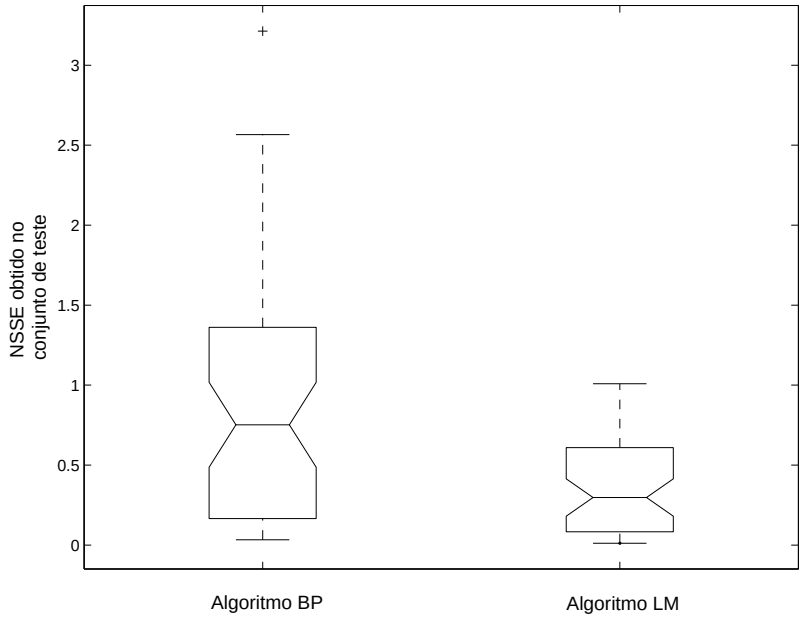

Figure 1. Gráfico Box-Plot com os erros NSSE obtidos pelos algoritmos BP e LM.

3.3. Meta-Atributos

No trabalho desenvolvido, um total de 10 meta-atributos foi usado para descrever os conjuntos de dados de problemas de regressão:

1. LogE - Log do número de exemplos de treinamento;
2. LogEA - Log da razão entre o número de exemplos e o número de atributos;
3. MinCA, MaxCA, MedCA e DesvCA - Mínimo, máximo, média e desvio padrão dos valores absolutos da correlação entre os atributos preditores e o atributo alvo;
4. MinCAtr, MaxCAtr, MedCAtr e DesvCAtr - Mínimo, máximo, média e desvio padrão dos valores absolutos da correlação entre pares de atributos preditores.

O meta-atributo LogE é um indicador do montante de dados disponíveis para treinamento, e LogEA, por sua vez, indica a dimensionalidade do conjunto de dados. Os meta-atributos MinCA, MaxCA, MedCA e DesvCA indicam a quantidade de informação relevante contida nos atributos preditores para a predição do atributo alvo. Os meta-atributos MinCAtr, MaxCAtr, MedCAtr e DesvCAtr, por sua vez, indicam o grau de redundância presente no conjunto de dados. Esse conjunto de meta-atributos foi escolhido, considerando características adotadas em trabalhos anteriores [Brazdil et al. 2003, Kalousis et al. 2004]. Como esse conjunto pode ser não ótimo, em trabalhos futuros novos meta-atributos serão considerados.

3.4. Meta-Exemplos

Cada meta-exemplo é associado a um problema e armazena: (1) os valores dos meta-atributos (as 10 características definidas na seção 3.3); e (2) o valor da classe, que indica um padrão de desempenho dos algoritmos BP e LM. Os possíveis valores de classe foram definidos através de um processo de clustering aplicado sobre o desempenho dos algoritmos. A motivação é encontrar grupos (ou clusters) de problemas com padrões similares para o desempenho obtido pelos algoritmos [Kalousis et al. 2004]. Os valores possíveis de classe correspondem aos grupos identificados no processo de clustering.

