Um Estudo sobre a Rotulação de Exemplos no Aprendizado Semissupervisionado Multivisão

Ígor Assis Braga, Edson Takashi Matsubara, Maria Carolina Monard

Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP) – São Carlos, SP – Brasil

{igorab,edsontm,mcmonard}@icmc.usp.br

Abstract. *Semi-supervised learning combines labeled and unlabeled data during the training phase.* CO-TRAINING *is a widely used semi-supervised learning algorithm which can be applied in domains where the training examples can be described by two different views, using a method to combine the two classifiers related to each view during the labeling step. Thus, it is important to make as few errors as possible while labeling examples during the training phase in order to avoid the degradation of the generated model. As* CO-TRAINING *treats both classifiers in an independent manner, some examples may not be equally labeled by the classifiers. This work proposes another method to combine the decision of both classifiers aiming to delay the labeling of this sort of examples. The method is illustrated using a well known dataset.*

Resumo. *O aprendizado semissupervisionado combina dados rotulados e não rotulados durante a fase de treinamento.* CO-TRAINING *é um algoritmo amplamente utilizado de aprendizado semissupervisionado, o qual pode ser aplicado em domínios nos quais os exemplos de treinamento são descritos por duas diferentes descrições, usando um método para combinar os classificadores relacionados a cada descrição durante o processo de rotulação. Desse modo, é importante evitar erros de rotulação durante a fase de treinamento para que o desempenho do algoritmo não degrade. Como* CO-TRAINING *trata ambos classificadores de modo independente, alguns exemplos podem não ser igualmente rotulados por esses classificadores. Neste trabalho, é proposto um outro método de combinação da decisão dos classificadores com o objetivo de atrasar a rotulação desse tipo de exemplos. O método proposto é ilustrado utilizando uma base de dados bastante conhecida na área.*

1. Introdução

Algoritmos de aprendizado semissupervisionado utilizam um pequeno número de exemplos rotulados juntamente com um número expressivo de exemplos não rotulados para a geração de classificadores. Uma abordagem ao aprendizado semissupervisionado que tem uma forte conexão com o aprendizado humano é a multivisão [Mitchell 1999], a qual é aplicável em domínios nos quais os exemplos podem ser descritos, no mínimo, de duas formas distintas. Na classificação de páginas *web*, por exemplo, é frequentemente possível inferir a classe de uma página tanto a partir do texto contido na página quanto a partir do texto contido nos *links* que apontam para essa página.

Um algoritmo de aprendizado que implementa a abordagem multivisão e que vem sendo amplamente utilizado pela comunidade é o CO-TRAINING [Blum e Mitchell 1998].

Nesse algoritmo, dois classificadores são inicialmente gerados, cada um usando uma descrição diferente dos exemplos rotulados. Depois disso, os exemplos não rotulados são classificados, e aqueles que o foram com alta confiança são inseridos no conjunto de exemplos rotulados. A partir desse conjunto incrementado de exemplos rotulados, dois classificadores são novamente gerados e o processo é repetido. Como os exemplos rotulados em cada iteração de CO-TRAINING são utilizados para a geração dos classificadores nas próximas iterações, é de fundamental importância que o processo de rotulação seja preciso. No entanto, o processo de rotulação do algoritmo original não considera a possibilidade de que os classificadores gerados independentemente em cada descrição dos exemplos possam discordar com alta confiança nas suas classificações. Isso pode levar a erros de rotulação na maioria dos problemas reais em que CO-TRAINING pode ser aplicado. Até onde sabemos, este problema ainda não foi considerado na literatura [veja, por exemplo, Gupta et al. 2008]

Neste trabalho, são propostas modificações ao processo de rotulação de CO-TRAINING, as quais tem como objetivo dar um tratamento diferenciado à rotulação de exemplos que são classificados diferentemente por cada classificador. Para entender essas novas estratégias de rotulação, o algoritmo CO-TRAINING é apresentado na próxima seção. Em seguida, na Seção 3, a função central no processo de rotulação do CO-TRAINING original é apresentada e suas possíveis deficiências são salientadas. Ainda na Seção 3, uma nova função que modifica o processo de rotulação original é proposta. Na Seção 4, é descrito um experimento ilustrativo das ideias contidas neste trabalho, utilizando uma tarefa de classificação de páginas *web*. Por último, são apresentados possíveis trabalhos futuros e as conclusões.

