Aprendizagem Ativa para Seleção de Exemplos em Meta-Aprendizado

Ricardo B. C. Prudêncio and Teresa B. Ludermir

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

rbcp@cin.ufpe.br, tbl@cin.ufpe.br

Abstract. *Meta-Learning has been used to select algorithms based on the features of the problems in which the algorithms can be applied. Each meta-example stores the features of a given problem and the performance information related to the candidate algorithms in the problem. The construction of a set of meta-examples may be costly, since the algorithm performance is usually defined through an empirical evaluation on the problem at hand. In this context, we proposed the use of Active Learning to select only the relevant meta-examples and hence, to reduce the need for empirical evaluations of the candidate algorithms. Experiments were performed using the kNN algorithm was as meta-learner and an uncertainty criteria was applied to select meta-examples. A significant gain in performance was yielded by selecting about 6% of the available meta-examples.*

Resumo. *Meta-Aprendizado tem sido usado com sucesso para selecionar algoritmos a partir das características dos problemas em que podem ser aplicados. Cada meta-exemplo armazena as características de um dado problema e as informações sobre o desempenho dos algoritmos candidatos no problema (e.g. precisão obtida). A construção de um conjunto de meta-exemplos pode ser custosa, uma vez que o desempenho dos algoritmos é usualmente definido através de avaliação empírica no problema em questão. Dentro desse contexto, propomos o uso de Aprendizagem Ativa para selecionar apenas os meta-exemplos mais relevantes, e assim, diminuir a necessidade de avaliações empíricas com os algoritmos candidatos. Experimentos foram realizados usando o algoritmo kNN como meta-aprendiz e um critério de incerteza foi aplicado para selecionar meta-exemplos. Um ganho significativo de desempenho foi obtido com cerca de 6% dos meta-exemplos disponíveis.*

1. Introdução

Em diversos domínios de aplicação, podemos observar uma ampla disponibilidade de algoritmos que podem ser aplicados para um mesmo tipo de problema. Como exemplo, no campo de Aprendizado de Máquina, temos uma grande diversidade de algoritmos candidatos para problemas de classificação e regressão. Uma questão importante que surge nesses domínios é a seleção de algoritmos adequados para cada problema, uma vez que nenhum algoritmo pode ser considerado o melhor independente do problema abordado [Kalousis et al. 2004].

As estratégias mais tradicionais para a seleção de algoritmos envolvem, em geral, conhecimento especialista, ou custosos processos de avaliação empírica

[Kalousis and Hilario 2003]. Meta-Aprendizado [Giraud-Carrier et al. 2004] surge dentro desse contexto como uma alternativa interessante, capaz de adquirir de forma automática conhecimento usado para seleção de algoritmos.

O conhecimento no Meta-Aprendizado é adquirido a partir de um conjunto de *meta-exemplos*, que armazena a experiência obtida na aplicação de um conjunto de algoritmos candidatos em problemas de aprendizado investigados no passado. Mais especificamente, cada meta-exemplo armazena: (1) as características que descrevem um dado problema (e.g. número de exemplos de treinamento, número de atributos, entropia do atributo classe,...); e (2) informações de desempenho dos algoritmos quando aplicados ao problema (e.g. melhor algoritmo, erros obtidos, tempos de execução,...). Um meta-aprendiz é um modelo de aprendizado usado para relacionar características dos problemas com o desempenho dos algoritmos candidatos.

Uma limitação do Meta-Aprendizado diz respeito ao processo de construção dos meta-exemplos. Para gerar um meta-exemplo a partir de um dado problema, é preciso realizar um processo de avaliação empírica (em geral, validação cruzada) para coletar as informações sobre o desempenho dos algoritmos candidatos. Embora a proposta do Meta-Aprendizado seja realizar essa avaliação empírica apenas para um conjunto limitado de problemas, o custo do processo de construção de meta-exemplos pode ser alto, dependendo, por exemplo, da complexidade dos algoritmos candidatos, da metodologia de avaliação empírica e da própria quantidade de problemas considerados.

