Analysis of the Threshold Variation of the *FlexCon-C* Algorithm for Semi-supervised Learning

Arthur C. Gorgônio[1], Cainan T. Alves[1], Amarildo J. F. Lucena[1], Flavius L. Gorgônio[1], Karliane M. O. Vale[1], Anne M. P. Canuto[2]

[1]Laboratório de Inteligência Computacional Aplicado a Negócios (LABICAN)
Universidade Federal do Rio Grande do Norte (UFRN)
Rua Joaquim Gregório, S/N – 59.300-000 – Caicó – RN – Brasil

[2]Departamento de Informática e Matemática Aplicada (DIMAp)
Universidade Federal do Rio Grande do Norte (UFRN)
Av. Sen. Salgado Filho, 300 – 59.078-970 – Natal – RN – Brasil

{arthurgorgonio,cainanalves,amarildolucena}@ufrn.edu.br,
flavius@dct.ufrn.br, karlianev@gmail.com, anne@dimap.ufrn.br

Abstract. *Semi-supervised learning algorithms are able to train classifiers from a small portion of initially labeled objects. The reliability of the classification process depends on several factors that include the type of classifier used and a set of parameters that customize them. One of the most important factors is a threshold that determines which instances are included per iteration, allowing to label only instances with high confidence values. This article analyzes different values for the variation factor of the FlexCon-C algorithm and measures the impact of this change on its accuracy. The results consider thirty different databases, four classifiers and five different percentages of pre-labeled data.*

Resumo. *Algoritmos de aprendizado semissupervisionado são capazes de treinar classificadores a partir de uma pequena parcela de objetos inicialmente rotulados. A confiabilidade do processo de classificação depende de vários fatores que incluem o tipo de classificador utilizado e um conjunto de parâmetros que os customiza. Um dos fatores mais importantes é um limiar que determina quais instâncias são incluídas por iteração, permitindo rotular apenas instâncias classificadas com alto valor de confiança. Este artigo analisa diferentes valores para o fator de variação do algoritmo FlexCon-C e mede o impacto dessa alteração na sua acurácia. Os resultados consideram trinta diferentes bases de dados, quatro classificadores e cinco diferentes percentuais de dados pré-rotulados.*

1. Introdução

A área de Aprendizado de Máquina (AM) concentra-se na questão de como construir programas de computador que aprendam automaticamente a partir de experiências [Mitchell 1997]. Quando uma máquina aprende a realizar escolhas e tomar decisões a partir de um conjunto de dados históricos, essa atividade é definida como AM. Atualmente, o grande volume das bases de dados contribui para tornar mais complexa a obtenção de informações necessárias ao processo de tomada de decisão.

O AM apresenta-se como uma possível solução para a análise de grandes volumes de dados, uma vez que estas análises demandam muito tempo quando desenvolvidas por

seres humanos. Além disso, pode ser necessário especialistas de domínio para que os dados sejam corretamente analisados. Por sua vez, uma máquina que possua a característica de aprendizado pode realizar esse processo de análise de forma mais rápida e com um custo reduzido. Justifica-se, assim, a necessidade de desenvolver algoritmos capazes de aprender padrões e realizar tais tarefas.

Tradicionalmente, o AM é dividido em dois tipos de tarefas, o aprendizado supervisionado e o aprendizado não supervisionado [Chapelle et al. 2006]. O aprendizado supervisionado concentra-se na tarefa de classificação, onde cada objeto é associado a um rótulo, que representa uma das classes existentes, a partir de instâncias cujo rótulo seja conhecido. Essa técnica é utilizada quando se deseja treinar a máquina para identificar a categoria à qual pertence um determinado objeto. O aprendizado não supervisionado, por sua vez, concentra-se na análise de agrupamentos (*clustering*) quando os objetos não estão rotulados, sendo a separação dos mesmos realizada a partir de suas similaridades.

O treinamento é uma etapa decisiva na obtenção de um modelo capaz de classificar corretamente os objetos com base em suas características. Um dos desafios da aplicação do AM está na etapa de treinamento, pois algumas bases de dados de aplicações reais são parcialmente rotuladas. Por isso, uma linha de pesquisa que tem ganhado atenção dos pesquisadores é o aprendizado semissupervisionado (do inglês, *semi-supervised learning - SSL*) que, a partir de uma pequena parcela de objetos classificados, consegue atribuir as classes aos dados restante. Portanto, a necessidade de possuir um conjunto de dados totalmente rotulado deixa de ser prioridade, pois com a inclusão dos novos exemplos ao conjunto de dados classificados, ao fim de cada iteração do algoritmo, o processo de rotulagem se torna incremental [Bouchachia 2012, Hady and Schwenker 2013].

Os algoritmos de SSL, como por exemplo o *self-training* e o *co-training*, são capazes de realizar o treinamento a partir de um conjunto pequeno de instâncias inicialmente rotuladas, classificando *n* instâncias por iteração [Zhu 2008]. Há variações destes algoritmos, como o *Flexible Confidence with Classifier (FlexCon-C)*, que fazem uso de um limiar com a intenção de inserir no conjunto dos dados rotulados somente as instâncias que possuem uma confiança mínima atribuída pelo classificador.

Uma desvantagem da utilização de um limiar fixo é que instâncias com baixo valor de confiança podem não ser incluídas no conjunto de rotulados, resultando na não classificação de todo o conjunto de dados. Sendo assim, uma das formas de resolver esse problema é utilizando uma taxa de controle que auxilie na variação do limiar, aumentando ou diminuindo esse valor, de forma a permitir a inclusão de todas as instâncias. O algoritmo *FlexCon-C* [Vale et al. 2018] faz uso de um fator de variação prefixado a cada execução e de um limiar ajustável a cada iteração, a partir do fator de variação.

O objetivo deste trabalho é comparar a eficácia do algoritmo *FlexCon-C* utilizando diferentes valores para o fator de variação e analisando o impacto dessa alteração na acurácia da classificação dos dados.

2. Referencial Teórico

Conforme mencionado anteriormente e apresentado na Figura 1, o SSL está na interseção dos aprendizados supervisionado e não supervisionado, onde uma parte dos exemplos possui o rótulo e outra não.

