Classificação automática de textos via aprendizado semissupervisionado baseado em uma única classe e representações em redes

Bruno Moraes Aranda[1], **Rafael Geraldeli Rossi**[2], **Franciene Duarte Gomes**[3]

[1]Unersidade Federal de Mato Grosso do Sul (UFMS)
Caixa Postal 210, CEP 79620-080 – Três Lagoas – MS – Brazil

{bruno.aranda,rafael.g.rossi,franciene.gomes}@ufms.br

Resumo. *Atualmente há uma quantidade massiva de textos sendo produzida no universo digital. Esse grande conjunto de textos pode conter conhecimento útil para diversas áreas, tanto acadêmicas quanto empresariais. Uma das formas para extração de conhecimento e gerenciamento de grandes volumes de texto é a classificação automática. Uma maneira de tornar mais atrativo e viável a utilização da classificação automática, é utilizando o aprendizado baseado em uma única classe (AMUC), no qual é aprendido um modelo de classificação considerando apenas documentos da classe de interesse do usuário. Porém, vale ressaltar que mesmo fazendo uso das técnicas de AMUC, uma grande quantidade de exemplos rotulados para classe de interesse precisa ser informada para uma classificação acurada, o que ainda pode inviabilizar o uso prático do AMUC. Pode-se então fazer uso do aprendizado semissupervisionado baseado em uma única classe (do inglês Positive and Unlabeled Learning – PUL), o qual faz uso de exemplos não rotulados para melhorar a performance de classificação. Entretanto as técnicas de PUL encontradas na literatura fazem uso de algoritmos que, em geral, não obtém performances de classificação satisfatórias ou superiores a outros algoritmos de aprendizado semissupervisionado. Dado isso, o objetivo desse projeto é a implementação e uso de técnicas de aprendizado semissupervisionado mais adequados para a classificação de textos, como as baseadas em redes. Foram executados experimentos em 10 coleções textuais considerando diferentes quantidades de exemplos rotulados. Observou-se que ao escolher algoritmos semissupervisionados mais adequados para o aprendizado semissupervisionado, obteve-se ganhos para todas as coleções de textos. Além disso, foi possível obter melhores resultados em comparação com algoritmos baseline e melhores resultados que algoritmos de AMUC quando utilizados apenas 1 exemplo rotulado para a maioria das coleções, observando-se assim que a utilização de exemplos não rotulados nos algoritmos de PUL contribuem para o aumento de performance de classificação.*

Palavras-chave: aprendizado semissupervisionado, aprendizado baseado em uma única classe, classificação de textos, aprendizado baseado em exemplos positivos e não rotulados.

1. Introdução

Atualmente há uma quantidade massiva de textos sendo produzida diariamente, como e-mails, relatórios, artigos, postagens em redes sociais e notícias, uma vez que o formato textual é comumente utilizado para armazenar e divulgar informações [Aggarwal 2018]. As informações contidas nos dados textuais podem ser úteis tanto na área acadêmica quanto na área empresarial, para compreensão do comportamento humano, análise de opinião pública, organização de informações e extração de conhecimento [Rossi et al. 2016, Ur-Rahman and Harding 2012]. Porém, devido à grande quantidade de informação textual produzida, torna-se humanamente impossível organizar, analisar e extrair conhecimento embutido nas informações textuais. Consequentemente, técnicas para realizar tais tarefas diminuindo a intervenção humana, como a classificação (rotulação) automática de textos, têm ganhado importância nos últimos anos.

Normalmente empregam-se algoritmos de aprendizado de máquina (AM) para "aprender" a reconhecer as classes de uma coleção de textos, com base no conteúdo e rótulos dos mesmos, e com isso viabilizar a classificação de novos textos de maneira automática [Aggarwal 2018, Rossi 2015, Sebastiani 2002]. Isso permite a automatização das tarefas mencionadas acima.

A grande maioria dos trabalhos para a classificação automática de textos consideram algoritmos de aprendizado de máquina multi-classe (AMMC) [Aggarwal 2018, Rossi 2015, Sebastiani 2002]. Nessa categoria de algoritmos, é necessário fornecer exemplos rotulados de todas as classes de um problema de classificação. A partir disso é aprendido um modelo de classificação capaz de atribuir rótulos a um texto não rotulado. Porém, neste tipo de aprendizado, o classificador irá rotular o texto obrigatoriamente em uma das classes que foram informadas durante o seu aprendizado. Entretanto, em muitas aplicações práticas, o usuário pode não conhecer todas as classes *a priori* do problema de classificação, novas classes podem surgir ao longo do tempo, ou ainda, o usuário pode querer fornecer apenas textos rotulados da classe de seu interesse para o algoritmo de AM.

Para lidar com os cenários informados acima, onde o fornecimento de exemplos rotulados apenas da classe de interesse é fundamental ou interessante para a aplicação prática da classificação automática, em aplicações como em sensores *web* ou sistemas de recomendação, ou ainda em situações onde há um desbalanceamento dos exemplos de treinamento muito acentuado, pode-se fazer uso do aprendizado de máquina baseado em uma única classe (AMUC), do inglês *one-class learning* [Kemmler et al. 2013, Li et al. 2009, Chandola et al. 2009, Tax 2001]. Nesse tipo de aprendizado, apenas a classe de interesse, também chamada de classe alvo (*target class*) ou classe positiva, é informada ao algoritmo de AM, sendo portanto um tipo de aprendizado supervisionado. Após o aprendizado, o algoritmo de classificação gerado é capaz de discriminar a classe de interesse das demais classes.

A maioria dos trabalhos na literatura utilizam o AMUC [Kemmler et al. 2013, Chandola et al. 2009, Tax 2001]. Neste cenário, o algoritmo irá extrair o padrão da classe de interesse considerando apenas exemplos rotulados da mesma, e a partir deste ponto, dado um exemplo a ser classificado, esse algoritmo irá inferir a confiança desse exemplo para a classe de interesse. Se essa confiança estiver acima de um limiar, esse exemplo será classificado como sendo da classe de interesse, caso contrário, será classificado como da

classe de não interesse, exemplo negativo ou ainda um *outlier*.

Vale ressaltar que para realizar uma classificação acurada, normalmente são requeridos muitos exemplos rotulados das classes do problema de classificação, inclusive para o AMUC [Tax 2001], o que pode inviabilizar o uso prático do mesmo. Para diminuir ainda esse custo de rotulação do usuário, pode-se fazer uso do aprendizado semissupervisionado baseado em uma única classe, também conhecido como aprendizado com exemplos positivos e não rotulados (ou do inglês *Positive and Unlabeled Learning*) [Bekker and Davis 2020, Sakai et al. 2017, Li et al. 2009, Elkan and Noto 2008]. Neste tipo de aprendizado, considera-se tanto os exemplos positivos rotulados quanto os não rotulados, de forma a tentar aumentar o número de exemplos da classe de interesse e consequentemente melhor caracterizar os exemplos desta classe.

Na literatura, a maneira mais comum de realizar o *PUL* é por meio de um *framework* contendo duas etapas [Bekker and Davis 2020]. A primeira etapa consiste na identificação de exemplos negativos confiantes, com o intuito de se obter os contraexemplos da classe de interesse, e (ou) exemplos positivos confiantes, objetivando-se aumentar a quantidade de exemplos da classe de interesse. Já na segunda etapa, uma vez que se têm-se exemplos positivos, negativos, e não rotulados, aplica-se um algoritmo de aprendizado semissupervisionado transdutivo multiclasse para rotular os exemplos não rotulados. Com isso, ao final do processo, têm-se o rótulo de todos os exemplos não rotulados, o que por si só pode ser usado como resultado do processo de classificação. Porém, pode-se ainda fazer uso dos exemplos rotulados obtidos para induzir um modelo de classificação utilizando qualquer algoritmo de aprendizado binário ou multiclasse [Rossi et al. 2017].

Apesar dos benefícios do uso do *PUL*, grande parte dos algoritmos utilizam algoritmos generativos (ou probabilísticos) e algoritmos como o Naïve Bayes e Rocchio na primeira etapa e os algoritmos *Expectation Maximization* (*EM*) e *Transductive Support Vector Machine* (*TSVM*) na segunda etapa [Zhang and Lee 2005, Li and Liu 2003, Liu et al. 2002]. Porém, vale ressaltar que em projetos anteriores de Iniciação Científica, trabalhos de conclusão de curso e outros estudos na literatura, apontam que tais métodos são menos acurados que outros existentes [Gôlo et al. 2019], como os baseados em redes ref:Rossi2017,ref:Rossi2016. Por exemplo, [Júnior and Rossi 2018] e [Gôlo et al. 2019] realizam uma extensa avaliação de algoritmos de AMUC, os quais são utilizados na primeira etapa do *framework PUL*, na classificação de textos e demonstram que abordagens probabilísticas obtiveram performances de classificação inferiores a outras abordagens, como as baseadas nos k-*Nearest Neighbors* (k-NN), k-*Means*, ou ainda uma proposta baseada no algoritmo *Inductive Model based on Bipartite Heterogeneous Networks* (*IMBHN*) [Rossi 2015]. Apenas em [Li et al. 2009] é utilizado abordagem baseada em k-Means na primeira etapa. Além disso, vale ressaltar que o uso de EM e TSVM proveem resultados inferiores a outras abordagens no aprendizado transdutivo semissupervisionado aplicado na classificação de textos, como as baseadas em redes [Rossi et al. 2017, Rossi et al. 2016, Rossi et al. 2014b, Chapelle et al. 2006]. Em [Ma and Zhang 2017] é apresentada uma abordagem baseada em redes, porém, a única rede consideradas é uma rede k-NN e o único algoritmo de aprendizado transdutivo utilizado é o *Learning with Local and Global Consistency* [Zhou et al. 2004].

