UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

**Construção Automática de Funções de Proximidade para
Redes de Termos usando Evolução Gramatical**

Felipe Provezano Coutinho

Dissertação de Mestrado do Programa de Pós-Graduação em Ciências
de Computação e Matemática Computacional (PPG-CMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Felipe Provezano Coutinho

Construção Automática de Funções de Proximidade para Redes de Termos usando Evolução Gramatical

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientadora: Profa. Dra. Solange Oliveira Rezende

USP – São Carlos
Outubro de 2019

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

C871c Coutinho, Felipe Provezano
Construção Automática de Funções de Proximidade
para Redes de Termos usando Evolução Gramatical /
Felipe Provezano Coutinho; orientadora Solange
Oliveira Rezende. -- São Carlos, 2019.
82 p.

Dissertação (Mestrado - Programa de Pós-Graduação
em Ciências de Computação e Matemática
Computacional) -- Instituto de Ciências Matemáticas
e de Computação, Universidade de São Paulo, 2019.

1. Evolução Gramatical. 2. Funções de Proximidade.
3. Rede de Termos. 4. Classificação de Documentos.
5. Mineração de Textos. I. Rezende, Solange
Oliveira, orient. II. Título.

Felipe Provezano Coutinho

Automatic Construction of Proximity Functions for Term Networks Using Grammatical Evolution

Master dissertation submitted to the Institute of
Mathematics and Computer Sciences – ICMC-USP,
in partial fulfillment of the requirements for the
degree of the Master Program in Computer Science
and Computational Mathematics. *FINAL VERSION*

Concentration Area: Computer Science and
Computational Mathematics

Advisor: Profa. Dra. Solange Oliveira Rezende

USP – São Carlos
October 2019

RESUMO

COUTINHO, F. P. **Construção Automática de Funções de Proximidade para Redes de Termos usando Evolução Gramatical**. 2019. 82 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

Com o crescente volume de dados textuais, a simples tarefa de rotulação de documentos torna-se onerosa. Algoritmos de aprendizado de máquina podem aprender com dados históricos e então serem capazes de realizar a tarefa de rotulação automática em novos documentos. Geralmente, os algoritmos supervisionados são utilizados. Nessa abordagem uma grande quantidade de exemplos é necessária para que o algoritmo seja capaz de identificar padrões e generalizar esse conhecimento com a rotulação de novos exemplos. Uma alternativa é o uso de algoritmos transdutivos, que necessitam apenas de poucos documentos rotulados para realizar a generalização. Geralmente, algoritmos transdutivos utilizam redes de documentos ou um modelo espaço vetorial para representação de documentos. A rede de documentos cresce a medida que novos documentos são adicionados, enquanto o modelo espaço vetorial sofre de problemas como a independência entre termos e alta esparsidade. Uma rede de termos é uma alternativa interessante aos modelos anteriores porque o número de termos tende a se estabilizar a medida que novos documentos são adicionados, a representação da relação entre termos é natural e é possível realizar a seleção de termos diminuir a rede e acelerar a classificação. O *Transductive Classification through Terms Networks* (TCTN) é estado-da-arte na classificação de documentos utilizando rede de termos, contudo um de seus hiperparâmetros é a função de proximidade que quantifica a proximidade entre termos e a escolha por uma função pode não ser trivial. Nessa dissertação, uma abordagem para construção automática de funções de proximidade para redes de termos é proposta, implementada e avaliada. A abordagem denominada *Grammatical Evolution for Automatically Design Interestingness Measures for Transductive Classification through Term Networks* (GE-TCTN) utiliza uma gramática livre de contexto para evolução de novas funções de proximidade através do algoritmo de otimização denominado Evolução Gramatical. Os resultados do GE-TCTN são comparados com os resultados do TCTN com objetivo de verificar se novas funções de proximidade construídas automaticamente pelo GE-TCTN são capazes de produzir resultados de classificação de documentos melhores do que os resultados produzidos utilizando-se funções de proximidade tradicionais da literatura em termos da medida de avaliação de classificação F^1 -ponderada. Além disso, o GE-TCTN é comparado aos algoritmos aprendizado transdutivo tradicionais e avaliados estatisticamente. Os resultados do GE-TCTN são competitivos com os algoritmos tradicionais e o GE-TCTN é capaz de produzir funções de proximidade que levam a resultados de classificação de documentos superiores ao TCTN.

Palavras-chave: Evolução Gramatical, Rede de Termos, Classificação de Documentos, Funções de Proximidade.

ABSTRACT

COUTINHO, F. P. **Automatic Construction of Proximity Functions for Term Networks Using Grammatical Evolution**. 2019. 82 p. Dissertação (Mestrado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2019.

With the increasing volume of textual data the simple task document labeling becomes costly. Machine learning algorithms can learn from historical data and then be able to perform the automatic labeling task on new documents. Generally, supervised algorithms are used for this. In this approach a large number of examples are required for the algorithm to be able to identify patterns and generalize this knowledge by labeling new examples. An alternative is the use of transductive algorithms that only require a few labeled documents to generalize. Usually, transductive algorithms use document networks or a vector space model for document representation. The document network grows as new documents are added and the vector space model suffers from problems such as term independence and high sparsity. A term network is an interesting alternative to previous models because the number of terms tends to stabilize as new documents are added, the representation of the relationship between terms is natural and it is possible to select terms to narrow the network and speed up classification. Transductive Classification through Terms Networks (TCTN) is state-of-the-art in document classification using term networks, however one of its hyperparameters is the proximity function that quantifies the proximity between terms and choosing a function may not be trivial. In this dissertation, an approach for automatic construction of proximity functions for term networks is proposed, implemented and evaluated. The approach called textit Grammatical Evolution for Automatically Design Proximity Functions for Transductive Classification through Term Networks (GE-TCTN) uses a context-free grammar to evolve new proximity functions through the optimization algorithm called Grammatical Evolution. The results produced by GE-TCTN are compared to TCTN's results to verify whether new proximity functions built automatically by GE-TCTN are capable of producing better document classification results than results produced by using traditional proximity functions of the literature. In addition, GE-TCTN is compared to traditional algorithms in the literature and statistically evaluated. GE-TCTN results are competitive with traditional algorithms and GE-TCTN is capable of producing proximity functions that lead to document classification results superior to TCTN in terms of F^1 -measure.

Keywords: Grammatical Evolution, Terms Network, Document Classification, Proximity Functions.

LISTA DE ILUSTRAÇÕES

Figura 1 – Ilustração de hiperplanos induzidos pelos algoritmos SVM e TSVM, respectivamente.	27
Figura 2 – Ilustração de uma rede de documentos.	27
Figura 3 – Exemplo de vizinhos mais próximos considerados pelos algoritmos k -vizinhos mais próximos mútuos (mKnn) e k -vizinhos mais próximos simétricos (sKnn).	29
Figura 4 – Exemplo de um indivíduo sendo decodificado de sua representação binária. O indivíduo tem genótipo 00001101 ₂ e fenótipo 13.	41
Figura 5 – Exemplo de indivíduo representado como uma árvore. Essa árvore representa a expressão simbólica $(* (+ X (/ Y 1.5)) (- Z 0.3))$	41
Figura 6 – Ilustração da seleção por giro de roleta. Neste exemplo, há cinco indivíduos, em que, cada um deles possui uma probabilidade de ser selecionado.	44
Figura 7 – Ilustração da seleção por torneio. Neste exemplo, há uma população de seis indivíduos e um torneio de $k = 3$ participantes é realizado. Primeiro, três indivíduos são selecionados para o torneio de maneira uniformemente aleatória. Em seguida, o indivíduo de maior aptidão é indicado como o vencedor do torneio.	45
Figura 8 – Cruzamento de subárvores. Dois progenitores trocam parte de sua estrutura e geram dois filhos. Os nós indicados para cruzamento estão marcados em cinza.	46
Figura 9 – Mutação de subárvores. O nó marcado com fundo cinza é selecionado para mutação de maneira aleatória. A subárvore que esse nó representa é substituída por outra gerada também de maneira aleatória.	47
Figura 10 – Exemplo de um conjunto de regras de uma gramática livre de contexto.	48
Figura 11 – Exemplo de um indivíduo codificado em genótipo.	49
Figura 12 – Processo de mapeamento genótipo-fenótipo do indivíduo representado na Figura 11.	49
Figura 13 – Processo de mapeamento genótipo-fenótipo do indivíduo representado na Figura 11 a partir de uma Árvore de Derivação.	50
Figura 14 – Visão geral da abordagem proposta.	55
Figura 15 – Regras de produção da gramática proposta.	57
Figura 16 – Exemplo de produção da função de proximidade Piatetsky-Shapiro por meio da gramática proposta neste trabalho.	58

LISTA DE TABELAS

Tabela 1	– Forma geral de uma matriz documento-termo.	24
Tabela 2	– Matriz de confusão para a classe c_i	35
Tabela 3	– Matriz de Contingência	36
Tabela 4	– Medidas de Interesse consideradas no TCTN	36
Tabela 5	– Resumo das características das coleções de documentos utilizadas nos experimentos: $ \mathcal{D} $ - número de documentos; $ \mathcal{T} $ - número de termos; $ \mathcal{C} $ - número de classes; $\sigma(\mathcal{C})$ - desvio padrão considerando a porcentagem das classes; $\max(\mathcal{C})$ - porcentagem da classe majoritária.	63
Tabela 6	– Resultados dos experimentos realizados. A primeira coluna contém o nome de cada coleção de documentos. A segunda, os resultados do Grid-TCTN, a terceira coluna contém os resultados do EM, a quarta coluna contém os resultados do algoritmo LLGC, a quinta coluna é o melhor indivíduo encontrado em 10 execuções do GE-TCTN e a última coluna contém a média e desvio padrão (em parênteses) de 10 execuções do GE-TCTN.	66
Tabela 7	– <i>Ranking</i> médio dos algoritmos (Teste de Friedman)	67
Tabela 8	– p -valores não ajustados e ajustados segundo o procedimento de Li (Teste de Friedman)	67
Tabela 9	– Exemplos de hiperparâmetros e funções de proximidade construídas pelo GE-TCTN.	67
Tabela 10	– Desvios-padrão dos resultados de classificação do Grid-TCTN, EM, LLGC e GE-TCTN considerando avaliações realizadas com a medida F^1 -ponderada.	81

SUMÁRIO

1	INTRODUÇÃO	15
1.1	Contextualização e Motivação	15
1.2	Hipótese e Objetivos	18
1.3	Principais Contribuições	18
1.4	Organização do Texto	19
2	CLASSIFICAÇÃO SEMISSUPERVISIONADA TRANSDUTIVA BASEADA EM REDES	21
2.1	Classificação de Documentos e Representação de Dados Textuais	22
2.1.1	<i>Classificação Semissupervisionada Transdutiva de Documentos utilizando Redes</i>	28
2.2	<i>Transductive Classification through Terms Networks (TCTN)</i>	33
2.3	Considerações Finais	37
3	GRAMÁTICA EVOLUTIVA	39
3.1	Algoritmos Evolutivos	39
3.1.1	<i>Representação</i>	40
3.1.2	<i>Inicialização</i>	41
3.1.3	<i>Função de Aptidão</i>	42
3.1.4	<i>População</i>	42
3.1.5	<i>Seleção</i>	43
3.1.5.1	<i>Seleção por Giro de Roleta</i>	43
3.1.5.2	<i>Seleção por Ranking</i>	44
3.1.5.3	<i>Seleção por Torneio</i>	44
3.1.6	<i>Operadores Genéticos</i>	45
3.1.6.1	<i>Cruzamento</i>	45
3.1.6.2	<i>Mutação</i>	47
3.1.7	<i>Condição ou Critério de Parada</i>	47
3.2	Evolução Gramatical	48
3.3	Considerações Finais	50
4	ABORDAGEM PROPOSTA: O <i>FRAMEWORK</i> GE-TCTN	53
4.1	GE-TCTN: Visão Geral	54
4.2	Definição da Gramática - 1ª Etapa	55

4.3	População Inicial e Construção da Rede - 2ª Etapa	58
4.4	Classificação Transdutiva Por Meio de Redes de Termos - 3ª Etapa	59
4.5	Avaliação dos Resultados - 4ª Etapa	59
4.6	Processo Evolutivo - 5ª Etapa	60
4.7	Considerações Finais	60
5	AVALIAÇÃO EXPERIMENTAL	61
5.1	Configuração Experimental	61
5.2	Resultados	65
5.3	Considerações Finais	68
6	CONCLUSÕES E TRABALHOS FUTUROS	69
	REFERÊNCIAS	73
APÊNDICE A	TABELAS DE RESULTADOS	81

INTRODUÇÃO

1.1 Contextualização e Motivação

Dados históricos são frequentemente utilizados com a finalidade de descoberta de informações úteis que podem auxiliar na tomada de decisões. Enquanto diversas técnicas para a análise de dados estão bem estabelecidas e ainda são muito utilizadas na área da Estatística, novos desafios surgem com o aumento massivo do volume de dados disponível. Além disso, grande parte dos dados estão em formato textual. Dados textuais são encontrados em diversas fontes. Desde notícias, e-mails, *blogs*, artigos, bate-papos, redes sociais, fóruns online até documentos governamentais. A manipulação, processamento e extração de informações de um grande volume de dados textuais são atividades manualmente custosas. Portanto, uma alternativa é a automação desse processo por meio do uso de computadores.

Algoritmos de extração de padrões a partir de conjuntos de dados (incluindo conjunto de dados textuais) são frequentemente denominados Algoritmos de Aprendizado de Máquina (AAM). AAM são cada vez mais utilizados na descoberta de informações úteis ou extração de padrões de dados textuais ([AGGARWAL, 2018](#)). A extração automática de padrões a partir de dados textuais é conhecida como Mineração de Textos. A tarefa de classificação de documentos é uma das mais conhecidas na área de Mineração de Textos ([AGGARWAL, 2018](#)). Essa tarefa consiste em rotular documentos dado um subconjunto de possíveis rótulos. Geralmente, um conjunto de documentos rotulados por um especialista é fornecido a um algoritmo que, “aprende” a partir desses dados a rotular novos documentos de maneira automática. Essa tarefa é particularmente útil na classificação de e-mails ([KIRITCHENKO; MATWIN, 2001](#)), previsão do movimento de ações ([NGUYEN; SHIRAI; VELCIN, 2015](#)), previsão de crimes ([GERBER, 2014](#)), previsão de eleições políticas ([PRASETYO; HAUFF, 2015](#)), recomendação de filmes ([RAJARAJESWARI et al., 2019](#)), entre outras.

Existem diversos tipos de AAM, porém algoritmos de aprendizado indutivo supervi-

sionado são os mais utilizados realizar a da classificação automática de documentos (ROSSI; REZENDE; LOPES, 2015). Contudo, algoritmos que seguem esse paradigma frequentemente necessitam de um grande volume de documentos rotulados para que seja possível a indução de um classificador útil (ABU-MOSTAFA; MAGDON-ISMAIL; LIN, 2012). Uma alternativa é a utilização de algoritmos de aprendizado semissupervisionados. Esses algoritmos são particularmente interessantes pelo fato de utilizarem informações de documentos rotulados e não-rotulados para classificação. Ainda, algoritmos transdutivos podem ser enriquecidos com a informação de relações entre objetos através do uso de grafos (ou redes). A classificação de documentos pode ser realizada de maneira semissupervisionada e transdutiva sem a utilização de grafos, contudo os algoritmos de aprendizado transdutivo baseados em grafos apresentaram uma performance superior a algoritmos não baseados em grafos (SUBRAMANYA; TALUKDAR, 2014) em termos de acurácia.

Algoritmos de aprendizado transdutivo baseado em grafos já são utilizados na classificação de dados textuais. Alguns exemplos são: *Semi-supervised Learning based on Gaussian Random Fields* (ZHU; GHAMRANI; LAFFERTY, 2003), *Learning with Local and Global Consistency* (ZHOU *et al.*, 2004) e *Label Propagation using Bipartite Heterogeneous Networks* (ROSSI; LOPES; REZENDE, 2014). As redes utilizadas em algoritmos de classificação transdutiva de dados textuais são compostas por documentos, termos ou uma combinação de documentos e termos (ZHU; GHAMRANI; LAFFERTY, 2003; ZHOU *et al.*, 2004; ROSSI; LOPES; REZENDE, 2014). Enquanto uma rede de documentos de uma coleção, parece ser a ideia mais natural durante a construção de uma estrutura que auxilie na classificação de dados textuais, em Rossi, Rezende e Lopes (2015) o *framework Transductive Classification through Terms Networks* (TCTN) é proposto com objetivo da utilização de redes de termos para classificação de documentos. Além disso, TCTN é pioneiro e foi capaz de produzir resultados competitivos na classificação transdutiva de documentos.

Uma das etapas mais importantes e sensíveis dos algoritmos de classificação transdutiva baseada em redes consiste na construção da rede (BERTON; LOPES, 2015). Nas redes de termos geradas pelo TCTN, os pesos das relações entre vértices são quantificados de acordo com uma função de proximidade definida a priori. Contudo, a escolha por uma função de proximidade deve ser realizada com cuidado, pois os resultados da classificação podem ser influenciados (ROSSI; REZENDE; LOPES, 2015). Além disso, uma vez que a quantidade de termos tende a se estabilizar em um número fixo, a quantidade de objetos na rede de termos tem tamanho limitado. Enquanto em outros tipos de redes que consideram documentos como objetos, o número de objetos cresce conforme o número de documentos aumenta. Outro detalhe apresentado em Rossi, Rezende e Lopes (2015), é a possibilidade de diminuir o número de objetos na rede de termos (sem efetuar a remoção de documentos) e manter a performance de classificação, o que não é possível em redes de documentos. É importante notar também que funções de proximidade da literatura não generalizam bem (TAN; KUMAR; SRIVASTAVA, 2002). Isso significa que funções de proximidade de propósito geral podem não capturar padrões diante da variação de

contexto. Além disso, construí-las manualmente para contextos diferentes, pode ser uma tarefa bastante custosa (BELLET; HABRARD; SEBBAN, 2015). Portanto, a construção automática de funções de proximidade apropriadas para cada coleção de documentos é desejada.

A necessidade de boas funções de distância entre objetos em algoritmos de aprendizado de máquina é frequente. Algoritmos como k -médiadas (OLIVEIRA *et al.*, 2017) e k -vizinhos mais próximos (COVER; HART, 1967) são apenas alguns exemplos de algoritmos nos quais esse parâmetro é crucial na obtenção de bons resultados.

O *framework* TCTN (ROSSI; REZENDE; LOPES, 2015) requer uma função de proximidade (GENG; HAMILTON, 2006; TAN; KUMAR; SRIVASTAVA, 2002) para determinar a relação entre dois termos. Em Rossi, Rezende e Lopes (2015), cinco funções de proximidade com características diferentes, foram utilizadas para realização de experimentos em coleções de documentos para validação dos resultados do TCTN. Contudo, a noção semântica de proximidade depende do contexto (BELLET; HABRARD; SEBBAN, 2015). Além disso, a decisão por uma função de proximidade existente pode não ser trivial (TAN; KUMAR; SRIVASTAVA, 2002). Uma solução é a delegação dessa decisão a uma máquina inteligente, capaz de construir funções de proximidade para cada contexto (coleção de documentos).

A preocupação com boas funções de distância para algoritmos de aprendizado não é uma novidade (XING *et al.*, 2003), em diversos contextos, aprender automaticamente uma função de distância ou proximidade que represente a noção de diferença e/ou semelhança entre objetos de uma base de dados de interesse ainda representa um desafio (BELLET; HABRARD; SEBBAN, 2015). Grande parte dos algoritmos de aprendizado de métricas consistem no aprendizado da matriz de Mahalanobis (BELLET; HABRARD; SEBBAN, 2015) e possuem complexidade de tempo quadrática em relação ao número de atributos. Outra parte dos algoritmos são baseados em *Kernel* e possuem complexidade quadrática em relação ao número de exemplos. Portanto, quando o número de atributos ou o número de exemplos é alto, grande parte das soluções disponíveis tornam-se impraticáveis. Além disso, grande parte das técnicas de aprendizado de métricas trabalham apenas com vetor de características no espaço R^n . Como técnicas para construção automática de funções de proximidade para construção de redes de termos não foram encontradas na literatura, é proposto um algoritmo capaz de realizar a construção automática de funções de proximidade para construção de redes de termos.