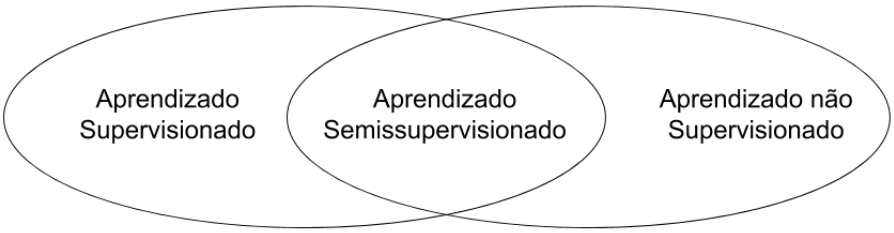

Figura 1. Divisão do Aprendizado
Fonte: O Próprio Autor (2018)

A abordagem supervisionada exige uma base de dados com todos os exemplos rotulados para gerar um classificador com uma eficácia maior. Porém, a abordagem semissupervisionada vem demonstrando ser relevante para muitas aplicações no mundo real, onde nem sempre é possível obter a base com todos os dados rotulados ou é necessário utilizar recursos financeiros e conhecimento dos profissionais com a finalidade de treinar um classificador com uma eficácia maior, o que pode vir a ser bastante oneroso. Então, com o SSL reduz-se o custo e esforço necessário para executar a rotulagem em um conjunto de dados [Bouchachia 2012, Hady and Schwenker 2013].

Além disto, essa abordagem pode ser utilizada tanto para as tarefas de classificação como as tarefas de *clustering*. Para a tarefa de classificação, os classificadores são utilizados para realizar predições sobre os dados adicionando no conjunto dos dados rotulados L. Quando a tarefa é de *clustering* os exemplos rotulados auxiliarão a definir os núcleos destes agrupamentos, obtendo assim uma eficácia melhor no processo [Chapelle et al. 2006].

Um algoritmo de SSL tem sua execução dada por: i) iniciar com um conjunto pequeno de dados rotulados; ii) treinar o classificador com os exemplos do conjunto dos rotulados L; iii) realizar, conforme suas previsões, a classificação dos exemplos que não possuem classe; iv) adicionar os exemplos no conjunto dos rotulados L [Chapelle et al. 2006, Grandvalet and Bengio 2004, Zhu and Goldberg 2009]. No Algoritmo 1 encontra-se o pseudocódigo do *self-training*.

Algoritmo 1: Pseudocódigo do *Self-Training*

entrada *dados rotulados* $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, *dados não rotulados* $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$
garanta $L \leftarrow \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ *e* $U \leftarrow \{\mathbf{x}_j\}_{j=l+1}^{l+u}$
1: **repita**
2: *Treinar* f *de* L *usando aprendizado supervisionado*
3: *Aplique* f *as instâncias não rotuladas em* U
4: *Remova um subconjunto* S *de* U*; adicione* $\{(\mathbf{x}, f(\mathbf{x}))|\mathbf{x} \in S\}a\ L$
5: **até que** $U = \emptyset$

Fonte: Adaptado de [Zhu and Goldberg 2009]

onde, L representa o conjunto de dados inicialmente rotulados, U representa o restante da base de dados contendo os exemplos não rotulados, f é o classificador que vai realizar o aprendizado de modo supervisionado a partir do conjunto L, S representa o subconjunto de U que será incluído em L, $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ o *i*-ésimo par elemento e sua respectiva classe

no conjunto dos rotulados e $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$ representa o *j*-ésimo elemento que pertence ao conjunto dos não rotulados.

3. Trabalhos Relacionados

Os trabalhos subsequentes propõem pesquisas que fazem uso do algoritmo *self-training* para problemas de classificação de rótulo único ou multirrótulo. Em [Rodrigues et al. 2013], foi utilizado o algoritmo *self-training* com métodos propostos pelos próprios autores para a classificação multirrótulo. O foco do trabalho foi reduzir a aleatoriedade na escolha dos exemplos durante a rotulagem, isso foi resolvido utilizando um limiar, aliado ao fator de confiança (o pertencimento de um exemplo a sua classe), para decidir se a instância deve ir para o conjunto dos rotulados ou não. Por fim, foi realizada uma análise comparativa entre os três métodos, utilizando-se cinco bases com seis métricas de avaliação.

[Rodrigues et al. 2014] identificaram que o uso do limiar de inclusão causou uma limitação no número de instâncias adicionadas, uma vez que uma quantidade bem menor de instâncias é adicionada, necessitando assim de uma quantidade maior de iterações. Como solução, foi proposto retirar esse limiar e fazer uso apenas do fator de confiança, ordenando os exemplos de forma decrescente com base nesse valor e selecionando *x* exemplos por iteração.

Em [Tanha et al. 2017], os autores demonstram que uma árvore de decisão (classificador supervisionado) não produz boas estimativas de probabilidade para as previsões quando aplicadas ao *self-training*. Foram desenvolvidas várias modificações nesse classificador com a finalidade de produzir uma estimativa melhor. As técnicas utilizadas foram: i) não poda da árvore; ii) correção de *Laplace*; iii) uma medida baseada em distância.

Em [Tao et al. 2018], foi proposto uma modificação do *self-training*, onde os autores utilizam dados rotulados e não rotulados para realizar a classificação dos dados, além de utilizar duas medidas para auxiliar o algoritmo, sendo elas: i) um método baseado na caminhada aleatória, utilizado junto ao classificador para gerar previsões confiáveis nos dados não rotulados; e ii) o sub-conjunto ótimo U' que gera mais instâncias no conjunto de treinamento.

No trabalho de [Vale et al. 2018], foi realizada uma comparação entre o algoritmo *self-training* e três variações desse algoritmo propostos pelos autores. Essas variações diferenciam-se por realizarem uma nova forma de calcular o limiar controlado dinamicamente, incluindo assim novos exemplos no conjunto dos rotulados. Os resultados das variações propostas obtiveram taxas de acerto melhores que o algoritmo do *self-training* original utilizado nos testes.