2. O Algoritmo CO-TRAINING

Ao longo deste trabalho, será considerado que duas descrições distintas dos exemplos podem ser obtidas. Assim, considere que o conjunto de m atributos $A = \{A_1, A_2, ..., A_m\}$, que descreve um conjunto de exemplos, pode ser dividido em dois subconjuntos de atributos X_1 e X_2, tal que $A = X_1 \cup X_2$ e $X_1 \cap X_2 = \emptyset$. Um exemplo $\mathbf{x}$ é representado pelo par $(\mathbf{x}_1, \mathbf{x}_2)$, no qual $\mathbf{x}_1$ e $\mathbf{x}_2$ são vetores cujas componentes representam, respectivamente, os valores dos atributos nos subconjuntos X_1 e X_2. Em outras palavras, $\mathbf{x}_1$ e $\mathbf{x}_2$ correspondem às duas descrições do exemplo $\mathbf{x}$. Um exemplo rotulado, por sua vez, é representado pela tripla $(\mathbf{x}_1, \mathbf{x}_2, y)$, na qual y é um rótulo proveniente do conjunto de classes Y.

O objetivo de CO-TRAINING — Algoritmo 1 — é aprender dois classificadores a partir de um conjunto L (*labeled*) de exemplos rotulados e de um conjunto U (*unlabeled*) de exemplos não rotulados. Na primeira iteração do algoritmo, dois classificadores iniciais h_1 e h_2 são gerados repassando o conjunto L a um *algoritmo-base* de aprendizado supervisionado. Cada um dos classificadores h_1 e h_2 é gerado considerando somente a descrição correspondente $\mathbf{x}_1$ ou $\mathbf{x}_2$ dos exemplos em L. A única exigência em relação ao algoritmo-base é que ele gere classificadores que emitam um valor de confiança (*score*) para as suas classificações.

Após a obtenção de h_1 e de h_2, alguns exemplos em U são escolhidos aleatoriamente e inseridos no conjunto U'. Em seguida, os classificadores h_1 e h_2 rotulam todos os exemplos em U' usando suas respectivas descrições $\mathbf{x}_1$ e $\mathbf{x}_2$, gerando os conjuntos L'_1

e L'_2. Os exemplos em L'_1 e em L'_2 que foram rotulados com alta confiança são selecionados pela função $melhoresExemplos$, e, posteriormente, transferidos para o conjunto L. Assim, a quantidade de exemplos rotulados em L é incrementada, e, na próxima iteração, dois novos classificadores h_1 e h_2 são gerados a partir do conjunto expandido de exemplos rotulados. As iterações do algoritmo continuam até que todos os exemplos já tenham sido rotulados ou algum outro critério de parada tenha sido atingido. Pode ser observado que o critério de seleção dos melhores exemplos assim como o número de exemplos de cada classe a serem rotulados em cada iteração do algoritmo são parâmetros importantes da função $melhoresExemplos$.

Algoritmo 1: CO-TRAINING

Entrada:

- conjunto L de exemplos rotulados
- conjunto U de exemplos não rotulados

repita

usar a descrição $\mathbf{x}_1$ dos exemplos em L para gerar um classificador h_1;
usar a descrição $\mathbf{x}_2$ dos exemplos em L para gerar um classificador h_2;
retirar aleatoriamente alguns exemplos de U e adicioná-los a U';
$L'_1 =$ todos os exemplos de U' rotulados por h_1;
$L'_2 =$ todos os exemplos de U' rotulados por h_2;
$L' = melhoresExemplos(L'_1, L'_2)$;
$L = L \cup L'$;
retirar de U' os exemplos em L';

até $L' = \emptyset$;

Saída: h_1, h_2

A análise do algoritmo CO-TRAINING tem sido realizada sob duas premissas. A primeira é a pressuposição de *compatibilidade*, que afirma que os classificadores h_1 e h_2 devem concordar na classificação dos exemplos. Já a segunda premissa afirma que as descrições dos exemplos não são *muito correlacionadas*, de tal forma que existam exemplos nos quais h_1 tenha alta confiança na rotulação mas h_2 não tenha e vice-versa. Considerando a validade dessas premissas durante a execução de CO-TRAINING, um exemplo que foi classificado com alta confiança por um dos classificadores pode-se tornar um exemplo rotulado bastante informativo para o aprendizado do outro classificador. Estudos teóricos [Balcan e Blum 2006, Balcan et al. 2005] e empíricos [Muslea et al. 2002, Nigam e Ghani 2000] mostram que, sob essas condições, e também relaxando um pouco a pressuposição de compatibilidade, o algoritmo CO-TRAINING obtém bons classificadores h_1 e h_2.