No nosso trabalho, aplicamos o algoritmo de k-Médias[3] sobre os erros NSSE obtidos por BP e LM nos 50 conjuntos de dados. Os objetos a serem agrupados consistem de pontos em um espaço bidimensional, onde cada ponto representa os erros NSSE obtidos pelos algoritmos BP e LM para um conjunto de dados específico. Obviamente, o agrupamento desses pontos poderia ser feito através de uma inspeção visual, uma vez que se trata de um espaço bidimensional. No entanto, aplicamos um algoritmo automático para obter uma maior segurança em relação aos grupos identificados. O algoritmo k-Médias foi executado com os valores de $k = 2$ e $k = 3$. Assim, a partir das duas execuções, teremos dois conjuntos de 50 meta-exemplos, etiquetados respectivamente com 2 e 3 possíveis valores de classe. A interpretação dos grupos identificados será apresentada logo abaixo.

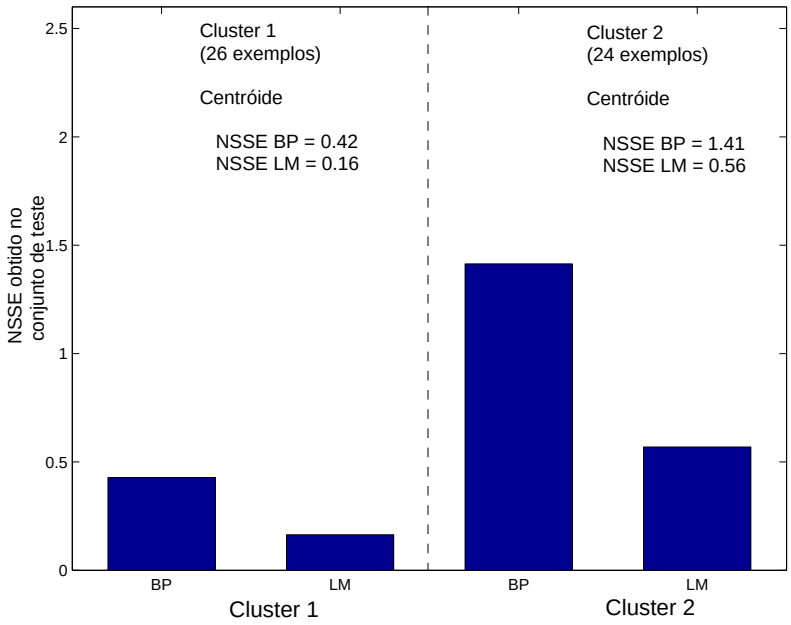

Figure 2. Representação dos centróides dos grupos identificados pelo algoritmo k-Médias com parâmetro k=2.

A figura 2 mostra os centróides (ponto central) dos 2 clusters encontrados pelo algoritmo k-Médias na sua primeira execução. O cluster 1 é formado por 26 problemas, cuja média dos erros NSSE de teste obtidos por BP e LM foi, respectivamente, 0,42 e 0,16. O cluster 2, por sua vez, é formado por 24 problemas, cujos erros NSSE para BP e LM foram, respectivamente, 1,41 e 0,56. O cluster 1 pode ser interpretado como um grupo de

[3]No nosso trabalho, usamos a implementação do algoritmo k-Médias disponível no ambiente WEKA [Witten and Frank 2003]

problemas em que tanto o algoritmo LM como o algoritmo BP obtiveram bons resultados no aprendizado, embora o algoritmo LM tenha se mostrado superior. O cluster 2, por sua vez, pode ser interpretado como um grupo de problemas em que o algoritmo BP obteve desempenho fraco, o algoritmo LM obteve desempenho intermediário, e a diferença entre os dois algoritmos foi maior que a observada no cluster 1.