Um método interessante e apropriado para a construção automática de funções é denominado Evolução Gramatical (EG) (RYAN; O'NEILL; COLLINS, 1998; BASGALUPP; BARROS; BARABASZ, 2014; BARROS *et al.*, 2013; MIQUILINI *et al.*, 2017). A EG é uma técnica de Programação Genética (PG) (KOZA, 1997) que utiliza uma gramática livre de contexto no mapeamento genótipo-fenótipo (O'NEILL; RYAN, 2001). Segundo McKay *et al.* (2010), a EG tornou-se uma das técnicas mais populares dentro da PG. Além disso, a técnica trouxe consigo inovações como a utilização de cromossomos de tamanhos variáveis e novas operações de mapeamento e tradução fenótipo-genótipo. Uma das vantagens da EG é a facilidade na inserção

de características do problema por meio da gramática para auxílio na busca por soluções viáveis. A hipótese que foi utilizada como guia para o desenvolvimento desse trabalho é descrita na seção a seguir.

1.2 Hipótese e Objetivos

É possível realizar a construção automática de funções de proximidade para o *framework* TCTN que permitem a geração de redes de termos que levam à uma melhor performance na classificação de documentos quando comparado à utilização de funções de proximidade selecionadas manualmente.

Como mencionado anteriormente, EGs já foram utilizados com a finalidade de construir automaticamente novas funções. Contudo, não foi encontrado nenhum trabalho com objetivo de construir funções de proximidade para quantificar proximidade entre termos. Construir funções de proximidade manualmente é uma tarefa custosa e não intuitiva em diversos contextos (BELLET; HABRARD; SEBBAN, 2015). Portanto, neste trabalho é proposta uma abordagem baseada em EG para a construção automática de funções capazes de quantificar a proximidade de termos durante a etapa de construção do grafo no *framework* TCTN que é utilizado para a classificação de documentos. Além disso, é importante que as redes construídas utilizando essas funções conduzam o TCTN a resultados de classificação de documentos melhores do que os obtidos utilizando funções da literatura.

Para validar a hipótese, foram definidos os objetivos apresentados a seguir.

1. Propor, desenvolver e avaliar um algoritmo baseado em Evolução Gramatical com o objetivo de construir automaticamente novas funções de proximidade para o *framework Transductive Classification Through Term Networks*;
2. Desenvolver e disponibilizar ferramentas computacionais que viabilizem a aplicação da abordagem proposta;
3. Aplicar a abordagem proposta em problemas reais.

1.3 Principais Contribuições

A principal contribuição deste trabalho é o *framework Grammatical Evolution for automatic construction of proximity functions for Transductive Classification through Term Networks* (GE-TCTN). Esse algoritmo permite a realização da classificação transdutiva de documentos através do TCTN sem a necessidade de especificar previamente uma função de proximidade. GE-TCTN aprende uma função de proximidade utilizando as coleções de documentos e realiza a classificação em seguida. A seguir estão listadas de maneira resumida as principais contribuições desse mestrado.

1. Proposta, desenvolvimento e avaliação de um *framework* denominado GE-TCTN para construção automática de funções de proximidade capazes de quantificar a proximidade entre termos para a construção de redes de termos utilizadas no *framework* TCTN;
2. Aplicação da abordagem proposta em 37 coleções de documentos reais;
3. Divulgação de uma ferramenta que permite a aplicação da abordagem proposta em novas coleções de documentos;

O algoritmo GE-TCTN facilita a realização da classificação de documentos para não especialistas, pois não exige do usuário o fornecimento de uma função de proximidade. Além disso, o algoritmo pode ser utilizado com a finalidade de melhorar a classificação de documentos ou apenas, verificar quais operações e operadores são frequentemente escolhidos pelo algoritmo, a fim de, facilitar a construção manual ou escolha por uma função de proximidade.

1.4 Organização do Texto

O restante dessa dissertação consiste de cinco capítulos. No Capítulo 2, a classificação transdutiva baseada em redes é discutida. Além disso, detalhes sobre o *framework* TCTN são apresentados. O Capítulo 3 contém detalhes sobre algoritmos evolutivos e sobre a gramática evolutiva. No Capítulo 4, a abordagem proposta denominada GE-TCTN é apresentada. O Capítulo 5 contém informações sobre configuração experimental e resultados obtidos. Por fim, no Capítulo 6, as conclusões, contribuições, publicações e trabalhos futuros são apresentados.

CLASSIFICAÇÃO SEMISSUPERVISIONADA TRANSDUTIVA BASEADA EM REDES

A classificação de documentos é um dos desafios da Mineração de Textos ([AGGARWAL, 2018](#)). Geralmente, a classificação é realizada por meio de técnicas de aprendizado de máquina supervisionado indutivo. O aprendizado supervisionado indutivo, embora relevante, é restrito a cenários em que um grande número de documentos rotulados deve ser fornecido previamente ao algoritmo de aprendizado ([ABU-MOSTAFA; MAGDON-ISMAIL; LIN, 2012](#)). Além disso, rotulação manual de documentos é custosa e, por vezes, não trivial.

Existem outras maneiras de se realizar o aprendizado de máquina. Na literatura, é possível encontrar algoritmos classificados como: (1) supervisionados, (2) não-supervisionados e (3) semissupervisionados. Algoritmos de aprendizado supervisionados utilizam dados históricos rotulados para treinar um classificador, o qual, espera-se ser capaz de classificar novos exemplos. Por exemplo, um conjunto de e-mails rotulados como SPAM ou NÃO-SPAM, pode ser utilizado para treinar um classificador capaz de rotular novos e-mails de maneira automática. Diferentemente, os algoritmos não supervisionados são utilizados para a descoberta de padrões na estrutura dos dados. Porém, nesse caso apenas dados não rotulados são utilizados. Certas estruturas naturalmente formadas pelos dados, podem ser encontradas pelos algoritmos não supervisionados e indicar a presença de um ou mais padrões. Os algoritmos semissupervisionados destacam-se por utilizar informações de documentos rotulados e não rotulados para realizar a classificação, rotulação automática ou tarefa de predição de classes. Essa classe de algoritmos, ainda pode ser dividida entre algoritmos baseados em redes e de aprendizado: (1) indutivo ou (2) transdutivo ([SUBRAMANYA; TALUKDAR, 2014](#)).

O objetivo do aprendizado semissupervisionado indutivo é encontrar uma função de classificação de novos dados a partir de dados históricos rotulados e não rotulados. Em contrapartida, no aprendizado transdutivo, dado um pequeno conjunto de dados rotulados e um conjunto maior de dados não rotulados, o objetivo é classificar os dados não rotulados. Esses dados, podem ser

encontrados em diversos formatos. Nesta dissertação, destaca-se o formato de dados textuais ou documentos.

Antes da aplicação de qualquer algoritmo de aprendizado, os dados geralmente devem ser pré-processados para satisfazer a estrutura utilizada pelas técnicas de aprendizado. Normalmente, documentos são representados por um modelo espaço-vetorial que consideram que termos e documentos são independentes. Uma outra forma de representação consiste na utilização de redes que facilitam a representação de relações entre documentos. Enquanto algoritmos que utilizam dados em modelo espaço-vetorial ainda são muito utilizados, algoritmos de aprendizado baseados em redes são úteis e podem produzir resultados melhores na tarefa de classificação (SUBRAMANYA; TALUKDAR, 2014; ROSSI, 2015; BERTON *et al.*, 2017).

Na literatura, existem diversos algoritmos de aprendizado semissupervisionado transdutivo que são utilizados para a classificação de documentos como: *Self-Training* (YAROWSKY, 1995), *Co-Training* (BLUM; MITCHELL, 1998), *Transductive Support Vector Machines* (TSVM) (JOACHIMS, 1999), *Expectation Maximization* (EM) (NIGAM *et al.*, 2000), *Gaussian Field and Harmonic Functions* (GFHF) (ZHU; GHAHRAMANI; LAFFERTY, 2003) e *LLGC Learning with Local and Global Consistency* (ZHOU *et al.*, 2004). Os algoritmos GFHF e LLGC consideram textos representados em redes, enquanto os demais algoritmos consideram textos representados em um modelo espaço-vetorial, em que, cada linha representa um documento e cada coluna um termo e os elementos da matriz. Além destes, o *framework* TCTN (ROSSI; REZENDE; LOPES, 2015) é estado-da-arte na classificação semissupervisionada transdutiva de documentos baseada em redes de termos. Neste capítulo, algoritmos de aprendizado semissupervisionado transdutivo baseados em redes, suas propriedades e detalhes do *framework* TCTN, bem como suas limitações são discutidas.

2.1 Classificação de Documentos e Representação de Dados Textuais

A classificação ou rotulação automática de documentos refere-se a uma tarefa preditiva. Na tarefa de predição, dado um novo documento a máquina deve classificá-lo como pertencente a um dos possíveis rótulos pré-especificados ao algoritmo (AGGARWAL, 2018). Por exemplo, ao receber um novo e-mail, o algoritmo deve classificá-lo ou prever se o e-mail é spam ou não é spam. A predição do rótulo de documentos pode ser realizada por algoritmos de aprendizado que usam dados históricos para realizar a classificação.

A classificação de dados textuais é geralmente realizada por algoritmos de aprendizado indutivo supervisionado. Nessa configuração, uma coleção de documentos rotulados $D^L = \{d_1, d_2, \dots, d_N\}$ é utilizada para indução de um classificador ou função F que seja capaz de rotular cada documento com seu respectivo rótulo, indicado em $C = \{C_{d_1}, C_{d_2}, \dots, C_{d_N}\}$. Portanto, $F : D^L \rightarrow C$. O processo de aproximação da função F é denominado processo de aprendizado.

Algoritmos tradicionais de aprendizado indutivo supervisionado que geralmente são utilizados na classificação de documentos incluem: *k-Nearest Neighbors* (KNN) (COVER; HART, 1967), *Support Vector Machines* (SVM) (JOACHIMS, 1999) e *Naive Bayes* (NB) (HAND; YU, 2001). Cada uma dessas técnicas segue um paradigma e partem de uma hipótese para realizar a classificação. Por exemplo, o algoritmo *k*-NN parte da hipótese que um documento pode ser classificado de acordo com o rótulo da maioria dos seus *k* vizinhos mais próximos. O SVM assume que os dados podem ser separados linearmente por meio de um hiperplano de separação. Já o Naive Bayes é um algoritmo de classificação probabilística que assume independência entre termos de uma coleção de textos. Contudo, as hipóteses assumidas pelos algoritmos supracitados nem sempre são verdadeiras (ZHU; GOLDBERG, 2009). Esse fato, pode prejudicar a performance do algoritmo na classificação de documentos.

A classificação de documentos também pode ser realizada utilizando documentos rotulados e não rotulados. Esse tipo de classificação é denominada classificação semissupervisionada. Formalmente, dada uma coleção de documentos $D = \{D^L, D^U\}$, pode ser dividida em duas partes: (1) coleção de documentos rotulados $D^L = \{d_1, d_2, \dots, d_L\}$ e (2) coleção de documentos não rotulados $D^U = \{d_{L+1}, d_{L+2}, \dots, d_N\}$. A classificação semissupervisionada indutiva utiliza D^L e D^U para aprender uma função de classificação de documentos. Por outro lado, na classificação transdutiva o objetivo é inferir rótulos para os documentos não rotulados (D^U) a partir das informações dos rótulos dos documentos rotulados (D^L) e da estrutura dos dados.

Algoritmos de aprendizado semissupervisionado geralmente utilizam documentos pré-processados e representados por meio de uma matriz D denominada *bag-of-words* (BoW) (ROSSI; REZENDE; LOPES, 2015). Isso ocorre porque esses algoritmos normalmente limitam seu funcionamento a uma estrutura de dados específica. Na estrutura BoW, N documentos são descritos em linhas, T termos em colunas e os elementos da matriz representam o peso do termo em relação ao documento. Além disso, essa matriz D é associada a um vetor C de rótulos de classe para cada um dos documentos. A Tabela 1 é uma generalização de uma matriz documento-termo. As linhas são documentos ($d_1, d_2, \dots, d_N$) e as colunas são termos ($t_1, t_2, \dots, t_M$). O peso de um termo t_i em um documento d_j é denotado por w_{d_j, t_i} . Além disso, cada um dos documentos é associado a um valor de classe. Portanto, o documento d_i é associado a um valor de classe c_{d_i} .

Algoritmos de aprendizado transdutivo baseados no modelo espaço vetorial (representado pela BoW), além de possuírem hipóteses intrínsecas ao método de aprendizado e classificação, assumem independência entre objetos (documentos ou termos). Algoritmos tradicionais de aprendizado transdutivo baseados no modelo espaço vetorial incluem: *Self-Training* (YAROWSKY, 1995), *Co-Training* (BLUM; MITCHELL, 1998), *Transductive Support Vector Machines* (TSVM) (JOACHIMS, 1999) e *Expectation Maximization* (EM) (NIGAM *et al.*, 2000). Nas próximas seções, os algoritmos listados são detalhados considerando-se que uma coleção de documentos $D = \{D^L, D^U\}$ é utilizada na classificação de documentos.

Tabela 1 – Forma geral de uma matriz documento-termo.

	t_1	t_2	t_3	$\dots$	t_{M-2}	t_{M-1}	t_M	Classe
d_1	w_{d_1,t_1}	w_{d_1,t_2}		$\dots$	$w_{d_1,t_{M-2}}$	$w_{d_1,t_{M-1}}$	w_{d_1,t_M}	c_{d_1}
d_2	w_{d_2,t_1}	w_{d_2,t_2}		$\dots$	$w_{d_2,t_{M-2}}$	$w_{d_2,t_{M-1}}$	w_{d_2,t_M}	c_{d_2}
d_3	w_{d_3,t_1}	w_{d_3,t_2}		$\dots$	$w_{d_3,t_{M-2}}$	$w_{d_3,t_{M-1}}$	w_{d_3,t_M}	c_{d_3}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
d_{N-2}	w_{d_{N-2},t_1}	w_{d_{N-2},t_2}		$\dots$	$w_{d_{N-2},t_{M-2}}$	$w_{d_{N-2},t_{M-1}}$	w_{d_{N-2},t_M}	$c_{d_{N-2}}$
d_{N-1}	w_{d_{N-1},t_1}	w_{d_{N-1},t_2}		$\dots$	$w_{d_{N-1},t_{M-2}}$	$w_{d_{N-1},t_{M-1}}$	w_{d_{N-1},t_M}	$c_{d_{N-1}}$
d_N	w_{d_N,t_1}	w_{d_N,t_2}		$\dots$	$w_{d_N,t_{M-2}}$	$w_{d_N,t_{M-1}}$	w_{d_N,t_M}	c_{d_N}

Fonte – ROSSI (2015).

Self-Training

Self-Training foi um dos primeiros algoritmos de aprendizado semissupervisionado transdutivo proposto. Esse algoritmo foi originalmente utilizado para desambiguação do significado de palavras em coleções textuais (YAROWSKY, 1995). No *Self-Training*, um classificador é induzido utilizando-se apenas uma pequena quantidade de exemplos rotulados (D^L). Em seguida, o classificador é utilizado para rotulação automática de exemplos não rotulados (D^U). Logo, os documentos não rotulados D^U (agora rotulados) que foram classificados com maior confiança são adicionados ao conjunto de documentos rotulados D^L . O classificador é retreinado utilizando-se D^L e o processo se repete.

No *Self-Training*, assume-se que as classificações de maior confiança estão corretas, contudo essa hipótese nem sempre é verdadeira (ZHU; GOLDBERG, 2009; ROSSI, 2015). Além disso, a complexidade de tempo do algoritmo pode ser um problema visto que o modelo de classificação deve ser induzido até que todos documentos não rotulados sejam classificados acima de um limiar que indica a confiança da classificação.

Co-Training

Co-Training é um algoritmo transdutivo que também pode ser utilizado na classificação automática de documentos. Essa técnica já foi utilizada em classificação de e-mails (KIRITCHENKO; MATWIN, 2001), classificação de sentimentos (WAN, 2009), *clustering* (KUMAR; DAUMÉ, 2011), adaptação de domínio (CHEN; WEINBERGER; BLITZER, 2011) e em jogos (LAMPLE; CHAPLOT, 2017). O algoritmo assume duas visões de uma coleção de documentos D , de modo que, $D = D_1 \times D_2$. Assim, D_1^L e D_2^L são conjuntos de documentos rotulados nas visões 1 e 2, respectivamente. Da mesma forma, D_1^U e D_2^U são conjuntos de documentos não rotulados nas visões 1 e 2 de D . Um modelo de classificação é treinado para cada uma das duas

visões considerando os documentos rotulados. Em seguida, os documentos não rotulados são classificados e os documentos que foram classificados com maior confiança são inseridos no conjunto de documentos rotulados da visão correspondente. Se há divergência entre rótulos nos documentos classificados, a classificação de maior confiança é retornada.

A maior diferença do *Co-Training* para o *Self-Training* está na hipótese da possibilidade de representar um conjunto de dados considerando duas visões independentes. Considerar duas visões pode trazer confiança aos resultados de classificação, contudo é uma hipótese difícil de ser satisfeita na prática. Além disso, não é difícil perceber que o *Co-Training* está sujeito a uma complexidade de tempo alta, pois deve induzir modelos de classificação e ordenar documentos pela confiança repetidas vezes (ROSSI, 2015).

Expectation Maximization

Expectation Maximization é um algoritmo iterativo que consiste de duas etapas principais: (1) *Expectation* e (2) *Maximization* (DEMPSTER; LAIRD; RUBIN, 1977). Uma abordagem transdutiva do EM combinada com *Multinomial Naive Bayes* (MNB) (HAND; YU, 2001) é apresentada em Nigam *et al.* (2000) e utilizada para a classificação de documentos. As premissas assumidas por essa abordagem são a de que os dados provêm de um modelo generativo e que existe independência entre termos. As premissas assumidas pelo EM nem sempre são verdadeiras, o que pode degradar a performance de classificação.

A abordagem de classificação de documentos adotada pelo MNB é baseada na ideia de cálculo de probabilidade posterior pela formulação de Naive Bayes com independência condicional entre termos. Portanto, suponha que $C = \{c_1, \dots, c_k\}$ é um conjunto de possíveis valores de classe ou rótulos para cada documento da coleção e $\mathbf{d}$ é um documento de $D = \{\mathbf{d}_1, \dots, \mathbf{d}_n\}$ composto por um subconjunto de $T = \{t_1, \dots, t_m\}$ termos. A probabilidade de um documento $\mathbf{d}$ pertencer a uma classe C_k é denotada pela Equação 2.1.

$$p(C_k|\mathbf{d}) = \frac{p(C_k)p(\mathbf{d}|C_k)}{p(\mathbf{d})}. \quad (2.1)$$

Ao assumir independência condicional entre termos, $p(\mathbf{d}|C_k)$ pode ser calculada como especificado na Equação 2.2.

$$p(\mathbf{d}|C_k) = \prod_{i=1}^m p(d_i|C_k) \quad (2.2)$$

Geralmente, o termo $p(\mathbf{d})$ da Equação 2.1 é omitido por se tratar de uma constante independente de C_k e a classe de um documento $\mathbf{d}$ com maior probabilidade é escolhida como a classe definitiva. Essa regra é conhecida como estimação máxima a posteriori (MAP). Além disso, é comum a aplicação de alguma técnica como a suavização de Laplace ou a suavização de Lidstone para correção de campos em que a frequência de ocorrência é zero para evitar

probabilidade nula (MANNING; RAGHAVAN; SCHÜTZE, 2010). Essa correção envolve a adição de um pseudo-valor aos campos das tabelas de frequência de co-ocorrência de termos e classes.

O algoritmo EM recebe uma coleção de documentos rotulados (D^L) e não rotulados (D^U) e então MNB é utilizado para o aprendizado de um classificador F . Esse classificador F é utilizado para a classificação probabilística de todos documentos não rotulados (D^U). Em seguida, os dados rotulados (D^L) e os dados recentemente rotulados (D^U) são utilizados para retreinar F , até que os parâmetros de F se estabilizem.