Dado os benefícios do AMUC semissupervisionado e dadas as lacunas da área como os algoritmos supervisionados utilizados na primeira etapa, e a ausência de algo-

ritmos com maior performance de classificação e/ou com menor custo computacional na segunda etapa, neste projeto foi implementado os algoritmos de AMUC com melhores performances na primeira e segunda etapas do *framework PUL*. As propostas foram comparadas diretamente com algoritmos da literatura, bem como o uso de algoritmos comumente utilizadas em ambas as etapas do *framework PUL*. Também foram realizadas comparações para verificar se de fato o uso de documentos não rotulado pelos algoritmos de PUL de fato contribuem para o aumento da performance de classificação.

A partir dos resultados coletados, observou-se que ao adotar algoritmos mais adequados em ambas etapas do *framework PUL*, obtêm-se melhores resultados de performance em todas as coleções de textos utilizadas quando comparado com algoritmos *baselines*. Além disso, observou-se melhores resultados em 7 de 10 coleções, quando comparados aos resultados obtidos através dos algoritmos AMUC, utilizando 1 exemplo rulado, e 6 de 10 coleções utilizando 5 exemplos rotulados. Portanto, nota-se os benefícios da utilização de exemplos não rotulados para classificação em cenários onde há poucos exemplos rotulados da classe de interesse (mais especificamente 1 e 5 exemplos).

O restante deste trabalho de conclusão de curso está dividido na seguinte forma. A seção 2 apresenta os conceitos necessários para o entendimento deste trabalho, junto aos trabalhos relacionado à área. Na seção 3 são apresentados os detalhes do método de pesquisa utilizado para atingir os objetivos do trabalho proposto, como as características das coleções e algoritmos utilizados, bem como o passo a passo para a obtenção dos resultados. Na Seção 4 são apresentados os resultados obtidos durante o desenvolvimento deste projeto e as discussões a cerca dos resultados. Por fim, na Seção 5 são apresentadas as considerações finais e são apontadas indicações para trabalhos futuros.

2. Trabalhos Relacionados e Embasamento Teórico

Nessa sessão, serão apresentados os trabalhos relacionados aos conceitos utilizados nesse projeto e o embasamento teórico necessário para o entendimento do mesmo, sendo a definição do AMUC, *framework PUL* e algoritmos utilizados.

2.1. Aprendizado de Maquina Baseado em Uma Única Classe

O Aprendizado de Maquina Baseado em Uma Única Classe (AMUC) pode ser encontrado na literatura de diversas maneiras, como: aprendizagem de conceito, detecção de anomalias e detecção de *outlier*. Essa abordagem, consiste no treinamento de modelos de aprendizagem, fornecendo exemplos da classe de interesse, que representem fortemente suas características, afim que a partir desses exemplos, o classificador possa determinar quais novos exemplos são ou não da classe de interesse [Bekker and Davis 2020, Wang et al. 2019, Khan and Madden 2014, Tax 2001].

Há diversos algoritmos de AMUC propostos na literatura, que possuem abordagens relacionadas a distância, *clusters*, *ensembles*, densidade, grafos, *deep learning*, e aprendizagem ativa [Chalapathy and Chawla 2019, Akoglu et al. 2015, Pimentel et al. 2018, Tan et al. 2006, Tax and Duin 2001]. Porém, ainda a poucos estudos presentes na literatura que utilizam AMUC para classificação de textos [Alam et al. 2020, Bekker and Davis 2020, Faustini and Covões 2019, Wang et al. 2019, Khan and Madden 2014].

Em [Faustini and Covões 2019] foi proposta uma nova abordagem chamada *DC-DistanceOCC*, afim de identificar notícias falsas. O modelo é uma adaptação do *Document-Class Distance* (DCDistance) [Ferreira et al. 2018]. A adaptação busca transformar o modelo originalmente pensado para redução de dimensionalidade em um AMUC. Os dados para a execução do experimento foram coletados através das plataformas *Twitter* e *WhatsApp* considerando o período das eleições gerais de 2018. Também foram utilizados outros conjuntos de dados já encontrados na literatura. O modelo sugerido teve resultados superiores ou semelhantes com outros modelos já propostos.

Em [Sonbhadra et al. 2020] é apresentado a extração de tendências de 45.000 artigos sobre Covid-19 envolvendo 75 categorias para auxiliar a comunidade cientifica na exploração sobre técnicas de prevenção e tratamento. Algoritmos de agrupamento como, k-*Means*, foram utilizados para o agrupamento de documentos considerados de categorias semelhantes. Os exemplos de cada cluster foram submetidos ao algoritmo OCSVM, considerando estes exemplos como sendo da classe de interesse. Portanto, se há k *clusters*, há k modelos OCSVM. A solução obteve mais de 89% de F_1 *Score*.

2.2. Aprendizado semissupervisionado baseado em uma única classe

O aprendizado semissupervisionado baseado em uma única classe (do inglês *Positive and Unlabeled Learning* – PUL), busca construir classificadores através de documentos rotulados e não rotulados da classe de interesse (aprendizagem semissupervisionada indutiva) ou para classificar exemplos conhecidos não rotulados (aprendizagem semissupervisionada transdutiva) [Jaskie and Spanias 2019]. A utilização dos exemplos não rotulados durante o treinamento do modelo busca melhorar a performance de classificação, como já visto na aprendizagem semissupervisionada multiclasse [Zhu and Goldberg 2009]. Ademais, exemplos não rotulados são facilmente coletados, e se faz necessário por parte do usuário, a classificação de uma menor quantidade de documentos. Por esses motivos o *framework PUL* tem ganhado atenção nos últimos anos [Jaskie and Spanias 2019, Zhang et al. 2019, Ma and Zhang 2017].

As abordagens mais frequentemente encontradas, são as que fazem uso do aprendizado em duas etapas. A primeira etapa, consiste na obtenção de exemplos positivos e negativos confiáveis. Feito isso, a segunda etapa consiste na aplicação de algoritmos de aprendizagem transdutiva, para inferir os rótulos nos documentos não rotulados restantes [Jaskie and Spanias 2019].

[de Souza et al. 2021] propôs uma adaptação da abordagem PUL-LP (*Positive and Unlabeled Learning by Label Propagation*) para a detecção noticias falsas. O identifica documentos positivos e negativos confiantes através de caminhos na rede, sendo que os documentos que na média possuem o menor caminho para os exemplos positivos são atribuídos ao conjunto de exemplos positivos confiáveis, e os de maior caminho são atribuídos ao conjunto de exemplos negativos confiáveis. Uma vez tendo exemplos positivos (incluindo os confiáveis), negativos e não rotulados, um processo de propagação de rótulos é realizado para rotulados os demais documentos da rede. Essa abordagem demonstrou ser capaz de superar os resultados de algoritmos de AMUC e de algoritmos binários para a maioria das coleções utilizadas.

2.3. Algoritmos

Na primeira etapa do *framework PUL* foram utilizados os seguintes algoritmos:

***k*-Means (*k*ME):** esse modelo [Tan et al. 2006] busca o posicionamento de centroides para geração de grupo de dados que damos o nome de *clusters*, sendo assim os dados encontrados em cada *cluster* apresenta maior proximidade com seu centroide. Centroides são pontos artificiais gerados aleatoriamente ao ser iniciado a execução do algoritmo e após este início, os documentos são delegados ao *cluster* referente ao centroide mais próximo, conseguinte, os centroides são recalculados, buscando a posição central do *cluster*. Dado um valor de k, k grupos devem ser criados, ou seja, k centroides serão estabelecidos. O processo de recalculo dos centroides se repete até que nenhum documento altere o grupo de dados a qual pertence, ou que o numero de iterações definido seja atingido.

***k*-Nearest Neighbors Density-based (*k*NND):** este é uma vertente do algoritmo *k-Nearest Neighbors* [Tan et al. 2006], comumente utilizado na área de aprendizado de máquina. Tem seu funcionamento a partir da identificação de exemplos próximos a classe de interesse através da utilização da densidade. Sendo assim, é possível prever se um documento pertence a classe de interesse quando se encontra em uma região de alta densidade. Para o cálculo de densidade, deve ser inferido o valor k (número de vizinhos), dado o valor de k a densidade é obtida através da média de proximidade dos k vizinhos mais próximos de um documento d_i, dado pela Equação 1:

$$densidade(d_i, k) = \frac{\sum_{d_j \in N_{(d_i,k)}} proximidade(d_i, d_j)}{\left|N_{(d_i,k)}\right|} \tag{1}$$

na qual $N_{(d_i,k)}$ se refere ao conjunto de vizinhos mais próximos de d_i. Para o cálculo de proximidade nesse projeto, foi utilizado a medida de similaridade Cosseno. O cálculo de proximidade entre os documentos d_i e d_j utiliza a medida de similaridade Cosseno, dado pela Equação 2:

$$proximidade(d_i, d_j) = \frac{\sum_{t_k \subset T} w_{d_i,t_k} \cdot w_{d_j,t_k}}{\sqrt{\sum_{t_k \subset T} w_{d_i,t_k}} \cdot \sqrt{\sum_{t_k \subset T} w_{d_j,t_k}}} \tag{2}$$

***Inductive Model based on Bipartite Heterogeneous Networks* (IMBHN):** esse modelo [Rossi et al. 2014a] busca induzir um classificador a partir de representações textuais em redes bipartidas, inferindo os *scores* de relevância dos termos para a classe de interesse, de forma que dado um novo documento, o *score* deste deverá ser próximo de 1 se o documento pertence a classe de interesse, conforme apresentado na Equação 3:

$$f(d_i) = \sum_{t_j \in \tau} w_{d_i t_j} \cdot F_{t_j}, \tag{3}$$

na qual F_{t_j} é o *score* de relevância para o termo t_j.