Dentro do contexto acima, apresentamos aqui uma proposta original onde Aprendizagem Ativa [Lindenbaum et al. 2004] é usada para a seleção meta-exemplos. Aprendizagem Ativa é um paradigma de Aprendizado de Máquina, usado em domínios em que é difícil ou custoso adquirir exemplos de treinamento. A motivação principal é reduzir o número de exemplos de treinamento, ao mesmo tempo mantendo o desempenho dos algoritmos de aprendizado. Na nossa aplicação, isso corresponde a diminuir o tamanho do conjunto de meta-exemplos, selecionando apenas os problemas considerados mais relevantes e, consequentemente, reduzindo o número de avaliações empíricas realizadas com os algoritmos candidatos.

Para avaliar a viabilidade da nossa proposta, implementamos um protótipo onde o algoritmo kNN (k-Nearest Neighbours) é usado como meta-aprendiz, e um critério baseado em incerteza na classificação [Lindenbaum et al. 2004] é usado para selecionar meta-exemplos. Experimentos foram realizados para um problema de seleção de algoritmos de regressão (usados especificamente para previsão de séries temporais). O algoritmo de Aprendizagem Ativa foi comparado com um algoritmo de seleção randômica, em uma base de 645 meta-exemplos. Os experimentos realizados revelaram que um ganho de desempenho foi obtido com a Aprendizagem Ativa, quando foram selecionados cerca de apenas 6% do conjunto total de meta-exemplos disponíveis.

O restante do trabalho é organizado como se segue. Seção 2 apresenta uma breve explanação sobre Meta-Aprendizado, com trabalhos desenvolvidos na literatura. Seção 3 descreve com mais detalhes a solução proposta e o protótipo implementado. Seção 4 apresenta os experimentos realizados e os resultados obtidos. Finalmente, a seção 5 conclui o artigo, apresentando considerações finais e os trabalhos futuros.

2. Meta-Aprendizado para Seleção de Algoritmos

Meta-Aprendizado é um arcabouço desenvolvido no campo de Aprendizado de Máquina com o objetivo de selecionar algoritmos de aprendizado [Giraud-Carrier et al. 2004]. Meta-Aprendizado usa exemplos empíricos para produzir um modelo de aprendizado (chamado meta-aprendiz) responsável por prever o desempenho dos algoritmos candidatos a partir de características dos problemas sendo abordados.

Em uma formulação mais simples de Meta-Aprendizado (adotada, por exemplo, em [Prudêncio et al. 2004, Leite and Brazdil 2005]), cada exemplo de treinamento, ou meta-exemplo, é relacionado a um problema de aprendizado e armazena: (1) as características descritivas do problema, os chamados *meta-atributos*; e (2) um valor de classe associado ao algoritmo que obteve o melhor desempenho para o problema, dentre um conjunto de algoritmos candidatos. Nessa formulação, o Meta-Aprendizado se torna simplesmente uma tarefa de classificação, onde os meta-atributos são usados como atributos preditores, e a classe associada ao melhor algoritmo corresponde ao atributo alvo a ser previsto. Os meta-atributos são, em grande parte dos trabalhos na área, estatísticas calculadas sobre os conjuntos de dados de treinamento [Engels and Theusinger 1998] (e.g., número de exemplos, número de atributos, correlações entre atributos,...). O meta-exemplo é etiquetado (i.e. o melhor algoritmo é definido), através de um experimento de validação cruzada com os algoritmos candidatos, usando os dados disponíveis do problema.

Diferentes abordagens têm sido propostas para adicionar novas capacidades ao processo de Meta-Aprendizado. No sistema NOEMON [Kalousis and Theoharis 1999], por exemplo, um conjunto de meta-aprendizes é usado não apenas para predizer o melhor algoritmo (como descrito acima), mas para fornecer uma ordenação (ou ranking) dos algoritmos candidatos. Nesse sistema, para cada par de algoritmos candidatos (X, Y) é construído um meta-aprendiz simples. Dado um novo problema de aprendizado, NOEMON coleta as respostas dos meta-aprendizes e credita pontos aos algoritmos candidatos dependendo das respostas fornecidas. Para um meta-aprendiz (X, Y), por exemplo, se 'X' é a classe prevista, então um ponto é creditado ao algoritmo X; caso contrário, um ponto é creditado ao algoritmo Y. A ordem final dos algoritmos candidatos é definida diretamente a partir do número total de pontos que cada algoritmo recebeu. Abordagens similares foram adotadas ainda em [Prudêncio and Ludermir 2004, Kalousis et al. 2004].