Transductive Support Vector Machines

Transductive Support Vector Machines (TSVM) é uma abordagem baseada no algoritmo de aprendizado supervisionado *Support Vector Machines* (SVM) (JOACHIMS, 1999). Esse algoritmo parte da hipótese de que os documentos não rotulados e de classes diferentes estão bem separados por uma margem grande. Basicamente o objetivo do SVM é encontrar um hiperplano de separação de classes que estejam o mais distante possível de documentos de classes diferentes. Este é um problema de otimização com restrições que pode ser resolvido usando Multiplicadores de Lagrange. O TSVM utiliza documentos rotulados e não rotulados na indução do hiperplano de separação e, diferentemente do SVM, se trata de um problema não convexo e de difícil solução exata (NP-difícil). A abordagem adotada pelo TSVM consiste de três passos principais: (1) Enumeração de todas as $2^{|D^U|}$ possibilidades de classificação de D^U ; (2) Construção de um classificador SVM considerando D^L e cada possibilidade de rotulação dos documentos de D^U e; (3) Escolha do classificador SVM que produz a maior margem entre documento e hiperplano. Como o problema de busca por um hiperplano de margem máxima considerando documentos rotulados e não rotulados é NP-difícil, diversas heurísticas são utilizadas na prática, como *Self-Labeling* (JOACHIMS, 1999), *Gradient Descent* (CHAPELLE; ZIEN, 2005) e CCCP-S³VM (COLLOBERT *et al.*, 2006). A Figura 1 é uma ilustração de hiperplanos induzidos pelos algoritmos SVM e TSVM, respectivamente para separar duas classes. Os círculos preenchidos representam documentos rotulados e os círculos vazios representam documentos não rotulados. O SVM é capaz de induzir um hiperplano apenas com uso de documentos rotulados (Figura 2(a)). O TSVM, por sua vez, utiliza documentos rotulados e não rotulados para induzir um hiperplano de separação de documentos de classes diferentes (Figura 2(b)).

O TSVM consegue lidar bem com exemplos de muitas dimensões, o que pode ser o motivo de estar entre os algoritmos mais utilizados na classificação de documentos. Contudo, a premissa da abordagem nem sempre é verdadeira, pois documentos de classes diferentes nem sempre estão bem separados. Não satisfazer tal premissa pode levar a resultados de classificação insatisfatórios.

Figura 1 – Ilustração de hiperplanos induzidos pelos algoritmos SVM e TSVM, respectivamente.

Fonte – ROSSI (2015).

Representação de Objetos utilizando Redes

Embora o modelo espaço vetorial seja intuitivo e se adapte a diversos algoritmos de classificação existentes na literatura, ele falha na representação da relação entre documentos e termos. A perda dessa informação pode influenciar negativamente os resultados de classificação (ROSSI, 2015). Uma alternativa natural, é a utilização de redes para representação dos documentos ou termos. Uma rede G é um grafo, representado por uma tripla $G = \{O, R, W\}$, em que O é um conjunto de objetos, R é um conjunto de relações e W é um conjunto de pesos das cada relações. A Figura 2 é uma ilustração de uma rede de N documentos. Cada um dos documentos representado por retângulos é um nó ou vértice do grafo. As relações entre documentos denotadas pelas linhas pontilhadas, representam as proximidades entre documentos. É possível observar que, nem sempre há uma conexão ou relação entre documentos (por exemplo, Doc 2 e Doc 3).

Figura 2 – Ilustração de uma rede de documentos.

Fonte – ROSSI (2015).

A representação de documentos por meio de redes torna-se interessante diante das falhas de representação que o modelo BoW apresenta. Nesse contexto, uma rede é definida como um grafo em que documentos, termos ou ambos podem ser modelados como objetos da rede. As arestas representam algum tipo de relação entre objetos da rede e o peso dessa relação é medido segundo alguma função previamente especificada, como uma função de proximidade entre documentos ou termos (ROSSI; REZENDE; LOPES, 2015). A representação da relação entre

documentos ou termos ocorre naturalmente em rede. Além disso, algoritmos de classificação baseados em redes apresentaram resultados interessantes e competitivos quando comparados aos algoritmos semissupervisionados não baseados em redes (SUBRAMANYA; TALUKDAR, 2014). O interesse por modelos de representação de documentos ainda melhores que os modelos existentes ainda é presente no meio científico (SINOARA *et al.*, 2019).

Uma rede de documentos pode ser utilizada para classificar de documentos utilizando o o algoritmo GFHF (ZHU; GHAHRAMANI; LAFFERTY, 2003) ou o algoritmo LLGC (ZHOU *et al.*, 2004). A utilização de rede de termos na classificação transdutiva é uma novidade introduzida por autores do *framework* TCTN (ROSSI; REZENDE; LOPES, 2015). Nesse *framework*, a classificação transdutiva de documentos só pode ser realizada após a construção de uma rede de termos. Uma rede de termos é um grafo não-direcionado e valorado em que, vértices são os termos dos documentos, as relações são definidas segundo algum critério de conexão (como vizinhança mais próxima, por exemplo) e as relações entre vértices são quantificadas de acordo com uma medida de proximidade predefinida. Na Subseção 2.1.1, o aprendizado semissupervisionado transutivo baseado em redes é definido.

2.1.1 Classificação Semissupervisionada Transdutiva de Documentos utilizando Redes

A classificação de documentos de maneira semissupervisionada e transdutiva baseada em redes é interessante pelo fato de utilizar informações de poucos documentos rotulados, informações de documentos não rotulados, representar naturalmente a relação entre documentos e produzir resultados de classificação competitivos (BELLET; HABRARD; SEBBAN, 2015). A classificação semissupervisionada baseada em redes é uma subclasse dos algoritmos semissupervisionados. Nesse modelo, o processo de classificação de documentos pode ser dividido em três etapas (SUBRAMANYA; TALUKDAR, 2014): (1) Construção da Rede, (2) Rotulação de um subconjunto de vértices, (3) Inferência dos rótulos de documentos não rotulados. Cada uma dessas etapas é detalhada a seguir.

(1) Construção da Rede

A construção da rede é uma das etapas mais importantes, pois pode afetar a qualidade dos resultados de classificação (BERTON; LOPES, 2015). Nessa etapa, uma rede de objetos¹ $N = \{O, R, W\}$, em que O é um conjunto de objetos, R um conjunto de relações entre esses objetos e W um conjunto de pesos de cada relação em R , deve ser construído sob as seguintes hipóteses: (1) N é não direcionada e W é simétrico; (2) $W_{ij} = 0$ indica a ausência de conexão entre os vértices i e j ; (3) não há *self-loops*² na rede.

¹ No contexto dessa dissertação, objetos podem ser termos, documentos ou ambos.

² *Self-loops* são arestas de um vértice para ele mesmo.

Para decidir se existe ou não uma conexão entre termos, um dos diversos algoritmos presentes na literatura podem ser utilizados, bem como um critério previamente definido. Algoritmos tradicionais para construção da rede incluem: k -vizinhos mais próximos mútuos, k -vizinhos mais próximos simétricos, vizinhança- ε e rede totalmente conectada (ROSSI, 2015).

Na rede totalmente conectada, a função de proximidade pode retornar valores acima de zero para todas as relações sem restrições, formando assim, uma rede totalmente conectada. No método denominado vizinhança- ε , uma relação entre dois termos (i, j) é estabelecida se a proximidade entre esses termos é superior a um limiar ε . A proximidade entre termos é medida segundo uma função de proximidade fornecida previamente. No caso dos k -vizinhos mais próximos simétricos uma relação entre dois termos (i, j) é estabelecida se i é vizinho mais próximo de j ou j é vizinho mais próximo de i . Finalmente, no k -vizinhos mais próximos mútuos uma relação entre dois termos (i, j) é estabelecida se i é vizinho mais próximo de j e j é vizinho mais próximo de i .

A Figura 3 é um exemplo de vizinhança estabelecida pelos algoritmos k -vizinhos mais próximos mútuos (mKnn) e k -vizinhos mais próximos simétricos (sKnn). Na Figura 3 há uma matriz binária em que as linhas e colunas são objetos e os componentes da matriz denotam o grau de proximidade entre esses objetos. Observa-se que no mKnn, apenas objetos que são mutualmente similares são considerados como vizinhos, enquanto que no sKnn a relação de vizinhança é estabelecida apenas considerando o fato de um dos objetos ser mais similar a outro.

Figura 3 – Exemplo de vizinhos mais próximos considerados pelos algoritmos k -vizinhos mais próximos mútuos (mKnn) e k -vizinhos mais próximos simétricos (sKnn).

	A	B	C	D	
A	0	1	0	1	mKnn = {(1,2), (2,1), (1,4), (4,1)} sKnn = {(1,2), (2,1), (1,4), (4,1), (2,3), (3,2), (3,1), (1,3), (4,2), (2,4), (4,3), (3,4)}
B	1	0	1	0	
C	1	0	0	0	
D	1	1	1	0	

Fonte – Autoria Própria.

Outro ponto importante na construção da rede é a definição de pesos entre as relações da rede. Os pesos das relações entre objetos geralmente são definidos por uma função de proximidade. A definição de uma função de proximidade pode não ser trivial, pois funções de propósito geral normalmente não generalizam bem porque a noção semântica de proximidade pode variar de acordo com o contexto de aplicação (BELLET; HABRARD; SEBBAN, 2015). Na próxima seção, funções de distância e proximidade são definidas, alguns dos problemas de funções de proximidade de propósito geral são discutidos e alternativas para esses impasses são

apresentadas.

Funções de Proximidade

Grande parte dos algoritmos de aprendizado necessita de uma função de proximidade que define o grau de relação entre objetos. Embora, existam funções de distância e proximidade de propósito geral como a distância Euclidiana e similaridade de Cosseno, estas falham em capturar características intrínsecas de diversos conjuntos de dados. De fato, a noção semântica de proximidade depende do contexto de aplicação da técnica (BELLET; HABRARD; SEBBAN, 2015).

Diante dos impasses supracitados, é natural que surja a ideia de construir funções de proximidade para conjuntos de dados de diferentes contextos. Contudo, a construção manual de funções de proximidade pode ser uma tarefa custosa e não trivial. Com essa justificativa, o aprendizado automático de métricas emergiu e tem atraído interesse nos últimos anos (XING *et al.*, 2003; YANG; JIN, 2006; BELLET; HABRARD; SEBBAN, 2015).

Uma métrica é uma função $dist : X \times X \rightarrow \mathbb{R}^+$ que define uma noção de distância entre pares de um conjunto de dados X e satisfaz as seguintes propriedades $\forall x_i, x_j, x_k \in X$:

1. $dist(x_i, x_j) \geq 0$ (Não-negatividade)
2. $dist(x_i, x_j) = dist(x_j, x_i)$ (Simetria)
3. $dist(x_i, x_j) \leq dist(x_i, x_k) + dist(x_k, x_j)$ (Desigualdade triangular)
4. $dist(x_i, x_j) = 0 \iff x_i = x_j$ (Identidade dos indiscerníveis)

Enquanto a função de distância é bem definida na literatura, não existe um padrão na definição de função de similaridade. Desse modo, uma função de similaridade pode ser qualquer função $sim : X \times X \rightarrow \mathbb{R}^+$ (BELLET; HABRARD; SEBBAN, 2015). Neste projeto, o termo mais genérico “função de proximidade” é utilizado para denotar similaridade ou distância entre dois objetos.

Geralmente, métricas para dados numéricos são da família de distâncias denominada Minkowski. Nesse caso, a métrica é induzida pelo parâmetro $p \geq 1$ que indica a norma da distância. A forma geral da distância de Minkowski, em que d é o número de atributos de X é definida na Equação 2.3.

$$dist_p(x_i, x_j) = \|x_i - x_j\|_p = \left(\sum_{k=1}^d |x_{ik} - x_{jk}|^p \right)^{1/p} \quad (2.3)$$

Algumas distâncias tradicionais podem ser induzidas ao ajustar o valor de p como se segue:

- Para $p = 1$, temos a distância de Manhattan: $dist_{man}(x_i, x_j) = \sum_{k=1}^d |x_{ik} - x_{jk}|$;
- Para $p = 2$, temos a distância Euclidiana: $dist_{euc}(x_i, x_j) = \left(\sum_{k=1}^d (|x_{ik} - x_{jk}|)^2 \right)^{1/2}$;
- Para $p = \infty$, temos a distância de Chebyshev: $dist_{che}(x_i, x_j) = \max_k |x_{ik} - x_{jk}|$.

Enquanto as distâncias supracitadas são consideradas de propósito geral, algoritmos de aprendizado de métricas têm o objetivo de produzir novas métricas a partir de uma base de dados pré-especificada. Geralmente, o aprendizado de métricas envolve a adaptação de uma função de propósito geral a um contexto específico. A distância de Mahalanobis dada pela Equação 2.4 é frequentemente utilizada na literatura do aprendizado de métricas (BELLET; HABRARD; SEBBAN, 2015).

$$dist_M = \sqrt{(x_i - x_j)^T M (x_i - x_j)} \quad (2.4)$$

O objetivo é encontrar a matriz de Mahalanobis $M \in \mathbb{R}^{n \times n}$. Uma desvantagem de métodos baseados na matriz de Mahalanobis consiste na complexidade de tempo na estimativa dos parâmetros da função, uma vez que a correlação dos atributos deve ser estimada (XING *et al.*, 2003). Métricas lineares, como a distância de Mahalanobis, são mais fáceis de otimizar e menos propensas ao *overfitting*, contudo falham em capturar variações não-lineares nos dados. Por outro lado, métricas não lineares como a distância χ^2 conseguem capturar variações não lineares nos dados, contudo a complexidade de otimização da métrica aumenta. Finalmente, o aprendizado de métricas, conhecidas como locais, consiste no aprendizado de múltiplas métricas locais, sendo essas lineares ou não lineares. Geralmente, esse último tipo é aplicado em problemas que envolvem dados heterogêneos (BELLET; HABRARD; SEBBAN, 2015).

Além dos métodos tradicionais (BELLET; HABRARD; SEBBAN, 2015), em Miquilini *et al.* (2017), um método para construção automática de funções de distância para algoritmos semissupervisionados baseados em grafo é proposto. O método denominado GEAD utiliza Gramática Evolutiva e meta-aprendizado para produção de novas funções de distância. A hipótese se baseia na ideia de que, se a função de distância funciona bem em uma pequena parte dos dados, funcionará também para a base de dados em sua totalidade. Essa abordagem pode ser utilizada para associação de pesos a relações de uma rede de documentos uma vez que documentos podem ser representados como vetores em um espaço vetorial, contudo não é útil em situações em que a co-ocorrência entre termos é considerada no cálculo de proximidade ou distância entre documentos, por exemplo.

O *framework* TCTN requer uma função de proximidade para atribuição de pesos nas relações entre objetos (termos) da rede que utiliza para classificação de documentos. Contudo, a função de proximidade requerida por TCTN deve considerar valores de co-ocorrência de termos. Funções que consideram tais valores costumam ser utilizadas com frequência na área de Regras de Associação e são, originalmente denominadas “Medidas de Interesse”. Até o

presente momento, não foi possível encontrar técnicas capazes de construir tais medidas de maneira automática. Um objetivo deste trabalho é a proposta, implementação e avaliação de uma abordagem para construção automática de medidas de interesse para construção de redes de termos utilizando o *framework* TCTN.

(2) Rotulação de um subconjunto de vértices

A rotulação inicial de documentos geralmente é realizada por um especialista capaz identificar o rótulo de cada documento. A rotulação de documentos é uma tarefa dispendiosa e consome muito tempo, contudo a indução de bons classificadores a partir de algoritmos de aprendizado indutivo supervisionado, normalmente, só é possível com a presença de muitos exemplos rotulados. Uma alternativa viável é a utilização de algoritmos de aprendizado semissupervisionado que são capazes de classificar documentos utilizando documentos rotulados e não rotulados. Um detalhe importante está no fato de utilizarem poucos documentos rotulados, quando comparado ao número de documentos não rotulados. Portanto, na classificação semissupervisionada o esforço de rotulação inicial de vértices (ou documentos/termos) é muito menor, pois apenas poucos exemplos precisam ser rotulados para o funcionamento do algoritmo.

(3) Inferência dos rótulos de documentos não rotulados

A inferência dos rótulos de documentos não rotulados é o grande objetivo dos algoritmos de classificação de documentos semissupervisionada baseada em redes. O *framework* TCTN utiliza um algoritmo denominado LLGC para inferência dos rótulos de documentos não rotulados, contudo existem outros algoritmos como o GFHF que também pode ser utilizado na classificação de documentos. Em seguida, ambos são descritos.

Gaussian Field and Harmonic Functions (GFHF)

Gaussian Field and Harmonic Functions (GFHF) (ZHU; GHAHRAMANI; LAFFERTY, 2003) é um algoritmo semissupervisionado que utiliza redes como estrutura para representar os documentos rotulados e não rotulados. As arestas do grafo possuem pesos que denotam proximidades entre vértices. A inferência de rótulos para documentos não rotulados pode ser realizada por meio da minimização da Equação 2.5.

$$Q(F) = \frac{1}{2} \sum_{d_i, d_j \in D} W_{d_i, d_j} (F_{d_i} - F_{d_j})^2 + \lim_{\mu \rightarrow \infty} \mu \sum_{d_i \in D^L} (F_{d_i} - y_{d_i})^2 \quad (2.5)$$

O termo $\lim_{\mu \rightarrow \infty}$ indica que os documentos rotulados devem manter seus rótulos iniciais, portanto apenas documentos não rotulados são classificados e o termo $W_{d_i, d_j} = \exp(-\sum_{c=1}^m (d_{i,c} - d_{j,c})^2 / \sigma_c^2)$ é uma função de base radial cuja abertura é controlada por σ_c . GFHF tem sido utilizado na segmentação de imagens médicas (GRADY; FUNKA-LEA, 2004), colorização de

imagens (LEVIN; LISCHINSKI; WEISS, 2004) e desambiguação do sentido de palavras (NIU; JI; TAN, 2005), sendo que neste último uma abordagem iterativa é utilizada para otimização da Equação 2.5. Essa abordagem iterativa é denominada *Label Propagation*.

Learning with Local and Global Consistency (LLGC)

Learning with Local and Global Consistency (LLGC) é outro algoritmo de classificação semissupervisionada transdutiva baseada em grafos, sendo que os seus resultados de classificação do LLGC foram superiores ao GFHF. O LLGC faz a inferência dos rótulos por meio da minimização da Equação 2.6

$$Q(F) = \frac{1}{2} \left(\sum_{i,j=1}^n W_{ij} \left\| \frac{1}{\sqrt{D_{ii}}} F_i - \frac{1}{\sqrt{D_{ij}}} F_j \right\| + \mu \sum_i \|F_i - Y_i\| \right) \quad (2.6)$$

em que μ é o parâmetro que controla o nível em que as informações iniciais devem ser mantidas durante o aprendizado. W é a matriz de afinidade, definida como $W_{d_i, d_j} = \exp(-\|d_{i,c} - d_{j,c}\| / 2\sigma_c^2)$, em que σ_c controla a abertura dessa função de base radial. F refere-se à matriz que contém os pesos de todos objetos. Portanto, $F_i = \langle f_1, f_2, \dots, f_l \rangle$ é o vetor de informação de classe do objetivo i e D é matriz diagonal, tal que $D = \sum_{j \in J} W_{ij}$.

Algoritmos de aprendizado semissupervisionado transdutivo baseados em rede podem ser utilizados para inferência do rótulo de documentos não rotulados. Na literatura, é possível encontrar diversos algoritmos de aprendizado transdutivo baseados em rede como o Mincut (BLUM; CHAWLA, 2001), GFHF (ZHU; GHAMRANI; LAFFERTY, 2003), LLGC (ZHOU *et al.*, 2004) e Tikhonov Regularization (BELKIN; MATVEEVA; NIYOGI, 2004), entre outros. Neste capítulo, os algoritmos GFHF e LLGC foram apresentados. O algoritmo LLGC apresentou resultados de classificação superiores ao GFHF e, portanto, foi utilizado no *framework* TCTN para classificação de documentos utilizando uma rede de termos. Contudo, alguns autores consideram as diferenças entre algoritmos não cruciais, enquanto a construção do grafo se trata de uma etapa mais sensível e de grande impacto nos resultados de classificação (ZHU; LAFFERTY; ROSENFELD, 2005; BERTON; LOPES; Vega-Oliveros, 2018). Na seção seguinte, detalhes da abordagem de classificação utilizada pelo TCTN são apresentados.

