

# Utilizando Algoritmos de Aprendizado Semi-supervisionados Multi-visão como Rotuladores de Textos

Edson Takashi Matsubara<sup>1</sup>, Maria Carolina Monard<sup>1</sup>, Gustavo E.A.P.A Batista<sup>2</sup>\*

<sup>1</sup>Laboratório de Inteligência Computacional  
Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo  
Av. do Trabalhador São-Carlense, 400 - Centro - Cx. Postal 668  
São Carlos - São Paulo - Brasil CEP 13560-970

<sup>2</sup>Faculdade de Engenharia de Computação  
Pontifícia Universidade Católica de Campinas  
Rodovia Dom Pedro I, km 136 - Parque das Universidades  
Campinas - São Paulo - Brasil CEP 13086-900

{edsontm, mcmonard, gbatista} at icmc usp br

**Abstract.** *The supervised learning approach to learn text classifiers usually requires a large number of labelled training examples. However, labelling is often manually performed, making this process costly and time-consuming. Multi-view semi-supervised learning algorithm have the potential of reducing the need of expensive labelled data whenever only a small set of labelled examples is available. In addition, these algorithms require a partitioned description of each example into distinct views. In this work we propose a method where these views can easily be obtained using simple and composed words from text data. Experimental results confirm the suitability of our proposal.*

**Resumo.** *O aprendizado de máquina supervisionado normalmente necessita de um número significativo de exemplos de treinamento para a indução de classificadores precisos. Entretanto, a rotulação de dados é freqüentemente realizada manualmente, o que torna esse processo demorado e caro. O aprendizado semi-supervisionado multi-visão tem o potencial de reduzir a necessidade de dados rotulados quando somente um pequeno conjunto de exemplos rotulados está disponível. No entanto, esses algoritmos necessitam de duas ou mais descrições de cada exemplo. Neste trabalho é proposta uma abordagem para exemplos texto, com a qual é simples obter essas descrições utilizando palavras simples e compostas. Os resultados experimentais confirmam a adequabilidade desta proposta.*

## 1. INTRODUÇÃO

Boa parte do conhecimento existente está em formato texto. Desse modo é bastante evidente a necessidade da aquisição automática de conhecimento em textos. Entretanto, os textos são naturalmente não estruturados e a maioria dos algoritmos de aprendizado de máquina necessitam de exemplos em um formato estruturado. Transformar textos (formato não estruturado) em tabelas (formato estruturado) tem sido uma abordagem amplamente utilizada para possibilitar o uso da maioria dos algoritmos de aprendizado que utilizam dados estruturados. No caso de aprendizado supervisionado é necessário uma

\*Trabalho realizado com auxílio financeiro da FAPESP e CAPES.

quantidade expressiva de dados rotulados para a indução de um bom classificador. Entretanto, na maioria das vezes, no mundo real, o número de exemplos rotulados é muito pequeno, quando inexistente. Desse modo, rotular dados tem sido um dos principais objetivos de uma área de pesquisa denominada aprendizado semi-supervisionado.

Boa parte dos algoritmos de aprendizado semi-supervisionado são multi-visão, ou seja, necessitam de duas ou mais descrições do conjunto de exemplos. Um exemplo pode ser descrito de diversas maneiras; por exemplo, um ser humano pode utilizar tanto a visão quanto o tato para identificar um objeto. No caso de Mineração de Textos (MT), um texto pode ser representado utilizando a frequência de palavras simples ou de palavras compostas que ocorrerem no texto. Tanto a visão e o tato no caso de um ser humano, quanto as palavras simples e compostas no caso de um texto podem ser consideradas como duas descrições diferentes, as quais podem cooperar entre si para representar um exemplo (objeto e texto respectivamente). Assim, os algoritmos multi-visão são restritos a conjunto de dados que possuam essa característica.

A utilização de aprendizado semi-supervisionado pode ser de grande utilidade para a comunidade de Mineração de Textos, sobretudo para incrementar o número de exemplos rotulados em bases nas quais esses exemplos são escassos para, após, rotular um número expressivo de exemplos, eles possam ser utilizados por qualquer algoritmo de aprendizado supervisionado. Neste trabalho é proposta uma abordagem simples para construir as visões de conjuntos de texto, as quais são necessárias para usar algoritmos de aprendizado semi-supervisionado multi-visão. Essa abordagem é avaliada utilizando um *córpus* em português com exemplos rotulados com duas e três classes, e o algoritmo CO-TRAINING. Os resultados experimentais obtidos confirmam a adequabilidade da nossa proposta.

Este trabalho está organizado da seguinte maneira: na Seção 2 são apresentados alguns dos principais trabalhos relacionados a aprendizado semi-supervisionado; na Seção 3 é apresentada a abordagem para a obtenção de duas ou mais descrições dos dados de qualquer base de texto por meio de *n-gram*; na Seção 4 é descrito o algoritmo CO-TRAINING; na Seção 5 são descritos alguns detalhes do pré-processamento dos *córpus* utilizados nos experimentos; na Seção 6 são apresentados os resultados das análises experimentais realizadas para avaliar a eficiência da abordagem proposta; por fim, as conclusões são apresentadas na Seção 7.