Em [Soares and Brazdil 2000, Brazdil et al. 2003], os autores usaram aprendizado baseado em instâncias (no nível meta) para produzir ranking de algoritmos candidatos levando em consideração múltiplos critérios de desempenho. No sistema Zoomed-Ranking [Soares and Brazdil 2000], cada meta-exemplo armazena, além dos meta-atributos, a precisão e o tempo de execução obtidos pelos algoritmos candidatos. Dado um novo problema, o Zoomed-Ranking recupera os meta-exemplos mais similares baseado na similaridade entre meta-atributos. O ranking dos algoritmos candidatos é então gerado, agregando as informações de desempenho obtidas pelos algoritmos para os problemas similares recuperados. Isso é feito através do uso de uma medida de avaliação multi-critério que combina precisão e tempo de execução obtidos por cada algoritmo.

A abordagem de Meta-Regressão [Bensusan and Alexandros 2001] tenta prever diretamente a precisão dos algoritmos (ou alternativamente o erro), em vez de simplesmente fornecer a classe que corresponde ao melhor algoritmo. Nesse caso, cada meta-exemplo deve armazenar o valor numérico da precisão obtida por cada algoritmo can-

didato. Em [Bensusan and Alexandros 2001], os autores usaram modelos de regressão linear para prever a precisão de 8 algoritmos de classificação, e obtiveram resultados aceitáveis. O meta-aprendiz, nesse caso, pode ser usado tanto para selecionar o algoritmo candidato com melhor precisão prevista, como para fornecer um ranking de algoritmos conforme a ordem das precisões previstas.

Os conceitos e técnicas de Meta-Aprendizado foram propostos originalmente para selecionar algoritmos em problemas de classificação. Em trabalhos recentes, o uso de Meta-Aprendizado tem sido extrapolado para outros domínios de aplicação. Por exemplo, em [Prudêncio and Ludermir 2004, dos Santos et al. 2004, Prudêncio and Ludermir 2006], os autores usaram diferentes técnicas de Meta-Aprendizado para selecionar modelos de previsão de séries temporais. Em [Tsoumakas et al. 2004], Meta-Aprendizado foi aplicado para configurar de forma automática sistemas de planejamento. Em tais domínios, Meta-Aprendizado se torna uma ferramenta para análise de experimentos realizados com um dado número de algoritmos em um conjunto relativamente grande de problemas do domínio. Nesse sentido, Meta-Aprendizado é um arcabouço mais geral que pode ser útil para a seleção de algoritmos relacionados a diferentes domínios de aplicação.

3. Aprendizagem Ativa para Seleção de Meta-Exemplos

Como visto, um passo importante para a geração de um conjunto de meta-exemplos é estimar o desempenho dos algoritmos candidatos nos conjuntos de dados disponíveis. A avaliação de desempenho é feita seguindo uma determinada metodologia de experimentos (e.g. validação cruzada). A informação resultante da avaliação de desempenho é usada então para definir os atributos alvo do Meta-Aprendizado (e.g. classe correspondente ao melhor algoritmo).

A geração dos meta-exemplos pode ser um processo custoso dependendo, por exemplo, da metodologia de experimentos adotada para avaliação de desempenho, do número de problemas disponíveis, e da quantidade e complexidade de algoritmos candidatos. Dentro desse contexto, propomos aqui o uso de Aprendizagem Ativa para seleção de exemplos usados no Meta-Aprendizado. Aprendizagem Ativa é um paradigma de Aprendizado de Máquina onde os algoritmos assumem algum tipo de controle sobre os exemplos usados no treinamento [Lindenbaum et al. 2004]. A principal motivação é reduzir o número de exemplos de treinamento, ao mesmo tempo mantendo a precisão do algoritmo. Aprendizagem Ativa é ideal em domínios onde é o processo de aquisição dos exemplos de treinamento é custoso, como exemplo, reconhecimento de imagens, classificação de textos, reconhecimento de fala, dentre outros.