2.2 Transductive Classification through Terms Networks (TCTN)

A classificação semissupervisionada transdutiva de documentos pode ser realizada utilizando-se uma rede de termos. Como definido na Seção 2.1, uma rede de termos é um grafo nos quais termos são objetos, arestas representam as relações entre termos e o peso das relações é definido segundo uma função de proximidade. Formalmente, uma rede de termos

é um grafo $N = \{T, R, W\}$, em que T representa um conjunto de termos de uma coleção de documentos, R as relações entre termos e W uma matriz de pesos entre as relações.

Em Rossi, Rezende e Lopes (2015), um *framework* denominado TCTN é proposto com intuito de adaptar o algoritmo LLGC (ZHOU *et al.*, 2004) ao problema de classificação de documentos utilizando-se uma rede de termos.

A rede de termos possui algumas vantagens em relação a rede de documentos. A rotulação de poucos documentos implica na rotulação de diversos termos. Além disso, é possível realizar a seleção de termos mais importantes, antes de realizar o aprendizado. Por exemplo, em Rossi, Rezende e Lopes (2015) termos foram selecionados a partir de um ranque construído com base no TF-IDF de cada termo. Os experimentos indicaram que a seleção de termos não afeta os resultados de classificação de maneira significativa.

O *framework* TCTN utiliza a informação da relação entre termos para classificação de documentos. Segundo ROSSI (2015), termos altamente relacionados tendem a pertencer a mesma classe. O *framework* pode ser dividido em quatro etapas, que são:

1. Construção da rede de termos;
2. Definição das informações de classe iniciais para os termos;
3. Propagação de informações de classe (LLGC);
4. Classificação de documentos.

A rede utilizada durante aplicação do TCTN consiste de um grafo em que os vértices são os termos e as relações entre os termos são as proximidades entre eles. Portanto, para a construção da rede de termos, é necessário definir um critério de conexão entre objetos da rede e uma função de proximidade (ou medida de interesse) entre termos deve ser fornecida para construção da rede de termos. Em Rossi, Rezende e Lopes (2015), alguns critérios como k -vizinhos mais próximos mútuos, k -vizinhos mais próximos simétricos, vizinhança- ε e rede totalmente conectada (ROSSI, 2015) são explorados. No método denominado vizinhança- ε , uma aresta (i, j) em que $i, j \in V$ é adicionada, apenas se a proximidade entre i e j for menor que um valor ε em que $\varepsilon \in \mathbb{R}^+$ é um parâmetro livre. No k -vizinhos mais próximos simétricos uma aresta (i, j) é adicionada se, e somente se i é vizinho mais próximo de j ou j é vizinho mais próximo de i . Finalmente, no k -vizinhos mais próximos mútuos uma aresta (i, j) é adicionada se, e somente se i é vizinho mais próximo de j e j é vizinho mais próximo de i .

A definição de informações de classe iniciais para os termos é realizada de acordo com o mecanismo proposto por Rossi, Rezende e Lopes (2015). A informação inicial de um termo t_j para a classe c_j é dada por:

$$f_{t_j c_j}^0 = \frac{\sum_{d_k \in D_L} \text{freq}(d_k, t_j) y_{d_k c_j}}{\sum_{d_k \in D_L} \text{freq}(d_k, t_j)} \quad (2.7)$$

em que D_L é o conjunto de documentos rotulados $y_{d_k c_j}$ é igual a 1 se o documento d_k pertence

à classe c_j , caso contrário $y_{d_k c_j}$ é igual a 0. E $freq(d_k, t_i)$ refere-se à frequência do termo t_i no documento d_k .

A propagação das informações de classes é realizada por meio do algoritmo LLGC (ZHOU *et al.*, 2004). Como explicado na Subseção 2.1.1, o algoritmo consiste na minimização da Equação 2.6. Contudo, o problema pode ser resolvido iterativamente aplicando-se a Equação 2.8.

$$F = \alpha S + (1 - \alpha)Y \quad (2.8)$$

em que $S = D^{1/2}WD^{1/2}$ e $\alpha \in (0, 1)$. $D^{1/2}$ é uma matriz em que cada elemento da diagonal é a soma dos elementos de cada linha.

Finalmente, a classificação de documentos é realizada utilizando das informações de classe que foram propagadas para os termos da rede. Portanto, a informação de classe de um documento não rotulado $d_i \in D^U$ para a classe c_j é dada pela função definida na Equação 2.9.

$$f_{i c_j} = \sum_{t_k \in T} freq(d_i, t_k) f_{t_k c_j} \quad (2.9)$$

A classe do documento $d_i \in D^U$ é definida pelo *argmin* do vetor $\vec{f}_{d_i}$. Por se tratar de um algoritmo não determinístico para diferentes configurações iniciais, TCTN geralmente é executado k vezes, a fim de, avaliar a qualidade dos resultados obtidos.

A avaliação final da classificação de documentos é realizada de acordo com uma métrica que utiliza dados de uma matriz de confusão. A matriz de confusão contém a contagem de classificações corretas (verdadeiros positivos (VP) e verdadeiros negativos (VN)) e classificações incorretas (falsos positivos (FP) e falsos negativos (FN)). A Tabela 2 representa uma matriz de confusão para a classe c_i .

Tabela 2 – Matriz de confusão para a classe c_i .

Classe c_i		Classe real	
		Positivo	Negativo
Classe predita	Positivo	VP_i	FP_i
	Negativo	FN_i	VN_i

Na literatura é possível encontrar diversas métricas de avaliação como: Acurácia, Erro, Precisão, Revocação, Micro- F^1 , Macro- F^1 e F^1 -ponderado (SEBASTIANI, 2002a; ROSSI, 2015). A Acurácia e o Erro são as métricas comuns na literatura e estão descritas nas equações 2.10 e 2.11, respectivamente.

$$\text{Acurácia}_{c_i} = \frac{VP_i + VN_i}{VP_i + VN_i + FP_i + FN_i} \quad (2.10) \quad \text{Erro}_{c_i} = \frac{FP_i + FN_i}{VP_i + VN_i + FP_i + FN_i} \quad (2.11)$$

Medidas de Interesse

Medidas de interesse (ou funções de proximidade) são amplamente utilizadas na mineração de regras de associação para quantificar a relevância de um padrão. No TCTN, essas medidas são utilizadas para quantificar conexões entre vértices da rede de termos, ou seja, calcular proximidade entre termos. Em [Rossi, Rezende e Lopes \(2015\)](#), cinco medidas de interesse com características diferentes foram selecionadas para execução dos experimentos. Contudo, nenhum método foi apresentado para indicar qual função de proximidade deve ser utilizada. De fato, geralmente, a escolha por uma medida de proximidade não é uma tarefa trivial ([TAN; KUMAR; SRIVASTAVA, 2002](#)). As medidas de interesse consideradas no trabalho de [Rossi, Rezende e Lopes \(2015\)](#) utilizam probabilidades de co-ocorrência entre termos e são baseadas na matriz de contingência apresentada na Tabela 3.

Tabela 3 – Matriz de Contingência

	t_j	$\neg t_j$	Total
t_i	$p(t_i, t_j)$	$p(t_i, \neg t_j)$	$p(t_i)$
$\neg t_i$	$p(\neg t_i, t_j)$	$p(\neg t_i, \neg t_j)$	$p(\neg t_i)$
Total	$p(t_j)$	$p(\neg t_j)$	1

Cada célula da matriz de contingência tem o seguinte significado.

- $p(t_i, t_j)$: probabilidade de co-ocorrência entre os termos t_i e t_j ;
- $p(t_i, \neg t_j)$: probabilidade de um termo t_i não co-ocorrer com um termo t_j ;
- $p(\neg t_i, t_j)$: probabilidade de um termo t_j não co-ocorrer com um termo t_i ;
- $p(\neg t_i, \neg t_j)$: probabilidade dos termos t_i e t_j não co-ocorrerem.

Algumas das medidas de interesse existentes na literatura e utilizadas por [Rossi, Rezende e Lopes \(2015\)](#), são descritas na tabela 4.

Tabela 4 – Medidas de Interesse consideradas no TCTN

ID	Função de Proximidade	Fórmula	Intervalo
1	<i>Piatetsky-Shapiro</i>	$p(t_i, t_j) - p(t_i)p(t_j)$	$[-0.25, 0.25]$
2	<i>Informação Mútua</i>	$p(t_i, t_j) \log_2 \left(\frac{p(t_i, t_j)}{p(t_i)p(t_j)} \right) + p(t_i, \neg t_j) \log_2 \left(\frac{p(t_i, \neg t_j)}{p(t_i)p(\neg t_j)} \right) + p(\neg t_i, t_j) \log_2 \left(\frac{p(\neg t_i, t_j)}{p(\neg t_i)p(t_j)} \right) + p(\neg t_i, \neg t_j) \log_2 \left(\frac{p(\neg t_i, \neg t_j)}{p(\neg t_i)p(\neg t_j)} \right)$	$[-1, 1]$
3	<i>Kappa</i>	$\frac{p(t_i, t_j) + p(\neg t_i, \neg t_j) - p(t_i)p(t_j) - p(\neg t_i)p(\neg t_j)}{1 - p(t_i)p(t_j) - p(\neg t_i)p(\neg t_j)}$	$[-1, 1]$
4	<i>Q de Yule</i>	$\frac{p(t_i, t_j)p(\neg t_i, \neg t_j) - p(\neg t_i, t_j)p(t_i, \neg t_j)}{p(t_i, t_j)p(\neg t_i, \neg t_j) + p(\neg t_i, t_j)p(t_i, \neg t_j)}$	$[-1, 1]$
5	<i>Suporte</i>	$p(t_i, t_j)$	$[0, 1]$

A literatura dispõe de diversas medidas de interesse de propósito geral ([GENG; HAMILTON, 2006](#)). O objetivo desse trabalho consiste em propor um *framework* para realizar a seleção da melhor função de proximidade de maneira automática a partir de um conjunto de medidas, ou seja, *framework* que construa novas medidas de interesse de maneira automática. Nesta dissertação, o termo função de proximidade será utilizado para denotar medidas de interesse.

2.3 Considerações Finais

Neste capítulo, foram apresentados conceitos relacionados à classificação de documentos por meio de algoritmos de aprendizado semissupervisionados transdutivos em especial, o *framework* TCTN foi descrito, bem como uma de suas limitações que é a necessidade da especificação de uma função de proximidade para construção de uma rede de termos que é utilizada na classificação de documentos. Para solucionar este problema uma abordagem baseada em Evolução Gramatical (RYAN; O'NEILL; COLLINS, 1998) é proposta nessa dissertação. Essa técnica faz parte dos algoritmos evolutivos e já foi utilizada para evolução de funções matemáticas (SOUSA; LORENA; BASGALUPP, 2017; MIQUILINI *et al.*, 2017). Geralmente, algoritmos evolutivos são aplicados em problemas os quais não se conhece muitas características do espaço de soluções ou em situações em que um esforço para evitar mínimos locais precisa ser realizado. A ideia central deste trabalho é utilizar Evolução Gramatical para construção automática de funções de proximidade (ou medidas de interesse) para algoritmos de classificação semissupervisionada transdutiva baseada em redes de termos. No próximo capítulo, detalhes sobre a Evolução Gramatical são apresentados.

GRAMÁTICA EVOLUTIVA

O TCTN requer que uma função de proximidade seja definida antes da construção da rede de termos que será utilizada para classificação de documentos. Na literatura, é possível encontrar diversas funções de proximidade, contudo escolher a função que levará o TCTN a produzir bons resultados de classificação pode ser uma tarefa não trivial. Além disso, é possível que novas funções de proximidade sejam necessárias uma vez que a noção de proximidade depende do contexto (coleção de textos) (BELLET; HABRARD; SEBBAN, 2015). A hipótese para solução deste problema, é a aplicação de um algoritmo denominado Evolução Gramatical que é uma meta-heurística baseada em população. Uma das vantagens da utilização de meta-heurísticas na solução de problemas decorre da capacidade de não ficar preso a soluções que representam máximos ou mínimos locais. A introdução da aleatoriedade costuma ser essencial nesse processo. Os algoritmos evolutivos proveem mecanismos que permitem explorar características úteis de soluções já encontradas e a introdução de novas características em soluções existentes com objetivo de produzir novas soluções. Neste trabalho, o algoritmo Evolução Gramatical é utilizado para construção automática de novas medidas de interesse. Assim, é importante entender como algoritmos evolutivos funcionam, quais são seus componentes e quais as particularidades da Evolução Gramatical.

Este capítulo está dividido da seguinte forma: na Seção 3.1, os algoritmos evolutivos são descritos. Na Seção 3.2, as diferenças da Evolução Gramatical são apresentadas. A Seção 3.3 contém as considerações finais sobre este capítulo.

3.1 Algoritmos Evolutivos

Algoritmos evolutivos utilizam mecanismos de otimização de soluções inspirados na Natureza como a mutação, reprodução e seleção. Estes algoritmos conseguem lidar com problemas complexos em que características do problema não são completamente entendidas. Basicamente, cada possível solução para o problema é representada por um indivíduo ou cromossomo. O

conjunto de possíveis soluções para o problema (cromossomos) é denominado população. O material genético de um cromossomo é chamado de genótipo, quando está em formato codificado e de fenótipo quando decodificado. Uma população é avaliada, submetida a operadores de cruzamento e mutação para geração de uma nova geração, que é novamente avaliada e o ciclo se repete até que um critério seja satisfeito. Esse ciclo é chamado de geração. Ao longo das gerações, espera-se que os indivíduos mais aptos sobrevivam. Em outras palavras, boas soluções devem permanecer ao longo das gerações. Na literatura, existem diversas implementações de algoritmos evolutivos. Algoritmos Genéticos (HOLLAND, 1992), Programação Genética (KOZA, 1997), Evolução Gramatical (RYAN; O'NEILL; COLLINS, 1998), Evolução Diferencial (PRICE; STORN; LAMPINEN, 2006) são alguns exemplos de algoritmos evolutivos que contém, cada um deles, suas particularidades. Independente da implementação, os componentes mais importantes dos algoritmos evolutivos são: (1) Representação, (2) Inicialização, (3) Função de aptidão, (4) População, (5) Seleção, (6) Operadores genéticos, (7) Condição ou critério de parada (EIBEN; SMITH, 2003). No restante dessa seção, detalhes sobre cada um sete dos componentes supracitados são apresentados.

3.1.1 Representação

Um indivíduo (ou cromossomo) de uma população geralmente é composto por um conjunto de genes, tem tamanho fixo e representa uma possível solução para o problema em questão. Por exemplo, um indivíduo pode ser visto computacionalmente como um vetor e os genes como componentes desse vetor. Geralmente, cromossomos assumem duas formas: (1) a forma codificada em genótipo e a forma (2) decodificada em fenótipo. Quando codificados, cromossomos assumem uma estrutura específica com objetivo de facilitar a aplicação dos operadores evolutivos (mutação e cruzamento). Quando decodificados, assumem uma forma de solução legível e útil. Cada cromossomo, geralmente é decodificado em um fenótipo para que, em seguida, seja avaliado a partir de uma função de aptidão.

A representação dos indivíduos pode ser realizada de diversas maneiras. Uma forma de representação tradicional e muito utilizada em algoritmos genéticos (HOLLAND, 1992) é a representação binária, em que cada cromossomo é uma sequência de *bits* (LINDEN, 2012). O que cada *bit* representa, depende do domínio do problema. Na Figura 4, o número inteiro 13 é representado em sua forma binária. Utilizando essa forma de representação, o indivíduo 13 é codificado no genótipo 00001101₂.

Outra forma de representação de indivíduos que é comum na Programação Genética (PG) (KOZA, 1997), é a representação por meio de árvore. A árvore utilizada por PGs geralmente representam novas funções ou programas de computador em linguagem LISP¹. Por exemplo, a expressão simbólica² $(* (+ X (/ Y 1.5)) (- Z 0.3))$, pode ser representado pela árvore da Figura

¹ Site oficial da linguagem LISP: <<https://lisp-lang.org>>

² Expressão simbólica é uma notação para descrever estruturas de dados do tipo árvore. Essa notação foi

Figura 4 – Exemplo de um indivíduo sendo decodificado de sua representação binária. O indivíduo tem genótipo 00001101₂ e fenótipo 13.

Fonte – Autoria Própria.

5.

Figura 5 – Exemplo de indivíduo representado como uma árvore. Essa árvore representa a expressão simbólica $(*(+X(/Y1.5))(-Z0.3))$.

Fonte – Adaptado de Wong e Leung (2006).

Os nós internos da árvore são denominados funções primitivas. Já os nós folha, são denominados terminais. No exemplo da Figura 5, os terminais são $\{X, Y, 1.5, Z, 0.3\}$ e o conjunto de funções primitivas é $\{*, +, -, /\}$. Uma propriedade interessante desse modelo de representação é a propriedade de fechamento. Assim, cada função primitiva deve aceitar qualquer terminal ou resultado de uma outra função como argumento.

3.1.2 Inicialização

A inicialização da população de soluções viáveis de um algoritmo evolutivo é geralmente realizada de maneira aleatória, em que valores para genes de cromossomos são selecionados aleatoriamente segundo alguma distribuição estatística considerando o domínio e restrições do problema. Porém, existem outras formas de inicialização, como a inicialização parcialmente enumerativa e a inicialização aleatória não-uniforme (GOLDBERG; HOLLAND, 1988).

popularizada com a linguagem de programação LISP (MCCARTHY, 1959).

Quando árvores são utilizadas na representação de indivíduos, a inicialização ainda pode ser feita de maneira aleatória ou utilizando técnicas de inicialização sensíveis (FAGAN; FENTON; O'NEILL, 2016). Algoritmos de PG e EG se beneficiam de populações de indivíduos bem diversificados (KOZA, 1997), portanto, técnicas que buscam satisfazer essa condição como *Ramped Half-and-Half* (RHH) (RYAN; AZAD, 2003) e *Position Independent Grow* (PI Grow) (FAGAN; FENTON; O'NEILL, 2016) são frequentemente utilizadas.

A população de indivíduos gerada pelo algoritmo RHH pode ser dividida em duas partes de mesmo tamanho: (1) indivíduos que atingem uma profundidade máxima pré-definida na árvore (denominado *Full*) e (2) indivíduos gerados aleatoriamente com profundidade menor que uma profundidade pré-especificada (denominado *Grow*). Por outro lado, o algoritmo PI Grow utiliza apenas o aspecto *Grow* e, na literatura, há indícios de que o PI Grow é mais promissor do que o RHH (FAGAN; FENTON; O'NEILL, 2016).

3.1.3 Função de Aptidão

Algoritmos evolutivos precisam determinar a qualidade dos indivíduos de uma população. Para isso, uma função denominada função de aptidão deve ser estabelecida. A função de aptidão pode ser considerada o componente dos algoritmos evolutivos que é mais próximo da natureza do problema (LINDEN, 2012), pois a sua definição depende do contexto de aplicação. Geralmente, a função de aptidão escolhida é de baixa complexidade computacional, pois sua utilização é frequente e o desempenho do algoritmo evolutivo pode ser afetado (GOLDBERG; HOLLAND, 1988). No contexto de classificação de documentos, medidas de performance como a Acurácia, Precisão *Recall*, Macro-F1, Micro-F1, F1-Ponderado, etc, podem ser utilizadas como função de aptidão.

Para ilustrar a definição de função de aptidão, considere o seguinte exemplo. Em um problema de otimização qualquer, indivíduos são representados por vetores em duas dimensões, ou seja $\vec{v}_1, \dots, \vec{v}_n \in \mathbb{R}^2$ em que cada componente $x_i \in [-5, 5]$. A função de aptidão é dada por $f(\vec{v}) = \sum_{i=1}^2 x_i^2$ e o objetivo é encontrar um vetor que retorne o maior valor dada a função de aptidão. Nesse caso, o indivíduo $\vec{v}_i = \langle 5, 5 \rangle$ é avaliado com o valor 50, que representa o valor ótimo global para esse problema, enquanto o vetor $\vec{v}_j = \langle 3, 3 \rangle$, é avaliado com valor 18 e representa uma solução viável, porém de menor qualidade ou menos apta do que o indivíduo v_i .