O foco deste trabalho é realizar uma análise comparativa entre três variantes de um algoritmo proposto em [Vale et al. 2018], diferindo-se deste por utilizar diferentes valores para a variação do limiar e aplicando a quatro classificadores sendo eles: rpartXse, Naive Bayes, *k*-NN e RIPPER. O diferencial deste trabalho em relação aos anteriores concentra-se no fato de mensurar a influência do fator de variação do limiar na classificação semissupervisionada, usando com base o algoritmo *FlexCon-C* proposto por [Vale et al. 2018].

4. Metodologia Proposta

O Algoritmo 2 apresenta a ideia geral do *FlexCon-C*, onde sua execução é dada por: i) separar os exemplos rotulados dos exemplos não rotulados; ii) treinar o classificador f utilizando o conjunto dos rotulados L; iii) aplicar f no conjunto dos não rotulados U, gerando assim uma matriz com o exemplo, sua respectiva classe e confiança do classificador; iv) selecionar todos os exemplos dado a regra de inserção; v) adicionar todos os exemplos de S em L; vi) utilizar o conjunto S para treinar um classificador f'; vii) aplicar o classificador f' em L; viii) atualizar o valor do limiar de inserção com base na acurácia obtida em f'.

O *FlexCon-C* decide se o exemplo deve ir para o conjunto dos rotulados realizando uma varredura nas regras de inserção de exemplos que são: i) os exemplos são da mesma classe e as confianças são maiores que o limiar; ii) os exemplos possuem a mesma classe, porém apenas uma de suas confianças é maior que o limiar; iii) os exemplos possuem classes diferentes e ambas as confianças são maiores que o limiar; e iv) os exemplos possuem classes diferentes e uma das confianças é maior que o limiar. A regra de inserção subsequente só é executada se, e somente se, executar a varredura e nenhum exemplo se aplicar na regra atual e caso não consiga selecionar nenhum exemplo e todas as regras foram percorridas, a confiança é alterada para o maior valor de confiança da predição da iteração corrente.

Algoritmo 2: Pseudo-Código do *FlexCon-C*

entrada *dados rotulados* $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$, *dados não rotulados* $\{\mathbf{x}_j\}_{j=l+1}^{l+u}$
garanta $L \leftarrow \{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ *e* $U \leftarrow \{\mathbf{x}_j\}_{j=l+1}^{l+u}$
1: **repita**
2: *Treinar* f *com os exemplos de* L *usando aprendizado supervisionado*
3: *Aplique* f *as instâncias não rotuladas em* U
4: *Remova um subconjunto* S *de* U *tal que o exemplo se enquadre com uma das regras inclusão de novos exemplos*
5: *Adicione* $\{(\mathbf{x}, f(\mathbf{x}))|\mathbf{x} \in S\}$ *a* L
6: *Treinar* f' *com os exemplos de* S *de forma supervisionada*
7: *Aplique* f' *as instâncias em* L
8: *Atualize o limiar de inclusão a partir da acurácia em* f'
9: **até que** $U = \emptyset$

Fonte: Adaptado de [Vale et al. 2018]

O *FlexCon-C* utiliza duas abordagens diferentes *FlexCon-C1* e *FlexCon-C2* para decidir qual rótulo será atribuído a cada exemplo. O algoritmo *FlexCon-C1* guarda a predição da primeira iteração para comparar com a predição da iteração corrente. Quando as predições convergem entre as classes, o exemplo é adicionado. Caso o exemplo seja selecionado para ser incluído no conjunto dos exemplos rotulados, a sua classe é definida através da quantidade de vezes que ele foi associado a uma determinada classe. Isso gera duas versões do mesmo algoritmo *FlexCon-C1*(s), que utiliza a soma das confianças obtidas desde a primeira iteração até a atual para realizar essa decisão; e *FlexCon-C1*(v), que faz uso de votação, onde são contabilizadas somente as ocorrências do exemplo em cada classe.

O algoritmo *FlexCon-C2* compara as predições do classificador da iteração corrente com a predição de um classificador supervisionado que é treinado com as instâncias inicialmente rotuladas. Quando as predições convergem entre as classes, o exemplo é adicionado, caso o exemplo selecionado para a inclusão não possua a mesma classe nas predições, o rótulo é selecionado verificando o rótulo do classificador supervisionado.

O limiar da iteração subsequente é calculado verificando se o conjunto dos exemplos adicionados na iteração atual possui uma quantidade mínima de exemplos de todas as classes, em caso positivo o classificador treina com os exemplos adicionados da iteração e testa com os exemplos inicialmente rotulados. Em caso negativo, o conjunto de treino fica se acumulando até possuir a quantidade mínima de exemplos por classe.

O limiar de inclusão da primeira iteração é de 95%, isto é, as instâncias que na primeira iteração possuírem uma confiança igual ou maior que este valor irão para o conjunto dos dados rotulados. A partir da segunda iteração, o algoritmo irá realizar o cálculo do novo limiar de inclusão considerando a acurácia do classificador e esse limiar foi variado conforme representado na Equação 1, proposta em [Vale et al. 2018].

$$conf(t_{i+1}) = \begin{cases} conf(t_i) - cr, se\ acc \geq mp + e \\ conf(t_i), se\ mp - e < acc < mp + e \\ conf(t_i) + cr, se\ acc \leq mp - e \end{cases} \quad (1)$$

onde, $conf(t_{i+1})$ é a confiança da próxima iteração, $conf(t_i)$ é a confiança da iteração atual, cr é a taxa de variação na qual o limiar é aplicado, acc é a acurácia do classificador, mp precisão mínima e e é uma variação de precisão aceitável que é permitida para definir uma estabilização em precisão.

Na implementação descrita em [Vale et al. 2018], o valor do cr foi fixado em 5%, isto é, o limiar variava em 5 pontos percentuais por iteração. Entretanto, este trabalho utilizou diversos valores do parâmetro cr dentro do intervalo $[2\%, 8\%]$, a fim de analisar o seu efeito na classificação dos dados semissupervisionados.