3. Função $melhoresExemplos$

A função $melhoresExemplos$ proposta por Blum e Mitchell [1998], a qual será denominada ORIGINAL neste trabalho, utiliza conjuntos de exemplos com duas classes $\{\oplus, \ominus\}$ e o algoritmo *Naive Bayes* (NB) como algoritmo base *score* para gerar h_1 e h_2. Os exemplos em L'_1 e em L'_2 são considerados levando em conta o grau de confiança de h_1 e h_2 na rotulação, e ORIGINAL retorna os p exemplos $\oplus$ e os n exemplos $\ominus$ rotulados com

maior confiança por h_1 e h_2 respectivamente. Consequentemente, o número máximo de exemplos que podem ser rotulados em cada iteração é $2p + 2n$[1].

Entretanto, a função ORIGINAL não verifica por rotulações conflitantes de $h_1(\mathbf{x}_1)$ e $h_2(\mathbf{x}_2)$ do mesmo exemplo $\mathbf{x}$. Para exemplificar a rotulação realizada por ORIGINAL, considere o seguinte conjunto U' que contém 10 exemplos — Tabela 1 — para os quais NB fornece uma aproximação da probabilidade do exemplo ser classificado como $\oplus$ ou $\ominus$ para $h_1(\mathbf{x}_1)$ e $h_2(\mathbf{x}_2)$ dadas, respectivamente, pelos pares de valores (p_1, n_1) e (p_2, n_2). Para esses pares de valores, é indicada a ordem de prioridade considerando, respectivamente, p_1 e p_2 como valores de confiança e o valor $0,5$ como limiar de decisão. Considere que, em cada iteração, são rotulados por cada classificador os $p = 1$ exemplos $\oplus$ e $n = 3$ exemplos $\ominus$ rotulados com maior confiança. Para os exemplos na Tabela 1, em cada descrição, o exemplo com prioridade 1 é classificado como $\oplus$ e os exemplos com prioridade 10, 9 e 8 como $\ominus$.

	Pri. $h_1(\mathbf{x}_1)$	$h_1(\mathbf{x}_1)$	Pri. $h_2(\mathbf{x}_2)$	$h_2(\mathbf{x}_2)$	ORIGINAL	Verdadeiro	NO	AVOID
e_1	2	(0.95, 0,05) $\oplus$	4	(0.70, 0.30) $\oplus$		$\oplus$	$\oplus$	$\oplus$
e_2	3	(0.70, 0.30) $\oplus$	3	(0.80, 0.20) $\oplus$		$\oplus$		
e_3	10	(0.25, 0.75) $\ominus$	2	(0.90, 0.10) $\oplus$	$\ominus$	$\ominus$		
e_4	7	(0.40. 0.60) $\ominus$	5	(0.51, 0.49) $\oplus$		$\ominus$		
e_5	1	(1.00, 0.00) $\oplus$	10	(0.00, 1.00) $\ominus$	$\oplus$	$\ominus$		
e_6	4	(0.65, 0.35) $\oplus$	7	(0.15, 0.85) $\ominus$		$\ominus$		$\ominus$
e_7	9	(0.30, 0.70) $\ominus$	8	(0.10, 0.90) $\ominus$	$\ominus$	$\ominus$	$\ominus$	$\ominus$
e_8	6	(0.45, 0.55) $\ominus$	6	(0.45, 0.55) $\ominus$		$\ominus$		
e_9	8	(0.35, 0.65) $\ominus$	1	(0.95, 0.05) $\oplus$	$\ominus$	$\oplus$		
e_{10}	5	(0.55, 0.45) $\oplus$	9	(0.05, 0.95) $\ominus$	$\ominus$	$\ominus$		$\ominus$

Tabela 1. Rotulação de exemplos pelas funções ORIGINAL, NOCONTENTION e AVOIDCONTENTION

Em outras palavras, $h_1(\mathbf{x}_1)$ classifica e_5 $\oplus$ e e_3, e_7, e_9 $\ominus$, enquanto $h_2(\mathbf{x}_2)$ classifica e_9 $\oplus$ e e_5, e_{10}, e_7 $\ominus$. Assim, pode ser observado que há conflito na rotulação dos exemplos e_5 e e_9. Caso os exemplos rotulados por $h_1(\mathbf{x}_1)$ sejam incorporados primeiramente ao conjunto L' de exemplos rotulados, os exemplos rotulados nessa iteração de CO-TRAINING com a função ORIGINAL seriam os exemplos e_5 $\oplus$ e e_3, e_7, e_9 $\ominus$ de $h_1(\mathbf{x}_1)$ e o exemplo e_{10} de $h_2(\mathbf{x}_2)$. Com esse critério, dos 5 exemplos rotulados nessa iteração, dois deles são rotulados errados — coluna ORIGINAL da Tabela 1.