A motivação do uso de Aprendizagem Ativa no nosso contexto é selecionar apenas os problemas mais relevantes para o Meta-Aprendizado, e assim diminuir o tempo gasto com a avaliação empírica dos algoritmos candidatos. A figura 1 mostra a arquitetura de sistema seguindo a solução proposta. O sistema tem basicamente três fases: fase de construção de meta-exemplos, fase de treinamento, e fase de uso, descritas a seguir

Na fase de construção de meta-exemplos, o módulo AA (Aprendizagem Ativa) inicialmente seleciona de uma base de problemas, aqueles que serão usados para a construção de meta-exemplos. Isso é feito através de um critério de seleção pré-definido. O conjunto de dados associado a cada problema selecionado é usado para a avaliação

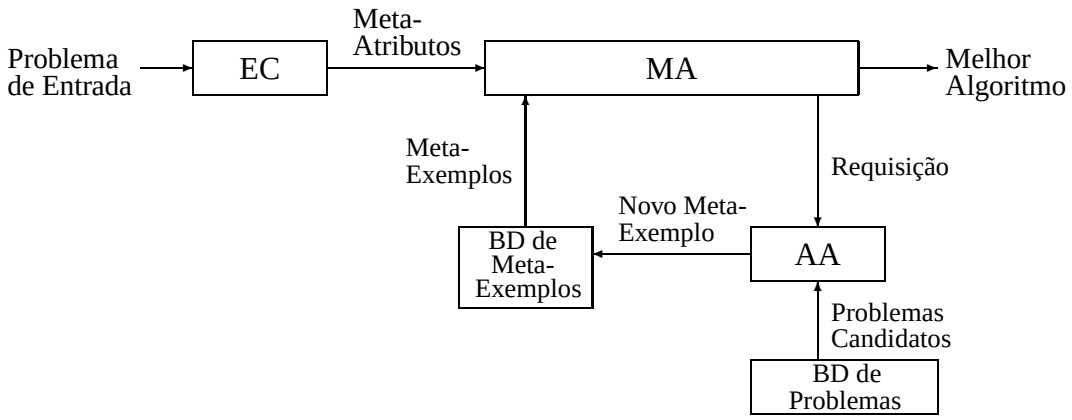

Figure 1. Arquitetura do Sistema.

empírica dos algoritmos candidatos, e os meta-exemplos construídos são armazenados em um banco de dados.

Na fase de treinamento, o módulo MA (Meta-Aprendiz) adquire conhecimento a partir do banco de meta-exemplos. O conhecimento adquirido associa meta-atributos com o desempenho obtido pelos algoritmos candidatos. Dependendo de algum critério pré-estabelecido (e.g. precisão obtida pelo meta-aprendiz), o módulo MA pode requisitar ao módulo AA que realize a seleção e construção de novos meta-exemplos. À medida que novos meta-exemplos são adquiridos, espera-se que o desempenho do módulo MA aumente.

Finalmente, na fase de uso, dado como entrada um novo problema e seu conjunto de dado, o módulo EC (Extrator de Características) computa os valores dos meta-atributos. De acordo com esses valores, o módulo MA realiza a ordenação e/ou seleção dos algoritmos candidatos. Para isso, o módulo MA usa o conhecimento previamente adquirido na fase de treinamento.

Para avaliar a solução proposta, implementamos um protótipo onde o algoritmo de kNN é usado como meta-aprendiz, e o critério proposto em [Lindenbaum et al. 2004], baseado em incerteza de classificação, é usado na aprendizagem ativa. O protótipo implementado foi usado no problema de seleção dos modelos de regressão Random Walk (RW) e Autoregressivo (AR), usados para a previsão de séries temporais. Como será visto, a base de problemas usados nos experimentos corresponde a 645 séries temporais anuais disponibilizadas na M3-Competition [Makridakis and Hibon 2000].

No que se segue, fornecemos uma descrição mais detalhada de cada módulo do protótipo implementado.

3.1. Extrator de Características

Na solução proposta, o módulo EC implementa p meta-atributos $X_1, \ldots, X_p$ usados para descrição de problemas. Cada problema e é descrito por um vetor $\mathbf{x} = (x^1, ..., x^p)$ no qual $x^j = X_j(e)$, para $j = 1, \ldots, p$.