Com a finalidade de avaliar a eficácia das variações propostas, foi realizada uma análise empírica, a partir da aplicação do algoritmo sobre um conjunto de 30 bases de dados. A Tabela 1 descreve os conjuntos de dados previamente selecionados, em termos do número de exemplos (Instâncias), de atributos e a quantidade de classes para cada conjunto de dados, além de indicar os tipos de dados como categórico (C), inteiro (I) ou real (R).

Cada uma destas bases de dados será dividida em dois sub-conjuntos i) para treinamento do classificador, com 75% das instâncias; e ii) para teste do classificador, com 25% das instâncias. Após essa etapa, dos 75% dos dados de treinamento será selecionado um percentual para permanecer com o rótulo 5%, 10%, 15%, 20% ou 25%. Uma vez separados os conjuntos dos rotulados e não rotulados, os dados poderão ser aplicados ao algoritmo para realização da tarefa de classificação. Para isso, pretende-se utilizar os classificadores Naive Bayes, rpartXse, RIPPER e o *k*-NN.

Tabela 1. Bases de Dados que serão utilizadas nos experimentos

Base de dados	Instâncias	Atributos	Classes	Tipo
Balance Scale	625	4	3	I
Blood Transfusion Service	748	5	2	R
Bupa	345	7	2	C, I, R
Car	1728	6	4	C, R
Cnae-9	1080	857	9	I
Connectionist-mines-vs-rocks	208	60	2	R
Flare	1389	10	6	C
Haberman	306	4	2	I
Handwritten Digits	10992	16	10	I
Hill Valley	606	101	2	R
Image Segmentation	2310	19	7	R
Indian Liver Patient	582	10	2	I, R
King-Rook vs King Pawn	3196	36	2	C
Leukemia Haslinger	100	50	2	R
Mammographic Mass	961	6	2	I
Mfeat-karhunen	2000	64	10	I, R
Mushroom	8124	22	2	C
Musk	6598	168	2	I
Ozone Level Detection	2536	73	2	R
Phishing	2456	30	3	I
Pima	768	9	2	I, R
Planning-relax	182	13	2	R
Seeds	210	7	3	R
Semeion	1593	256	10	I
Spectf Heart	267	14	2	I
Tic-tac-toe	958	9	2	C
Twonorm	7400	21	2	R
Vehicle	946	18	4	I
Waveform	5000	40	3	R
Wilt	4839	6	2	R

5. Resultados Obtidos

Esta seção apresenta a análise dos resultados obtidos no experimento. Inicialmente, foram tabuladas as acurácias médias e o desvio padrão obtidos após a aplicação de cada classificador individualmente. No segundo momento, foram utilizados gráficos para avaliar a acurácia do classificador mediante a variação do parâmetro cr

As Tabelas 2-5 apresentam, respectivamente, as acurácias médias obtidas no conjunto de bases de dados utilizando as variações do *FlexCon-C* para os classificadores Naive Bayes, rpartXse (Árvore de Decisão), RIPPER e *k*-NN. Em cada uma das células dessas tabelas estão representadas a acurácia e o desvio padrão obtidos pelo classificador, após a aplicação das variações no parâmetro cr. Também são apresentados os diferentes percentuais de exemplos inicialmente rotulados para cada uma das variações do algoritmo

FlexCon-C.

Como apresentado na Tabela 2, onde foi utilizado o classificador Naive Bayes, os melhores valores do parâmetro cr concentraram-se entre 5% e 6%. No algoritmo *FlexCon-C1(s)*, o fator de variação do limiar obteve os melhores resultados quando pré-fixado em 6%, com resultados superiores em 3 dos 5 casos. Tanto no algoritmo *FlexCon-C1(v)* como no *FlexCon-C2* o melhor valor para o fator de variação foi 5%, com 5 dos 5 casos em ambos os algoritmos. De modo geral, observa-se que o melhor valor do parâmetro cr para o classificador Naive Bayes foi de 5%, sendo superior em 10 dos 15 casos.

Como apresentado na Tabela 3, onde foi utilizado o classificador rpartXse, os melhores valores do parâmetro cr concentraram-se em 2%, 4% e 6%. No algoritmo *FlexCon-C1(s)*, o fator de variação do limiar obteve os melhores resultados quando pré-fixado em 7%, com resultados superiores em 2 dos 5 casos. Entretanto, nos algoritmos *FlexCon-C1(v)* e *FlexCon-C2* o melhor valor para o fator de variação foi 4%, com 2 dos 5 casos em ambos algoritmos. De modo geral, observa-se que o melhor valor do parâmetro cr para o classificador rpartXse foi de 4%, sendo superior em 5 dos 15 casos.

Como apresentado na Tabela 4, onde foi utilizado o classificador RIPPER, os melhores valores do parâmetro cr concentraram-se em 5%. No algoritmo *FlexCon-C1(s)*, somente a análise da acurácia não é eficaz para condizer o melhor valor. Por sua vez, no algoritmo *FlexCon-C1(v)* o melhor valor para o fator de variação foi 5%, com 3 dos 5 casos. Para o algoritmo *FlexCon-C2* dois valores são melhores, em 2% e 5%, sendo superiores em 2 dos 5 casos. De modo geral, observa-se que o melhor valor do parâmetro cr para o classificador RIPPER foi de 5%, sendo superior em 5 dos 15 casos.