Exemplos que são classificados diferentemente por h_1 e por h_2 com alta confiança, tais como os exemplos e_5, e_3, e_6 e e_9 na Tabela 1, são chamados de *pontos de contenção*. Se um ponto de contenção tiver sido selecionado pela função ORIGINAL, então ele passa a integrar o conjunto L de exemplos rotulados, o que não é desejável, pois não é possível decidir o rótulo de um ponto de contenção sem informação adicional. Deve ser observado que essa situação pode acontecer ainda que a maioria dos exemplos sejam compatíveis, pois, nas iterações iniciais de CO-TRAINING, o número de exemplos rotulados disponíveis para gerar $h_1(\mathbf{x}_1)$ e $h_2(\mathbf{x}_2)$ é pequeno.

Matsubara [2004] propõe uma função $melhoresExemplos$, a qual será denominada NOCONTENTION neste trabalho, que *não* rotula pontos de contenção, considerando o valor de confiança dado pelos classificadores h_1 e h_2 somente a exemplos que são rotulados com a mesma classe. Definido um limiar $l > 0,5$ (o valor desse limiar l é definido

[1] $2p + 2n$ também é a quantidade mínima de exemplos que devem ser retirados de U e adicionados a U' em cada iteração do CO-TRAINING original.

pelo usuário), para exemplos rotulados $\oplus$, deve-se verificar que $p_1 \geq l$ e $p_2 \geq l$, enquanto que, para exemplos rotulados $\ominus$, deve-se verificar $p_1 \leq 1 - l$ e $p_2 \leq 1 - l$. Os exemplos que verificam essas condições são ordenados considerando o valor de $p_1 + p_2$. Logo após, até p exemplos positivos com maior valor de $p_1 + p_2$ e até n exemplos negativos com menor valor de $p_1 + p_2$ são selecionados para serem inseridos em L. Assim, o número máximo de exemplos que a função NOCONTENTION pode rotular em cada iteração de CO-TRAINING é $p + n$. Por exemplo, considerando os exemplos na Tabela 1, $l = 0.6$, $p = 1$ e $n = 3$, os exemplos e_1 e e_2 (nessa ordem de prioridade) verificam as condições para serem classificados como $\oplus$, e somente o exemplo e_7 como $\ominus$. Portanto o exemplo e_1 seria rotulado $\oplus$ e o e_7 $\ominus$, os quais são os rótulos verdadeiros desses dois exemplos — coluna NO da Tabela 1.

Entretanto, a função NOCONTENTION é muito conservadora na rotulação de exemplos, pois exige no mínimo que eles sejam classificados da mesma maneira por h_1 e por h_2. Também, não consegue rotular todos os exemplos em U pois, após várias iterações, pode não haver exemplos que verificam as condições requeridas por NOCONTENTION. Porém, usando NOCONTENTION com um alto limiar l, poderiam ser rotulados exemplos com bastante segurança, incrementando o número de exemplos em L tal que seria possível utilizar após qualquer algoritmo de aprendizado supervisionado para gerar classificadores utilizando somente os exemplos em L.

Neste trabalho é proposta uma outra solução, denominada AVOIDCONTENTION e descrita a seguir, na qual a inserção de pontos de contenção é *evitada* sempre que possível. Isso se traduz em selecionar até p exemplos positivos com maior valor de $p_1 + p_2$ e até n exemplos negativos com menor valor de $p_1 + p_2$. A diferença entre NOCONTENTION e AVOIDCONTENTION é que a primeira não insere em L os exemplos que foram rotulados diferentemente por cada classificador em uma mesma iteração, enquanto que a segunda permite inserir em L os exemplos classificados de maneira diferente em cada visão, mas que muito provavelmente não chegam a constituir pontos de contenção. Desse modo, a função AVOIDCONTENTION não é tão conservadora quanto NOCONTENTION, mas também é mais precavida que a função ORIGINAL, pois dá prioridade aos exemplos que não são pontos de contenção. Voltando ao exemplo da Tabela 1, a ordem dos exemplos segundo os valores decrescentes de p_1+p_2 seria e_1, e_2, e_9, e_3, e_5, e_4, e_8, e_6, e_{10} e e_7. Assim, AVOIDCONTENTION rotula e_1 $\oplus$ e e_6, e_{10} e e_7 $\ominus$, os quais são os rótulos verdadeiros para esses quatro exemplos — coluna AVOID da Tabela 1. Observe que exemplos como e_6 e e_{10}, que são rotulados como $\ominus$ com alta confiança por h_2 e como $\oplus$ com menos confiança por h_1, não são considerados por NOCONTENTION, mas são rotulados como $\ominus$ por AVOIDCONTENTION.