## 2. TRABALHOS RELACIONADOS

Os algoritmos de aprendizado semi-supervisionado podem ser divididos em: visão única e visão múltipla (Muslea, 2002). Os algoritmos de visão única normalmente utilizam apenas um classificador e podem ser subdivididos em: transdutivos (Vapnik, 1998), variações de *Expectation Maximization* (EM) (Nigam and Ghani, 2000), algoritmos que utilizam conhecimento de fundo (Wagstaff et al., 2001) e algoritmos de *clustering* com o uso de “sementes” (Sanches, 2003). Os algoritmos de visão múltipla utilizam dois ou mais classificadores e normalmente são variações do algoritmo CO-TRAINING (Blum and Mitchell, 1998). CO-TRAINING fornece a primeira formalização de aprendizado multi-visão; ele requer que as duas visões sejam compatíveis, *i.e.*, todos os exemplos devem possuir o mesmo rótulo em cada visão, e as visões devem ser independentes, ou seja, deve ser possível realizar a indução de um classificador preciso sem depender da outra visão.

Diversos algoritmos multi-visão baseados na idéia de CO-TRAINING surgiram desde então. Goldman and Zhou (2000) propõem uma variação de CO-TRAINING que utiliza dois algoritmos indutores com *bias* diferentes para a construção das duas visões. Nigam et al. (2000), utilizando as idéias do EM e CO-TRAINING, propõem o CO-EM. Em

Muslea (2002) é proposto o CO-TESTING, um algoritmo multi-visão interativo que permite ao usuário rotular os exemplos que possuem classificações distintas fornecidas por cada visão. Ainda em Muslea et al. (2002) é proposto uma extensão do CO-TESTING, chamado CO-EMT, que surgiu da união de CO-EM e CO-TESTING. O algoritmo original de CO-TRAINING (Blum and Mitchell, 1998) utiliza *Naive Bayes* como algoritmo base. Em (Kiritchenko and Matwin, 2002; Kockelkorn et al., 2003) é apresentado uma versão de CO-TRAINING que utiliza *Support Vector Machines* (SVM) como algoritmo base. Nos resultados experimentais descritos nesses trabalhos, o CO-TRAINING com SVM supera o CO-TRAINING original (*Naive Bayes*). Brefeld and Scheffer (2004) propõem uma versão de CO-EM e SVM que, segundo os resultados experimentais apresentados, supera os resultados obtidos pelos outros algoritmos.

No entanto, no trabalho de Cozman et al. (2003) mostra, por meio de uma análise matemática, como o incremento de exemplos não rotulados, em alguns casos, pode aumentar o erro de classificação. Outros trabalhos mostram alguns poucos experimentos indicando que o uso de muitas classes pode prejudicar o desempenho de algoritmos semi-supervisionado. Em Ghani (2002) é feito uma análise comparando duas classes com multi-classe além de propor o uso de *Error-Correcting Output Coding* para melhoria do desempenho em problemas multi-classe.

O aprendizado multi-visão é amplamente utilizado em aplicações reais de Mineração de Texto. Classificação de páginas de internet (Blum and Mitchell, 1998), reconhecimento de entidades nominais (Collins and Singer, 1999), indução de *wrappers* (Muslea et al., 2002) e classificação de emails (Kiritchenko and Matwin, 2002) são alguns exemplos. Entretanto, ao observar essas aplicações de MT nota-se que a construção das duas visões é altamente dependente do domínio da aplicação. Em (Blum and Mitchell, 1998) é utilizado como primeira visão os textos contidos em páginas da internet, e como segunda visão os links de internet que apontam para essas páginas. Ao trocar de domínio de aplicação, como classificação de emails, não é possível utilizar a mesma visão utilizada em páginas da internet já que os emails não possuem links. Em (Kiritchenko and Matwin, 2001) que trata do problema de classificação de emails, é utilizado o cabeçalho do email como primeira visão e o corpo do email como segunda visão. Assim, a obtenção das visões diferentes do conjunto de exemplos pode ser realizada de diversas maneiras. Neste trabalho é proposta uma abordagem simples e genérica para a obtenção dessas duas visões, a qual pode ser utilizada sempre que os exemplos consistirem de documentos texto.

### 3. CONSTRUINDO DUAS DESCRIÇÕES

No Laboratório de Inteligência Computacional (LABIC<sup>1</sup>) foi desenvolvida a ferramenta computacional PRETEXT (Matsubara et al., 2003) que tem como objetivo realizar o pré-processamento de textos utilizando a abordagem *bag-of-words*. Nessa abordagem, cada documento é representado como um vetor das palavras que ocorrem no documento. PRETEXT permite a construção de *n-grams*,  $n = 1, 2, \dots$ , no qual  $n$  refere-se a um termo composto por  $n$  palavras que ocorrem sequencialmente no documento.