Tabela 2. Resultado da acurácia com o classificador Naive Bayes

Alg.	**Exemplos Inicialmente Rotulados**	**Variação no parâmetro cr**						
		2%	**3%**	**4%**	**5%**	**6%**	**7%**	**8%**
FlexCon-C1(s)	5%	$66,83 \pm 17,76$	**67,11** $\pm 17,87$	**67,11** $\pm 17,61$	$66,83 \pm 17,59$	$67,09 \pm 17,73$	**67,11** $\pm 17,73$	$66,83 \pm 17,72$
	10%	$69,39 \pm 15,81$	$69,38 \pm 15,81$	$69,73 \pm 15,99$	$68,49 \pm 15,99$	**69,81** $\pm 16,01$	$69,59 \pm 16,08$	$69,14 \pm 16,08$
	15%	$68,84 \pm 17,80$	$68,85 \pm 17,80$	$69,36 \pm 17,77$	$68,10 \pm 17,82$	$69,32 \pm 17,82$	**69,37** $\pm 17,80$	$68,82 \pm 17,72$
	20%	$70,75 \pm 16,01$	$70,72 \pm 16,02$	$71,15 \pm 16,26$	$69,97 \pm 16,24$	**71,20** $\pm 16,20$	$71,19 \pm 16,21$	$70,75 \pm 16,27$
	25%	$70,36 \pm 17,79$	$70,45 \pm 17,79$	$70,54 \pm 18,71$	$69,26 \pm 18,65$	**70,61** $\pm 18,65$	$70,55 \pm 18,64$	$70,36 \pm 17,89$
FlexCon-C1(v)	5%	$66,80 \pm 17,80$	$66,98 \pm 17,57$	$66,95 \pm 17,61$	**67,35** $\pm 17,54$	$66,82 \pm 17,72$	$66,82 \pm 17,72$	$66,83 \pm 17,72$
	10%	$69,36 \pm 15,79$	$69,14 \pm 16,13$	$69,21 \pm 16,10$	**69,56** $\pm 16,18$	$69,15 \pm 16,08$	$69,12 \pm 16,08$	$69,14 \pm 16,08$
	15%	$68,83 \pm 17,78$	$68,83 \pm 17,78$	$68,86 \pm 17,82$	**69,24** $\pm 17,82$	$68,78 \pm 17,77$	$68,81 \pm 17,76$	$68,82 \pm 17,72$
	20%	$70,62 \pm 16,27$	$70,60 \pm 16,28$	$70,62 \pm 16,28$	**70,87** $\pm 16,22$	$70,72 \pm 16,27$	$70,73 \pm 16,27$	$70,75 \pm 16,27$
	25%	$70,33 \pm 17,84$	$70,33 \pm 17,93$	$70,76 \pm 17,43$	**70,93** $\pm 17,46$	$70,88 \pm 17,37$	$70,36 \pm 17,89$	$70,36 \pm 17,89$
FlexCon-C2	5%	$66,49 \pm 18,48$	$66,59 \pm 18,31$	$66,51 \pm 18,40$	**66,98** $\pm 18,33$	$66,52 \pm 18,39$	$66,53 \pm 18,39$	$66,53 \pm 18,39$
	10%	$69,15 \pm 16,07$	$69,18 \pm 16,12$	$69,17 \pm 16,13$	**69,49** $\pm 16,15$	$69,12 \pm 16,03$	$69,09 \pm 16,05$	$69,09 \pm 16,05$
	15%	$68,81 \pm 17,71$	$68,82 \pm 17,80$	$68,85 \pm 17,82$	**69,22** $\pm 17,81$	$68,77 \pm 17,71$	$68,81 \pm 17,69$	$68,81 \pm 17,69$
	20%	$70,37 \pm 16,14$	$70,36 \pm 16,20$	$70,39 \pm 16,19$	**70,84** $\pm 16,22$	$70,41 \pm 16,18$	$70,42 \pm 16,18$	$70,44 \pm 16,18$
	25%	$70,43 \pm 17,87$	$70,35 \pm 17,92$	$70,45 \pm 17,92$	**70,53** $\pm 17,92$	$70,52 \pm 17,95$	$70,50 \pm 17,96$	$70,50 \pm 17,96$

Fonte: O Próprio Autor (2018)

Como apresentado na Tabela 5, onde foi utilizado o classificador *k*-NN, o melhor valor do parâmetro cr concentrou-se em 4%. No algoritmo *FlexCon-C1(s)*, o fator de variação do limiar obteve os melhores resultados quando pré-fixado em 4%, com resultados superiores em 3 dos 5 casos. Por sua vez, no algoritmo *FlexCon-C1(v)* os melhores valores para o fator de variação foram 4% e 5%, com 2 dos 5 casos em cada um deles. Para o algoritmo *FlexCon-C2*, o valor de 7% foi superior em 2 dos 5 casos, representando a melhor escolha. De modo geral, observa-se que o melhor valor do parâmetro cr para o classificador *k*-NN foi de 4%, sendo superior em 6 dos 15 casos.

Tabela 3. Resultado da acurácia com o classificador *rpartXse*

Alg.	Exemplos Inicialmente Rotulados	Variação no parâmetro cr						
		2%	3%	4%	5%	6%	7%	8%
FlexCon-C1(s)	5%	$69,25 \pm 17,69$	$69,18 \pm 18,65$	$69,22 \pm 18,63$	$69,35 \pm 17,58$	**69,38** $\pm 18,48$	**69,38** $\pm 18,48$	$69,37 \pm 18,58$
	10%	**75,45** $\pm 13,40$	$75,30 \pm 13,77$	$74,99 \pm 13,87$	$75,42 \pm 13,40$	$75,44 \pm 13,50$	$75,43 \pm 13,45$	$75,31 \pm 14,24$
	15%	$76,58 \pm 13,56$	$76,57 \pm 13,72$	$76,82 \pm 14,02$	$76,78 \pm 13,84$	$76,90 \pm 13,85$	**76,91** $\pm 13,87$	$76,11 \pm 14,83$
	20%	$77,22 \pm 13,52$	**77,26** $\pm 13,71$	$76,57 \pm 14,60$	$76,56 \pm 14,41$	$77,01 \pm 14,14$	$77,03 \pm 14,13$	$77,00 \pm 13,83$
	25%	$78,45 \pm 13,21$	$78,49 \pm 13,20$	**78,59** $\pm 12,94$	$78,46 \pm 13,08$	$78,41 \pm 13,06$	$78,52 \pm 12,84$	$78,45 \pm 13,48$
FlexCon-C1(v)	5%	$69,47 \pm 18,61$	$69,39 \pm 18,58$	**69,59** $\pm 18,57$	$69,55 \pm 17,67$	$69,37 \pm 18,58$	$69,37 \pm 18,58$	$69,37 \pm 18,58$
	10%	$75,33 \pm 14,32$	**75,40** $\pm 14,17$	$75,35 \pm 14,25$	$75,37 \pm 14,10$	$75,28 \pm 14,27$	$75,24 \pm 14,25$	$75,31 \pm 14,24$
	15%	$76,11 \pm 14,83$	$76,16 \pm 14,79$	**76,17** $\pm 14,82$	$76,10 \pm 14,74$	$76,11 \pm 14,84$	$76,16 \pm 14,84$	$76,11 \pm 14,83$
	20%	**77,16** $\pm 13,80$	$76,98 \pm 13,92$	$77,05 \pm 13,85$	$77,08 \pm 13,48$	$77,05 \pm 13,77$	$77,04 \pm 13,75$	$77,00 \pm 13,83$
	25%	$78,48 \pm 13,51$	$78,43 \pm 13,54$	$78,48 \pm 13,55$	$78,48 \pm 13,49$	**78,50** $\pm 13,46$	$78,45 \pm 13,50$	$78,45 \pm 13,48$
FlexCon-C2	5%	$70,22 \pm 18,35$	$70,33 \pm 18,23$	**76,45** $\pm 13,08$	$70,33 \pm 17,58$	$70,32 \pm 18,14$	$70,32 \pm 18,14$	$70,32 \pm 18,14$
	10%	$74,74 \pm 14,61$	$77,66 \pm 14,01$	**77,87** $\pm 13,31$	$74,81 \pm 14,43$	$74,54 \pm 14,58$	$74,74 \pm 14,51$	$74,43 \pm 14,80$
	15%	**76,47** $\pm 14,56$	$76,10 \pm 14,75$	$74,33 \pm 13,86$	$76,28 \pm 14,53$	$76,44 \pm 14,47$	$76,38 \pm 14,49$	$76,23 \pm 14,70$
	20%	$76,91 \pm 14,44$	$77,00 \pm 14,43$	$74,33 \pm 13,86$	**77,06** $\pm 14,12$	$76,74 \pm 14,95$	$76,75 \pm 14,62$	$76,75 \pm 14,69$
	25%	$77,94 \pm 13,68$	$67,74 \pm 16,44$	$67,74 \pm 16,44$	$78,56 \pm 13,20$	**78,65** $\pm 13,16$	$78,36 \pm 13,43$	$78,54 \pm 13,33$