4. Experimento Ilustrativo

Para ilustrar as ideias apresentadas na seção anterior, o algoritmo CO-TRAINING foi executado utilizando as funções ORIGINAL, NOCONTENTION e AVOIDCONTENTION utilizando o conjunto de exemplos COURSES[2], o qual também foi usado por Blum e Mitchell [1998] na apresentação do CO-TRAINING original. Essa base contém 1051 exemplos referentes a páginas *web* coletadas de quatro universidades americanas. As páginas estão

[2] http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-51/www/co-training/data

em formato html e estão dividas em duas classes: páginas referentes às disciplinas oferecidas nessas universidades (*course*) e páginas dessas universidades que não se referem a disciplinas (*non-course*).

Dada uma página da base COURSES, a sua primeira descrição é formada pelo texto contido na página, enquanto que a sua segunda descrição é formada pelo texto contido nos *links* que apontam para a página. Neste trabalho, a primeira descrição é sempre referida como *texto*, e a segunda descrição como *links*. Foi observado que nem todas as páginas na descrição *links* continham algum conteúdo, e, portanto, ao realizar o pré-processamento dessa base, esses exemplos foram retirados do conjunto de exemplos. Após essa limpeza, o número de exemplos da base original foi reduzido para 1038, sendo 221 (21%) da classe *course* e 817 (79%) da classe *non-course*.

A base COURSES foi pré-processada utilizando a ferramenta PRETEXT[3] e transformada em uma tabela atributo-valor. Inicialmente, utilizando a lista de *stopwords* padrão do PRETEXT, os textos foram transformados em *stems*, que constituem os possíveis atributos. Para calcular o valor desses atributos, foi utilizada a medida *tf* (frequência de ocorrência do *stem*). Em seguida, foram aplicados os cortes de *Luhn* [Luhn 1958], nos quais é possível especificar a frequência mínima e máxima dos *stems* da coleção. Neste trabalho, foi utilizado o valor 2 para realizar o corte mínimo em ambas descrições, *i.e.* todos os *stems* que aparecem apenas 1 vez em toda a coleção foram retirados da tabela atributo-valor que representa esses documentos. O número máximo de aparições de um *stem* não foi limitado.

Todos os exemplos da base COURSES estão rotulados. Assim, é possível executar CO-TRAINING em um modo de *simulação*, *i.e.*, ocultando o rótulo de um número expressivo de exemplos e verificando se a rotulação realizada por CO-TRAINING confere com o rótulo verdadeiro dos exemplos. Esse aspecto de avaliação permite verificar se há uma degradação na rotulação dos exemplos, o que resultaria também na degradação do desempenho dos classificadores h_1 e h_2 induzidos. As medidas interessantes, nesse caso, são: o número de exemplos inicialmente rotulados $|L_{ini}|$; o número de exemplos inicialmente não rotulados $|U_{ini}|$; o número de exemplos rotulados ao fim da execução $|L_{fim}|$; o número de exemplos rotulados erroneamente $\#Errados$; a proporção de exemplos rotulados erroneamente $\%Errados$, dada pela Equação 1; e a proporção de exemplos rotulados ao fim da execução $\%Rotulados$, dado pela Equação 2.

$$\%Errados = \frac{\#Errados}{|L_{fim}| - |L_{ini}|} \tag{1}$$

$$\%Rotulados = \frac{|L_{fim}| - |L_{ini}|}{|U_{ini}|} \tag{2}$$

Além de verificar o desempenho na rotulação, foi realizado um processo de 10-*fold cross-validation* para estimar o desempenho dos classificadores produzidos pelas 3 versões de $melhoresExemplos$. Em cada uma das 10 iterações do processo de *cross-validation*, o conjunto L foi criado selecionando 30 exemplos do conjunto de treinamento, sendo 6 da classe *course* e 24 da classe *non-course*. Além disso, o conjunto U' inicial foi

[3] http://www.icmc.usp.br/~edsontm/pretext/pretext.html

composto por 75 exemplos de U selecionados aleatoriamente de acordo com a distribuição das classes, assim como foi feito em [Blum e Mitchell 1998].