Já na segunda etapa, que trata do aprendizado semissupervisonado transdutivo, foram utilizados os seguintes algoritmos:

***Transductive Classification based on Bipartite Heterogeneous Network* (TCBHN):** Esse modelo busca realizar uma classificação transdutiva, a partir da utilização de redes bipartidas, para obtenção de *scores* de relevância dos termos para a classes

positiva e negativa. O algoritmo realiza um processo de propagação de rótulos para os documentos não rotulados e um processo de otimização utilizando o algoritmo *Least Mean Squares* para inferir os *scores* de relevâncias dos termos. Os processos de propagação e otimização são realizados repetidas vezes até que a classificação dos exemplos não rotulados estabilize, ou seja não haja alterações das classes dos documentos [Rossi et al. 2016].

***Transductive Support Vector Machines* (TSVM):** esse modelo busca induzir através de exemplos rotulados e não rotulados um hiperplano de separação de margem máxima [Vapnik and Vapnik 1998, Joachims 1999]. Sendo os exemplos não rotulados os responsáveis por mover hiperplano de separação para regiões não densas. Levando em consideração a posição dos exemplos não rotulados em relação ao hiperplano, são delegados os rótulos aos exemplos.

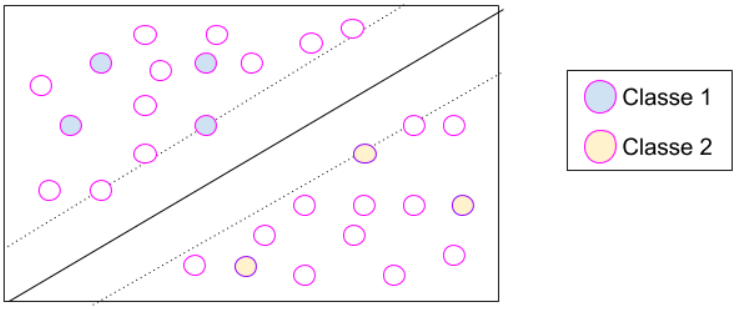


Figura 1. Representação de execução do algoritmo TSVM.
Fonte: Autoria Própria - baseada em [Rossi 2015]

Por fim, também foi implementado nesse projeto o algoritmo de *PUL* RC-SVM [Li and Liu 2003], para fins de comparação com a literatura. O mesmo faz o uso do algoritmo Rocchio [Rocchio 1971] na primeira etapa, para geração de protótipos de documentos positivos e não rotulados. Dado isso, os documentos não rotulados que apresentam maior similaridade aos protótipos dos documentos não rotulados do que aos documentos da classe de interesse, são escolhidos como negativos. A similaridade é calculada através da medida de similaridade Cosseno, dada pela Equação 2. Na segunda etapa, é aplicado o algoritmo SVM [Vapnik 2013] para classificação dos documentos não rotulados restantes. Para que isso seja feito, o SVM é treinado com os documentos positivos e negativos extraídos da etapa anterior, sendo assim utilizado para classificar os exemplos não rotulados. Os documentos classificados como negativos pelo SVM, são retirados do grupo de documentos não rotulados e inseridos no grupo de exemplos negativos, e assim, o SVM é treinado novamente. Esse processo se repete iterativamente até que nenhum dos documentos do grupo de não rotulados seja classificado como negativo.

3. Método de Pesquisa

O método de pesquisa adotado para atingir os objetivos deste projeto foi divido em 4 etapas, a saber: (i) coleta de coleções de textos para que se pudesse realizar a avaliação experimental; (ii) representação das coleções de textos em um formato estruturado para que os algoritmos de aprendizado de máquina possam processá-las; (iii) implementação dos algoritmos propostos bem como os algoritmos utilizados na comparação; e (iv)

realização da avaliação experimental. Os detalhes dos passos utilizados no método de pesquisa utilizado neste projeto são apresentados nas próximas subseções.

3.1. Coleta de Coleções de Textos

Foram coletadas 10 bases presentes em um *benchmarking* que possuem coleções de diversos domínios, que podem ser encontradas em [Rossi et al. 2013]. As bases coletadas possuem os seguintes domínios: Científico (CI), Novos Artigos (NA), Documentos Médicos (DM), Páginas *Web* (PW), Documentos da TREC (DT).

Um ponto que deve ser levado em consideração é que as bases utilizadas nesse projeto são bases multi-classe. Isso se deve a baixa disponibilidade de bases com características de única classe. Porém essa lacuna foi sanada no processo de avaliação experimental, onde em cada iteração uma classe é selecionada como de interesse, enquanto as outras são consideradas como classes de não interesse. Mais detalhes quanto às características das coleções utilizadas nesse trabalho são apresentadas na próxima seção.

3.2. Representação das Coleções de Textos

Para que seja possível a aplicação dos algoritmos, é necessário que as coleções de textos passem por um pré-processamento, para que sejam estruturadas e assim interpretadas pelos algoritmos de aprendizado de máquina [Rossi et al. 2015]. Nesse projeto, os textos foram estruturados e representados através do modelo espaço-vetorial [Salton 1989] com a técnica de *bag-of-words*, por ser um técnica que apresenta boa performance, baixa complexidade e ampla utilização na literatura. Nessa estrutura, cada documento é representado por um vetor e em cada posição está o peso de cada termo que representa uma característica da coleção. Afim de determinar o peso dos termos nesse projeto, foi-se utilizado o esquema de pesos *Term Frequency Inverse Document Frequency* (*TF-IDF*) [Salton 1989].

Para gerar uma representação mais concisa de forma a prover melhores resultados para os algoritmos de aprendizado de máquina, alguns passos são comumente utilizados, como:

- **Padronização dos termos**: consiste na retirada da acentuação das palavras contidas no documento junto a padronização de caixas.
- **Remoção de *Stopwords***: remoção de termos irrelevantes para determinar a classe de um documento, como conectivos, preposições e verbos de estado. Exemplos de *Stopwords* são: as, e, os, de, para, com, sem, foi.
- **Radicalização de termos**: consiste em utilizar o radical do termo ao invés do termo completo, de forma que as variações de gênero número e grau não geram diferentes atributos na representação no modelo espaço-vetorial (exemplo: `pedra` e `pedreira` viram `pedr`). Foi utilizado o algoritmo de Porter para a radicalização dos termos [Karaa and Gribâa 2013].

Na Tabela 1 são apresentadas as características detalhadas das coleções de textos coletadas após os passos mencionados anteriormente, sendo: número de domínios que a coleção de texto possuí $|\mathcal{D}|$, número de termos que o documento possui $|\mathcal{T}|$, número médio de termos por documento $|\overline{\mathcal{T}}|$, número de classes $|\mathcal{C}|$, esparsidade de matriz **(E.M)**, desvio padrão tendo em vista o percentual da ocorrência das classes $\sigma(\mathcal{C})$, porcentagem de documentos pertencentes à classe majoritária $max(\mathcal{C})$ e seus domínios [Rossi et al. 2013].

Tabela 1. Características e propriedades das coleções de textos.

Coleção	$\|\mathcal{D}\|$	$\|\mathcal{T}\|$	$\|\overline{\mathcal{T}}\|$	E.M.	$\|\mathcal{C}\|$	$\sigma(\mathcal{C})$	$max(\mathcal{C})$	Domínio
CSTR	299	1726	54.27	96.86%	4	15.89%	42.81%	CI
La2s	3075	12433	144.52	98.84%	6	8.59%	29.43%	NA
Oh0	1003	3183	52.50	98.35%	10	5.33%	19.34%	DM
Oh5	918	3013	54.43	98.19%	10	3.72%	16.23%	DM
Oh10	1050	3239	55.63	98.28%	10	4.25%	15.71%	DM
Oh15	3101	54142	17.46	99.97%	10	1.26%	5.06%	DM
SyskillWebert	334	4340	93.15	97.85%	4	10.75%	41.02%	PW
Tr11	414	6430	281.66	95.62%	9	9.80%	31.88%	DT
Tr12	313	5805	273.59	95.29%	8	7.98%	29.71%	DT
Tr21	336	7903	469.86	94.05%	6	25.88%	68.75%	DT

3.2.1. Avaliação experimental

Nesta subseção são apresentados os detalhes a respeito das medidas de performance de classificação utilizadas, como foram gerados os valores dessas medidas, bem como são apresentados os parâmetros dos algoritmos utilizados para a obtenção dos resultados.

3.2.2. Critérios de Avaliação

Para a execução dos experimentos, primeiro foi necessário simular que as coleções multi-classe coletadas tivessem características de bases de uma única classe, devido à baixa disponibilidade de coleções de textos com essas características. Para isso, a primeira etapa do processo de avaliação consiste em selecionar uma classe dentre as disponíveis na coleção e elenca-la como de interesse, considerando todas as outras classes como exemplos sem categoria. Para cada classe de interesse selecionada, foram executados experimentos considerando 1, 5 e 10 exemplos positivos, os quais foram selecionados de maneira aleatória dentre os exemplos da classe de interesse. Já os exemplos restantes da classe de interesse, bem como os exemplos das demais classes foram considerados como não rotulados. Para cada quantidade de exemplos positivos considerada o processo foi repetido 10 vezes. Vale ressaltar que esse procedimento é repetido para cada classe da coleção, e ao final, os resultados obtidos consideram as 10 repetições para cada classe.