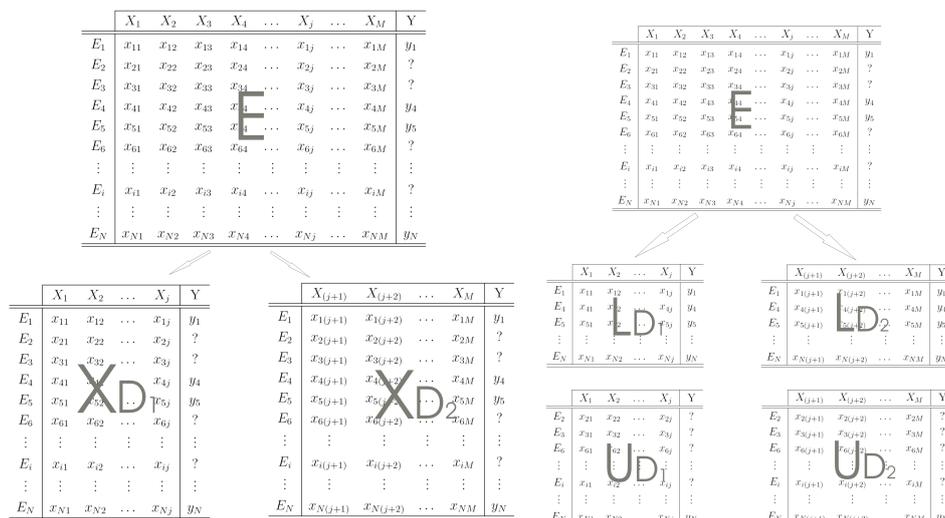
Neste trabalho é proposto utilizar, para cada descrição dos dados de um algoritmo multi-visão, os termos compostos por um determinado número de palavras, ou seja, cada descrição dos dados é composta por todos os *n-gram* (com  $n$  fixo) formados a partir do texto. Essa proposta é avaliada utilizando *1-gram* para a primeira visão e *2-gram* para a segunda visão. Entretanto, pode-se utilizar qualquer valor de  $n$  para cada uma das visões, desde que o valor de  $n$  seja distinto entre elas. Um pouco mais formalmente, o

---

<sup>1</sup><http://labic.icmc.usp.br>.

conjunto de atributos  $\mathbf{X}$  que descreve os exemplos consiste de dois subconjuntos  $\mathbf{X}_{D_1}$  e  $\mathbf{X}_{D_2}$ , representados neste por trabalho 1-gram e 2-gram, os quais constituem as duas descrições dos textos (exemplos) tal que  $\mathbf{X} = \mathbf{X}_{D_1} \cup \mathbf{X}_{D_2}$  e  $\mathbf{X}_{D_1} \cap \mathbf{X}_{D_2} = \emptyset$  como ilustra a Figura 1(a). Nessa figura, por simplicidade, é considerado que  $\mathbf{X}_{D_1} = \{X_1, X_2, \dots, X_j\}$  e  $\mathbf{X}_{D_2} = \{X_{j+1}, X_{j+2}, \dots, X_M\}$ . Exemplos com valor de  $Y$  igual a “?” representam exemplos não rotulados.

Além da separação em duas descrições, o conjunto de exemplos  $E = \{E_1, \dots, E_N\}$  deve ser separado em dois subconjuntos  $L$  e  $U$ , também conhecidos como conjunto de exemplos rotulados (*Labeled*) e não-rotulados (*Unlabeled*), respectivamente. O subconjunto  $L \subset E$  que contém exemplos que possuem o atributo classe conhecido é, por sua vez, dividido em duas descrições  $L_{D_1}$  e  $L_{D_2}$ , tal que  $L = L_{D_1} \cup L_{D_2}$  e  $L_{D_1} \cap L_{D_2} = \emptyset$ . Analogamente, o subconjunto  $U \subset E$  que contém os exemplos nos quais o atributo classe é desconhecido é também dividido em duas descrições  $U_{D_1}$  e  $U_{D_2}$ , tal que  $U = U_{D_1} \cup U_{D_2}$  e  $U_{D_1} \cap U_{D_2} = \emptyset$ . Os dados de entrada do algoritmo de CO-TRAINING consistem desses quatro conjuntos de exemplos  $L_{D_1}$ ,  $L_{D_2}$ ,  $U_{D_1}$  e  $U_{D_2}$ . Na Figura 1(b) encontram-se ilustrados esses quatro conjuntos de exemplos, os quais são submetidos ao algoritmo CO-TRAINING descrito a seguir.



(a) Conjunto de exemplos  $E$  dividido em duas descrições  $\mathbf{X}_{D_1}$  e  $\mathbf{X}_{D_2}$ .