Fonte: O Próprio Autor (2018)

Tabela 4. Resultado da acurácia com o classificador RIPPER

Alg.	Exemplos Inicialmente Rotulados	Variação no parâmetro cr						
		2%	3%	4%	5%	6%	7%	8%
FlexCon-C1(s)	5%	$69,58 \pm 16,15$	$69,48 \pm 16,05$	**69,70** $\pm 16,20$	$69,65 \pm 16,19$	$69,65 \pm 16,19$	$69,64 \pm 16,19$	$69,56 \pm 16,10$
	10%	$72,28 \pm 14,08$	$72,12 \pm 14,48$	$72,17 \pm 14,07$	$72,24 \pm 14,36$	$72,28 \pm 14,23$	**72,31** $\pm 14,29$	$72,01 \pm 14,34$
	15%	$74,01 \pm 13,60$	$73,99 \pm 13,67$	$73,45 \pm 13,96$	$73,25 \pm 14,13$	$73,39 \pm 14,13$	$73,43 \pm 14,14$	**74,04** $\pm 13,79$
	20%	**77,02** $\pm 12,22$	$76,60 \pm 12,70$	$76,48 \pm 12,83$	$76,62 \pm 12,81$	$76,58 \pm 12,83$	$76,65 \pm 12,75$	$76,93 \pm 12,30$
	25%	$78,21 \pm 13,08$	**78,39** $\pm 12,93$	$78,08 \pm 13,12$	$77,79 \pm 13,18$	$77,73 \pm 12,94$	$78,00 \pm 13,11$	$77,85 \pm 12,92$
FlexCon-C1(v)	5%	$69,54 \pm 17,11$	$69,46 \pm 16,04$	$69,57 \pm 16,10$	**69,62** $\pm 16,10$	$66,82 \pm 17,31$	$69,56 \pm 16,10$	$69,56 \pm 16,10$
	10%	$72,30 \pm 14,14$	$72,18 \pm 14,38$	**72,52** $\pm 14,29$	$72,42 \pm 14,23$	$69,15 \pm 15,95$	$72,01 \pm 14,34$	$72,01 \pm 14,34$
	15%	$73,95 \pm 13,81$	$73,99 \pm 13,67$	$73,56 \pm 13,92$	**74,15** $\pm 13,97$	$68,78 \pm 17,46$	$73,92 \pm 13,83$	$74,04 \pm 13,79$
	20%	$76,97 \pm 12,46$	$76,60 \pm 12,70$	$76,98 \pm 12,24$	**76,99** $\pm 12,17$	$70,72 \pm 15,83$	$76,83 \pm 12,37$	$76,93 \pm 12,30$
	25%	$78,21 \pm 12,94$	**78,31** $\pm 12,96$	$77,99 \pm 13,01$	$78,00 \pm 12,99$	$70,88 \pm 17,00$	$77,90 \pm 12,85$	$77,85 \pm 12,92$
FlexCon-C2	5%	$67,77 \pm 16,43$	$67,74 \pm 15,49$	$67,74 \pm 15,49$	**67,93** $\pm 15,51$	$67,76 \pm 15,50$	$67,76 \pm 15,50$	$67,76 \pm 15,50$
	10%	**71,87** $\pm 14,62$	$71,30 \pm 14,73$	$71,30 \pm 14,73$	$71,68 \pm 14,81$	$71,14 \pm 14,96$	$71,14 \pm 14,96$	$71,14 \pm 14,96$
	15%	$74,51 \pm 13,64$	$73,95 \pm 13,74$	$73,95 \pm 13,74$	$74,14 \pm 14,25$	$74,25 \pm 13,54$	$74,55 \pm 12,91$	**74,65** $\pm 12,71$
	20%	$76,17 \pm 13,20$	$75,92 \pm 13,32$	$75,92 \pm 13,32$	**76,31** $\pm 13,11$	$75,75 \pm 12,97$	$75,75 \pm 12,97$	$75,74 \pm 12,97$
	25%	**77,99** $\pm 13,35$	$77,66 \pm 13,40$	$77,66 \pm 13,40$	$77,96 \pm 13,39$	$77,55 \pm 13,50$	$77,48 \pm 13,52$	$77,42 \pm 13,56$