3.1.4 População

Algoritmos evolutivos são também conhecidos como meta-heurísticas de otimização baseadas em população. A população é um conjunto de indivíduos que, por sua vez, representam possíveis soluções para o problema. Por exemplo, se uma população de cem indivíduos distintos é gerada, cem possíveis soluções para o problema também são geradas. Na maioria dos algoritmos evolutivos, o tamanho da população é definido como um parâmetro fixo.

Embora intuitivo, é interessante mencionar que, o tamanho da população pode afetar o desempenho do AE, uma vez que, cada indivíduo deve ser decodificado e avaliado. Contudo, uma população pequena pode não ser suficiente para a realização da exploração do espaço de soluções. Além disso, é interessante que a população contenha diversidade, ou seja, indivíduos devem ser diferentes para que combinações genéticas realizadas pelos operadores evolutivos permitam exploração do espaço de busca (mutação) e o aproveitamento de padrões úteis de cada indivíduo (cruzamento) (EIBEN; SMITH, 2003).

3.1.5 Seleção

Indivíduos de uma população devem ser selecionados, tanto para o processo de reprodução, quanto para o processo de sobrevivência. Geralmente, indivíduos mais aptos possuem mais chances de reproduzirem e sobreviverem (EIBEN; SMITH, 2003). Na literatura, observa-se frequentemente a expressão "indivíduo mais apto" ou "de maior aptidão". As soluções modeladas em algoritmos evolutivos tendem a ser de maximização de um ou mais objetivos. Isso porque, o termo aptidão traz a ideia de, quanto mais apto melhor. Para manter-se consistente aos termos da área de computação evolutiva, os problemas de minimização são frequentemente convertidos em problemas de maximização. Além disso, é importante ressaltar que as técnicas de seleção tradicionais independem de uma representação de indivíduos específica. Nas subseções seguintes, técnicas tradicionais para seleção de indivíduos são apresentadas.

3.1.5.1 Seleção por Giro de Roleta

A seleção por giro de roleta é um dos métodos mais populares. Neste método, a seleção dos indivíduos é realizada levando em consideração o valor retornado pela função de aptidão de cada indivíduo (EIBEN; SMITH, 2003). A probabilidade p de um indivíduo i ser selecionado é definida pela Equação 3.1.

$$p_i = \frac{f_i}{\sum_{j=1}^{\mu} f_j} \quad (3.1)$$

em que f_i é a função de aptidão aplicada ao indivíduo i e μ o total de indivíduos. A Figura 6 é uma ilustração desse método de seleção. Nesta Figura, cinco indivíduos podem ser selecionados, contudo a probabilidade de seleção não é uniforme. Por exemplo, o indivíduo 1 possui 5% de chance de ser selecionado, enquanto o indivíduo 3 possui 37% de chance. Se é desejado selecionar k indivíduos, a roleta deve girar k vezes e o indivíduo correspondente ao intervalo indicado pela roleta é selecionado.

Problemas desse tipo de seleção incluem a convergência prematura e a incapacidade de lidar com valores de aptidão negativos (GOLDBERG, 2013). A convergência prematura ocorre quando indivíduos mais aptos assumem grande parte da probabilidade de serem selecionados, afetando negativamente as chances do algoritmo explorar novas soluções.

Figura 6 – Ilustração da seleção por giro de roleta. Neste exemplo, há cinco indivíduos, em que, cada um deles possui uma probabilidade de ser selecionado.

Fonte – Autoria Própria.

3.1.5.2 Seleção por Ranking

A seleção por *ranking* é um método criado para combater a convergência prematura inerente de métodos baseados na função de aptidão, como a seleção por giro de roleta (KOZA, 1997). Este método proporciona uma diversidade maior, portanto costuma ser mais efetivo na busca por soluções (MITCHELL, 1998). Neste método de seleção, os indivíduos são ordenados e numerados com base no valor de aptidão. Posteriormente, essa ordem é utilizada na definição das probabilidades dos indivíduos serem selecionados.

3.1.5.3 Seleção por Torneio

O torneio é um método de seleção simples, de baixa complexidade e de fácil implementação em ambientes paralelos. Neste método, k indivíduos são selecionados aleatoriamente segundo alguma distribuição pré-definida (geralmente uniforme) com reposição para participar do torneio. Em seguida, o indivíduo de maior aptidão é selecionado como vencedor do torneio e escolhido para reprodução ou sobrevivência (YU; GEN, 2010). É possível também, utilizar uma distribuição de probabilidades para seleção de um dos indivíduos participantes para o torneio. O método deve ser repetido até que o número de indivíduos necessários para reprodução seja atingido.

A Figura 7 é uma ilustração de um exemplo da aplicação da seleção por meio de um torneio. Neste exemplo, há uma população de seis indivíduos em sua forma codificada (vetor binário). Três indivíduos são selecionados de maneira uniforme para participar do torneio. O indivíduo mais apto é declarado vencedor do torneio e selecionado para reprodução ou sobrevivência. Neste caso, o indivíduo 3 foi selecionado como o mais apto, pois seu valor de aptidão é superior aos demais participantes do torneio.

Figura 7 – Ilustração da seleção por torneio. Neste exemplo, há uma população de seis indivíduos e um torneio de $k = 3$ participantes é realizado. Primeiro, três indivíduos são selecionados para o torneio de maneira uniformemente aleatória. Em seguida, o indivíduo de maior aptidão é indicado como o vencedor do torneio.

Fonte – Autoria Própria.

3.1.6 Operadores Genéticos

Os operadores genéticos são responsáveis pela produção de novos indivíduos. Os operadores genéticos mais utilizados são o cruzamento e a mutação. O operador de mutação atua na exploração de novas regiões do espaço de busca, enquanto o operador de cruzamento atua na troca de material genético entre indivíduos. Nas subseções seguintes, detalhes sobre os operadores de cruzamento e mutação são apresentados.

3.1.6.1 Cruzamento

O cruzamento opera em função de dois ou mais indivíduos que são selecionados para trocar informações entre si e, assim, gerar descendentes que terão chances de fazer parte da próxima geração (SIMON, 2013).

A forma mais simples de cruzamento é denominada cruzamento de um ponto. Nesse caso, um ponto no genótipo dos indivíduos é escolhido e os *bits* desse ponto em diante são trocados com os *bits* de outro indivíduo. Esse método tem a tendência de manter pequenos blocos de genes, enquanto os maiores tem grande probabilidade de serem destruídos (ESHELMAN; CARUANA; SCHAFFER, 1989).

O cruzamento de dois pontos utiliza dois pontos no genótipo e todos os genes entre esses pontos são trocados com o outro indivíduo selecionado. Uma vantagem desse tipo de cruzamento é que a probabilidade de manter e propagar blocos de genes interessantes é maior (MITCHELL, 1998).

Outra forma de cruzamento é o cruzamento uniforme. Neste método, genes de um indivíduo são substituídos por genes de outro indivíduo na mesma posição com probabilidade p . Este método se assemelha à mutação e pode ser altamente destrutivo à blocos de genes

interessantes (MITCHELL, 1998).

O cruzamento também pode ser aplicado diretamente em indivíduos representados por uma árvore. Quando indivíduos estão codificados nesse formato, o cruzamento entre dois indivíduos geralmente é realizado por um tipo de cruzamento denominado cruzamento de subárvores (KOZA, 1997). Esse operador naturalmente produz dois descendentes que representam soluções válidas para o problema. Essa propriedade é interessante pelo fato de evitar computações desnecessárias (ABID; ABDELMALEK; HAMIDA, 2012).

Figura 8 – Cruzamento de subárvores. Dois progenitores trocam parte de sua estrutura e geram dois filhos. Os nós indicados para cruzamento estão marcados em cinza.

Fonte – Adaptado de Abid, Abdelmalek e Hamida (2012)

A Figura 8, é um exemplo de cruzamento de subárvores entre dois indivíduos, denominados Progenitor₁ e Progenitor₂. Um dos nós internos de cada um dos progenitores é escolhido aleatoriamente segundo alguma distribuição como pontos de cruzamento. Em seguida, ocorre a troca de subárvore entre ambos indivíduos nos pontos indicados, gerando assim, dois novos indivíduos. Esses indivíduos foram rotulados como Filho₁ e Filho₂. A subárvore circulada em cada um dos filhos representa, parte da estrutura que foi recebida de um outro progenitor. Por exemplo, a subárvore circulada no Filho₂, representa a parte da estrutura que foi recebida do Progenitor₁.

3.1.6.2 Mutação

Na mutação, genes do próprio indivíduo são alterados com objetivo de alcançar regiões do espaço de busca que ainda não foram exploradas. O maior benefício da mutação é proporcionar a diversidade da população. Assim como no cruzamento, a mutação pode ser realizada de diversas maneiras. A forma mais comum consiste na alteração de um gene aleatoriamente. Geralmente, para cada gene do indivíduo um número é gerado aleatoriamente, se esse número for menor que um valor t denominado taxa de mutação, o gene sofre uma modificação também aleatória. Outra forma de realizar a mutação consiste na troca de genes de duas posições diferentes.

Uma das formas de realizar a mutação em indivíduos representados por uma árvore, consiste em escolher um dos nós de maneira aleatória, segundo alguma distribuição e em seguida iniciar a construção de uma subárvore aleatória.

Figura 9 – Mutação de subárvores. O nó marcado com fundo cinza é selecionado para mutação de maneira aleatória. A subárvore que esse nó representa é substituída por outra gerada também de maneira aleatória.

Fonte – Adaptado de [Abid, Abdelmalek e Hamida \(2012\)](#)

Por exemplo, na Figura 9, um nó (em cinza) é selecionado para mutação. Uma nova árvore é gerada aleatoriamente de acordo com algum algoritmo, como o PI Grow. Essa nova árvore é inserida na posição do nó anteriormente selecionado para mutação, atuando assim como uma subárvore do indivíduo. Por fim, uma nova árvore resulta desse processo.

3.1.7 Condição ou Critério de Parada

Um critério de parada ou convergência deve ser estabelecido para que o algoritmo evolutivo finalize sua execução. Uma vez que atingir o ótimo global pode ser custoso ou mesmo impossível, critérios que trazem a certeza de serem satisfeitos são utilizados ([EIBEN; SMITH, 2003](#)). Um método tradicional consiste em estabelecer um número fixo de gerações (iterações) do algoritmo. Outro, define o término da execução quando se passaram g gerações sem melhora. Enfim, diversos métodos podem ser utilizados. Se há um conhecimento prévio do valor ótimo para o problema, este pode ser definido como critério de parada, senão outro critério deve ser estabelecido.

3.2 Evolução Gramatical

A Evolução Gramatical (EG) é um Algoritmo Genético (AG) capaz de evoluir funções ou programas de computador. A EG usa uma gramática formal livre de contexto no processo de mapeamento de genótipo para fenótipo (FENTON *et al.*, 2017). Essa gramática é, geralmente, escrita na forma de Backus Naur (BNF), deve respeitar as restrições do problema e dita as regras de produção de novos indivíduos. O algoritmo de Programação Genética (PG) tradicional proposto por Koza (1997) é capaz de evoluir e retornar funções ou programas de computador. Contudo, grande parte das PGs são capazes de evoluir apenas programas em linguagem LISP. A EG possui um sistema que permite a evolução de programas em qualquer linguagem. Na literatura, é possível encontrar artigos científicos que demonstram que a Evolução Gramatical e Programação Genética são úteis na construção de programas e funções matemáticas (SOUSA; LORENA; BASGALUPP, 2017; MIQUILINI *et al.*, 2017; BARMPALEXIS *et al.*, 2011; VANNESCHI *et al.*, 2012).

Na EG, indivíduos admitem tamanho variável e são mapeados de genótipos para fenótipos através do uso de uma gramática livre de contexto pré-definida. O restante do processo de otimização se trata de aplicação de um AG. Gramáticas livre de contexto são definidas como uma 4-tupla (V, Σ, R, S) , tal que (1) V é um conjunto finito de variáveis; (2) Σ é um conjunto finito e disjunto de V chamado de terminais; (3) R é um conjunto finito de regras e (4) $S \in V$ é a variável de início (SIPSER, 2006). Um exemplo de uma gramática G na forma BNF é dada a seguir: $\Sigma = \{X, +, -, /, *\}$, $V = \{<expr>, <op>, <pre_op>, <var>\}$, $S = <expr>$. O conjunto de regras é definido conforme ilustrado na Figura 10. É importante observar que a gramática possui um conjunto de regras que podem ser divididos em subconjuntos. Por exemplo, a variável $<expr>$ dispõe de 4 opções que podem ser escolhidas para substituí-la.

Figura 10 – Exemplo de um conjunto de regras de uma gramática livre de contexto.

(1) $<expr>$	$::= <expr><op><expr>$ (A)
	$(<expr><op><expr>)$ (B)
	$<pre_op> (<expr>)$ (C)
	$<var>;$ (D)
(2) $<op>$	$::= +$ (A)
	$-$ (B)
	$/$ (C)
	$*$; (D)
(3) $<pre_op>$	$::= +$ (A)
	$-;$ (B)
(4) $<var>$	$::= X;$ (A)

Fonte – Adaptado de Painter (2006).

A partir da gramática G , é possível realizar o mapeamento genótipo-fenótipo de um

indivíduo qualquer. Por exemplo, considere o indivíduo representado na Figura 11.

Figura 11 – Exemplo de um indivíduo codificado em genótipo.

220	203	20	3	109	215	104	220
-----	-----	----	---	-----	-----	-----	-----

Fonte – Adaptado de Painter (2006).

O processo de mapeamento inicia-se com a variável de início S , que possui apenas a regra $\langle \text{expr} \rangle$. Essa regra, possui 4 opções de expansão. O primeiro gene não utilizado, da esquerda para a direita, do genótipo representado pela Figura 11 é utilizado na escolha de uma das 4 opções disponíveis. Para isso, é feito $220 \bmod 4$ e é obtido o resultado 0. Observe que o operador $\bmod$, representa o resto da divisão de 220 por 4, nesse caso. Portanto, a primeira regra $\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$ deve ser escolhida. Em seguida, um símbolo não-terminal é escolhido ($\langle \text{expr} \rangle$ nesse caso) e o ciclo se repete. A Figura 12 é uma ilustração do processo de mapeamento genótipo-fenótipo do indivíduo representado na Figura 11. Inicialmente, a variável de início é escolhida e o primeiro símbolo não-terminal é selecionado para substituição. Ao longo das iterações, o primeiro gene do indivíduo é utilizado para decisão do símbolo que irá substituir o símbolo não-terminal selecionado. É importante destacar que, quando a regra possui apenas uma opção para substituição, não é necessário ler um valor de gene. Esse fenômeno ocorre com a regra $\langle \text{var} \rangle$. Sempre que essa regra aparece, é substituída por X , porque essa é a única opção disponível na gramática pré-definida na Figura 10.

Figura 12 – Processo de mapeamento genótipo-fenótipo do indivíduo representado na Figura 11.

1	220	203	20	3	109	215	104	220	Escolher a variável de início	$\langle \text{expr} \rangle$
2	220	203	20	3	109	215	104	220	$220 \bmod 4 = 0$	$\langle \text{expr} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$
3	220	203	20	3	109	215	104	220	$203 \bmod 4 = 3$	$\langle \text{var} \rangle \langle \text{op} \rangle \langle \text{expr} \rangle$
4	220	203	20	3	109	215	104	220	$20 \bmod 4 = 0$	$X \langle \text{op} \rangle \langle \text{expr} \rangle$
5	220	203	20	3	109	215	104	220	$3 \bmod 4 = 3$	$X + \langle \text{expr} \rangle$

Resultado: $X+X$

Fonte – Adaptado de Painter (2006).

Enquanto a distinção genótipo-fenótipo de um indivíduo pode não ficar muito clara em AGs, em EGs esse processo ganha destaque. De fato, o sucesso dos EGs depende de uma gramática bem definida (O'NEILL; RYAN, 2001). Além disso, não é difícil perceber que, os mesmos operadores genéticos tradicionais utilizados pelos AGs pode ser utilizado diretamente sobre os indivíduos de EGs em sua forma codificada (genótipos).

Uma outra forma de representação de indivíduos em EGs é chamada de Árvore de Derivação (AD) (FENTON *et al.*, 2017). ADs são geradas implicitamente durante o processo de mapeamento genótipo-fenótipo, evitam a produção de indivíduos inválidos e dispensam o uso do genótipo. Nesse caso, o mapeamento começa com a variável de início e , a cada etapa, uma opção para substituição do símbolo não-terminal é realizada de maneira aleatória.

Figura 13 – Processo de mapeamento genótipo-fenótipo do indivíduo representado na Figura 11 a partir de uma Árvore de Derivação.

A Figura 13 representa um exemplo de derivação do fenótipo $X + X$. Inicialmente, a variável de início $\langle \text{expr} \rangle$ é utilizada (S na raiz da árvore). Em seguida, uma das opções de substituição da regra é escolhida aleatoriamente (de acordo com alguma distribuição, geralmente Uniforme). Esse processo é repetido até que uma possível solução seja produzida. É interessante observar que, um genótipo linear como o da Figura 11 pode ser gerado a partir da árvore de derivação ilustrado na Figura 13.

3.3 Considerações Finais

Um dos algoritmos capazes de lidar com problemas tão complexos quando a construção de novas medidas de interesse são os algoritmos evolutivos. Na literatura é possível encontrar diversos algoritmos da classe dos evolutivos. Neste trabalho, é dado destaque para a Evolução Gramatical pela sua capacidade de lidar com problemas tão complexos quanto evoluir funções de proximidade para construção de redes de termos. Algoritmos de Evolução Gramatical exigem a definição de uma gramática livre de contexto que engloba as restrições do problema e é utilizada no mapeamento genótipo-fenótipo. No Capítulo 4, os conceitos apresentados neste capítulo são

utilizados para o detalhamento da abordagem proposta que incluem a definição da gramática livre de contexto, do método de inicialização de indivíduos, dos parâmetros do EG, do critério de parada e dos operadores genéticos.

ABORDAGEM PROPOSTA: O *FRAMEWORK* GE-TCTN

A classificação de documentos por meio de algoritmos semissupervisionados transdutivos é interessante pelo fato de não requerer a rotulação de uma grande quantidade de exemplos rotulados. O *framework* denominado *Transductive Classification through Terms Network* (TCTN) proposto por [Rossi, Rezende e Lopes \(2015\)](#) é capaz de classificar documentos por meio de uma rede de termos construída com auxílio de uma função de proximidade pré-definida. Essa função de proximidade é utilizada para quantificar as relações entre objetos da rede (termos). Essa relação entre objetos na rede de termos, pode ser vista como a medida da proximidade entre termos. Entretanto, o significado de proximidade é dependente de contexto (conjunto de dados), de modo que, funções ou medidas de proximidade de propósito geral, podem não trazer essa noção de proximidade entre objetos no contexto em que estão sendo utilizadas ([BELLET; HABRARD; SEBBAN, 2015](#)). Em [Rossi, Rezende e Lopes \(2015\)](#), cinco funções de proximidade foram utilizadas e diversos experimentos foram realizados. Os autores perceberam que funções de proximidade diferentes podem afetar o resultado da classificação, contudo a escolha por uma função de proximidade pode não ser uma tarefa trivial. Além disso, funções de proximidade de propósito geral, podem não generalizar bem, de modo que, uma função de proximidade pode prover um bom resultado de classificação para uma coleção, mas ruim para outra coleção de documentos. Diante disso, a construção automática de funções de proximidade para classificação de documentos, por meio de um algoritmo transdutivo torna-se uma ideia interessante, embora seja um desafio.