(b) Descrições  $\mathbf{X}_{D_1}$  e  $\mathbf{X}_{D_2}$  divididas em sub-conjuntos  $L_{D_1}$ ,  $U_{D_1}$  e  $L_{D_2}$ ,  $U_{D_2}$ .

Figura 1: Criação de duas descrições de um conjunto de dados

#### 4. O ALGORITMO CO-TRAINING

No primeiro passo do CO-TRAINING — Algoritmo 1 —, antes do laço, são criados dois subconjuntos  $U'_{D_1}$  e  $U'_{D_2}$ , tal que  $U' = U'_{D_1} \cup U'_{D_2}$  e  $U'_{D_1} \cap U'_{D_2} = \emptyset$ , sendo  $U'$  um subconjunto de, geralmente, poucos exemplos de  $U$ , ou seja, esses dois subconjuntos de exemplos não rotulados  $U'_{D_1}$  e  $U'_{D_2}$  são subconjuntos de  $U_{D_1}$  e  $U_{D_2}$ , respectivamente. Os exemplos que compõem  $U'_{D_1}$  e  $U'_{D_2}$  são removidos de  $U_{D_1}$  e  $U_{D_2}$  a fim de verificar as condições  $U'_{D_1} \cap U_{D_1} = \emptyset$  e  $U'_{D_2} \cap U_{D_2} = \emptyset$  criando assim conjuntos disjuntos. Após cada iteração, com os conjuntos de exemplos rotulados  $L_{D_1}$  e  $L_{D_2}$  são induzidos dois classificadores  $h_{D_1}$  e  $h_{D_2}$ , respectivamente, os quais são utilizados para rotular exemplos não rotulados de  $U'_{D_1}$  e  $U'_{D_2}$ . No fim desse processo,  $R'_{D_1}$  e  $R'_{D_2}$  contém os conjuntos de exemplos rotulados nesse passo da iteração, os quais são argumentos da função

*melhores/2* que é responsável pela seleção de exemplos que foram “melhor” rotulados, e com o mesmo rótulo, pelos respectivos classificadores  $h_{D_1}$  e  $h_{D_2}$ .

---

### Algoritmo 1 Cotraining

---

**Input:**  $L_{D_1}, L_{D_2}, U_{D_1}, U_{D_2}, k$   
**Output:**  $L_{D_1}, L_{D_2}$   
 Construir  $U'_{D_1}$  e  $U'_{D_2}$  como descrito;  
 $U_{D_1} = U_{D_1} - U'_{D_1}$ ;  
 $U_{D_2} = U_{D_2} - U'_{D_2}$ ;  
**for**  $i = 0$  to  $k$  **do**  
     induzir o classificador  $h_{D_1}$  a partir dos exemplos de treinamento  $L_{D_1}$ ;  
     induzir o classificador  $h_{D_2}$  a partir dos exemplos de treinamento  $L_{D_2}$ ;  
      $R'_{D_1}$  = todos os exemplos de  $U'_{D_1}$  rotulados com o classificador  $h_{D_1}$ ;  
      $R'_{D_2}$  = todos os exemplos de  $U'_{D_2}$  rotulados com o classificador  $h_{D_2}$ ;  
      $(R_{D_1}, R_{D_2}) = \text{melhores}(R'_{D_1}, R'_{D_2})$ ;  
     **if**  $R_{D_1} = \emptyset$  **then Return**  $L_{D_1}, L_{D_2}$  ;  
      $L_{D_1} = L_{D_1} \cup R_{D_1}$ ;  
      $L_{D_2} = L_{D_2} \cup R_{D_2}$ ;  
     **if**  $U_{D_1} = \emptyset$  **then Return**  $L_{D_1}, L_{D_2}$  **else**  
         retirar aleatoriamente alguns exemplos de  $U_{D_1}$  e os exemplos  
         respectivos de  $U_{D_2}$  e adicioná-los a  $U'_{D_1}$  e  $U'_{D_2}$  respectivamente;  
     **end**  
**end**  
**Return**  $L_{D_1}, L_{D_2}$ ;

---

Em outras palavras, a função *melhores/2* ignora exemplos que foram rotulados diferentemente pelos classificadores  $h_{D_1}$  e  $h_{D_2}$ . Posteriormente, é calculado o *ranking* dos exemplos rotulados como o produto das probabilidades das classes fornecidas pelos classificadores  $h_{D_1}$  e  $h_{D_2}$ . Os exemplos são ordenados pelo *ranking* em forma decrescente e a função *melhores/2* seleciona os melhores exemplos (os primeiros do *ranking*) de cada classe. Após a execução da função *melhores/2*, os exemplos que foram “melhor” rotulados são adicionados aos conjuntos de exemplos de treinamento  $L_{D_1}$  e  $L_{D_2}$ . Caso ainda existam exemplos não rotulados, são utilizados mais exemplos ainda não rotulados em  $U_{D_1}$  e  $U_{D_2}$  os quais são adicionados aos conjuntos  $U'_{D_1}$  e  $U'_{D_2}$ , respectivamente, e o processo é repetido.