Fonte: O Próprio Autor (2018)

Tabela 5. Resultado da acurácia com o classificador *k*-NN

Alg.	Exemplos Inicialmente Rotulados	Variação no parâmetro cr						
		2%	3%	4%	5%	6%	7%	8%
FlexCon-C1(s)	5%	$74,09 \pm 16,36$	$74,09 \pm 16,33$	$74,27 \pm 16,46$	$74,19 \pm 16,40$	**74,35** $\pm 16,16$	$74,30 \pm 16,32$	$74,08 \pm 16,45$
	10%	$77,70 \pm 13,51$	$77,99 \pm 12,98$	**78,22** $\pm 13,07$	$78,22 \pm 13,11$	$78,19 \pm 13,13$	$78,22 \pm 13,14$	$77,75 \pm 13,43$
	15%	$79,39 \pm 13,16$	$79,53 \pm 13,19$	**79,85** $\pm 13,47$	$79,69 \pm 13,39$	$79,64 \pm 13,43$	$79,69 \pm 13,39$	$79,51 \pm 13,11$
	20%	$77,96 \pm 15,85$	$77,99 \pm 15,72$	$78,01 \pm 15,88$	$77,93 \pm 15,69$	**78,10** $\pm 15,82$	$77,96 \pm 15,82$	$78,00 \pm 15,76$
	25%	$79,99 \pm 14,10$	$79,96 \pm 14,10$	**80,29** $\pm 14,35$	$80,18 \pm 14,18$	$79,90 \pm 14,46$	$80,08 \pm 14,48$	$80,00 \pm 14,15$
FlexCon-C1(v)	5%	$74,09 \pm 16,11$	$74,09 \pm 16,33$	$74,14 \pm 16,31$	**74,17** $\pm 16,33$	$74,08 \pm 16,45$	$74,08 \pm 16,45$	$74,08 \pm 16,45$
	10%	$77,82 \pm 13,72$	$77,99 \pm 12,98$	**78,06** $\pm 12,97$	$77,89 \pm 13,12$	$77,87 \pm 13,19$	$77,85 \pm 13,20$	$77,75 \pm 13,43$
	15%	$79,43 \pm 13,24$	$79,53 \pm 13,19$	**79,57** $\pm 13,17$	$79,53 \pm 12,91$	$79,44 \pm 13,06$	$79,45 \pm 13,06$	$79,51 \pm 13,11$
	20%	**78,02** $\pm 15,88$	$77,99 \pm 15,72$	$77,99 \pm 15,72$	$77,91 \pm 15,83$	$77,92 \pm 15,84$	$77,94 \pm 15,88$	$78,00 \pm 15,76$
	25%	$79,98 \pm 14,19$	$79,96 \pm 14,10$	$79,99 \pm 14,09$	**80,03** $\pm 14,18$	$79,94 \pm 14,20$	$79,91 \pm 14,23$	$80,00 \pm 14,15$
FlexCon-C2	5%	$73,63 \pm 15,89$	$73,70 \pm 16,11$	$73,66 \pm 16,15$	$73,83 \pm 15,95$	$73,85 \pm 15,94$	$73,85 \pm 15,94$	**73,88** $\pm 15,88$
	10%	$76,74 \pm 14,28$	$76,71 \pm 13,91$	$76,71 \pm 14,00$	$76,75 \pm 13,84$	$76,80 \pm 13,74$	**76,85** $\pm 13,73$	$76,65 \pm 13,82$
	15%	$78,58 \pm 13,36$	$78,53 \pm 13,28$	**78,60** $\pm 13,28$	$78,23 \pm 13,72$	$78,58 \pm 13,40$	$78,55 \pm 13,39$	$78,53 \pm 13,41$
	20%	**77,08** $\pm 15,65$	$77,07 \pm 15,50$	$77,03 \pm 15,51$	$76,99 \pm 15,54$	$77,03 \pm 15,51$	$76,96 \pm 15,72$	$76,96 \pm 15,72$
	25%	$79,89 \pm 14,46$	$79,89 \pm 14,41$	$79,89 \pm 14,36$	$79,89 \pm 14,46$	$79,89 \pm 14,38$	**79,93** $\pm 14,35$	$79,89 \pm 14,42$

Fonte: O Próprio Autor (2018)

(a) *FlexCon-C1 (s)* - Naive Bayes (b) *FlexCon-C1 (s)* - rpartXse (c) *FlexCon-C1 (s)* - RIPPER

(d) *FlexCon-C1 (s)* - *k*-NN (e) *FlexCon-C1 (v)* - Naive Bayes (f) *FlexCon-C1 (v)* - rpartXse

(g) *FlexCon-C1 (v)* - RIPPER (h) *FlexCon-C1 (v)* - *k*-NN (i) *FlexCon-C2* - Naive Bayes

(j) *FlexCon-C2* - rpartXse (k) *FlexCon-C2* - RIPPER (l) *FlexCon-C2* - *k*-NN

Figura 2. Desempenho do algoritmo *FlexCon-C*

As Figuras 2(a)-(d) apresentam os gráficos gerados utilizando o algoritmo *FlexCon-C1(s)*, para cada um dos percentuais de exemplos rotulados. Utilizando-se o *FlexCon-C1(s)* com o classificador Naive Bayes (Figura 2(a)), obteve-se a melhor acurácia com 20% de exemplos rotulados e o valor do fator *cr* pré-fixado em 4%. Por sua vez, as Figuras 2(b)-(d) – que apresentam o desempenho dos classificadores rpartXse, RIPPER e

k-NN, respectivamente – demonstram que quando utilizado o valor de 25% das instâncias rotuladas, obtém-se maiores acurácias. Para o classificador rpartXse os diferentes valores do cr quase não influenciam, porém para o RIPPER o melhor percentual é de 3% e para o *k*-NN o melhor percentual é de 4%.