Após a definição dos exemplos positivos, de acordo com a quantidade e classe selecionados, os demais exemplos (considerando os das classes de não interesse) são considerados como exemplos não rotulados e de teste. Tais exemplos serão avaliados para definir a performance de classificação. Como métricas de performance, foi utilizada a F_1 *Score*. A métrica F_1 *Score*, é dada por:

$$F_1 = \frac{2 \cdot Precisão \cdot Recall}{Precisão + Recall} \tag{4}$$

$$Precisão = \frac{VP}{VP + FP} \tag{5}$$

$$Revocação = \frac{VP}{VP + FN} \tag{6}$$

sendo VP (verdadeiros positivos) o número de documentos classificados positivos corretamente, FN (falsos negativos) o número de documentos positivos classificados como negativos e FP (falsos positivos) o número de documentos negativos classificados como positivos. Vale ressaltar que os resultados apresentados na Seção 4 consideram a média dos valores de F_1 *Score* obtida através das 10 interações citadas acima.

3.2.3. Parâmetros de execução dos algoritmos

A seguir estão descritos os parâmetros utilizados para execução dos algoritmos abordados:

- ***kNN-Density***: $k \in \{1,3,5,7,9,11,13,15,17,19\}$, utilizando o cosseno como medida de similaridade.
- ***k-Means***: $k \in \{1,3,5,7,9,11,13,15,17,19\}$, utilizando a medida de proximidade similaridade cosseno e com máximo de iterações definido em 100.
- ***IMBHN***: o número máximo de iterações foi definido em 100, *Error Correction Rate*: $\{0.01, 0.05, 0.1, 0.5\}$ e *Minimum Mean Squared Error*: $\{0.001, 0.005, 0.01, 0.05\}$
- ***TCBHN***: O máximo de iterações globais foi definido em 10 e as locais em 100, *Error Correction Rate*: $\{0.01, 0.05, 0.1, 0.5\}$ e *Minimum Mean Squared Error*: $\{0.01\}$
- ***TSVM***: C $\in \{0.01, 0.1, 1.0, 10.0\}$, com o máximo de iterações definido em 50.
- ***RCSVM***: Para a primeira etapa de execução o algoritmo *Rocchio* utiliza: $\alpha \in \{00.2, 0.4, 0.6, 0.8, 1.0\}$ e β = 1-α. Para a segunda etapa de execução o algoritmo *SVM* utiliza: C $\in \{0.1, 1.0, 10, 100, 1000\}$

Vale ressaltar que os algoritmos *kNN-Density*, *k-Means* e *IMBHN* também foram avaliados como algoritmos de AMUC e não só como parte do *framework PUL*. Isso foi feito para analisar se o uso de exemplos não rotulados foi útil para aumentar a performance de classificação.

É importante ressaltar que outros algoritmos de aprendizado transdutivo semissupervisionados foram considerados neste trabalho, como o *Learning with Local and Global Consistency* [Zhou et al. 2004], [**?**] [Zhou et al. 2004] e [**?**] [Ji et al. 2010]. Entretanto, não foi possível obter resultados a tempo para estes algoritmos. Além disso, é importante ressaltar que as abordagens propostas neste projeto se tratam da utilização dos algoritmos k**ME,** K**NND e IMBHN** na primeira etapa, e o algoritmo **TCBHN** na segunda etapa. Sendo assim, os algoritmos Rocchio e TSVM estão presentes a fim de demonstrar a disparidade de performance com a abordagem proposta, sendo essa a integração considerada *baseline* pela literatura pois esses algoritmos são utilizados em outras abordagens.

4. Resultados e Discussões

Para execução dos experimentos foram utilizadas as bases CSTR, La2s, Oh0, Oh10, Oh15, Oh5, SyskillWebert, Tr11, Tr12 e TR21 [Rossi et al. 2013], sendo as mesmas representadas através do modelo espaço-vetorial, utilizando-se da técnica *bag-of-words* com os pesos dos termos determinados através da medida estatística $TF\text{-}IDF$. Os algoritmos tiveram sua execução baseada em diferentes quantidades de exemplos rotulados, sendo elas: 1, 5 e 10 exemplos rotulados.

Na Tabela 2 e Figura 2, são apresentados os resultados obtidos através da execução dos algoritmos, considerando a base CSTR. Observa-se que a abordagem proposta kME-TCBHN apresenta melhor performance, quando considerado 1 exemplo rotulado. É importante ressaltar que a abordagem IMBHN-TCBHN tem uma performance similar, com a disparidade de apenas $0,002$ pontos. Já quando considerado 5 exemplos, o algoritmo IMBHN-TCBHN tem a melhor performance, sendo essa de $0,7155$ pontos. No entanto, quando considerado 10 exemplos rotulados, o algoritmo IMBHN (AMUC) apresenta maior performance, porém, vale ressaltar que os algoritmos IMBHN-TCBHN, kNND-TCBHN e kME-TCBHN tiveram performances superiores as outras abordagens de AMUC. Ademais, observa-se que o algoritmo RC-SVM apresentou uma performance inferior a todas as outras abordagens, e em todos os cenários (1, 5 e 10 exemplos).

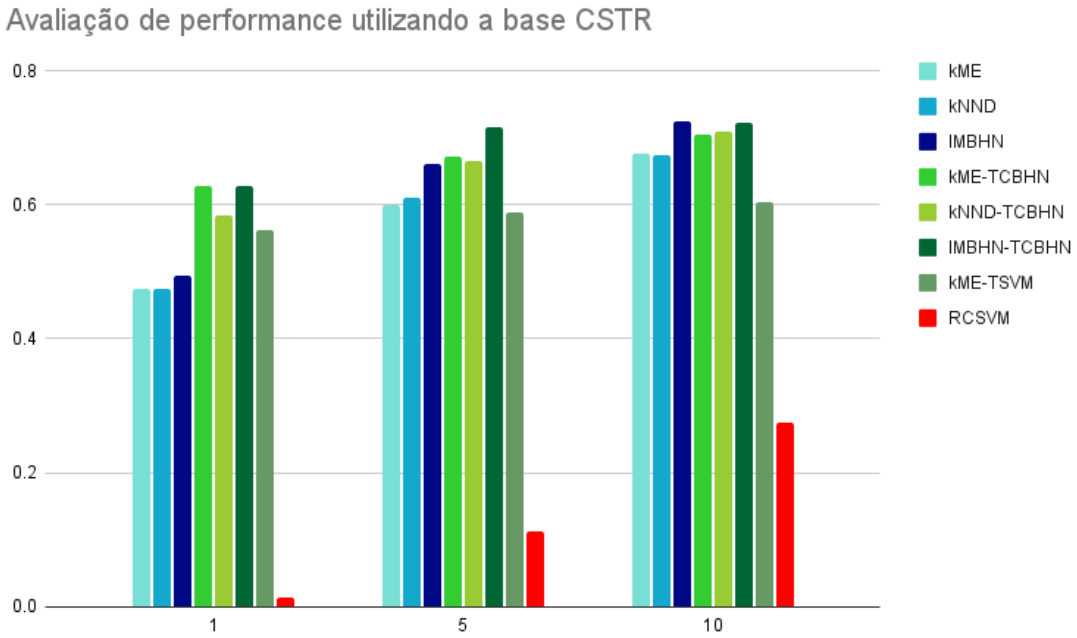
Tabela 2. Resultados experimentais considerando a base CSTR

Algoritmos	1	5	10
kME	0,4752	0,6006	0,6757
kNND	0,4752	0,6104	0,6739
IMBHN	0,4953	0,6607	**0,7256**
kME-TCBHN	**0,6276**	0,6717	0,7059
kNND-TCBHN	0,5846	0,6651	0,7094
IMBHN-TCBHN	0,6274	**0,7155**	0,7228
kME-TSVM	0,5623	0,5896	0,6037
RCSVM	0,0139	0,1124	0,2748

Na Tabela 3 e Figura 3, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base La2s. Observa-se que o algoritmo proposto kME-TCBHN teve grande destaque, considerando essa base. O mesmo teve performance superior a todas as outras abordagens, em todos os cenários experienciados. Vale ressaltar que até o momento de confecção desse relatório, o algoritmo kME-TSVM não encerrou sua execução utilizando essa base de textos, sendo assim, os resultados ainda devem ser coletados.

Na Tabela 4 e Figura 4, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base Oh0. Observa-se que todas as abordagens propostas têm performances superiores às abordagens de AMUC e ao algoritmo *baseline* presente na literatura (RC-SVM), em cenários de apenas 1 exemplo rotulado. Já no cenário de 5 exemplos rotulados, o algoritmo IMBHN-TCBHN se destaca com a maior performance, sendo sua pontuação igual a $0,6619$. Ademais, quando considerado 10 exemplos rotulados, a abordagem AMUC IMBHN se destaca. Vale ressaltar que o algoritmo proposto pela literatura apresentou performance inferior em todos os cenários.

Figura 2. Gráfico comparativo de resultados utilizando-se da base CSTR



Autoria própria.