A Evolução Gramatical (EG) (apresentada no Capítulo 3) é um Algoritmo Genético (AG) capaz de lidar com problemas tão complexos, quanto ao de construção automática de funções e programas completos em qualquer linguagem. Na literatura, é possível encontrar diversos trabalhos, em que os autores utilizaram esse algoritmo para evoluir funções ([RYAN; O'NEILL; COLLINS, 1998](#); [TSOULOS; GAVRILIS; GLAVAS, 2008](#); [SOUSA; LORENA;](#)

BASGALUPP, 2017; ASSUNÇÃO *et al.*, 2017b; ASSUNÇÃO *et al.*, 2017a; MIQUILINI *et al.*, 2017). A ideia para solução do problema da construção automática de funções de proximidade para classificação de documentos consiste no uso de um EG em uma amostra da coleção de documentos. Ao longo das gerações a EG irá evoluir funções (indivíduos), e ao final do processo, a melhor função é retornada. A hipótese é que essa função pode ser utilizada para aumentar a performance de classificação do *framework* TCTN. O *framework* proposto neste trabalho foi nomeado como *Grammatical Evolution for automatically design interestingness measures for Transductive Classification through Term Networks* (GE-TCTN).

Este capítulo está dividido da seguinte forma: na Seção 4.1 uma visão geral da abordagem proposta é apresentada. Na Seção 4.2 detalhes sobre a gramática livre de contexto do EG são discutidos. A Seção 4.3 contém a abordagem para construção da rede de termos e população inicial. A Seção 4.4 contém informações sobre o mecanismo de classificação transdutivo utilizado no GE-TCTN. Na Seção 4.5 o método para avaliação de resultados é apresentado. A Seção 4.6 contém uma descrição das técnicas utilizadas para evolução dos indivíduos e a Seção 4.7 contém as considerações finais deste Capítulo.

4.1 GE-TCTN: Visão Geral

Neste trabalho, é proposto, implementado e avaliado um *framework* de Evolução Gramatical capaz de construir automaticamente funções de proximidade para construção de redes de termos realizada pelo *framework* TCTN. A abordagem proposta é dividida em 5 etapas: (1) Definição da Gramática, (2) População Inicial e Construção da Rede, (3) Classificação Transdutiva Por Meio de Redes de Termos, (4) Avaliação dos Resultados e (5) Processo Evolutivo. Na Figura 14 é apresentada uma ilustração da visão geral da abordagem proposta e o fluxo de informação entre as etapas. Os detalhes de cada uma das etapas serão apresentados nas próximas seções.

A EG precisa de uma gramática livre de contexto para realizar o mapeamento genótipo-fenótipo de indivíduos. Assim, uma gramática precisa ser definida. A construção da gramática é realizada manualmente. Contudo, uma vez definida, a gramática é capaz de produzir diversas e diferentes funções. Na definição da gramática, houve-se a preocupação de permitir com que as funções de proximidade utilizadas em Rossi, Rezende e Lopes (2015) também pudessem ser construídas. É esperado que o processo evolutivo seja capaz de gerar novas funções de proximidade que sejam promissoras na construção das redes de termos. É importante observar que a gramática está presente tanto na geração da população inicial como no processo evolutivo, uma vez que o mapeamento fenótipo-genótipo e validação dos indivíduos precisam ser realizados.

Cada indivíduo é mapeado em uma função de proximidade que, por sua vez, é utilizada na construção das redes de termos. Observe na Figura 14 que SM_k refere-se a medida de proximidade mapeada a partir do indivíduo k e essa medida é utilizada para quantificar a proximidade entre dois termos t_i e t_j . Em seguida, o *framework* TCTN é utilizado para realizar a

Figura 14 – Visão geral da abordagem proposta.

classificação transdutiva de documentos em uma amostra da coleção de textos ou documentos. Então, os resultados de classificação são validados utilizando alguma medida de performance de classificação, como Acurácia, *Recall* ou F^1 -ponderada. A partir dos resultados da validação, é possível verificar se alguma das funções de proximidade foi capaz de atuar na produção de resultados de classificação satisfatórios. Essa medida é então utilizada na classificação de todos os documentos da coleção.

Neste trabalho, um limite de g de gerações será utilizado para garantir que o algoritmo irá terminar e o melhor indivíduo será retornado. Enquanto o limite de gerações não for atingido, indivíduos são selecionados probabilisticamente para sofrerem ação dos operadores evolutivos, uma nova população é gerada e o processo se repete sobre essa nova população.

Ao longo das próximas seções, detalhes de cada etapa do GE-TCTN são apresentados.

4.2 Definição da Gramática - 1ª Etapa

A primeira etapa do projeto consiste na definição de uma gramática livre de contexto. A gramática é utilizada no mapeamento fenótipo-genótipo da Evolução Gramatical. Por meio dela, é possível fornecer informações sobre o problema a ser resolvido a fim de apoiar o processo de busca por soluções satisfatórias.

Na literatura, é possível encontrar diversas funções de proximidade que podem ser utilizadas no algoritmo TCTN (ROSSI, 2015). Entretanto, funções de proximidade tradicionais não generalizam bem e influenciam no resultado da classificação (BELLET; HABRARD; SEBBAN, 2015; ROSSI; REZENDE; LOPES, 2015). No trabalho de Rossi, Rezende e Lopes (2015), as medidas Suporte, Q de Yule, Informação Mútua, Kappa e Piatetsky-Shapiro foram utilizadas

como funções de proximidade para a realização da classificação de documentos (ou textos). A gramática livre de contexto foi definida manualmente. Contudo, uma única gramática pode ser utilizada em diferentes coleções de textos se for capaz de produzir muitas funções diferentes. Além disso, houve-se a preocupação em definir uma gramática capaz de gerar as funções de proximidade supracitadas, uma combinação delas ou novas funções.

Como explicado no Capítulo 3, uma gramática livre de contexto é definida como uma 4-tupla (V, Σ, R, S) , tal que (1) V é um conjunto finito de variáveis; (2) Σ é um conjunto finito e disjunto de V chamado de terminais; (3) R é um conjunto finito de regras e (4) $S \in V$ é a variável de início (SIPSER, 2006). Neste trabalho, cada um dos itens dessa 4-tupla é definido da seguinte forma:

- $S = \langle \text{start} \rangle$
- $V = \{ \langle \text{sc} \rangle, \langle \text{op1} \rangle, \langle \text{op2} \rangle, \langle \text{n} \rangle, \langle \text{t1} \rangle, \langle \text{t2} \rangle \}$
- $\Sigma = \{ +, -, /, *, ^, \sin, \text{sqrt}, \tanh, \text{sqrt}, \ln, \log_2, p(t_i), p(\neg t_i), p(t_j), p(\neg t_j), p(t_i, t_j), p(t_i, \neg t_j), p(\neg t_i, t_j), p(\neg t_i, \neg t_j), 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$.

As relações entre termos na rede de termos utilizada pelo TCTN é estabelecida de acordo com uma função de proximidade que, por sua vez, utiliza as probabilidades de co-ocorrência entre termos da coleção de textos ou documentos. Os símbolos $p(t_i)$, $p(\neg t_i)$, $p(t_j)$, $p(\neg t_j)$, $p(t_i, t_j)$, $p(t_i, \neg t_j)$, $p(\neg t_i, t_j)$ e $p(\neg t_i, \neg t_j)$ representam esses valores e são detalhados a seguir.

- $p(t_i)$: probabilidade de ocorrência do termo t_i
- $p(\neg t_i)$: probabilidade de não ocorrência do termo t_i
- $p(t_j)$: probabilidade de ocorrência do termo t_j
- $p(\neg t_j)$: probabilidade de não ocorrência do termo t_j
- $p(t_i, t_j)$: probabilidade de co-ocorrência entre os termos t_i e t_j ;
- $p(t_i, \neg t_j)$: probabilidade de um termo t_i não co-ocorrer com um termo t_j ;
- $p(\neg t_i, t_j)$: probabilidade de um termo t_j não co-ocorrer com um termo t_i ;
- $p(\neg t_i, \neg t_j)$: probabilidade dos termos t_i e t_j não co-ocorrerem.

As operações matemáticas $+$, $-$, $/$, $*$ e $^$, são operações básicas da aritmética, soma, subtração, divisão, multiplicação e potência, respectivamente. Em seguida temos, $\sin$, $\tanh$, sqrt , $\ln$ e $\log_2$, que são as funções matemáticas seno, tangente hiperbólica, raiz quadrada, logaritmo natural e logaritmo na base 2, respectivamente. O conjunto de regras de produção é definido conforme representado pela Figura 15.

Figura 15 – Regras de produção da gramática proposta.

$\langle \text{start} \rangle$	$::=$	$\langle \text{sc} \rangle$	(0)
$\langle \text{sc} \rangle$	$::=$	$((\langle \text{sc} \rangle) \langle \text{op1} \rangle (\langle \text{sc} \rangle))$	(0)
	$\parallel$	$((\langle \text{sc} \rangle) \langle \text{op1} \rangle (\langle n \rangle \langle n \rangle . \langle n \rangle \langle n \rangle))$	(1)
	$\parallel$	$((\langle n \rangle \langle n \rangle . \langle n \rangle \langle n \rangle) \langle \text{op1} \rangle \langle \text{sc} \rangle)$	(2)
	$\parallel$	$\langle \text{op2} \rangle (\langle \text{sc} \rangle)$	(3)
	$\parallel$	$(\langle t2 \rangle * \langle t2 \rangle)$	(4)
	$\parallel$	$((\langle t2 \rangle * \langle t2 \rangle) \langle \text{op1} \rangle (\langle \text{sc} \rangle))$	(5)
	$\parallel$	$\langle t1 \rangle$	(6)
$\langle \text{op1} \rangle$	$::=$	$+$	(0)
	$\parallel$	$-$	(1)
	$\parallel$	$*$	(2)
	$\parallel$	$/$	(3)
	$\parallel$	$\wedge$	(4)
$\langle \text{op2} \rangle$	$::=$	$\sin$	(0)
	$\parallel$	$\tanh$	(1)
	$\parallel$	sqrt	(2)
	$\parallel$	$\ln$	(3)
	$\parallel$	$\log_2$	(4)
$\langle t1 \rangle$	$::=$	$p(t_i, t_j)$	(0)
	$\parallel$	$p(t_i, \neg t_j)$	(1)
	$\parallel$	$p(\neg t_i, t_j)$	(2)
	$\parallel$	$p(\neg t_i, \neg t_j)$	(3)
$\langle t2 \rangle$	$::=$	$p(t_i)$	(0)
	$\parallel$	$p(\neg t_i)$	(1)
	$\parallel$	$p(t_j)$	(2)
	$\parallel$	$p(\neg t_j)$	(3)
$\langle n \rangle$	$::=$	0	(0)
	$\parallel$	1	(1)
	$\parallel$	2	(2)
	$\parallel$	3	(3)
	$\parallel$	$\dots$	(4)
	$\parallel$	9	(9)

Figura 16 – Exemplo de produção da função de proximidade Piatetsky-Shapiro por meio da gramática proposta neste trabalho.

$\langle \text{start} \rangle \Rightarrow \langle \text{sc} \rangle$	(0)
$\Rightarrow ((\langle \text{sc} \rangle) \langle \text{op1} \rangle (\langle \text{sc} \rangle))$	(0)
$\Rightarrow ((\langle \text{t1} \rangle) \langle \text{op1} \rangle (\langle \text{sc} \rangle))$	(6)
$\Rightarrow ((p(t_i, t_j)) \langle \text{op1} \rangle (\langle \text{sc} \rangle))$	(0)
$\Rightarrow ((p(t_i, t_j)) - (\langle \text{sc} \rangle))$	(1)
$\Rightarrow ((p(t_i, t_j)) - ((\langle \text{t2} \rangle * \langle \text{t2} \rangle)))$	(4)
$\Rightarrow ((p(t_i, t_j)) - (((p(t_i)) * \langle \text{t2} \rangle)))$	(0)
$\Rightarrow ((p(t_i, t_j)) - (((p(t_i)) * p(t_j))))$	(2)

Fonte: Autoria Própria.

É importante ressaltar que, a gramática proposta é capaz de produzir as funções de proximidade utilizadas por Rossi, Rezende e Lopes (2015), bem como novas funções. Por exemplo, a medida Piatetsky-Shapiro, pode ser gerada escolhendo se as regras de produção na seguinte ordem: 0,0,6,0,1,4,0,2. A Figura 16 é uma ilustração do processo de produção dessa função a partir da gramática supracitada. O processo inicia-se pela variável de início $\langle \text{start} \rangle$ e segue o processo de substituição de variáveis da esquerda para a direita de acordo com a regra escolhida para a substituição.

4.3 População Inicial e Construção da Rede - 2ª Etapa

A população inicial pode ser gerada de maneira aleatória segundo alguma distribuição, ou de acordo com algum algoritmo mais sofisticado como *Ramped Half-and-Half* (RHH) (RYAN; AZAD, 2003) ou *Position Independent Grow* (PI Grow) (FAGAN; FENTON; O'NEILL, 2016). Neste trabalho, a técnica PI Grow foi utilizada por se tratar do estado da arte na geração de populações para o algoritmo de Gramática Evolutiva. Os indivíduos não possuem tamanho fixo e são mapeados em funções de proximidade com base na gramática definida na etapa anterior. As medidas são importantes porque são parâmetros para algoritmos de construção de rede.

A função de proximidade não é só importante, como necessária construção da rede de termos. Para o estabelecimento das conexões entre termos pode-se utilizar uma das diversas metodologias presentes na literatura (ZHU; LAFFERTY; ROSENFELD, 2005), como rede totalmente conectada, k -vizinhos mais próximos mútuos, k -vizinhos mais próximos simétricos e vizinhança- ϵ . As técnicas de k -vizinhos permitem limitar o grau de conexões de um nó, enquanto na vizinhança- ϵ esse tipo de controle não é possível.

4.4 Classificação Transdutiva Por Meio de Redes de Termos - 3ª Etapa

O algoritmo TCTN utiliza a informação da relação entre termos para classificação de documentos. Essa informação de relação entre termos, é obtida após a construção de uma rede de termos na primeira etapa do TCTN, conforme apresentado no Capítulo 2. A rede de termos utilizada pelo TCTN consiste de um grafo em que objetos são os termos e as relações entre os termos são as proximidades entre eles. Neste trabalho, a função de proximidade é construída com base em uma gramática livre de contexto e refinada no processo evolutivo do algoritmo de Evolução Gramatical proposto. Cada indivíduo representa uma nova função de proximidade. Indivíduos passam pelo processo evolutivo na forma codificada em genótipos e são utilizados e avaliados em sua forma decodificada em fenótipos. Em resumo, cada indivíduo (nova função) é fornecido como entrada para o TCTN, a rede de termos é construída e a classificação é realizada em seguida. Essa classificação é realizada em uma amostra da coleção de textos.

4.5 Avaliação dos Resultados - 4ª Etapa

Com o objetivo de identificar boas funções de proximidade, os resultados produzidos pelo algoritmo TCTN na etapa anterior precisam ser avaliados. No contexto dos algoritmos evolutivos, essa etapa refere-se à função de aptidão. Nesse caso, pode ser utilizada como função de aptidão qualquer medida capaz de quantificar a qualidade de classificação como a Precisão, Revocação, Acurácia, Micro- F^1 , Macro- F^1 , F^1 -ponderada.

Quando uma métrica é utilizada para avaliação de um classificador em um problema de múltiplas classes, geralmente a média das avaliações é considerada. A medida F^1 corresponde à média harmônica da Precisão e *Recall*, entretanto na F^1 -ponderada a Precisão e *Recall* são ponderadas pela proporção de exemplos em cada classe. As Equações 4.1, 4.3 e 4.2 representam as medidas F^1 -ponderada, Precisão e *Recall*, para uma classe c_i , respectivamente. Em cada equação, $num(c_i)$ representa o número de exemplos da classe c_i e os símbolos VP_{c_i} , FN_{c_i} and FP_{c_i} significam verdadeiro positivo, falso negativo e falso positivo da classe c_i , respectivamente (SEBASTIANI, 2002b).

$$F^1\text{-ponderada} = 2 \times \frac{\text{Precisão} \times \text{Recall}}{\text{Precisão} + \text{Recall}} \quad (4.1)$$

$$\text{Recall}_{c_i} = \frac{num(c_i)}{\sum_{c_j \in C} num(c_j)} \times \frac{VP_{c_i}}{VP_{c_i} + FN_{c_i}} \quad (4.2)$$

$$\text{Precisão}_{c_i} = \frac{num(c_i)}{\sum_{c_j \in C} num(c_j)} \times \frac{VP_{c_i}}{VP_{c_i} + FP_{c_i}} \quad (4.3)$$

4.6 Processo Evolutivo - 5ª Etapa

Como discutido no Capítulo 3, diversos critérios de parada podem ser definidos para a Evolução Gramatical. Neste projeto, um número fixo de g gerações é definido e ao final do processo o melhor indivíduo será retornado como resposta para o problema. Enquanto o critério de parada do algoritmo não é satisfeito, os indivíduos da população atual são selecionados por meio de um método de seleção para serem submetidos aos operadores evolutivos: reprodução e mutação. Neste trabalho, o método de seleção Torneio foi escolhido, por ser um método de simples implementação e de complexidade de tempo baixa. Em cada geração, uma nova população é gerada e utilizada para substituir a população antiga (método de substituição geracional).

Sobre os operadores evolutivos, a literatura contém diversas opções que poderiam ser utilizadas neste trabalho. Contudo, há uma preferência pelos operadores de cruzamento e mutação de subárvores porque esses operadores produzem apenas indivíduos válidos. Esses operadores foram discutidos no Capítulo 3.

4.7 Considerações Finais

O *framework* GE-TCTN consiste em diversas etapas que foram detalhadas na Seção 4.1. Em Rossi, Rezende e Lopes (2015), uma rede de termos é utilizada para a classificação de documentos. A função de proximidade escolhida para quantificar relações entre termos influencia no resultado da classificação e funções de proximidade da literatura podem não apresentar bons resultados. Além disso, a qualidade dos resultados pode variar entre coleções de documentos diferentes. Portanto, construir automaticamente uma função de proximidade para cada conjunto de documentos é desejado. Enquanto construir manualmente funções de proximidade que levam a resultados de classificação satisfatórios ou melhores do que os já obtidos por meio de funções de proximidade de propósito geral pode ser uma tarefa custosa, o GE-TCTN tem o objetivo de encontrar tais medidas automaticamente por meio de um algoritmo de Evolução Gramatical aplicado em uma amostra da coleção de textos. Ao final do processo, a medida encontrada pode ser utilizada na classificação de toda a coleção de textos. No próximo capítulo, os resultados da classificação do GE-TCTN em 37 coleções de dados textuais são apresentados e comparados estatisticamente aos resultados obtidos pelos algoritmos EM, LLGC com rede de documentos e os resultados do *framework* TCTN com Busca em Grade para definição dos hiperparâmetros.

AVALIAÇÃO EXPERIMENTAL

Para analisar o *framework* GE-TCTN, seus resultados são comparados com resultados de outros algoritmos da literatura, incluindo o TCTN (ROSSI; REZENDE; LOPES, 2015), utilizando a busca em Grade (Grid-TCTN) para a otimização de hiperparâmetros. A abordagem proposta também faz otimização de hiperparâmetros, em que se destaca a construção automática da função de proximidade utilizada na construção de uma rede de termos para classificação transdutiva conforme explicado no Capítulo 4.

O restante deste capítulo está organizado em três seções. A Seção 5.1 contém uma descrição das coleções de documentos junto de suas características, dos hiperparâmetros utilizados no GE-TCTN e dos hiperparâmetros utilizados nos algoritmos Grid-TCTN, EM e LLGC que foram comparados a abordagem proposta. Na Seção 5.2, os resultados obtidos pelos algoritmos são apresentados e comparados por meio de análise estatística. Finalmente, a Seção 5.3, contém as considerações finais sobre os resultados obtidos.

5.1 Configuração Experimental

O *framework* GE-TCTN foi implementado em Python 3¹ a partir do módulo PonyGE2 (FENTON *et al.*, 2017). Da mesma forma, o *framework* TCTN e a técnica de Busca em Grade foram implementados na linguagem Python 3. Todos experimentos foram realizados em um computador equipado com GNU/Linux, 64GB de memória ram e 32 processadores.