Em [Matsubara et al. 2006], é verificado experimentalmente que uma escolha de p e n em uma proporção muito diferente da distribuição natural das classes tende a degradar o desempenho de CO-TRAINING. Desse modo, os valores dos parâmetros p e n foram determinados considerando a distribuição das classes do conjunto COURSES, já que essa distribuição é conhecida no modo de simulação. Para a execução de CO-TRAINING com a função ORIGINAL, os valores escolhidos foram $p = 1$ e $n = 3$, assim como foi feito em [Blum e Mitchell 1998]. Para que as funções NOCONTENTION e AVOIDCONTENTION tenham a oportunidade de rotular o mesmo número de exemplos que ORIGINAL, foram escolhidos os valores $p = 2$ e $n = 6$. No entanto, é importante ressaltar que em uma execução real de CO-TRAINING, somente é conhecido o rótulo de poucos exemplos, e inferir a distribuição das classes a partir de um conjunto de exemplos muito pequeno pode não ser válido.

Depois de finalizada a execução do algoritmo CO-TRAINING em um determinado conjunto de treinamento, foi obtido um classificador combinado a partir dos dois classificadores h_1 e h_2 gerados em cada descrição. Desse modo, foi possível calcular, no respectivo conjunto de teste, um único valor da área sob a curva ROC [Fawcett 2004], a qual permite estimar a probabilidade de sucesso de um classificador *score* independentemente do limiar de decisão escolhido. O classificador combinado utilizado neste trabalho é o mesmo utilizado em [Blum e Mitchell 1998] — Algoritmo 2. A probabilidade de um exemplo $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$ do conjunto de teste pertencer a uma classe y é calculada como o produto entre as probabilidades $P(y|\mathbf{x}_1)$ e $P(y|\mathbf{x}_2)$ obtidas pelo *Naive Bayes* em cada descrição. O exemplo $\mathbf{x}$ recebe, então, a classe y que maximiza o referido produto.

Algoritmo 2: Classificador combinado de Blum e Mitchell [1998]

Entrada: $\mathbf{x}_1, \mathbf{x}_2$
para cada $y \in \{\oplus, \ominus\}$ **faça**
  $P(y|\mathbf{x}) = P(y|\mathbf{x}_1)P(y|\mathbf{x}_2)$;
fim
Saída: $argmax_{y \in \{\oplus, \ominus\}} P(y|\mathbf{x})$

4.1. Resultados e Análise

Os resultados experimentais apresentados nesta seção têm por objetivo ilustrar o comportamento das funções NOCONTENTION e AVOIDCONTENTION em relação à função ORIGINAL. Como já mencionado anteriormente, a função ORIGINAL pode, em algumas situações, rotular pontos de contenção e as funções NOCONTENTION e AVOIDCONTENTION tentam evitá-los. Acredita-se que esses pontos, quando inseridos no conjunto de exemplos rotulados, podem degradar o desempenho do algoritmo.

Para ilustrar como CO-TRAINING comporta-se utilizando a função ORIGINAL na base COURSES, foi medido, sobre as 10 partições, o número médio de exemplos erroneamente rotulados ($\#Errados$) em relação aos valores de $p_1 + p_2$ — Figura 1. Valores de $p_1 + p_2$ próximos de 0 indicam exemplos rotulados $\ominus$ por h_1 e h_2 com alta confiança, enquanto que valores próximos de 2 indicam exemplos rotulados $\oplus$ por h_1 e h_2 com alta confiança. Valores de $p_1 + p_2$ na vizinhança de 1 indicam exemplos que potencialmente,

mas não necessariamente, são pontos de contenção. Para ilustrar, considere os exemplos e_3, e_4, e_5 e e_6 da Tabela 1 que possuem respectivamente valores de $p_1 + p_2$ iguais a $1,15$, $1,01$, 1 e $0,80$. Esses quatro exemplos são rotulados diferentemente por h_1 e h_2 e ficariam próximos de 1 no histograma, mas, além do exemplo e_5, somente e_3 e e_6 poderiam também ser considerados pontos de contenção, já que o exemplo e_4 não é rotulado com alta confiança por nenhum classificador. Esse gráfico mostra uma grande concentração de exemplos erroneamente rotulados em valores de $p_1 + p_2$ próximos de 1. Como exemplos são rotulados quando há alta confiança na sua rotulação, os exemplos cujos valores de $p_1 + p_2$ são próximos de 1 são provavelmente pontos de contenção.