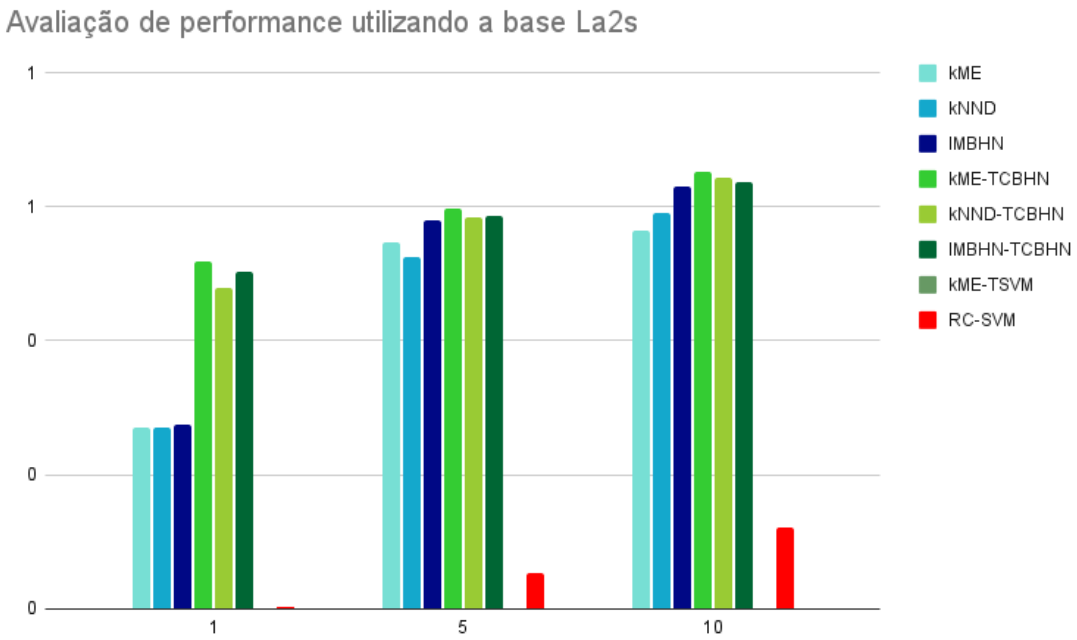
Tabela 3. Resultados experimentais considerando a base La2s

Algoritmos	1	5	10
kME	0,271	0,5469	0,5655
kNND	0,271	0,5242	0,5904
IMBHN	0,2748	0,5807	0,6315
kME-TCBHN	**0,518**	**0,5975**	**0,6534**
kNND-TCBHN	0,4791	0,5847	0,6434
IMBHN-TCBHN	0,5032	0,5858	0,6379
kME-TSVM			
RC-SVM	0,0034	0,0524	0,1213

Tabela 4. Resultados experimentais considerando a base Oh0

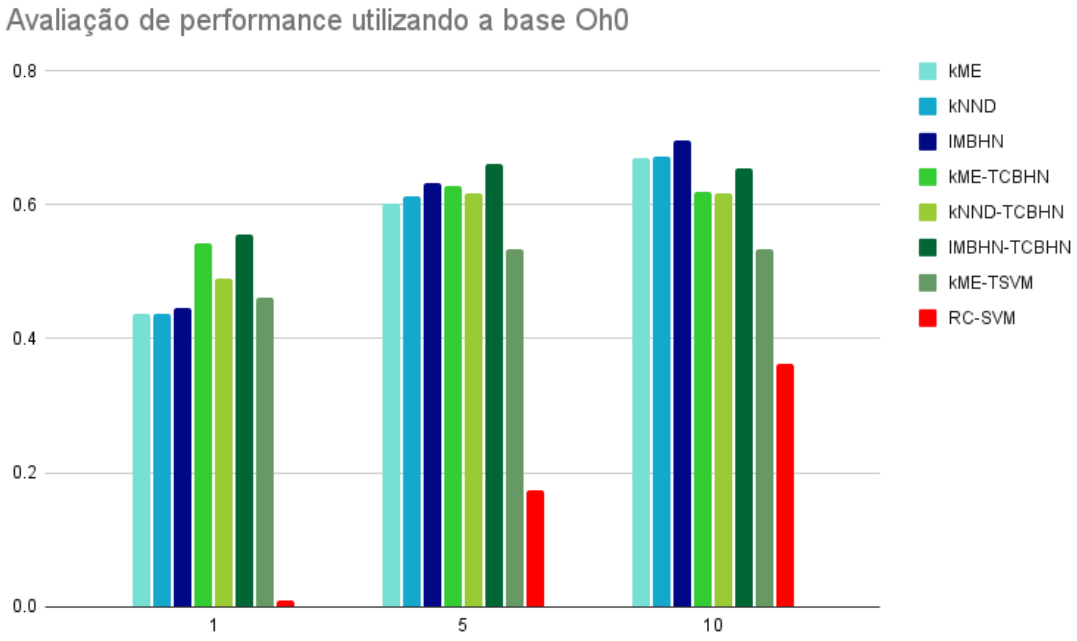
Algoritmos	1	5	10
kME	0,4380	0,6017	0,6702
kNND	0,4380	0,6124	0,6717
IMBHN	0,4456	0,6333	**0,6971**
kME-TCBHN	0,5417	0,6283	0,6206
kNND-TCBHN	0,4911	0,6185	0,6177
IMBHN-TCBHN	**0,5550**	**0,6619**	0,6538
kME-TSVM	0,4611	0,5346	0,5350
RC-SVM	0,0093	0,1730	0,3625

Figura 3. Gráfico comparativo de resultados utilizando-se da base La2s



Autoria própria.

Figura 4. Gráfico comparativo de resultados utilizando-se da base Oh0



Autoria própria.

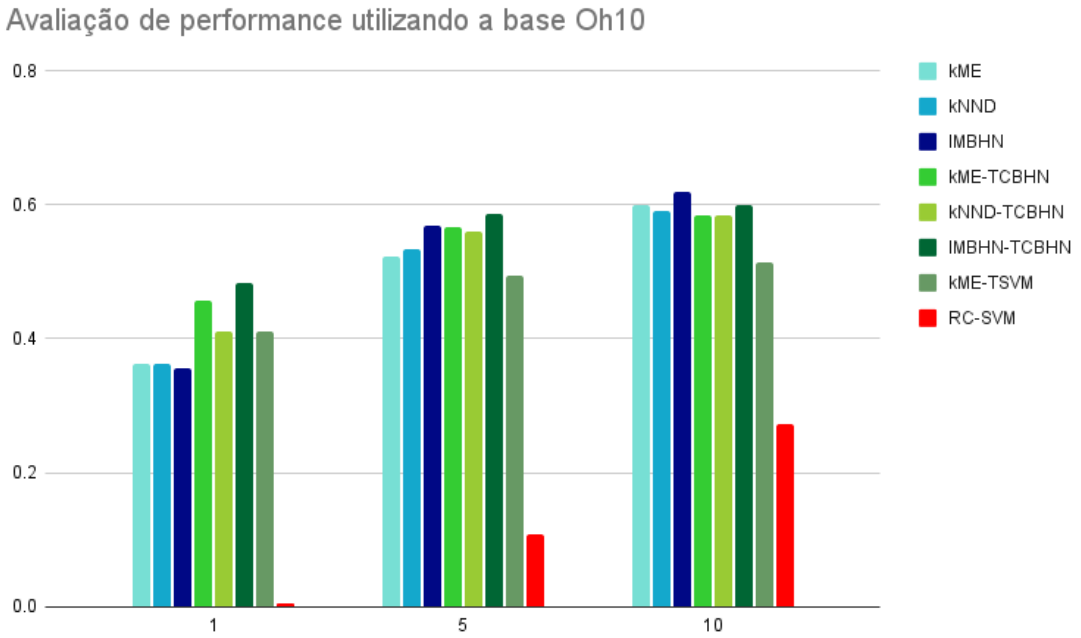
Na Tabela 5 e Figura 5, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base Oh10. Nota-se que a abordagem IMBHN-

TCBHN teve performance superior nos cenários de 1 e 5 exemplos rotulados, deste modo, é importante citar que as outras abordagens propostas também tiveram performance superior ás outras abordagens, quando expostas ao cenário de apenas 1 exemplo rotulado. Já ao tratarmos do cenário de 10 exemplos rotulados, o algoritmo de AMUC IMBHN apresentou melhor performance. Ademais, o algoritmo proposto pela literatura teve a performance mais baixa em todos os cenários.

Tabela 5. Resultados experimentais considerando a base Oh10

Algoritmos	1	5	10
kME	0,36230	0,52350	0,60060
kNND	0,36230	0,53380	0,59220
IMBHN	0,35710	0,56940	**0,61950**
kME-TCBHN	0,45650	0,56690	0,58450
kNND-TCBHN	0,41030	0,56070	0,58500
IMBHN-TCBHN	**0,48380**	**0,58640**	0,59900
kME-TSVM	0,41050	0,49440	0,51410
RC-SVM	0,0050	0,1072	0,2726

Figura 5. Gráfico comparativo de resultados utilizando-se da base Oh10



Autoria própria.

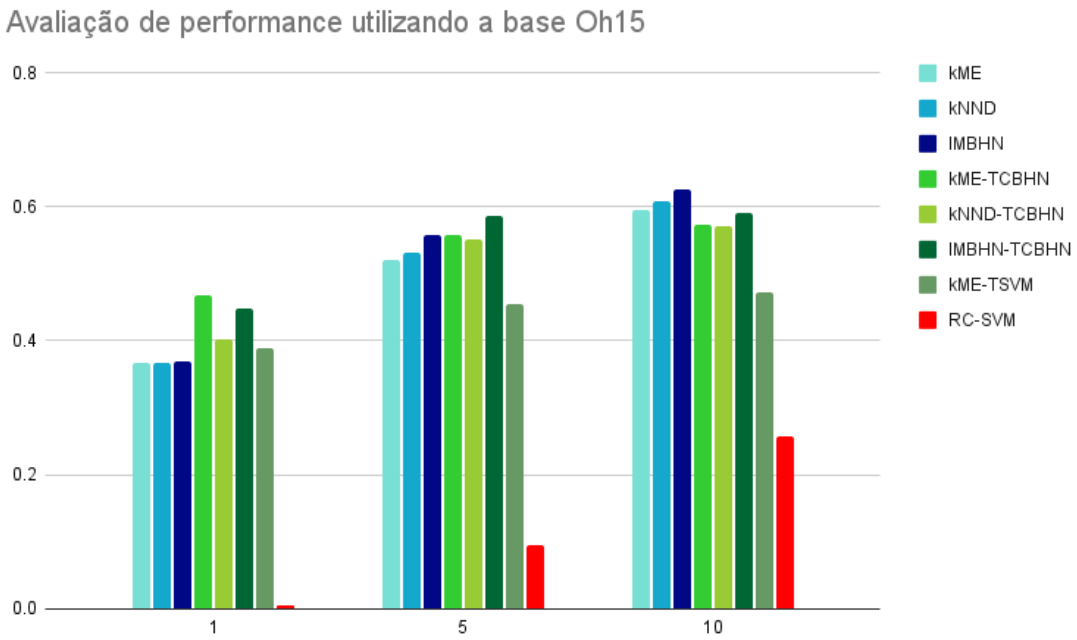
Na Tabela 6 e Figura 6, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base Oh15. Observa-se que em cenários de apenas 1 exemplo rotulado, o algoritmo kME-TCBHN apresenta maior performance, sendo sua pontuação de $0,4686$. Vale ressaltar que nesse cenário todas as outras abordagens propostas tiveram performance superior aos algoritmos de AMUC e ao algoritmo proposto pela literatura. Já em cenários de 5 exemplos rotulados o algoritmo IMBHN-TCBHN

apresenta a melhor performance. Ademais, ao considerarmos 10 exemplos rotulados, o algoritmo IMBHN teve a melhor performance observada.