No protótipo implementado, usamos $p = 4$ características para descrever as séries anuais da M3-Competition:

1. Tamanho (X_1): número de observações da série;
2. Tendência Básica (X_2): inclinação do modelo de regressão linear ajustado para os dados da série;
3. Percentagem de Turning Points (X_3): Z_t é um turning point se $Z_{t-1} < Z_t > Z_{t+1}$ ou $Z_{t-1} > Z_t < Z_{t+1}$. Essa característica indica o grau de oscilação da série;
4. Primeiro Coeficiente de Autocorrelação (X_4): valores altos para essa característica sugerem que o valor da série em um ponto influencia fortemente o valor da série no ponto seguinte.

As quatro características implementadas nesse módulo correspondem a estatísticas calculadas diretamente a partir dos dados disponíveis da série temporal. Como esse conjunto é possivelmente não ótimo, em futuras implementações serão considerados outros meta-atributos.

3.2. Meta-Aprendiz

No protótipo implementado, utilizamos o algoritmo kNN (k-Nearest Neighbors) como um meta-aprendiz em que as classes associadas aos melhores algoritmos são previstas a partir dos meta-atributos. Esse meta-aprendiz corresponde a formulação simples de Meta-Aprendizado apresentada na seção 2. Em [Brazdil et al. 2003], os autores apresentam vantagens para o uso de algoritmos baseados em instâncias, como o kNN, no processo de Meta-Aprendizado. Em especial, quando um novo meta-exemplo se torna disponível, ele pode ser facilmente integrado aos resultados existentes sem a necessidade de reiniciar o aprendizado do meta-aprendiz.

Descrevemos aqui o meta-aprendiz usado no protótipo. Seja $E = \{e_1, \ldots, e_n\}$ um conjunto de n problemas, onde cada problema é descrito pelos p meta-atributos $X_1, \ldots, X_p$, e associado a um atributo C que indica o melhor algoritmo para o problema, dentre os algoritmos candidatos.

Seja $D = \{c_1, \ldots, c_L\}$ o domínio da variável C onde cada valor de classe $c_l \in D$ representa um algoritmo candidato. Desta forma, cada problema $e_i \in E$ ($i = 1, \ldots, n$) é representado como um vetor $\mathbf{x}_i = (x_i^1, \ldots, x_i^p)$, onde $x_i^j = X_j(e_i)$ ($j = 1, \ldots, p$), e é associado à classe $C(e_i) = c_l$, onde $c_l \in D$. O meta-exemplo associado a e_i é definido então como o par $(\mathbf{x}_i, C(e_i))$.

Dado um novo problema descrito pelo vetor $\mathbf{x} = (x^1, \ldots, x^p)$, um número de k meta-exemplos são recuperados do conjunto de treinamento de acordo com a distância entre os meta-atributos. A função de distância usada no protótipo implementado foi a Norma L_1 sem peso, definida como:

$$dist(\mathbf{x}, \mathbf{x}_i) = \sum_{j=1}^{p} \frac{|x^j - x_i^j|}{max_i(x_i^j) - min_i(x_i^j)} \quad (1)$$

A previsão do melhor algoritmo é feito de acordo com o número de ocorrências (i.e. votos) de cada valor de classe no conjunto de meta-exemplos recuperados.

3.3. Aprendizagem Ativa

Como visto, o módulo Meta-Aprendiz dispõe para seu treinamento de um conjunto E de problemas *etiquetados*, i.e. problemas para os quais o valor do melhor algoritmo foi

definido. O módulo de Aprendizagem Ativa (AA), por sua vez, dispõe de um conjunto $\widetilde{E}$ de problemas *não-etiquetados*, que consiste no conjunto de problemas para os quais não se conhece o melhor algoritmo. Assim, o objetivo principal do módulo AA é incrementalmente selecionar problemas de $\widetilde{E}$ que serão usados para a construção de novos meta-exemplos.

No protótipo implementado, o módulo AA implementa um critério de seleção de exemplos não-etiquetados específico para o algoritmo kNN. Mais especificamente, adotamos um dos critérios propostos em [Lindenbaum et al. 2004], baseado na *incerteza de classificação*. O princípio desse critério é descrito como se segue.