No total, trinta e sete coleções de documentos reais foram utilizadas na avaliação do GE-TCTN. As coleções de documentos pertencem a diferentes domínios como: E-mails (EM), documentos médicos (DM), artigos de notícias (AN), documentos científicos (DC), análise de sentimentos (AS), documentos de conferências de recuperação de informações (TR) e páginas da *web* (PW) (ROSSI; MARCACINI; REZENDE, 2013). A Tabela 5 contém um resumo das

¹ Site oficial da linguagem Python: <<https://www.python.org>>

características das coleções de documentos consideradas nessa dissertação. As colunas dessa tabela são: nome da coleção, domínio, número de documentos ($|\mathcal{D}|$), número de termos ($|\mathcal{T}|$), número médio de termos por classe ($\overline{|\mathcal{T}|}$), número de classes ($|\mathcal{C}|$), desvio padrão considerando a porcentagem de documentos pertencentes a cada classe da coleção ($\sigma(\mathcal{C})$) e porcentagem de documentos pertencentes a classe majoritária ($\max(\mathcal{C})$).

Em Rossi, Rezende e Lopes (2015) foi demonstrado que a aplicação de uma seleção de atributos é possível sem muito efeito na qualidade dos resultados. Neste projeto, 10% dos termos mais bem classificados de acordo com a soma do TF-IDF (SALTON, 1989) foram selecionados. Esse procedimento também acelera o processo de classificação, uma vez que, a variável que mais influencia o tempo de execução do TCTN é o número de termos (ROSSI *et al.*, 2012; ROSSI, 2015).

Parte dos algoritmos de classificação semissupervisionados transdutivo de documentos requer a definição de alguns hiperparâmetros antes da execução da tarefa de predição de rótulos. É comum que a escolha destes hiperparâmetros seja uma tarefa não-trivial porque frequentemente os valores que levarão aos melhores resultados são não intuitivos. Portanto, neste trabalho foi utilizada uma técnica de otimização denominada Busca em Grade para determinação desses hiperparâmetros. Esse procedimento, escolhe os melhores hiperparâmetros de conjuntos de possíveis hiperparâmetros pré-definidos. Para cada coleção, a otimização de hiperparâmetros foi realizada em uma amostra de 20% dos documentos (10% de documentos rotulados e 10% de documentos para validação, sendo que a quantidade de documentos é a mesma para cada classe e os documentos são selecionados de maneira aleatória). O restante dos documentos, 80%, são considerados como documentos não rotulados. O conjunto de valores de hiperparâmetros que produzir o melhor resultado de classificação de uma média de 10 execuções do algoritmo (em 10 amostras diferentes) é escolhido para então ser utilizado na classificação dos documentos não rotulados. Neste trabalho, a medida F^1 -ponderada, apresentada no Capítulo 4 foi escolhida para quantificar a performance de classificação de cada algoritmo. No GE-TCTN, a função de proximidade (MI) é construída automaticamente pela Evolução Gramatical, sendo que, a EG é executada por 10 vezes para cada combinação dos hiperparâmetros k e α . Os algoritmos Grid-TCTN e LLGC seguem a mesma abordagem para otimização de seus hiperparâmetros. O algoritmo EM foi aplicado com suavização de Laplace e não foi necessária a otimização de hiperparâmetros, nesse caso, apenas 10% dos documentos rotulados e 80% dos documentos não rotulados foram utilizados. As configurações das abordagens GE-TCTN, Grid-TCTN, LLGC e EM são resumidas a seguir:

- **GE-TCTN:** as configurações da evolução gramatical são: tamanho da população = 100, número de gerações = 100, probabilidade de cruzamento = 75%, probabilidade de mutação = 10%, tamanho torneio = 5, elitismo = 2. E a Busca em Grade considera $k = \{1, 7, 17, 37, 57\}$ e $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

Tabela 5 – Resumo das características das coleções de documentos utilizadas nos experimentos: $|\mathcal{D}|$ - número de documentos; $|\mathcal{T}|$ - número de termos; $|\mathcal{C}|$ - número de classes; $\sigma(\mathcal{C})$ - desvio padrão considerando a porcentagem das classes; $\max(\mathcal{C})$ - porcentagem da classe majoritária.

Coleção	Domínio	$ \mathcal{D} $	$ \mathcal{T} $	$ \overline{\mathcal{T}} $	$ \mathcal{C} $	$\sigma(\mathcal{C})$	$\max(\mathcal{C})$
20ng	EM	18808	45434	76.47	20	0.52	5.31
CSTR	DC	299	1726	54.27	4	18.89	42.81
Classic4	DC	7095	7749	35.28	4	13.70	45.16
Dmoz-Computers-500	PW	9500	5011	10.83	19	0.00	5.26
Dmoz-Health-500	PW	6500	4217	12.40	13	0.00	7.69
Dmoz-Science-500	PW	6000	4821	11.52	12	0.00	9.63
Dmoz-Sports	PW	13500	5682	11.87	27	0.00	3.70
FBIS	AN	2463	2001	159.24	17	5.66	26.54
Hitech	AN	2301	12942	141.93	6	8.25	26.21
IrishEconomicSentiment	AS	1660	8659	112.65	3	6.83	39.46
La1s	AN	3204	13196	144.64	6	8.22	29.43
La2s	AN	3075	12433	144.83	6	8.59	29.43
LATimes	AN	6279	10020	42.19	6	8.38	29.43
New3s	AN	9558	26833	234.53	44	1.32	7.28
NFS	DC	10524	3888	6.56	16	3.82	13.39
Oh0	MD	1003	3183	52.50	10	5.33	19.34
Oh5	MD	918	3013	54.43	10	3.72	16.23
Oh10	MD	1050	3239	55.64	10	4.25	15.71
Oh15	MD	913	3101	59.30	10	4.27	17.20
Ohscal	MD	11162	11466	60.38	10	2.66	14.52
Ohsumed-400	MD	9200	13512	55.15	12	0.00	4.35
Opinosis	AS	6457	2693	7.56	51	1.42	8.18
Re0	AN	1504	2887	51.73	13	11.56	40.43
Re1	AN	1657	3759	52.70	25	5.54	22.39
Re8	AN	7674	8901	35.31	8	18.24	51.12
Reuters-21578	AN	8723	14035	81.72	120	5.75	45.29
Reviews	AN	4069	22927	183.10	5	12.80	34.11
SyskillWebert	PW	334	4340	93.16	4	10.75	41.02
Tr11	TD	414	6430	281.66	9	9.80	31.88
Tr12	TD	313	5805	273.60	8	7.98	29.71
Tr21	TD	336	7903	469.86	6	25.88	68.75
Tr23	TD	204	5833	385.29	6	15.58	44.61
Tr31	TD	927	10129	268.50	7	13.37	37.97
Tr45	TD	690	8262	280.58	10	6.69	23.19
WAP	PW	1560	8461	141.33	20	5.20	21.86
WebACE	PW	3900	8881	43.15	21	8.44	35.74
WebKB	PW	8282	22892	89.78	7	15.19	45.45

- **Grid-TCTN**: os conjuntos de valores para cada hiperparâmetro são $k = \{1, 7, 17, 37, 57\}$, $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $MI = \{(1) \text{ Piatetsky Shapiro}, (2) \text{ Mutual Information}, (3) \text{ Kappa}, (4) \text{ Yule's Q and } (5) \text{ Support}\}$.
- **LLGC**: os conjuntos de valores para cada hiperparâmetro do LLGC são $k = \{1, 7, 17, 37, 57\}$, $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $MI = \{(1) \text{ Piatetsky Shapiro}, (2) \text{ Mutual Information}, (3) \text{ Kappa}, (4) \text{ Yule's Q and } (5) \text{ Support}\}$.
- **EM**: os conjuntos de valores para cada hiperparâmetro do EM são: suavização de Laplace $= \alpha = \{1.0\}$, componente por classe $= \{1\}$.

Para avaliar a relevância estatística dos resultados, o teste de Friedman $1 \times N$ com pós-teste de Li (LI, 2008; GARCÍA *et al.*, 2010) e 95% de confiança foi aplicado com auxílio da ferramenta KEEL² (ALCALÁ-FDEZ *et al.*, 2009). O teste de Friedman $1 \times N$ é não paramétrico, baseado em *ranking* e aconselhado para a comparação de resultados de um classificador (Top-1 no *ranking* de Friedman) com resultados de N outros classificadores.

O teste de Friedman admite duas hipóteses: H_0 , em que não há diferença estatística entre os resultados dos algoritmos ou H_1 , em que há diferença estatística entre os algoritmos. Para isso, um *ranking* é gerado de acordo com os resultados de cada coleção de documentos e a média dos *ranking* é utilizada para o cálculo da probabilidade dos resultados serem provenientes de uma mesma distribuição (p -valor). Esse valor pode ser interpretado como a probabilidade de observação de duas amostras coletadas de uma população que segue a mesma distribuição. Se o p -valor for menor do que um nível de significância (α), então a hipótese nula (H_0) pode ser rejeitada. Neste projeto foi considerado $\alpha = 0.05$.

O pós-teste de Li (LI, 2008) é um procedimento que pode ser realizado após a rejeição da hipótese nula do teste de Friedman. Esse teste permite realizar a comparação $1 \times N$ entre o algoritmo mais bem ranqueado no *ranking* médio do teste de Friedman e os demais algoritmos. Em Li (2008), é proposta uma abordagem de rejeição da hipótese nula (H_0) em duas etapas. Considere k o número de classificadores sendo comparados e i o índice dos p -valores após serem ordenados em ordem crescente, então:

1ª Etapa Rejeite todas as H_i hipóteses se $p_{k-1} \leq \alpha$, caso contrário aceite H_k e prossiga para a segunda etapa.

2ª Etapa Rejeite todas as hipóteses restantes com $p_i \leq (1 - p_{k-1}) / (1 - \alpha) \alpha$.

Alternativamente α pode ser diretamente comparado aos valores de p -ajustados segundo Li (2008). O ajuste dos p -valores é realizado da seguinte forma: $p_{Li} = p_i / (p_i + 1 - p_{k-1})$. Assim, quando $p_{Li} \leq \alpha$, a hipótese nula é rejeitada.

² <<http://www.keel.es/>>

5.2 Resultados

Neste capítulo, os resultados dos experimentos realizados são apresentados e discutidos. A Tabela 6 contém os resultados da classificação de documentos considerando a medida F^1 —ponderada explicado no Capítulo 4. A primeira coluna contém o nome das coleções de documentos consideradas, a segunda coluna os resultados do Grid-TCTN, a terceira coluna contém os resultados do algoritmo *Expectation Maximization* (EM), a quarta coluna contém os resultados do algoritmo *Learning with Local and Global Consistency* (LLGC), a quinta coluna é o melhor indivíduo encontrado em 10 execuções do GE-TCTN e a sexta coluna é a média seguida do desvio padrão da aptidão indivíduos em 10 execuções do GE-TCTN. Sublinhados estão os melhores resultados entre todos classificadores e em negrito estão os resultados do GE-TCTN que superaram os resultados do Grid-TCTN. É importante pontuar que, a segunda (Grid-TCTN) e a quinta coluna (GE-TCTN) representam a média de 10 execuções do TCTN considerando 10 amostras diferentes de 10% de documentos rotulados aleatoriamente em cada execução. Os desvios-padrão de cada resultado estão na Tabela 10 do Apêndice A.

Os resultados da Tabela 6 indicam que o GE-TCTN é capaz de construir automaticamente funções de proximidade que são úteis na tarefa de classificação de documentos por meio do TCTN. De um total de trinta e sete coleções de documentos, os resultados do GE-TCTN foram superiores aos resultados do Grid-TCTN em trinta e cinco, ou seja, em aproximadamente 94.59% das coleções. O Grid-TCTN escolhe uma função de proximidade de um conjunto de possíveis medidas (5 nesse caso) e o GE-TCTN constrói automaticamente a função de proximidade utilizada pelo TCTN. Apenas em 2 casos o GE-TCTN foi inferior ao Grid-TCTN, contudo os resultados ainda são bem similares, com uma diferença máxima de 0.0015 na coleção NFS. Observa-se também, que em 29 coleções, a média das 10 execuções do GE-TCTN também foi superior ao Grid-TCTN o que indica a consistência da abordagem proposta. As maiores diferenças absolutas entre resultados ocorrem nas coleções Tr31, Tr11, New3s e Tr45 enquanto as menores ocorrem nas coleções La2s, NFS, Dmoz-Science e La1s.

Em uma comparação geral, com todos os algoritmos (Grid-TCTN, EM e LLGC), o GE-TCTN é superior em 17 coleções, sendo que o GE-TCTN (avg.) é também superior em 15 coleções, o LLGC é superior em 8 coleções, o EM apresenta resultados superiores aos demais algoritmos em 12 coleções. Quando o domínio das coleções é considerado, pode-se observar que o GE-TCTN apresenta uma melhor performance de classificação em coleções de artigos de notícias (AN) e os piores resultados em páginas da *web* (PW) quando comparado a todos os outros algoritmos. Quando comparado apenas ao Grid-TCTN, o GE-TCTN apresenta os melhores resultados em coleções de documentos de conferências de recuperação de informações (TD) e piores resultados em páginas da Web (PW).

O teste estatístico de Friedman indica a superioridade dos resultados do GE-TCTN quando comparado aos demais algoritmos no *ranking* médio do teste de Friedman (apresentado na Tabela 7) e também a diferença estatística entre as abordagens, pois o p -valor obtido no teste

Tabela 6 – Resultados dos experimentos realizados. A primeira coluna contém o nome de cada coleção de documentos. A segunda, os resultados do Grid-TCTN, a terceira coluna contém os resultados do EM, a quarta coluna contém os resultados do algoritmo LLGC, a quinta coluna é o melhor indivíduo encontrado em 10 execuções do GE-TCTN e a última coluna contém a média e desvio padrão (em parênteses) de 10 execuções do GE-TCTN.

Coleções	Grid-TCTN	EM	LLGC	GE-TCTN	GE-TCTN (avg)
20ng	0.7313	<u>0.8167</u>	0.7259	0.7438	0.7411 (0.0022)
Classic4	0.7317	<u>0.9353</u>	0.8955	0.7961	0.7649 (0.0221)
CSTR	0.7443	<u>0.7947</u>	0.7069	0.7760	0.7467 (0.0110)
Dmoz-Computers	0.5812	0.5806	0.5720	0.5836	0.5828 (0.0004)
Dmoz-Health	0.7085	<u>0.7227</u>	0.6747	0.7117	0.7072 (0.0074)
Dmoz-Science	0.5904	0.5606	<u>0.7397</u>	0.5898	0.5893 (0.0005)
Dmoz-Sports	0.7886	0.6888	0.7199	0.7969	0.7924 (0.0030)
FBIS	0.4778	0.5529	<u>0.6016</u>	0.5081	0.4965 (0.0070)
Hitech	0.6929	0.6314	0.6494	0.6989	0.6933 (0.0035)
IrishEconomicSentiment	0.5711	0.5786	<u>0.6208</u>	0.6119	0.5830 (0.0181)
La1s	0.8080	<u>0.8153</u>	0.7758	0.8082	0.7942 (0.0159)
La2s	0.8218	0.8161	0.7881	0.8240	0.8219 (0.0010)
LATimes	0.7744	0.7785	0.7430	0.7802	0.7786 (0.0018)
New3s	0.2847	0.6229	<u>0.6998</u>	0.4455	0.3977 (0.0589)
NFS	0.6844	<u>0.6949</u>	0.6378	0.6829	0.6819 (0.0005)
Oh0	0.7603	<u>0.8127</u>	0.6767	0.7647	0.7577 (0.0044)
Oh10	0.7181	<u>0.7478</u>	0.6365	0.7256	0.7203 (0.0037)
Oh15	0.7194	<u>0.7772</u>	0.6343	0.7233	0.7170 (0.0062)
Oh5	0.7722	0.7784	0.6755	0.7866	0.7817 (0.0065)
Ohscal	0.7392	0.6503	0.6333	0.7489	0.7446 (0.0037)
Ohsumed-400	0.4263	0.3611	0.1779	0.4305	0.4284 (0.0014)
Opinions	0.5360	<u>0.5652</u>	0.5388	0.5604	0.5594 (0.0010)
Re0	0.6683	0.6830	<u>0.7062</u>	0.6733	0.6654 (0.0066)
Re1	0.6224	0.5344	0.4819	0.6247	0.6145 (0.0104)
Re8	0.9023	0.8587	0.8980	0.9114	0.9089 (0.0025)
Reuters-21578	0.8023	0.7984	0.8082	0.8237	0.8112 (0.0070)
Reviews	0.9070	0.8364	0.8893	0.9178	0.9103 (0.0089)
Syskillwebert	0.8224	<u>0.9259</u>	0.9178	0.8378	0.8161 (0.0084)
Tr11	0.2306	0.6540	<u>0.7610</u>	0.3983	0.2664 (0.0604)
Tr12	0.4023	0.5684	<u>0.7013</u>	0.4993	0.4391 (0.0300)
Tr21	0.7036	0.5284	0.3269	0.7793	0.7067 (0.0323)
Tr23	0.4327	0.4773	0.5093	0.5305	0.5112 (0.0246)
Tr31	0.6651	0.8406	0.8755	0.8880	0.8767 (0.0246)
Tr45	0.3896	0.6157	<u>0.6546</u>	0.5296	0.4426 (0.0334)
Wap	0.5806	<u>0.7536</u>	0.7068	0.6003	0.5843 (0.0074)
WebACE	0.8808	0.7557	0.8373	0.8858	0.8831 (0.0020)
WebKB	0.5312	0.4300	0.3634	0.5629	0.5340 (0.0143)

é de 0.000374, com $\alpha = 0.05$ e distribuição qui-quadrado com 3 graus de liberdade estatísticos.

Tabela 7 – *Ranking* médio dos algoritmos (Teste de Friedman)

Algoritmo	<i>Ranking</i>
GE-TCTN	1.8378
EM	2.3243
LLGC	2.8649
Grid-TCTN	2.9730

O pós-teste de Li também é aplicado pois a hipótese nula no teste de Friedman foi rejeitada (uma vez que $p\text{-valor} < \alpha$). A Tabela 8 contém os p -valores não ajustados (segunda coluna) e p -valores ajustados segundo o procedimento de Li (p_{Li}). Os resultados dessa tabela indicam a diferença estatística do GE-TCTN para as abordagens Grid-TCTN e LLGC.

Tabela 8 – p -valores não ajustados e ajustados segundo o procedimento de Li (Teste de Friedman)

i	Algoritmos	p -valores não ajustados	p_{Li}
1	Grid-TCTN	0.000156	0.000174
2	LLGC	0.000622	0.000695
3	EM	0.105058	0.105058

A Tabela 9 contém alguns exemplos de hiperparâmetros e funções de proximidade construídas pelo GE-TCTN. É possível observar que as medidas possuem aspectos não-triviais e não-intuitivos como a divisão pelo número real 25.32 ou o uso de expoentes como 41.30 e 71.03. Se há interesse na escolha por uma função de proximidade da literatura ou na construção manual de uma função de proximidade, essas características não intuitivas podem ser analisadas posteriormente para auxílio na tomada de decisões. Por exemplo, os termos $p(\neg t_i, \neg t_j)$ e $p(t_i, t_j)$, ocorrem 79 e 66 vezes, respectivamente, entre as melhores funções de proximidade retornada pelo GE-TCTN. Grande parte das funções é uma combinação das probabilidades de co-ocorrência, mas para algumas coleções de documentos, apenas o termo $p(\neg t_i, \neg t_j)$ é utilizado. Se há interesse em criar novas funções de proximidade, os termos encontrados na função retornada pelo GE-TCTN pode ser um indicativo da relevância dos mesmos na especificação de proximidade no contexto em questão.

Tabela 9 – Exemplos de hiperparâmetros e funções de proximidade construídas pelo GE-TCTN.