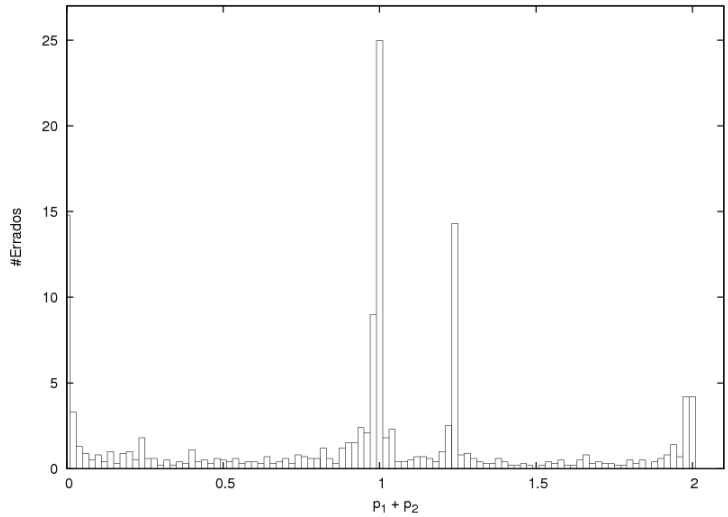

Figura 1. Número médio de exemplos erroneamente rotulados ($\#Errados$) *versus* $p_1 + p_2$

Utilizando o mesmo conjunto de dados, foi executado CO-TRAINING com as funções NOCONTENTION e AVOIDCONTENTION com o objetivo de observar a variação do número de exemplos incorretamente rotulados para cada uma das diferentes funções estudadas neste trabalho. Na Figura 2, são mostrados os resultados do número de exemplos rotulados errados *versus* a quantidade de exemplos rotulados, $|L|$, ao longo das iterações. Pode ser observado que o número de exemplos rotulados incorretamente pelas funções que possuem mecanismos para evitar os pontos de contenção é menor que o número de exemplos rotulados incorretamente por ORIGINAL. As funções AVOIDCONTENTION e NOCONTENTION possuem desempenho parecido e diferem apenas nas iterações finais de CO-TRAINING, nas quais NOCONTENTION interrompe o processo de rotulação quando atinge aproximadamente 770 exemplos rotulados, enquanto AVOIDCONTENTION continua o processo de rotulação com um incremento mais acentuado na rotulação de exemplos incorretos.

Na Tabela 2 são sumarizados os resultados finais das execuções, na qual: $melhoresExemplos$ representa as funções utilizadas por CO-TRAINING; L_{fim} o número de exemplos rotulados; $\#Errados$ a média do número de exemplos rotulados incorretamente; $\%Errados$ a média da percentagem do número de exemplos rotulados; $\%Rotulados$ a porcentagem de exemplos rotulados; e AUC a média da área sob a curva ROC para o classificador combinado. Números entre parênteses correspondem a desvios-padrão. Ainda que se observe em NOCONTENTION o menor valor de $\%Errados$, vale

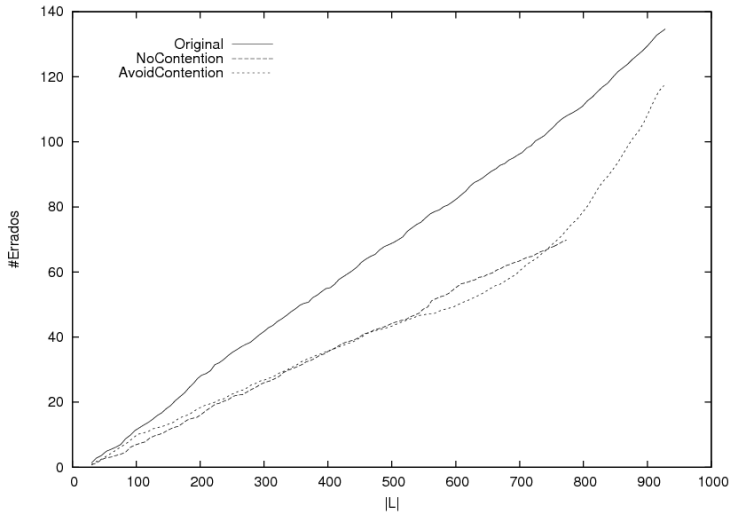

Figura 2. Número médio de exemplos erroneamente rotulados ($\#Errados$) *versus* número de exemplos rotulados ($|L|$)

ressaltar que essa função interrompe o processo de rotulação com uma média de 773 exemplos rotulados devido aos exemplos não rotulados remanescentes terem sido classificados diferentemente por h_1 e por h_2. De qualquer modo, as funções ORIGINAL e AVOIDCONTENTION são diretamente comparáveis, e observa-se que AVOIDCONTENTION tem menor valor de $\%Errados$ e maior valor de AUC do que ORIGINAL.