Tabela 6. Resultados experimentais considerando a base Oh15

Algoritmos	1	5	10
kME	0,3681	0,5217	0,5956
kNND	0,3681	0,5324	0,6081
IMBHN	0,3686	0,5589	**0,6271**
kME-TCBHN	**0,4686**	0,5585	0,5741
kNND-TCBHN	0,4014	0,5522	0,5703
IMBHN-TCBHN	0,4489	**0,5872**	0,5913
kME-TSVM	0,3891	0,4556	0,4717
RC-SVM	0,0047	0,0938	0,2563

Figura 6. Gráfico comparativo de resultados utilizando-se da base Oh15



Autoria própria.

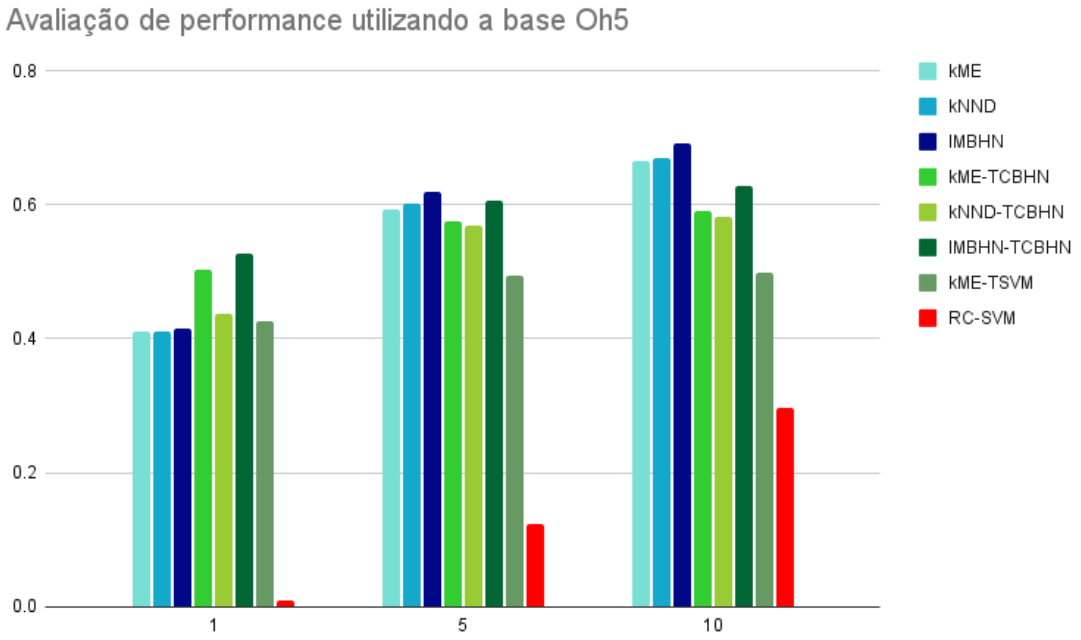
Na Tabela 7 e Figura 7, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base Oh5. Ao considerarmos apenas 1 exemplo rotulado, as abordagens propostas apresentam performances superiores, dando destaque ao algoritmo IMBHN-TCBHN o qual apresentou maior pontuação $(0,5271)$. Porém, ao analisar os outros cenários, nota-se que o algoritmo IMBHN tem a maior performance apresentada.

Na Tabela 8 e Figura 8, estão representados os resultados obtidos através da execução dos algoritmos, considerando a base SyskillWebert. Ao analisarmos os cenários

Tabela 7. Resultados experimentais considerando a base Oh5

Algoritmos	1	5	10
kME	0,4108	0,5925	0,6651
kNND	0,4108	0,6010	0,6706
IMBHN	0,4164	**0,6194**	**0,6924**
kME-TCBHN	0,5043	0,5752	0,5911
kNND-TCBHN	0,4369	0,5691	0,5818
IMBHN-TCBHN	**0,5271**	0,6072	0,6284
kME-TSVM	0,4266	0,4934	0,4995
RC-SVM	0,0085	0,1234	0,2964

Figura 7. Gráfico comparativo de resultados utilizando-se da base Oh5



Autoria própria.

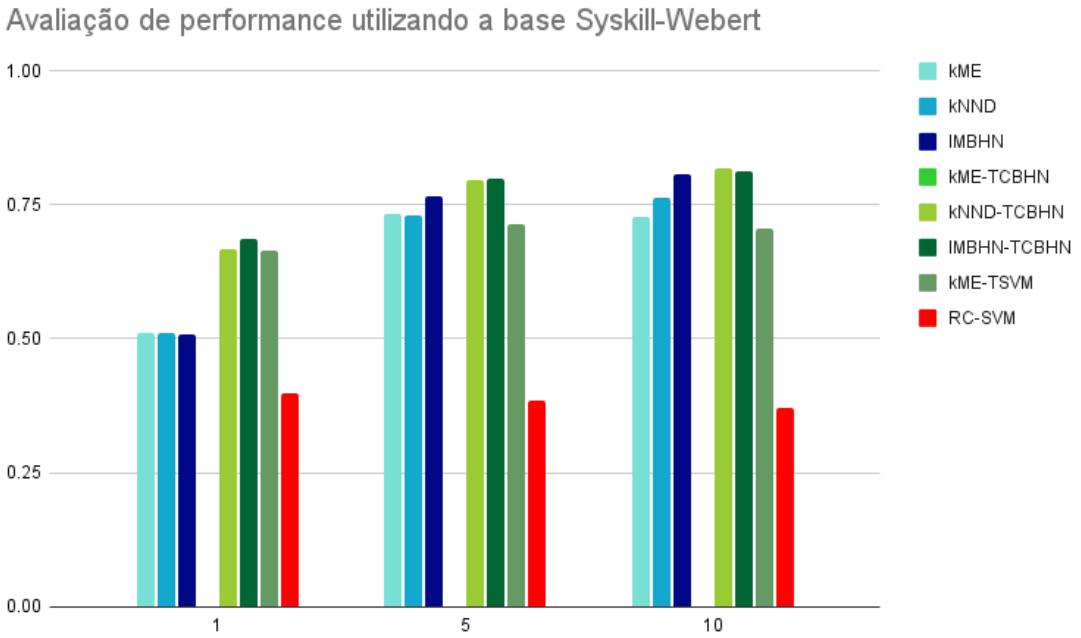
de 1 e 5 exemplos rotulados, notamos a melhor performance do algoritmo IMBHN-TCBHN. No entanto, em cenários de 10 exemplos rotulados o algoritmo kNND-TCBHN apresenta maior performance, sendo o algoritmo IMBHN-TCBHN o segundo que melhor performou, apresentando uma pequena diferença de $0,0059$ pontos. Vale ressaltar que até o momento de confecção desse relatório, o algoritmo kME-TCBHN não encerrou sua execução utilizando essa base de textos, sendo assim, os resultados ainda devem ser coletados.

Nas Tabelas 9, 10 e 11, e Figuras 9, 10 e 11 estão representados os resultados obtidos através da execução dos algoritmos, considerando as bases Tr11, Tr12 e Tr21. Observa-se que nessas bases os algoritmos propostos não conseguiram bater os algoritmos com a abordagem AMUC, sendo o algoritmo IMBHN o qual apresentou maior performance em todos os cenários. Vale ressaltar que até o momento de confecção desse

Tabela 8. Resultados experimentais considerando a base SyskillWebert

Algoritmos	1	5	10
kME	0,5098	0,7342	0,7283
kNND	0,5098	0,7298	0,7630
IMBHN	0,5084	0,7659	0,8085
kME-TCBHN			
kNND-TCBHN	0,6667	0,7960	**0,8193**
IMBHN-TCBHN	**0,6869**	**0,7993**	0,8134
kME-TSVM	0,6658	0,7132	0,7063
RC-SVM	0,3971	0,3853	0,3698

Figura 8. Gráfico comparativo de resultados utilizando-se da base SyskillWebert



Autoria própria.