Cada exemplo não-etiquetado é classificado pelo algoritmo de aprendizado usando os exemplos etiquetados até o momento. Um valor de incerteza de classificação é então definido para cada exemplo não-etiquetado, e finalmente o exemplo não-etiquetado com maior valor de incerteza é selecionado. A incerteza de classificação, considerando o algoritmo kNN, é definida em [Lindenbaum et al. 2004] como a razão entre: (1) a distância entre o exemplo e o seu vizinho etiquetado mais próximo; e (2) a soma das distâncias entre o exemplo e os vizinhos etiquetados mais próximos de classes diferentes. Alto valor de incerteza nesse caso implica que um exemplo não-etiquetado tem vizinhos com distâncias similares, mas com classes conflitantes. Assim, etiquetar esse exemplo implicaria em diminuir a incerteza de classificação na vizinhança do exemplo.

Formalmente, seja $\widetilde{E}$ o conjunto de problemas não-etiquetados, e seja E o conjunto de problemas etiquetados. Seja E_l o conjunto de problemas associados à classe c_l, i.e. $E_l = \{e_i \in E | C(e_i) = c_l\}$. Dado o conjunto E, a incerteza de classificação de um problema não-etiquetado $\widetilde{e} \in \widetilde{E}$ é definida como:

$$\mathcal{S}(\widetilde{e}|E) = \frac{\min_{e_i \in E} dist(\widetilde{\mathbf{x}}, \mathbf{x}_i)}{\sum_{l=1}^{L} \min_{e_i \in E_l} dist(\widetilde{\mathbf{x}}, \mathbf{x}_i)} \tag{2}$$

Na fórmula acima, $\widetilde{\mathbf{x}}$ é a descrição do problema $\widetilde{e}$. O módulo AA seleciona então o problema $\widetilde{e}^* \in \widetilde{E}$ de maior valor de incerteza, ou seja:

$$\widetilde{e}^* = argmax_{\widetilde{e} \in \widetilde{E}} \mathcal{S}(\widetilde{e}|E) \tag{3}$$

Finalmente, o problema selecionado é etiquetado (i.e. definição do atributo classe $C(\widetilde{e}^*)$), através da avaliação empírica dos algoritmos candidatos usando os dados disponíveis do problema. O novo meta-exemplo é gerado então com a descrição do problema, e o seu valor da classe associado.

No protótipo implementado, etiquetar uma série temporal selecionada pelo módulo AA corresponde a avaliar empiricamente os modelos de previsão RW e AR, e definir qual o melhor modelo. Para isso, a seguinte metodologia de avaliação foi adotada. Inicialmente, dada uma série temporal, os seus dados são divididos em duas partes: período de ajuste e período de teste. O período de ajuste corresponde aos dados iniciais da série, e é usado para calibrar os parâmetros dos modelos candidatos. O período de teste corresponde aos últimos 6 anos da série (como sugerido pelos autores da M3-Competition), e é usado para avaliar o desempenho de previsão dos modelos já calibrados.

O valor da classe é então definido como o modelo que obteve a menor média de erros de previsão absolutos, obtida nos dados do período de teste.

4. Experimentos e Resultados

Nos experimentos realizados, usamos as 645 séries anuais da M3-Competition, relacionadas a domínios econômicos e demográficos. Assim, 645 meta-exemplos foram produzidos.

Para avaliar a utilidade do módulo AA, realizamos um experimento de validação cruzada com 3 partições. A cada passo da validação cruzada, 215 meta-exemplos (uma partição) é usada como teste, e 430 meta-exemplos (as duas partições restantes) são fornecidos para o módulo AA para seleção de acordo com o critério definido na seção 3.3. A cada meta-exemplo selecionado, computamos o erro de classificação do módulo MA para a partição de teste. Para cada passo da validação cruzada, o módulo AA selecionou meta-exemplos incrementalmente até o limite de 100 meta-exemplos selecionados, e assim uma curva de 100 erros de classificação foi obtida. Finalmente, a curva média dos erros de classificação foi calculada, considerando os três passos da validação cruzada.