Note que a função *melhores/2* exige do classificador um valor de confiança para cada exemplo rotulado. Como mencionado, no artigo original de CO-TRAINING é utilizado o algoritmo *Naive Bayes*, o qual, ao classificar um exemplo, fornece a probabilidade do exemplo pertencer a cada uma das classes possíveis. Entre essas classes possíveis, a classe de maior probabilidade é a classe do exemplo. Essa probabilidade da classe do exemplo é aqui denominada de valor de confiança. O usuário pode definir um valor mínimo aceitável para esse valor de confiança. Dessa maneira, todos os exemplos rotulados que possuem o valor de confiança menor que o definido pelo usuário são descartados.

## 5. Pré-processamento dos textos

O subcórpus para a construção da base de dados para execução dos experimentos foi obtida utilizando o Lácio-web<sup>2</sup> do Núcleo Interinstitucional de Linguística Computacional

<sup>2</sup><http://www.nilc.icmc.usp.br/lacioweb/>

(NILC USP-São Carlos). No Lácio-web estão disponibilizados diversos corpúscos que podem ser acessados e copiados livremente. Entre os corpúscos existentes no Lácio-web, o Lácio-ref possibilita a construção de subcorpúscos de acordo com a necessidade do usuário. A versão atual do Lácio-ref possui 4.156.816 ocorrências composta por textos organizados em cinco gêneros (informativo, científico, prosa, poesia e drama), vários tipos de textos (por exemplo, reportagens, artigos, crônicas, cartas), vários domínios (como educação, engenharia, política) e alguns meios de distribuição (revista, internet, livro, por exemplo). O Lácio-ref é disponibilizado para pesquisa com geração de subcorpúscos para download em dois formatos: um com cabeçalho em XML contendo dados bibliográficos e de catalogação, e outro com o título, subtítulo, autoria e texto cru.

Para avaliar o método de construção de multi-visões, foi construído um subcorpúscos utilizando os cadernos *dinheiro*, *esporte* e *informática* do jornal Folha de São Paulo. Cada caderno representa o rótulo da classe para o algoritmo de aprendizado — Tabela 1. O subcorpúscos construído é do gênero informativo, contendo textos de diferentes domínios do meio jornalístico. Os tipos de textos mais comuns encontrados nesse subcorpúscos são notícias, reportagens ou artigos, todos coletados em 1994.

caderno	abreviação	# textos
dinheiro	din	279
esporte	ept	302
informática	inf	253

**Tabela 1: Cadernos do Jornal Folha de SP utilizados como classe**

Combinando esses cadernos foram construídos 4 conjuntos de textos: três conjuntos, *din-ept*, *din-inf*, *ept-inf*, com duas classes, e um conjunto, *din-ept-inf*, com três classes — Tabela 2. Os conjuntos de textos foram devidamente separados nesses 4 conjuntos de textos e pré-processados, *i.e.*, transformados em tabela atributo-valor separadamente. Essa etapa foi realizada com a ferramenta de pré-processamento PRETEXT<sup>3</sup> (Matsubara et al., 2003), com a qual primeiramente foram transformadas as palavras dos textos em *stems* (remoção de sufixos de um termo, ou mesmo na transformação de um verbo para sua forma no infinitivo). Os *stems* foram agrupados dois a dois para formarem os *2-gram* necessários para a construção da segunda visão do CO-TRAINING. As tabelas atributo-valor foram construídas utilizando a abordagem *bag-of-words* (*bow*). Nessa abordagem, cada texto é representado por um vetor de palavras ou termos, sendo que cada palavra ou termo está associado a um valor numérico. O valor numérico é fornecido por uma medida geralmente associada à frequência de ocorrência da palavra ou termos nos textos. Nos experimentos realizados foi utilizada a medida mais simples que consiste da contagem dos termos nos textos, também conhecida como medida *term frequency* (*tf*).