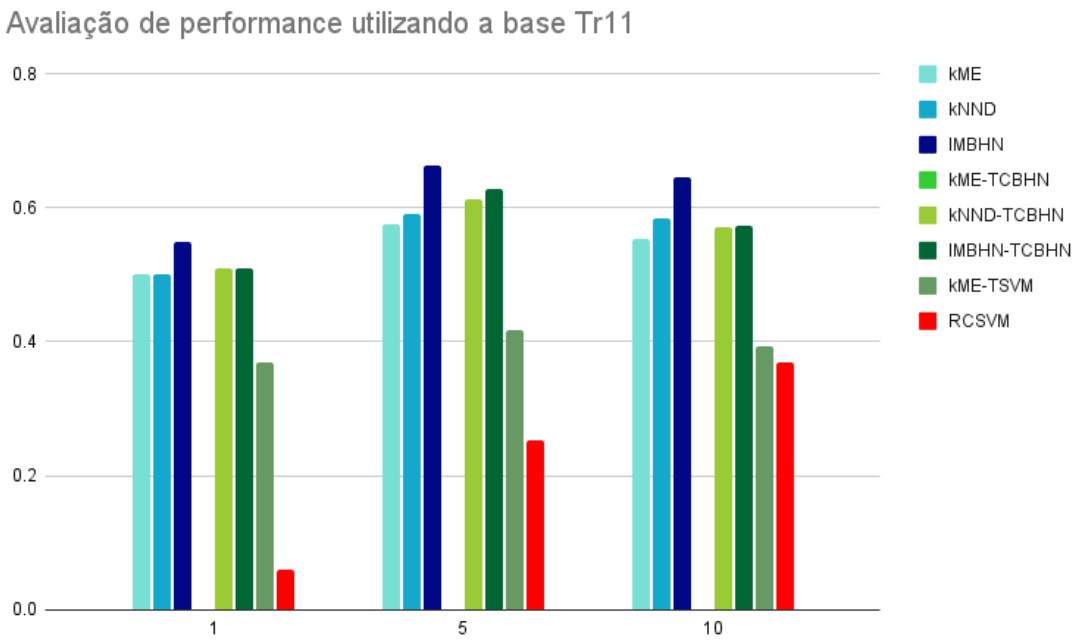
relatório, o algoritmo kME-TCBHN não encerrou sua execução utilizando as bases de textos Tr11 e Tr12, sendo assim, os resultados ainda devem ser coletados.

Analisando os resultados apresentados nessa seção, podemos observar que em todas as bases utilizadas, todos os algoritmos de *PUL* propostos, de fato tiveram performance superior ao algoritmo proposto pela literatura. Ao compararmos os algoritmos propostos com os de abordagem AMUC, é possível notar que ao considerarmos 1 exemplo rotulado, 7 de 10 coleções de texto apresentaram melhores performances ao utilizar-se do *framework PUL* com abordagens mais adequadas, os algoritmos que obtiveram melhores performances nesse cenário foram: IMBHN-TCBHN tendo 4 pontuações superiores e kME-TCBHN tendo 3 pontuações superiores. Ao considerarmos 5 exemplos rotulados, as abordagens propostas obtiveram melhores resultados em 6 de 10 coleções, sendo as abordagens que tiveram maiores resultados: IMBHN-TCBHN tendo 5 pontuações supe-

Tabela 9. Resultados experimentais considerando a base Tr11

Algoritmos	1	5	10
kME	0,5000	0,5750	0,5537
kNND	0,5000	0,5905	0,5848
IMBHN	**0,5494**	**0,6644**	**0,6468**
kME-TCBHN			
kNND-TCBHN	0,5098	0,6125	0,5720
IMBHN-TCBHN	0,5097	0,6292	0,5735
kME-TSVM	0,3693	0,4171	0,3943
RCSVM	0,0602	0,2521	0,3683

Figura 9. Gráfico comparativo de resultados utilizando-se da base Tr11

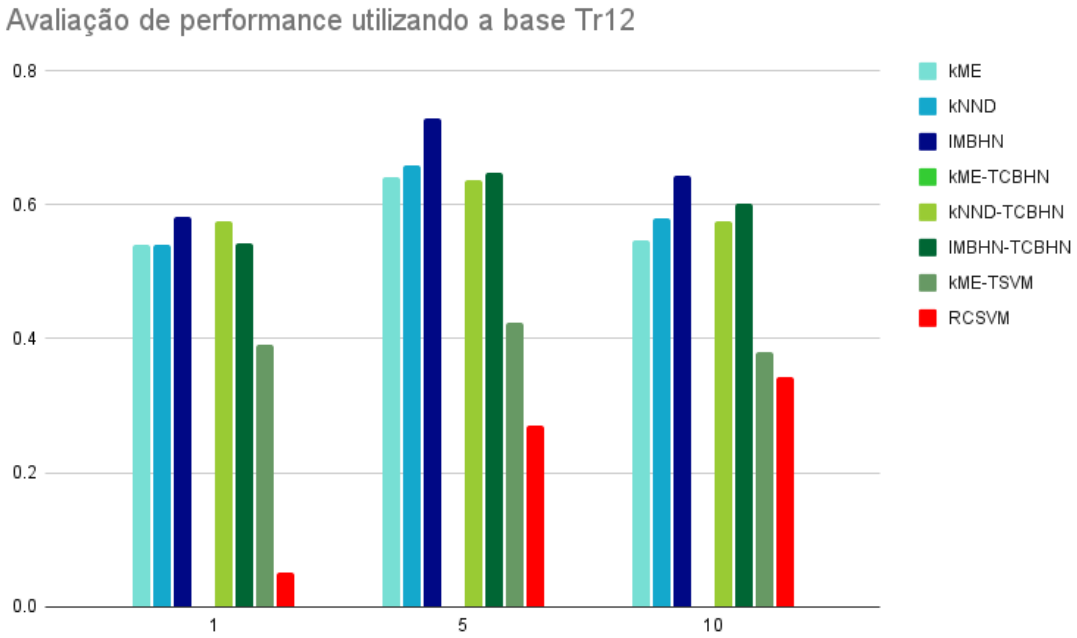


Autoria própria.

Tabela 10. Resultados experimentais considerando a base Tr12

Algoritmos	1	5	10
kME	0,5396	0,6425	0,5471
kNND	0,5396	0,6587	0,5807
IMBHN	**0,5827**	**0,7301**	**0,6436**
kME-TCBHN			
kNND-TCBHN	0,5747	0,6377	0,5758
IMBHN-TCBHN	0,5437	0,6476	0,603
kME-TSVM	0,3916	0,4243	0,3809
RCSVM	0,0500	0,2709	0,3420

Figura 10. Gráfico comparativo de resultados utilizando-se da base Tr12



Autoria própria.

Tabela 11. Resultados experimentais considerando a base Tr21

Algoritmos	1	5	10
kME	0,4827	0,4640	0,4062
kNND	0,4827	0,4892	0,4174
IMBHN	**0,5150**	**0,5114**	**0,4610**
kME-TCBHN	0,4864	0,4864	0,4226
kNND-TCBHN	0,4711	0,4854	0,4245
IMBHN-TCBHN	0,4432	0,4885	0,4203
kME-TSVM	0,354	0,3397	0,2851
RCSVM	0,0425	0,1818	0,2135

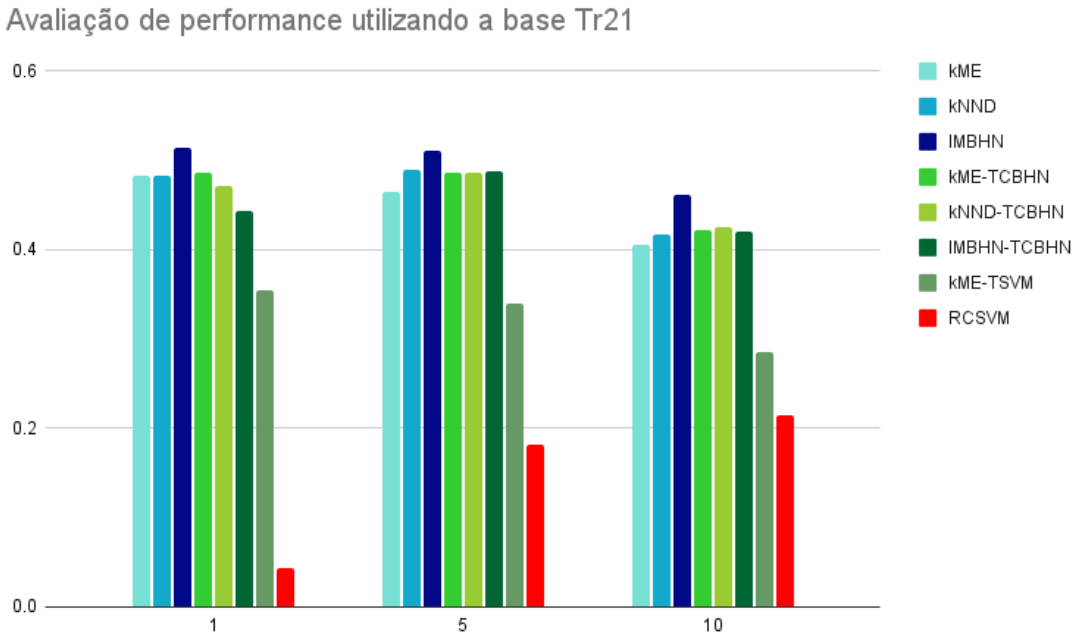
riores e kME-TCBHN tendo 1 pontuação superior. Já nos cenários contendo 10 exemplos rotulados a abordagem AMUC IMBHN foi predominante, apresentando melhor performance em 8 de 10 coleções.

Observa-se que em cenários de 1 e 5 exemplos rotulados, a utilização de documentos não rotulados, podem sim melhorar a performance de classificação através da utilização do *framework PUL*, com abordagem mais adequadas.

Na Tabela 12, estão apresentados os domínios das bases de textos junto a porcentagem de vezes em que os algoritmos propostos, utilizando-se do *framework PUL* superaram as abordagens AMUC. Vale ressaltar que todos os cenários foram considerados (1, 5 e 10 exemplos rotulados da classe de interesse).

Constata-se ao analisar as informações contidas na Tabela 12, que para maioria

Figura 11. Gráfico comparativo de resultados utilizando-se da base Tr21



Autoria própria.