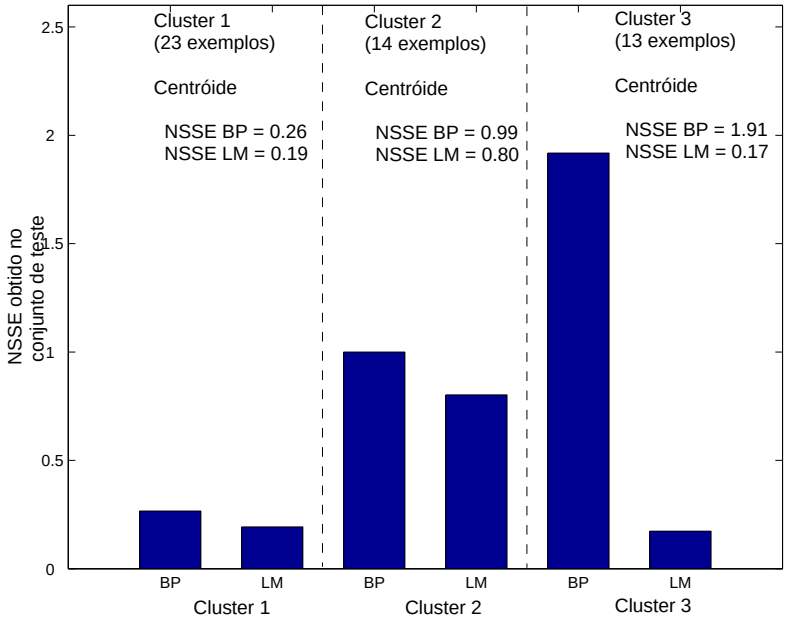

Figure 3. Representação dos centróides dos grupos identificados pelo algoritmo k-Médias com parâmetro k=3.

A figura 3 mostra os centróides dos 3 clusters encontrados pelo algoritmo k-Médias na execução com $k = 3$. O cluster 1 é formado por 23 problemas, e é representado pelo centróide com erros NSSE iguais a 0,26 e 0,19 para BP e LM, respectivamente. Esse cluster pode ser interpretado como um grupo de problemas em que os dois algoritmos avaliados obtiveram um bom desempenho. O cluster 2 é formado por 14 problemas, com centróide igual a 0,99 e 0,80, e pode ser interpretado como um grupo de problemas em que os dois algoritmos apresentam desempenho relativamente fraco. O cluster 3, formado por 13 exemplos e com centróide igual a 1,91 e 0,17, corresponde a problemas em que os algoritmos apresentaram resultados bastante distintos. Nesse grupo, o algoritmo BP teve um desempenho muito fraco, enquanto que o algoritmo LM obteve um bom desempenho.

3.5. Experimentos e Resultados com Meta-Aprendizado

Nessa seção, será avaliado o desempenho do processo de Meta-Aprendizado para os 2 conjuntos de meta-exemplos gerados na seção 3.4. Nesses experimentos, usamos como meta-aprendiz três classificadores implementados no ambiente Weka [Witten and Frank 2003], cada um representando uma família diferente de algoritmos de aprendizado: o algoritmo IBk, equivalente ao algoritmo k-NN; o algoritmo J48, uma variação do algoritmo C4.5 para árvores de decisão; e o algoritmo Naive Bayes (NB). Os algoritmos foram avaliados através de validação cruzada com 10 partições. Os parâmetros de cada algoritmo foram definidos como os valores *default* utilizados no ambiente Weka.

Table 1. Resultados obtidos para o conjunto de meta-exemplos com 2 classes.

	IBk	J48	NB	Default
Taxa de Erro	76%	84%	76%	52%
I.C. (95%)	[74,3%; 77,7%]	[82,6%; 85,4%]	[74,3%; 77,7%]	[50%; 54%]

Table 2. Resultados obtidos para o conjunto de meta-exemplos com 3 classes.

	IBk	J48	NB	Default
Taxa de Erro	84%	92%	76%	46%
I.C. (95%)	[82,6%; 85,4%]	[90.9%; 93.1%]	[74.3%; 77.7%]	[44%; 48%]

As tabelas 1 e 2 apresentam os resultados obtidos com o uso dos três classificadores como meta-aprendiz, considerando respectivamente os conjuntos de meta-exemplos com 2 e 3 valores de classe. Como base de comparação, apresentamos também o erro *default*, obtido por um classificador que sempre associa a um exemplo de teste, o valor da classe majoritária nos exemplos de treinamento (i.e. a classe mais freqüente).

Como pode ser visto, para os dois conjuntos de meta-exemplos, os classificadores avaliados apresentaram um ganho de desempenho quando comparados ao classificador default. Isso indica que os meta-atributos usados para descrição dos problemas contêm, de fato, informação útil para prever o desempenho dos algoritmos BP e LM. O ganho de desempenho sobre o classificador default foi estatisticamente confirmado, analisando o intervalo de 95% de confiança dos erros de classificação. Detalhes sobre a construção do intervalo de confiança podem ser encontrados em [Mitchel 1997].