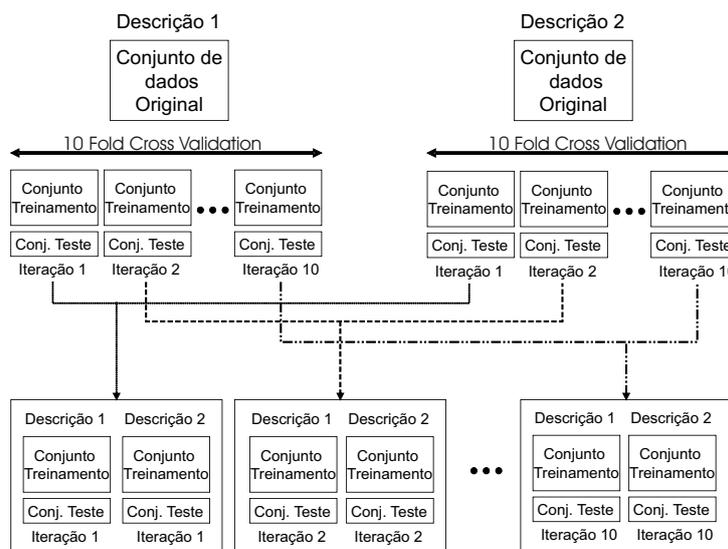
Com as duas visões, *1-gram* e *2-gram*, devidamente transformadas, foram realizados os cortes de Luhn, o qual considera que os termos muito frequentes e pouco frequentes são pouco relevantes para descrever o texto. A Tabela 2 apresenta os cortes máximo e mínimo de cada conjunto. Para *1-gram* foram retirados os termos que aparecem uma ou duas vezes ou os termos que apareceram mais de 600 vezes. Para *2-gram* os conjuntos com duas classes foram pré-processados retirando os termos que aparecem menos de quatro vezes. Para o conjunto *din-ept-inf* com três classes, para *1-gram* foram retirados os termos que apareceram menos que três vezes e mais que 900 vezes. Para *2-gram* foram retirados os termos que apareceram menos de quatro vezes.

<sup>3</sup>Disponível para *download* em [www.icmc.usp.br/~edsontm/](http://www.icmc.usp.br/~edsontm/)

conjunto		visão	corte min	corte max	#atr. depois corte
din-ept	581	1-gram	3	600	4129
		2-gram	4	-	1577
din-inf	532	1-gram	3	600	3831
		2-gram	4	-	1450
ept-inf	555	1-gram	3	600	3815
		2-gram	4	-	1302
din-ept-inf	834	1-gram	3	900	5180
		2-gram	4	-	2301

**Tabela 2: Cortes de Lunh**

Como mencionado, CO-TRAINING utiliza duas descrições do conjunto de exemplos, as quais devem se corresponder uma a uma (devem ser compatíveis). Assim, o método de re-amostragem *10-fold cross-validation* foi adaptado para essa particularidade do algoritmo, como ilustrado na Figura 2. Após o pré-processamento das bases de documentos, cada uma das descrições do conjunto de exemplos é particionada em 10 subconjuntos, sendo que cada subconjunto contém exemplos correspondentes a cada uma das descrições. Essas partições são sempre pareadas uma a uma, tal que as mesmas partições de ambas as descrições, em cada iteração, são utilizadas para treinamento (90%) e teste (10%). Todos os experimentos com CO-TRAINING, descritos a seguir, foram avaliados utilizando *10-fold cross-validation*, no qual os 10 *folds* foram construídos como ilustrado na Figura 2.



**Figura 2: Representação gráfica da construção dos 10-folds para CO-TRAINING**

## 6. Análise experimental

Na implementação realizada (Matsubara and Monard, 2004b), CO-TRAINING pode ser executado em dois modos diferentes: modo simulação e modo real. A principal diferença entre esses modos de execução refere-se ao conjunto de exemplos não rotulados  $U$ . No modo simulação, o conjunto  $U$  é construído artificialmente a partir de um conjunto de exemplos rotulados. Dessa maneira, é possível verificar, a cada iteração do algoritmo, se os rótulos atribuídos por CO-TRAINING aos exemplos “não rotulados” coincidem com os verdadeiros rótulos desses exemplos. Todos os experimentos com CO-TRAINING realizados neste trabalho foram executados no modo simulação. Foi utilizado o algoritmo *Naive Bayes* para construir, em cada iteração do algoritmo, os classificadores  $h_{D_1}$ ,  $h_{D_2}$  e

o classificador combinado  $h$  o qual é construído multiplicando as probabilidades obtidas de  $h_{D_1}$  e  $h_{D_2}$ .

O algoritmo possui diversos parâmetros possibilitando uma ampla gama de variações nos resultados. Boa parte desses parâmetros foram avaliados em (Matsubara and Monard, 2004a) no qual são apresentados resultados experimentais obtidos variando o tamanho do  $L_{ini}$ , alterando a proporção de exemplos de cada classe a ser inserida em  $L$ , entre diversos outros experimentos. Com base nesses resultados foram definidos os parâmetros para a execução dos experimentos aqui apresentados — Tabela 3, utilizando as bases previamente descritas.

critério de parada	$U = \emptyset$
tamanho inicial do conjunto de exemplos rotulados ( $ L_{ini} $ )	10 exemplos
valor mínimo de confiança	0.6

**Tabela 3: Parâmetros utilizados nos experimentos com CO-TRAINING**

O critério de parada do algoritmo é acionado quando o conjunto de exemplos não rotulados  $U$  estiver vazio ou não for possível rotular mais exemplos com o valor mínimo de confiança estipulado. O número inicial de exemplos rotulados  $|L_{ini}| = 10$ , representa aproximadamente 2% dos dados para os conjuntos com duas classes. Com relação ao conjunto de teste, ele consiste, em cada uma das execuções realizadas usando a adaptação de *10-fold cross validation* ilustrada na Figura 2, de um arquivo `.test` que contém os 10% dos exemplos de teste para os respectivos 90% de exemplos de treinamento.