Tabela 12. Melhores performances de abordagens propostas

Domínios	CI	NA	DM	PW	DT
Percentual de ganhos	66,67%	100%	58,33%	66,67%	0%

dos domínios utilizados, os algoritmos de PUL propostos obtiveram performances superiores, dando destaque para o domínio NA, onde as abordagens propostas apresentaram melhores performances em todas as coleções de textos e em todos os cenários experienciados. Porém, vale ressaltar que para coleções de domínio DT, as abordagens propostas não são recomendadas, pois não obtiveram performances superiores as abordagens AMUC.

5. Conclusões

Dada a grande quantidade de informações textuais produzidas diariamente, os algoritmos de classificação de textos se mostram cada vez mais necessários. Como já descrito na Seção 1, um dos grandes problemas a ser enfrentado em um espaço amostral real, é o não conhecimento de todas as classes, prejudicando assim os algoritmos atualmente mais utilizados (algoritmos de aprendizado multi-classe). Nota-se também que há necessidade de algoritmos mais eficientes que os de abordagem AMUC em cenários com poucos exemplos da classe de interesse. Além disso, mesmo com o uso de abordagens *PUL* para diminuir a quantidade de exemplos rotuladas requeridas pelos algoritmos de AMUC, observa-se que comumente são utilizados algoritmos que já demonstraram obter performances de classificação inferiores à outras abordagens.

Com isso, este trabalho de conclusão de curso buscou adotar algoritmos de AMUC com melhores performances em ambas as etapas do *framework PUL*, comparando-os dire-

tamente com propostas apresentadas na literatura. Buscou-se também verificar se de fato a utilização de documentos não rotulados pelos algoritmos de PUL influenciam positivamente na performance de classificação. Ao verificar os resultados coletados, observa-se que ao adotar algoritmos mais adequados nas etapas do *framework PUL*, obtém-se melhores performances em comparação com as as abordagens propostas pela literatura. Além disso, nota-se que em ambientes com apenas 1 e 5 exemplos rotulados, os exemplos não rotulados contribuem para aumentar a performance de classificação.

Como trabalhos futuros pretende-se implementar algoritmos de aprendizado profundo para a comparação com os resultados obtidos neste trabalho, além de verificar o impacto do uso de *word embeddings* para gerar as representações das coleções de textos.

Referências

Aggarwal, C. (2018). *Machine Learning for Text*. Springer International Publishing.

Akoglu, L., Tong, H., and Koutra, D. (2015). Graph based anomaly detection and description: A survey. *Data Min. Knowl. Discov.*, 29(3):626–688.

Alam, S., Sonbhadra, S. K., Agarwal, S., and Nagabhushan, P. N. (2020). One-class support vector classifiers: A survey. *Knowl. Based Syst.*, 196:105754.

Bekker, J. and Davis, J. (2020). Learning from positive and unlabeled data: a survey. *Machine Learning*, 109(4):719–760.

Chalapathy, R. and Chawla, S. (2019). Deep learning for anomaly detection: A survey. *CoRR*, abs/1901.03407.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15.

Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press.

de Souza, M. C., Nogueira, B. M., Rossi, R. G., Marcacini, R. M., dos Santos, B. N., and Rezende, S. O. (2021). A network-based positive and unlabeled learning approach for fake news detection. *Machine Learning*.

Elkan, C. and Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proc. ACM Int. Conf. Knowledge Discovery and Data Mining*, pages 213–220. ACM.

Faustini, P. and Covões, T. F. (2019). Fake news detection using one-class classification. *2019 8th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 592–597.

Ferreira, C. H. P., de Medeiros, D. M. R., and de França, F. O. (2018). Dcdistance: A supervised text document feature extraction based on class labels. *CoRR*, abs/1801.04554.

Gôlo, M., Marcacini, R., and Rossi, R. (2019). Uma extensa avaliação empírica de técnicas de pré-processamento e algoritmos de aprendizado supervisionado de uma classe para classificação de texto. In *Anais do XVI Encontro Nacional de Inteligência Artificial e Computacional*, pages 262–273. SBC.

Jaskie, K. and Spanias, A. (2019). Positive and unlabeled learning algorithms and applications: A survey. In *2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pages 1–8. IEEE.

Ji, M., Sun, Y., Danilevsky, M., Han, J., and Gao, J. (2010). Graph regularized transductive classification on heterogeneous information networks. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 570–586. Springer-Verlag.

Joachims, T. (1999). Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pages 200–209.

Júnior, D. D. S. and Rossi, R. G. (2018). Classificação automática de textos utilizando aprendizado supervisionado baseado em uma única classe. In *26º Simpósio Internacional de Iniciação Científica e Tecnológica da USP*, volume 1, pages 1–1.

Karaa, W. B. A. and Gribâa, N. (2013). Information retrieval with porter stemmer: a new version for english. In *Advances in computational science, engineering and information technology*, pages 243–254. Springer.

Kemmler, M., Rodner, E., Wacker, E.-S., and Denzler, J. (2013). One-class classification with gaussian processes. *Pattern Recognition*, 46(12):3507–3518.

Khan, S. S. and Madden, M. G. (2014). One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(3):345–374.

Li, X. and Liu, B. (2003). Learning to classify texts using positive and unlabeled data. In *IJCAI*, volume 3, pages 587–592.

Li, X.-L., Yu, P. S., Liu, B., and Ng, S.-K. (2009). Positive unlabeled learning for data stream classification. In *Proc. 2009 SIAM International Conference on Data Mining*, pages 259–270. SIAM.

Liu, B., Lee, W. S., Yu, P. S., and Li, X. (2002). Partially supervised classification of text documents. In *Proc. Int. Conf. Machine Learning*, volume 2, pages 387–394. Citeseer.

Ma, S. and Zhang, R. (2017). Pu-lp: A novel approach for positive and unlabeled learning by label propagation. In *Proc. Int. Conference on Multimedia & Expo Workshops*, pages 537–542. IEEE.

Pimentel, T., Monteiro, M., Viana, J., Veloso, A., and Ziviani, N. (2018). A generalized active learning approach for unsupervised anomaly detection. *stat*, 1050:23.

Rocchio, J. (1971). Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323.

Rossi, R. G. (2015). *Classificação automática de textos por meio de aprendizado de máquina baseado em redes*. PhD thesis, Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação. `http://sites.labic.icmc.usp.br/ragero/thesis/`.

Rossi, R. G., de Andrade Lopes, A., de Paulo Faleiros, T., and Rezende, S. O. (2014a). Inductive model generation for text classification using a bipartite heterogeneous network. *Journal of Computer Science and Technology*, 3(29):361–375.

Rossi, R. G., de Andrade Lopes, A., and Rezende, S. O. (2017). Using bipartite heterogeneous networks to speed up inductive semi-supervised learning and improve automatic text categorization. *Knowledge-Based Systems*, 132:94–118.

Rossi, R. G., Lopes, A. A., and Rezende, S. O. (2014b). A parameter-free label propagation algorithm using bipartite heterogeneous networks for text classification. In *Proc. Symposium on Applied Computing*, pages 79–84. ACM.

Rossi, R. G., Lopes, A. d. A., and Rezende, S. O. (2016). Optimization and label propagation in bipartite heterogeneous networks to improve transductive classification of texts. *Information Processing & Management*, 52(2):217–257.

Rossi, R. G., Marcacini, R. M., and Rezende, S. O. (2013). Benchmarking text collections for classification and clustering tasks. Technical Report 395, Institute of Mathematics and Computer Sciences, University of Sao Paulo.

Rossi, R. G., Rezende, S. O., and de Andrade Lopes, A. (2015). Term network approach for transductive classification. In *Int. Conf. Intelligent Text Processing and Computational Linguistics*, pages 497–515.

Sakai, T., du Plessis, M. C., Niu, G., and Sugiyama, M. (2017). Semi-supervised classification based on classification from positive and unlabeled data. In *Proc. Int. Conference on Machine Learning*, pages 2998–3006. JMLR. org.

Salton, G. (1989). Automatic text processing: The transformation, analysis, and retrieval of. *Reading: Addison-Wesley*, 169.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47.

Sonbhadra, S. K., Agarwal, S., and Nagabhushan, P. N. (2020). Target specific mining of covid-19 scholarly articles using one-class approach. *Chaos, Solitons, and Fractals*, 140:110155 – 110155.

Tan, P.-N., Steinbach, M., and Kumar, V. (2006). Introduction to data mining, pearson education. *Inc., New Delhi*.

Tax, D. M. and Duin, R. P. (2001). Combining one-class classifiers. In *Proc. Int. Workshop on Multiple Classifier Systems*, pages 299–308. Springer.

Tax, D. M. J. (2001). *One-class classification: Concept learning in the absence of counter-examples*. PhD thesis, Technische Universiteit Delft.

Ur-Rahman, N. and Harding, J. A. (2012). Textual data mining for industrial knowledge management and text classification: A business oriented approach. *Expert Systems with Applications*, 39(5):4729–4739.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Vapnik, V. and Vapnik, V. (1998). Statistical learning theory. 1st edwiley. *Boston, USA*.

Wang, H., Bah, M. J., and Hammad, M. (2019). Progress in outlier detection techniques: A survey. *IEEE Access*, 7:107964–108000.

Zhang, C., Ren, D., Liu, T., Yang, J., and Gong, C. (2019). Positive and unlabeled learning with label disambiguation. In *Proceedings of the Twenty-Eighth International*

Joint Conference on Artificial Intelligence, IJCAI-19, pages 4250–4256. International Joint Conferences on Artificial Intelligence Organization.

Zhang, D. and Lee, W. S. (2005). A simple probabilistic approach to learning from positive and unlabeled examples. In *Proc. 5th Annual UK Workshop on Computational Intelligence*, pages 83–87.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, volume 16, pages 321–328.

Zhu, X. and Goldberg, A. B. (2009). *Introduction to Semi-Supervised Learning*. Morgan and Claypool Publishers.