Como base de comparação, realizamos experimentos com o método de seleção randômica, onde os meta-exemplos foram selecionados de maneira aleatória do conjunto de meta-exemplos disponíveis. Segundo [Lindenbaum et al. 2004], apesar de pouco sofisticado, o método de seleção randômica tem a vantagem de explorar de maneira uniforme o espaço de exemplos, o que pode proporcionar boa competitividade. A metodologia de experimentos para avaliação do método randômico foi a mesma usada para a avaliação do módulo AA.

Na figura 2, apresentamos a curva média de erros obtidos pelo kNN, considerando o uso da aprendizagem ativa e do método de seleção randômica, e levando em consideração ainda três diferentes configurações do algoritmo kNN (com k = 1, 3 e 5 vizinhos mais próximos). As execuções de kNN com $k = 1$ e $k = 3$ obtiveram um padrão similar de resultados. Na parte inicial das curvas (que corresponde até aproximadamente os 25 primeiros exemplos selecionados), podemos observar uma forte alternância entre o desempenho da aprendizagem ativa e o da seleção randômica. No entanto, a partir de um certo número de meta-exemplos selecionados, o desempenho da aprendizagem ativa passa a ser constantemente superior ao desempenho da seleção randômica. Isso ocorre nos pontos 25 e 22 respectivamente para as execuções com $k = 1$ e $k = 3$.

Para a execução com $k = 5$, o desempenho da aprendizagem ativa é superior ao desempenho da seleção randômica em praticamente todos os pontos das curvas. De fato, apenas no ponto 1 e no ponto 27, o desempenho da seleção randômica é superior em valores absolutos. De um modo geral, considerando as três configurações do kNN, o ganho de desempenho alcançado com o uso da aprendizagem ativa passa a ser constante após a seleção de aproximadamente 6% do total de 430 meta-exemplos disponíveis.

5. Conclusão

Nesse artigo, apresentamos uma proposta original onde Aprendizagem Ativa é usada para a seleção de exemplos de treinamento para Meta-Aprendizado. Para testar a viabilidade da proposta, implementamos um protótipo onde um critério de seleção de exemplos foi