Na Tabela 4 é apresentado, para cada conjunto de dados, um resumo dos resultados obtidos, média e desvio padrão, ao final da execução do CO-TRAINING. São apresentados: o número de iterações realizadas pelo algoritmo CO-TRAINING (# iterações), o tamanho do conjunto de exemplos rotulados ao final da execução ( $|L_{fim}|$ ), o número total de exemplos rotulados errados (# Errados), e o percentual de exemplos rotulados errados (% Errados). Como pode ser observado o número de exemplos rotulados errados é baixo, sobretudo considerando que os classificadores foram gerados incrementalmente a partir de uma base inicial de somente 10 exemplos rotulados. Para o conjunto de dados com duas classes com o pior desempenho, `din-inf`, o algoritmo errou em média 8.8 entre 423.6 exemplos, isto é, aproximadamente 2% dos exemplos foram rotulados errados. Para a base com três classes, `din-ept-inf`, o algoritmo errou em média 16.7 entre 656 exemplos, menos que 3%.

conjunto	# iterações	$ L_{fim} $	# Errados	% Errados
<code>din-ept</code>	116	461.6 (2.7)	4.0 (1.1)	0.9% (0.2%)
<code>din-inf</code>	106	423.6 (0.7)	8.8 (1.6)	2.1% (0.4%)
<code>ept-inf</code>	110	433.4 (2.3)	2.7 (1.2)	0.6% (0.3%)
<code>din-ept-inf</code>	111	656.0 (4.4)	16.7 (3.4)	2.6% (0.5%)

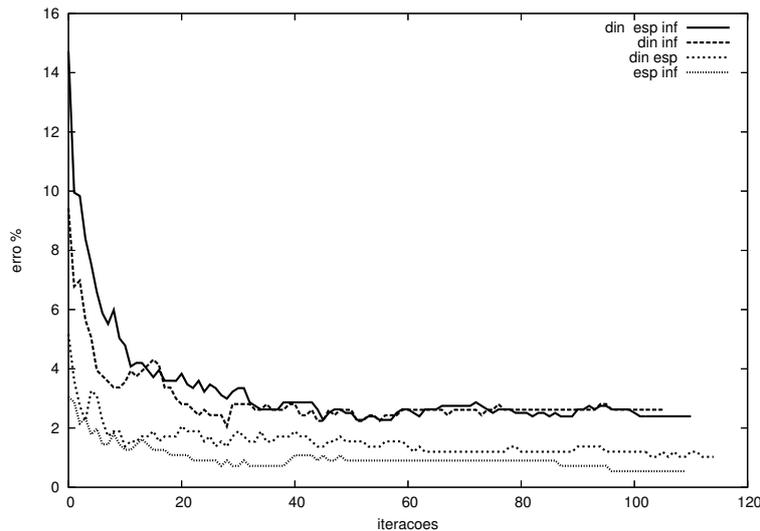
**Tabela 4: Resumo do número de exemplos rotulados e o número de exemplos rotulados errados**

Na Tabela 5 é mostrado o erro médio na primeira e última iteração dos classificadores induzidos,  $h_{D_1}$  e  $h_{D_2}$ , e do classificador combinado  $h$  para cada conjunto de documentos; o número entre parênteses indica o desvio padrão. Vale observar que na primeira iteração do CO-TRAINING os classificadores são construídos utilizando apenas 10 exemplos rotulados e, como esperado, o erro nas primeiras iterações é alto mas diminui com o número de iterações, *i.e.*, quando incrementa o número de exemplos rotulados. O erro dos classificadores combinados  $h$  dos quatro conjuntos são apresentados na Figura 3. Todos os quatro conjuntos tiveram resultados semelhantes sendo bastante

evidente o decréscimo do erro dos classificadores nos quatro conjuntos de dados durante as iterações.

conjunto	$h_{D_1}$		$h_{D_2}$		$h$	
	primeira	última	primeira	última	primeira	última
din-ept	5.51 (4.66)	1.03 (1.44)	29.24 (8.74)	3.79 (2.67)	5.17 (4.45)	1.03 (1.44)
din-inf	9.43 (4.88)	2.62 (2.35)	32.70 (4.88)	6.95(4.36)	9.43 (5.19)	2.62 (2.35)
ept-inf	2.86 (3.02)	0.36 (0.77)	24.75 (7.13)	2.53 (2.29)	3.05 (3.01)	0.54 (0.87)
din-ept-inf	13.53 (3.95)	2.28 (1.64)	45.19 (5.79)	6.12 (2.15)	14.73 (3.08)	2.40 (1.49)

**Tabela 5: Erro médio (%) dos classificadores na primeira e última iteração**



**Figura 3: Classificador combinado para os 4 conjuntos de exemplos**

## 7. Conclusões

Rotular textos é uma tarefa árdua, demorada e muitas vezes realizada manualmente. Neste trabalho foi apresentado o aprendizado semi-supervisionado como um método para auxiliar nessa tarefa quando existem poucos exemplos rotulados. No entanto, boa parte dos algoritmos de aprendizado semi-supervisionados são multi-visão e necessitam que os dados fornecidos como entrada consistam de diferentes descrições que, por sua vez, devem ser compatíveis e suficientemente independentes. A construção das duas descrições dos dados é altamente dependente do domínio de aplicação. Neste trabalho foi apresentada uma abordagem para a construção de duas ou mais descrições de dados obtidas a partir de documentos textos que satisfazem essas condições. As descrições dos dados são obtidas na fase de pré-processamento dos textos, por meio da criação de *n-grams*.