4. Conclusão

Nesse trabalho, investigamos o uso de Meta-Aprendizado para adquirir conhecimento relacionando o desempenho de algoritmos de treinamento de redes MLP, com as características dos problemas de aprendizado. As contribuições do presente trabalho podem ser destacadas em dois contextos diferentes. No contexto de Meta-Aprendizado, aplicamos seus conceitos e técnicas em um estudo de caso ainda não investigado na literatura. Na área RNAs, investigamos uma estratégia que pode ser usada de forma efetiva para analisar o desempenho dos algoritmos de treinamento para redes MLP.

A viabilidade da proposta foi verificada com meta-exemplos gerados a partir da aplicação dos algoritmos BP e LM em 50 problemas de regressão. Três classificadores foram usados como meta-aprendizes para predizer a classe de desempenho dos algoritmos de treinamento. Os experimentos que avaliaram esses classificadores apresentaram resultados promissores.

Alguns trabalhos futuros podem ser apontados. Inicialmente, destacamos que as taxas de aprendizado dos algoritmos BP e LM assumiram os valores default definidos no toolbox NNET. Assim, em trabalhos futuros, pretendemos variar os valores das taxas de aprendizado, e analisar o efeito dessa variação no desempenho dos algoritmos. No mesmo sentido, podemos investigar o efeito do número de neurônios ocultos no desempenho dos algoritmos. No presente trabalho, consideramos apenas os resultados obtidos com a melhor rede. Trabalhos futuros incluem ainda o uso de outros algoritmos de treinamento,

o aumento no número de meta-atributos e a aplicação de outras abordagens de Meta-Aprendizado capazes de avaliar, por exemplo, os tempos de execução dos algoritmos.

References

Bensusan, H. and Alexandros, K. (2001). Estimating the predictive accuracy of a classifier. In *Proceedings of the 12th European Conference on Machine Learning*, pages 25–36.

Brazdil, P., Soares, C., and da Costa, J. (2003). Ranking learning algorithms: Using IBL and meta-learning on accuracy and time results. *Machine Learning*, 50(3):251–277.

Demuth, H. and Beale, M. (1993). *Neural Network Toolbox: For use with MATLAB: User's Guide*. The Mathworks.

dos Santos, P., Ludermir, T. B., and Prudêncio, R. B. C. (2004). Selection of time series forecasting models based on performance information. In *4th International Conference on Hybrid Intelligent Systems*, pages 366–371.

Giraud-Carrier, C., Vilalta, R., and Brazdil, P. (2004). Introduction to the special issue on meta-learning. *Machine Learning*, 54(3):187–193.

Kalousis, A., Gama, J., and Hilario, M. (2004). On data and algorithms - understanding inductive performance. *Machine Learning*, 54(3):275–312.

Kalousis, A. and Theoharis, T. (1999). Noemon: Design, implementation and performance results of an intelligent assistant for classifier selection. *Intelligent Data Analysis*, 3(5):319–337.

Leite, R. and Brazdil, P. (2005). Predicting relative performance of classifiers from samples. In *Proceedings of the 22nd International Conference on Machine Learning*.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathmatics*, II(2):164–168.

Michie, D., Spiegelhalter, D., and Taylor, C. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.

Mitchel, T., editor (1997). *Machine Learning*. MacGraw Hill, New York.

Prechelt, L. (1994). A set of neural network benckmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Information, Universitä Karlsruhe, Karlsruhe, Germany.

Prudêncio, R. B. C. and Ludermir, T. B. (2004). Meta-learning approaches to selecting time series models. *Neurocomputing*, 61:121–137.

Prudêncio, R. B. C., Ludermir, T. B., and de Carvalho, F. A. T. (2004). A modal symbolic classifier to select time series models. *Pattern Recognition Letters*, 25(8):911–921.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by backpropagating errors. *Nature*, 323:533–536.

Vilalta, R. and Drissi, Y. (2002). A perspective view and survey of meta-learning. *Journal of Artificial Intelligence Review*, 18(2):77–95.

Witten, I. H. and Frank, E., editors (2003). *WEKA: machine learning algorithms in Java*. University of Waikato, New Zealand.