$melhoresExemplos$	$\|L_{fim}\|$	$\#Errados$	$\%Errados$	$\%Rotulados$	AUC
ORIGINAL	934,0 (0,0)	134,7 (53,3)	14,9	100,0	0,8 (0,2)
NOCONTENTION	773,3 (45,5)	69,8 (43,0)	9,4	82,8	0,8 (0,2)
AVOIDCONTENTION	934,0 (0,0)	117,3 (51,1)	12,6	100,0	0,9 (0,1)

Tabela 2. Resultados da rotulação nas 3 versões de $melhoresExemplos$

5. Conclusão e Trabalhos Futuros

CO-TRAINING é um algoritmo de aprendizado semissupervisionado multivisão que incrementalmente rotula os exemplos de treinamento. No artigo original em que CO-TRAINING é proposto, a função que seleciona os melhores exemplos a serem inseridos no conjunto de exemplos rotulados permite que também sejam selecionados alguns exemplos que são denominados pontos de contenção, os quais possuem um grande chance de degradar o desempenho de CO-TRAINING.

Neste trabalho foi apresentado um breve estudo sobre o efeito desses pontos de contenção no algoritmo original proposto por Blum e Mitchell [1998], bem como duas possíveis soluções para evitar a inserção de pontos de contenção no conjunto de exemplos rotulados. Uma possível solução é apresentada no trabalho de Matsubara [2004], denominada NOCONTENTION, e uma outra solução, denominada AVOIDCONTENTION, é apresentada neste trabalho. A solução NOCONTENTION, apesar de não rotular pontos de contenção, usa uma abordagem conservadora que também deixa de rotular exemplos que podem ser úteis para o aprendizado de um dos classificadores. A solução AVOIDCONTENTION, ainda que não tão conservadora quanto NOCONTENTION, é mais precavida que a solução original de CO-TRAINING, pois posterga a rotulação de pontos de contenção.

A rotulação realizada utilizando os três métodos foi ilustrada utilizando a base de dados COURSES, usada por Blum e Mitchell [1998]. Nesse estudo, foi bastante evidente que postegar a rotulação de pontos de contenção pôde contribuir de maneira significativa no desempenho do algoritmo. Como trabalho futuro, pretende-se fazer uma avaliação experimental abrangente utilizando um grande número de conjuntos de dados.

Agradecimentos

Gostaríamos de agradecer à FAPESP e ao CNPq pelo apoio a este trabalho. Também aos revisores, pelas valiosas sugestões e críticas.

Referências

Balcan, M.-F. e Blum, A. (2006). An augmented PAC model for semi-supervised learning. In *Semi-Supervised Learning (Adaptive Computation and Machine Learning)*, páginas 397–420.

Balcan, M.-F., Blum, A., e Yang, K. (2005). "CO-TRAINING and expansion: Towards bridging theory and practice". In *NIPS '04: Advances in Neural Information Processing Systems 17*, páginas 89–96.

Blum, A. e Mitchell, T. (1998). "Combining labeled and unlabeled data with CO-TRAINING". In *COLT '98: Proceedings of the 11th Annual Conference on Computational Learning Theory*, páginas 92–100.

Fawcett, T. (2004). ROC graphs: Notes and practical considerations for researchers. Relatório técnico, HP Laboratories. `http://home.comcast.net/~tom.fawcett/public_html/papers/ROC101.pdf`.

Gupta, S., Kim, J., Grauman, K., e Mooney, R. (2008). "Watch, listen & learn: CO-TRAINING on captioned images and videos". In *ECML/PKDD '08: Proceedings of the 2008 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, páginas 457–472.

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165.

Matsubara, E. T. (2004). O algoritmo de aprendizado semi-supervisionado CO-TRAINING e sua aplicação na rotulação de documentos. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo. `http://www.teses.usp.br/teses/disponiveis/55/55134/tde-19082004-092311/`.

Matsubara, E. T., Monard, M. C., e Prati, R. C. (2006). "On the class distribution labelling step sensitivity of CO-TRAINING". In *IFIP AI '06: Artificial Intelligence in Theory and Practice*, páginas 199–208.

Mitchell, T. M. (1999). "The role of unlabeled data in supervised learning". In *Proceedings of the 6th International Colloquium on Cognitive Science*, páginas 1–8.

Muslea, I., Minton, S., e Knoblock, C. (2002). "Active + semi-supervised learning = robust multi-view learning". In *ICML '02: Proceedings of the 19th International Conference on Machine Learning*, páginas 435–432.

Nigam, K. e Ghani, R. (2000). "Analyzing the effectiveness and applicability of CO-TRAINING". In *CIKM '00: Proceedings of the 9th International Conference on Information and Knowledge Management*, páginas 86–93.