A abordagem proposta foi avaliada utilizando quatro conjuntos de dados. Considerando o número de exemplos rotulados errado e o erro dos classificadores induzidos, pode-se concluir que o algoritmo semi-supervisionado CO-TRAINING atingiu bons resultados em todos os casos. Com apenas 10 exemplos rotulados foi possível rotular quase totalmente os conjuntos de dados com uma média de erro sempre menor que 3%. Esse bom desempenho é refletido na precisão dos classificadores induzidos, na qual é bastante evidente o decréscimo no erro dos classificadores. Assim, consideramos apropriada a abordagem aqui proposta para construir as visões necessárias a fim de utilizar algoritmos de aprendizado semi-supervisionado multi-visão para auxiliar na rotulação de textos. Como trabalhos futuros serão consideradas outras medidas, tais como tf-idf, booleano, tf-linear, implementadas no PRETEXT, com o objetivo de verificar a influência dessas medidas no processo de rotulação de textos.

## Referências

- Blum, A. and Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proc. 11th Annu. Conf. on Comput. Learning Theory*, pages 92–100. ACM Press, New York, NY.
- Brefeld, U. and Scheffer, T. (2004). Co-EM Support Vector Learning. In *Proceedings of the International Conference in Machine Learning*. Morgan Kaufmann.
- Collins, M. and Singer, Y. (1999). Unsupervised models for named entity classification. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.
- Cozman, F. G., Cohen, I., and Cirelo, M. C. (2003). Semi-supervised learning of mixture models. In Fawcett, T. and Mishra, N., editors, *Proceedings of the 20th International Conference (ICML 2003)*, pages 99–106. AAAI Press.
- Ghani, R. (2002). Combining labeled and unlabeled data for multiclass text categorization. In *Proceedings of the 19th International Conference (ICML 2002)*, pages 187–194.
- Goldman, S. and Zhou, Y. (2000). Enhancing supervised learning with unlabeled data. In *Proceedings of the International Conference on Machine Learning*, pages 327–334. Morgan Kaufmann.
- Kiritchenko, S. and Matwin, S. (2001). Email classification with co-training. In *Conference of the Centre for Advanced Studies on Collaborative research*, page 8. IBM Press.
- Kiritchenko, S. and Matwin, S. (2002). Email classification with co-training. Technical report, University of Ottawa.
- Kockelkorn, M., Lüneburg, A., and Scheffer, T. (2003). Using transduction and multi-view learning to answer emails. In *Proceedings of the European Conference on Principle and Practice of Knowledge Discovery in Databases*, pages 266–277. Springer-Verlag.
- Matsubara, E. T., Martins, C. A., and Monard, M. C. (2003). Pretext: Uma ferramenta para pré-processamento de textos utilizando a abordagem *bag-of-words*. Technical Report 209, ICMC-USP.
- Matsubara, E. T. and Monard, M. C. (2004a). Análise experimental do algoritmo de aprendizado de máquina semi-supervisionado CO-TRAINING. Technical Report 235, ICMC-USP.
- Matsubara, E. T. and Monard, M. C. (2004b). Projeto e implementação do algoritmo de aprendizado de máquina semi-supervisionado CO-TRAINING. Technical Report 229, ICMC-USP.
- Muslea, I. (2002). Active Learning With Multiple Views. Phd Dissertation, University Southern California.
- Muslea, I., Minton, S., and Knoblock, C. (2002). Active + semi-supervised learning = robust multi-view learning. In *International Conference on Machine Learning*, pages 435–432. Morgan Kaufmann.
- Nigam, K. and Ghani, R. (2000). Analyzing the effectiveness and applicability of co-training. In *Conference on Information and Knowledge Management*, pages 86–93.
- Nigam, K., McCallum, A. K., Thrun, S., and Mitchell, T. M. (2000). Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134.
- Sanches, M. K. (2003). Aprendizado de máquina semi-supervisionado: proposta de um algoritmo para rotular exemplos a partir de poucos exemplos rotulados. Dissertação de Mestrado, ICMC-USP.
- Vapnik, V. (1998). *Statistical learning theory*. John Wiley & Sons.
- Wagstaff, K., Cardie, C., Rogers, S., and Schroedl, S. (2001). Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–584.