As Figuras 2(e)-(h) apresentam os gráficos gerados utilizando o algoritmo *FlexCon-C1(v)* para cada um dos percentuais de exemplos rotulados. Na Figura 2(e) observa-se que utilizando o Naive Bayes com os percentuais entre 4% e 6%, o melhor valor de exemplos inicialmente rotulados é de 25% porém, para o restante do intervalo é melhor utilizar 20% de exemplos inicialmente rotulado. Por sua vez, nas Figuras 2(f)-(h) estão representados os gráficos com os classificador rpartXse, RIPPER e *k*-NN, respectivamente, nos três casos o quando é utilizado 25% das instâncias rotuladas obtém a maior acurácia. Tanto o classificador rpartXse como o *k*-NN possuem uma variação muito baixa na acurácia para os diferentes valores do cr. Porém, para o RIPPER o valor do cr em 6% resulta numa queda significativa da acurácia para todos os percentuais de exemplos rotulados.

As Figuras 2(i)-(l) apresentam os gráficos gerados utilizando o algoritmo *FlexCon-C2* para cada um dos percentuais de exemplos rotulados. Na Figura 2(i), quando utilizado o classificador Naive Bayes, 20% de exemplos rotulados e o valor do fator cr pré-fixado em 5%, obteve-se a melhor acurácia. Por sua vez, quando aplicado o classificador rpartXse, os melhores resultados foram obtidos utilizando-se 10% de exemplos rotulados e com os valores do cr variando entre 3% até 4% (Figura 2(j)); porém, com o percentual de exemplos inicialmente rotulados de 25%, os melhores valores de acurácia para cr ficaram entre 5% e 8%. A Figura 2(k), que apresenta os resultados do classificador RIPPER, mostra uma melhor eficácia quando utilizado um conjunto de 25% de exemplos inicialmente rotulados e valor do cr pré-fixado em 2% ou 5%. Para o classificador *k*-NN (Figura 2(l)), com o conjunto de 25% dos exemplos inicialmente rotulados, obteve-se acurácias em torno de 80% com qualquer valor de cr.

De maneira geral, observando-se todos os valores relacionados nas Tabelas 2-5, é possível verificar que o valor de cr fixado em 5% obteve desempenho superior em 18 dos 60 experimentos realizados, o que corresponde a 30% dos casos. O segundo melhor valor para o parâmetro cr foi de 4%, sendo superior em 23% dos casos, o que representa 14 dos 60 experimentos realizados.

6. Discursões e Trabalhos Futuros

Algoritmos de aprendizado semissupervisionado são sensíveis a vários parâmetros que influenciam diretamente na sua eficácia. Neste trabalho foi avaliada a influência da variação do parâmetro cr na acurácia do algoritmo de aprendizado semissupervisionado FlexCon-C. Tal parâmetro representa um limiar (fator de inclusão) que determina como novas instâncias são incluídas por iteração.

Os resultados obtidos demonstram que o parâmetro cr não apresenta um comportamento semelhante para todos os casos, variando em função do classificador utilizado e dos percentuais de dados previamente rotulados. Percebe-se que o parâmetro é menos sensível a mudanças quando utilizados os classificadores *k*-NN e RIPPER, mas torna-se bem mais instável quando utilizados o rpartXse e o Naive Bayes.

Para os classificadores Naive Bayes e RIPPER, os melhores resultados foram obtidos com o valor de 5% para o parâmetro cr, enquanto que nos classificadores rpartXse e *k*-NN

o melhor valor de cr foi de 4%. Analisando-se os gráficos, também percebe-se uma maior estabilidade quando utilizado o classificador *k*-NN, para todos os métodos testados. Por outro lado, o classificador Naive Bayes apresentou-se instável em relação a todos os métodos testados. Por fim, verificou-se que o método *FlexCon-C1* possui desempenho mais estável quando associado ao classificador rpartXse, enquanto que o classificador RIPPER apresentou-se estável com os métodos *FlexCon-C1(s)* e *FlexCon-C2*.

Como propostas de trabalhos futuros, sugere-se a utilização de outros classificadores e uma investigação mais detalhada em valores intermediários do parâmetro cr dentro das faixas de valores considerada, principalmente entre 4% e 5%.

Referências

Bouchachia, A. (2012). Learning with partional supervision. In *Machine Learning*, volume 3, pages 1880–1888. Information Science Reference, Hershey PA 17033.

Chapelle, O., Schölkopf, B., and Zien, A. (2006). *Semi-Supervised Learning*. The MIT Press, Cambridge, Massachusetts, London.

Grandvalet, Y. and Bengio, Y. (2004). Semi-supervised learning by entropy minimization. In *Proceedings of the 17th International Conference on Neural Information Processing Systems*, NIPS'04, pages 529–536, Cambridge, MA, USA. MIT Press.

Hady, M. F. A. and Schwenker, F. (2013). Semi-supervised learning. In *Handbook on Neural Information Processing*, pages 215–239. Springer, Berlin Heidelberg.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Boston, Massachusetts, London.

Rodrigues, F. M., Canuto, A. M. P., and Santos, A. M. (2014). Confidence factor and feature selection for semi-supervised multi-label classification methods. *International Joint Conference on Neural Networks*, pages 864 – 871.

Rodrigues, F. M., Santos, A. M., and Canuto, A. M. P. (2013). Using confidence values in multi-label classification problems with semi-supervised learning. *International Joint Conference on Neural Networks*, pages 1 – 8.

Tanha, J., van Someren, M., and Afsarmanesh, H. (2017). Semi-supervised self-training for decision tree classifiers. *International Journal of Machine Learning and Cybernetics*, 8(1):355–370.

Tao, Y., Zhang, D., Cheng, S., and Tang, X. (2018). Improving semi-supervised self-training with embedded manifold transduction. *Transactions of the Institute of Measurement and Control*, 40(2):363–374.

Vale, K. M. O., de P. Canuto, A. M., de Medeiros Santos, A., da Luz e Gorgônio, F., de M. Tavares, A., Gorgônio, A. C., and Alves, C. T. (2018). Automatic adjustment of confidence values in self-training semi-supervised method. *International Joint Conference on Neural Networks*.

Zhu, X. (2008). Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.

Zhu, X. and Goldberg, A. B. (2009). *Introduction to semi-supervised learning*. Morgan & Claypool Publishers, San Rafael, California, 3 edition.