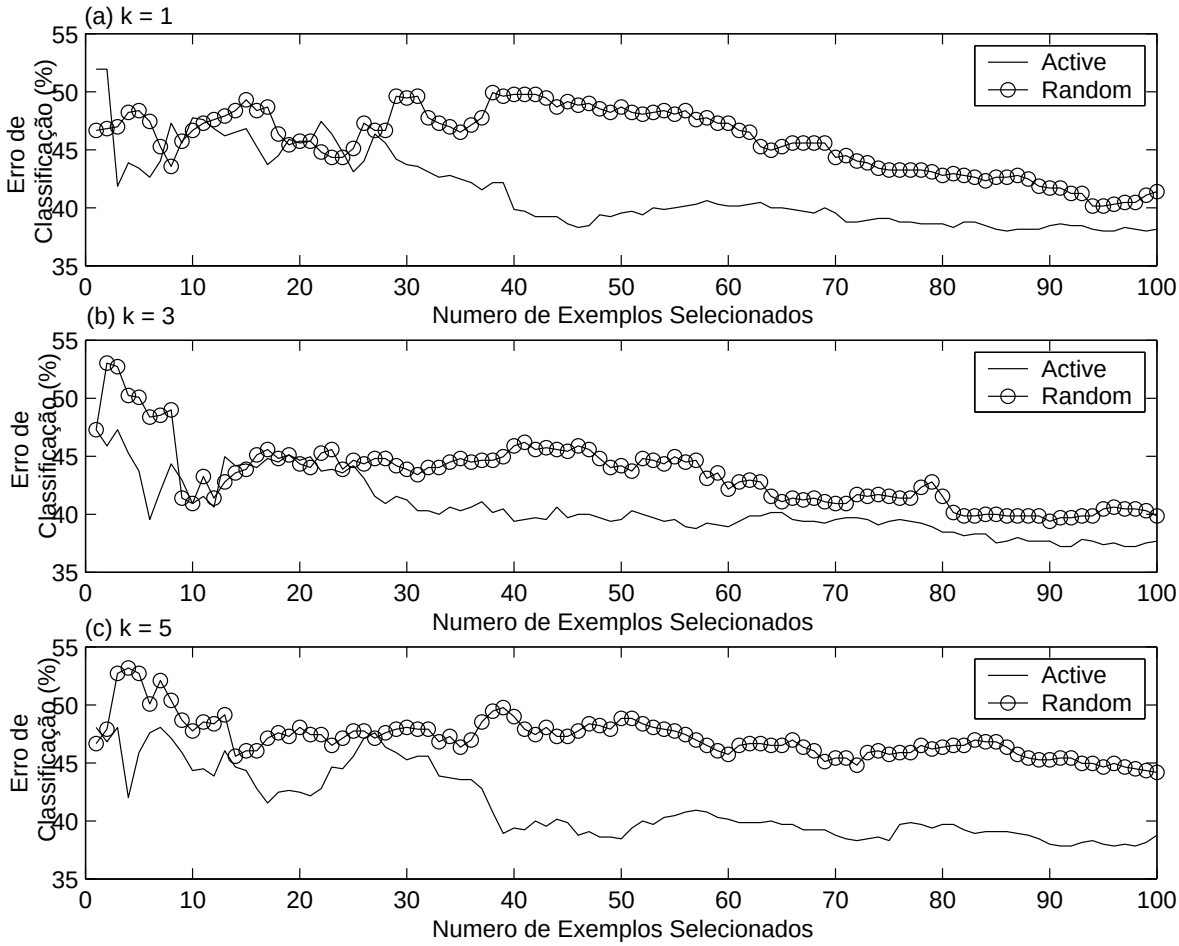

Figure 2. Curva dos erros de classificação obtidas com kNN (k = 1,3 e 5).

aplicado em um meta-aprendiz baseado no algoritmo kNN. Experimentos foram realizados no problema de seleção de modelos de séries temporais, com uma base de 645 séries anuais. Os experimentos mostraram que a Aprendizagem Ativa teve um desempenho superior a um método de seleção randômica, a partir da seleção de uma fração relativamente pequena do conjunto total de meta-exemplos disponíveis.

Destacamos aqui que a despeito da originalidade da proposta e dos bons resultados dos experimentos preliminares, o trabalho apresenta limitações. Um ponto importante a investigar é o uso de outros métodos de Aprendizagem Ativa para o algoritmo kNN. Embora o critério de seleção de exemplos aplicado no protótipo seja simples de implementar e tenha proporcionado um ganho de desempenho no meta-aprendizado, ele não é o critério que apresenta em geral os melhores resultados da literatura para o algoritmo kNN [Lindenbaum et al. 2004].

Outra questão importante é a investigação de métodos de Aprendizagem Ativa gerais, não restritos ao algoritmo kNN. Embora o kNN seja amplamente aplicado como meta-aprendiz, o uso de outros algoritmos de aprendizado também tem sido investigado. Finalmente, citamos como trabalhos futuros a aplicação de Aprendizagem Ativa em outras abordagens de Meta-Aprendizado, como as citadas na seção 2 deste artigo.

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.

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.

Engels, R. and Theusinger, C. (1998). Using a data metric for preprocessing advice for data mining applications. In Prade, H., editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, pages 430–434. John Wiley & Sons.

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 Hilario, M. (2003). Representational issues in meta-learning. In *Proceedings of the 20th International Conferente on Machine Learning*, pages 313–320.

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*.

Lindenbaum, M., Markovitch, S., and Rusakov, D. (2004). Selective sampling for nearest neighbor classifiers. *Machine Learning*, 54:125–152.

Makridakis, S. and Hibon, M. (2000). The M3-competition: Results, conclusions and implications. *International Journal of Forecasting*, 16(4):451–476.

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. and Ludermir, T. B. (2006). A machine learning approach to define weights for linear combination of forecasts. In *16th International Conference on Artificial Neural Networks*, pages 274–283.

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.

Soares, C. and Brazdil, P. (2000). Zoomed ranking - selection of classification algorithms based on relevant performance information. *Lecture Notes in Computer Science*, 1910:126–135.

Tsoumakas, G., Vrakas, D., Bassiliades, N., and Vlahavas, I. (2004). Lazy adaptive multicriteria planning. In *Proceedings of the 16th European Conference on Artificial Intelligence, ECAI04*, pages 693–697.