Coleção	k	alpha	Função de Proximidade
Dmoz-Computers	1	0.1	$p(t_i, t_j) + (p(\neg t_i, t_j) * (p(\neg t_i, \neg t_j)^{41.30}))$
New3s	1	0.9	$p(t_i, t_j)^{(p(\neg t_i, t_j) - p(t_i, t_j) - p(t_i, \neg t_j) - (p(t_i, t_j) / (p(\neg t_i, \neg t_j) / 25.32)))}$
SyskillWebert	17	0.5	$p(t_i, t_j) / (p(\neg t_i, t_j) + p(\neg t_i, \neg t_j)^{29.62})$
Tr11	37	0.9	$p(\neg t_i, \neg t_j)^{(p(\neg t_i, \neg t_j) / (p(\neg t_i, \neg t_j)^{59.45}))}$
Wap	7	0.7	$p(t_i, t_j)^{(p(\neg t_i, \neg t_j) / (p(\neg t_i, \neg t_j) * p(\neg t_i, \neg t_j)^{71.03}))}$

5.3 Considerações Finais

Nesta Seção, foram apresentados os resultados dos experimentos com Grid-TCTN, LLGC, EM e GE-TCTN. Os resultados indicam que o GE-TCTN é capaz de produzir resultados superiores do que o Grid-TCTN que utiliza apenas funções de proximidade de propósito geral. As funções de proximidade produzidas pelo GE-TCTN possuem características não intuitivas e não triviais que podem ser analisadas posteriormente para auxílio da escolha ou construção manual de uma função de proximidade. Os resultados do GE-TCTN também foram comparados aos resultados de classificação dos algoritmos transdutivos LLGC, EM e Grid-TCTN. Em geral o GE-TCTN apresentou resultados competitivos quando comparado aos demais algoritmos. A relevância dos resultados também foi comparada estatisticamente. Nesse caso, o teste de Friedman indicou a superioridade do GE-TCTN no *ranking* médio do teste e o procedimento de Li diferença estatística entre as abordagens. A partir dos dados apresentados na Tabela 8, foi possível identificar que, GE-TCTN produziu resultados estatisticamente diferentes dos resultados produzidos pelos algoritmos LLGC e Grid-TCTN, contudo o mesmo não pode ser afirmado em relação ao EM. Ainda sim, é possível inferir que o GE-TCTN apresentou resultados interessantes, competitivos com algoritmos da literatura e estatisticamente superiores ao Grid-TCTN, que é o principal objetivo desse projeto.

CONCLUSÕES E TRABALHOS FUTUROS

A classificação de documentos é geralmente realizada por meio de algoritmos de aprendizado supervisionado. Um problema desses algoritmos é a necessidade de muitos documentos rotulados para aprendizado de um classificador acurado. Uma alternativa é realizar a classificação de documentos de maneira semissupervisionada transdutiva. O *framework* TCTN (ROSSI; REZENDE; LOPES, 2015) é uma abordagem competitiva, que utiliza redes de termos para classificação de documentos em que um dos hiperparâmetros da abordagem é a função de proximidade (ou medida de interesse). A escolha por uma função de proximidade pode não ser uma tarefa trivial. Além disso, a noção semântica de proximidade pode ser diferente para cada coleção de documentos. Assim, foi levantada nessa dissertação a seguinte questão de pesquisa: "É possível realizar a construção automática de funções de proximidade para o *framework* TCTN que permitem a geração de redes de termos que levam à uma melhor performance na classificação de documentos quando comparado à utilização de funções de proximidade criadas manualmente?". Para responder essa questão, foi proposto, implementado e avaliado um *framework* denominado GE-TCTN que utiliza Evolução Gramatical para construção automática de funções de proximidade a partir de uma gramática livre de contexto. Os resultados da abordagem proposta foram comparados ao *framework* TCTN com Busca em Grade (Grid-TCTN) e aos algoritmos LLGC e EM, e avaliados estatisticamente.

Os resultados indicaram que o GE-TCTN é capaz de construir funções de proximidade para cada coleção de documentos e produzir resultados de classificação superiores ou, ao menos, competitivos aos resultados obtidos com uso de funções de proximidade tradicionais no Grid-TCTN. Esses indícios são verdadeiros tanto para o melhor indivíduo, quanto para a média de 10 execuções do GE-TCTN o que também indica a consistência dos resultados. É importante ressaltar que, o GE-TCTN fica incumbido de construir e selecionar a melhor função de proximidade em cada coleção de documentos, facilitando o uso do *framework*, enquanto no Grid-TCTN deve-se estabelecer um conjunto de funções de proximidade antes da execução do algoritmo.

O GE-TCTN também foi comparado aos algoritmos transdutivos LLGC e EM. Em geral, a abordagem proposta foi superior na maioria dos resultados e apresentou diferença estatisticamente significativa segundo o teste de Friedman com pós-teste de Li dos resultados obtidos pelo Grid-TCTN e LLGC.

Principais Resultados e Contribuições

As contribuições e resultados alcançados nessa dissertação estão resumidas a seguir.

- Definição de uma gramática livre de contexto que pode ser utilizada para produção de funções de proximidade que utilizam a co-ocorrência entre termos em seus operandos.
- O *framework* GE-TCTN é a principal contribuição dessa dissertação. O GE-TCTN é capaz de construir automaticamente novas funções de proximidade com base em uma gramática livre de contexto.
- Resultados, comparações e análise da aplicação do GE-TCTN, LLGC, EM e Grid-TCTN em 37 coleções de documentos de diferentes domínios. Os resultados foram validados estatisticamente de acordo com o teste de Friedman e pós-teste de Li.

A gramática livre de contexto foi construída com base nas cinco funções de proximidade utilizadas em [Rossi, Rezende e Lopes \(2015\)](#). Contudo a mesma pode ser reutilizada ou modificada para atender a necessidades específicas. O GE-TCTN pode ser utilizado em novas coleções de documentos de diferentes domínios. Se há o interesse na construção de uma função de proximidade manualmente, os resultados do GE-TCTN podem indicar a relevância da presença de um termo na função. Esse indicativo pode ser interessante para tomada de decisão sobre incluir ou não um termo na função ou para o auxílio na escolha por uma função de proximidade da literatura. Os resultados entre diferentes abordagens pode ser útil na escolha por uma abordagem de classificação semissupervisionada.

Publicações e Submissões

As publicações e submissões de artigos científicos que resultaram dessa dissertação estão listadas a seguir.

- **COUTINHO, F. P.; ROSSI, R. G.; REZENDE, S. O.** A Grammatical Evolution Algorithm to Automatically Design Similarity Measures for Term Networks. São Carlos. *Encontro Paulista de Pós-graduandos em Computação*, 2017. (Publicado - Pôster).

- **COUTINHO, F. P.; ROSSI, R. G.; REZENDE, S. O.** Similarity Function Learning for Term Networks based on Grammatical Evolution. *III Workshop de Sistemas de Informação da SACSIS*. Rio Paranaíba. 2019. (*Publicado - Pôster*).
- **COUTINHO, F. P.; REZENDE, S. O.; ROSSI, R. G.** GA-TCTN: A Framework For Hyper-Parameter Optmization And Text Classification Using Transductive Classificaton Through Term Networks. *ABRICOM: Learning and NonLinear Models*. v. 17, n. 2, 2019. (*Aceito para publicação - Artigo de Periódico - Qualis B5*).
- **COUTINHO, F. P.; ROSSI, R. G.; REZENDE, S. O.** Grammatical Evolution for Automatically Design Similarity Functions for Transductive Classification through Term Networks. (*Em fase final de revisão - Artigo de Periódico - Qualis A1*).

Trabalhos Futuros

Como trabalhos futuros pretende-se considerar outras abordagens para otimização dos hiperparâmetros k e α ou a otimização de todos os parâmetros utilizando a Evolução Gramatical. Além disso, pretende-se realizar experimentos considerando novas gramáticas, que permitam a construção de novas funções de proximidade. Outro ponto interessante é identificação de padrões nas funções de proximidade selecionadas para cada domínio de aplicação. Ainda, é interessante a realização de novos experimentos, contudo, considerando um número menor e fixo de documentos rotulados, pois essa configuração pode ser útil em cenários que o número de documentos rotulados é pequeno.

REFERÊNCIAS

ABID, F.; ABDELMALEK, W.; HAMIDA, S. B. Dynamic hedging using generated genetic programming implied volatility models. **Genetic Programming: New Approaches and Successful Applications**, IntechOpen, p. 141–172, 2012. Citado nas páginas 46 e 47.

ABU-MOSTAFA, Y. S.; MAGDON-ISMAIL, M.; LIN, H.-T. **Learning from data**. New York: AMLBook, 2012. v. 4. 15-27 p. Citado nas páginas 16 e 21.

AGGARWAL, C. C. **Machine Learning for Text: An Introduction**. Berlim: Springer International Publishing, 2018. 1–16 p. Citado nas páginas 15, 21 e 22.

ALCALÁ-FDEZ, J.; SÁNCHEZ, L.; GARCÍA, S.; JESUS, M. J. del; VENTURA, S.; GARRELL, J. M.; OTERO, J.; ROMERO, C.; BACARDIT, J.; RIVAS, V. M.; FERNÁNDEZ, J. C.; HERRERA, F. Keel: a software tool to assess evolutionary algorithms for data mining problems. **Soft Computing**, Springer-Verlag, Berlim, v. 13, n. 3, p. 307–318, 2009. ISSN 1433-7479. Citado na página 64.

ASSUNÇÃO, F.; LOURENÇO, N.; MACHADO, P.; RIBEIRO, B. Automatic generation of neural networks with structured grammatical evolution. In: IEEE. **Congress on Evolutionary Computation (CEC)**. Spain, 2017. p. 1557–1564. Citado nas páginas 53 e 54.

_____. Towards the evolution of multi-layered neural networks: A dynamic structured grammatical evolution approach. In: ACM. **Proceedings of the Genetic and Evolutionary Computation Conference**. Berlim, 2017. p. 393–400. Citado nas páginas 53 e 54.

BARMPALEXIS, P.; KACHRIMANIS, K.; TSAKONAS, A.; GEORGARAKIS, E. Symbolic regression via g in the optimization of a controlled release pharmaceutical formulation. **Chemo-metrics and Intelligent Laboratory Systems**, Elsevier, v. 107, n. 1, p. 75–82, 2011. Citado na página 48.

BARROS, R. C.; BASGALUPP, M. P.; CERRI, R.; SILVA, T. S. da; CARVALHO, A. C. de. A grammatical evolution approach for software effort estimation. In: ACM. **Proceedings of the 15th annual conference on Genetic and evolutionary computation**. New York, 2013. p. 1413–1420. Citado na página 17.

BASGALUPP, M. P.; BARROS, R. C.; BARABASZ, T. A grammatical evolution based hyper-heuristic for the automatic design of split criteria. In: ACM. **Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation**. New York, 2014. p. 1311–1318. Citado na página 17.

BELKIN, M.; MATVEEVA, I.; NIYOGI, P. Regularization and semi-supervised learning on large graphs. In: SHAWE-TAYLOR, J.; SINGER, Y. (Ed.). **Learning Theory**. Berlim, Heidelberg: Springer Berlim Heidelberg, 2004. p. 624–638. ISBN 978-3-540-27819-1. Citado na página 33.

BELLET, A.; HABRARD, A.; SEBBAN, M. Metric learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan & Claypool Publishers, v. 9, n. 1, p. 1–151, 2015. Citado nas páginas 17, 18, 28, 29, 30, 31, 39, 53 e 55.

- BERTON, L.; FALEIROS, T. de P.; VALEJO, A.; VALVERDE-REBAZA, J.; LOPES, A. de A. Rgcli: Robust graph that considers labeled instances for semi-supervised learning. **Neurocomputing**, Elsevier, v. 226, p. 238 – 248, 2017. Citado na página [22](#).
- BERTON, L.; LOPES, A. A.; Vega-Oliveros, D. A. A comparison of graph construction methods for semi-supervised learning. In: IEEE. **International Joint Conference on Neural Networks (IJCNN)**. Rio de Janeiro, 2018. p. 1–8. Citado na página [33](#).
- BERTON, L.; LOPES, A. de A. Graph construction for semi-supervised learning. In: **IJCAI. Proceedings of the 24th International Joint Conference on Artificial Intelligence**. Buenos Aires, 2015. Citado nas páginas [16](#) e [28](#).
- BLUM, A.; CHAWLA, S. Learning from labeled and unlabeled data using graph mincuts. In: **Proceedings of the Eighteenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers, 2001. p. 19–26. Citado na página [33](#).
- BLUM, A.; MITCHELL, T. Combining labeled and unlabeled data with co-training. In: **ACM. Proceedings of the 11th Annual Conference on Computational Learning Theory**. New York, NY, USA, 1998. p. 92–100. Citado nas páginas [22](#) e [23](#).
- CHAPELLE, O.; ZIEN, A. Semi-supervised classification by low density separation. In: **AISTATS. International Conference on Artificial Intelligence and Statistics**. Barbados, 2005. p. 57–64. Citado na página [26](#).
- CHEN, M.; WEINBERGER, K. Q.; BLITZER, J. C. Co-training for domain adaptation. In: **NIPS. Proceedings of the 24th International Conference on Neural Information Processing Systems**. USA, 2011. (NIPS'11), p. 2456–2464. ISBN 978-1-61839-599-3. Citado na página [24](#).
- COLLOBERT, R.; SINZ, F.; WESTON, J.; BOTTOU, L. Large scale transductive svms. **Journal of Machine Learning Research**, v. 7, n. Aug, p. 1687–1712, 2006. Citado na página [26](#).
- Cover, T.; Hart, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, p. 21–27, 1967. Citado na página [23](#).
- COVER, T. M.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, Menlo Park, v. 13, n. 1, p. 21–27, 1967. Citado na página [17](#).
- DEMPSTER, A. P.; LAIRD, N. M.; RUBIN, D. B. Maximum likelihood from incomplete data via the em algorithm. **Journal of the Royal Statistical Society: Series B (Methodological)**, Wiley Online Library, v. 39, n. 1, p. 1–22, 1977. Citado na página [25](#).
- EIBEN, A. E.; SMITH, J. E. **Introduction to evolutionary computing**. Berlim: Springer, 2003. v. 53. Citado nas páginas [40](#), [43](#) e [47](#).
- ESHELMAN, L. J.; CARUANA, R. A.; SCHAFFER, J. D. Biases in the crossover landscape. In: **Proceedings of the 3rd International Conference on Genetic Algorithms**. San Francisco, CA, USA: Morgan Kaufmann Publishers, 1989. p. 10–19. ISBN 1-55860-006-3. Citado na página [45](#).
- FAGAN, D.; FENTON, M.; O'NEILL, M. Exploring position independent initialisation in grammatical evolution. In: IEEE. **Congress on Evolutionary Computation (CEC)**. Spain, 2016. p. 5060–5067. Citado nas páginas [42](#) e [58](#).

FENTON, M.; MCDERMOTT, J.; FAGAN, D.; FORSTENLECHNER, S.; HEMBERG, E.; O'NEILL, M. Ponyge2: Grammatical evolution in python. In: ACM. **Proceedings of the Genetic and Evolutionary Computation Conference Companion**. Berlim, 2017. p. 1194–1201. Citado nas páginas 48, 50 e 61.

GARCÍA, S.; FERNÁNDEZ, A.; LUENGO, J.; HERRERA, F. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. **Information Sciences**, Elsevier, v. 180, n. 10, p. 2044–2064, 2010. Citado na página 64.

GENG, L.; HAMILTON, H. J. Interestingness measures for data mining: A survey. **ACM Computing Surveys (CSUR)**, ACM, New York, v. 38, n. 3, p. 9, 2006. Citado nas páginas 17 e 36.

GERBER, M. S. Predicting crime using twitter and kernel density estimation. **Decision Support Systems**, Elsevier, v. 61, p. 115–125, 2014. Citado na página 15.

GOLDBERG, D. E. **The design of innovation: Lessons from and for competent genetic algorithms**. Berlim: Springer, 2013. v. 7. Citado na página 43.

GOLDBERG, D. E.; HOLLAND, J. H. Genetic algorithms and machine learning. **Machine learning**, Springer, Berlim, v. 3, n. 2, p. 95–99, 1988. Citado nas páginas 41 e 42.

GRADY, L.; FUNKA-LEA, G. Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In: SONKA, M.; KAKADIARIS, I. A.; KYBIC, J. (Ed.). **Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis**. Berlim, Heidelberg: Springer Berlim Heidelberg, 2004. p. 230–245. ISBN 978-3-540-27816-0. Citado na página 32.

HAND, D. J.; YU, K. Idiot's bayes—not so stupid after all? **International statistical review**, Wiley Online Library, v. 69, n. 3, p. 385–398, 2001. Citado nas páginas 23 e 25.

HOLLAND, J. H. Genetic algorithms. **Scientific American**, Scientific American, a division of Nature America, Inc., v. 267, n. 1, p. 66–73, 1992. Citado na página 40.

JOACHIMS, T. Transductive inference for text classification using support vector machines. In: ICML. **Proceedings of the Sixteenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers, 1999. v. 99, p. 200–209. Citado nas páginas 22, 23 e 26.

KIRITCHENKO, S.; MATWIN, S. Email classification with co-training. In: **Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research**. Toronto: IBM Press, 2001. p. 301–312. Citado nas páginas 15 e 24.

KOZA, J. R. **Genetic Programming**. Cambridge: MIT Press, 1997. Citado nas páginas 17, 40, 42, 44, 46 e 48.

KUMAR, A.; DAUMÉ, H. A co-training approach for multi-view spectral clustering. In: ICML. **Proceedings of the 28th International Conference on Machine Learning**. Washington, 2011. p. 393–400. Citado na página 24.

LAMPLE, G.; CHAPLOT, D. S. Playing fps games with deep reinforcement learning. In: AIII. **31st Conference on Artificial Intelligence**. São Francisco, 2017. Citado na página 24.

- LEVIN, A.; LISCHINSKI, D.; WEISS, Y. Colorization using optimization. In: ACM. **ACM transactions on graphics (tog)**. [S.l.], 2004. v. 23, n. 3, p. 689–694. Citado na página 33.
- LI, J. A two-step rejection procedure for testing multiple hypotheses. **Journal of Statistical Planning and Inference**, v. 138, p. 1521–1527, 2008. Citado na página 64.
- LINDEN, R. **Algoritmos genéticos (3ª edição)**. Rio de Janeiro: Ciência Moderna, 2012. Citado nas páginas 40 e 42.
- MANNING, C.; RAGHAVAN, P.; SCHÜTZE, H. Introduction to information retrieval. **Natural Language Engineering**, Cambridge University Press, v. 16, n. 1, p. 260, 2010. Citado na página 26.
- MCCARTHY, J. Recursive functions of symbolic expressions and their computation by machine. 1959. Citado na página 41.
- MCKAY, R. I.; HOAI, N. X.; WHIGHAM, P. A.; SHAN, Y.; O’NEILL, M. Grammar-based genetic programming: a survey. **Genetic Programming and Evolvable Machines**, Berlim, v. 11, n. 3-4, p. 365–396, 2010. Citado na página 17.
- MIQUILINI, P.; ROSSI, R. G.; QUILES, M. G.; MELO, V. V. de; BASGALUPP, M. P. Automatically design distance functions for graph-based semi-supervised learning. In: IEEE. **Trust-com/BigDataSE/ICSS**. Sydney, 2017. p. 933–940. Citado nas páginas 17, 31, 37, 48, 53 e 54.
- MITCHELL, M. **An introduction to genetic algorithms**. Cambridge: MIT press, 1998. Citado nas páginas 44, 45 e 46.
- NGUYEN, T. H.; SHIRAI, K.; VELCIN, J. Sentiment analysis on social media for stock movement prediction. **Expert Systems with Applications**, Elsevier, v. 42, n. 24, p. 9603–9611, 2015. Citado na página 15.
- NIGAM, K.; MCCALLUM, A. K.; THRUN, S.; MITCHELL, T. Text classification from labeled and unlabeled documents using em. **Machine Learning**, Kluwer Academic Publishers, Norwell, MA, USA, v. 39, n. 2-3, p. 103–134, maio 2000. ISSN 0885-6125. Citado nas páginas 22, 23 e 25.
- NIU, Z.-Y.; JI, D.-H.; TAN, C. L. Word sense disambiguation using label propagation based semi-supervised learning. In: ACL. **Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics**. Stroudsburg, PA, USA, 2005. p. 395–402. Citado na página 33.
- OLIVEIRA, G.; COUTINHO, F.; CAMPELLO, R. J.; NALDI, M. C. Improving k-means through distributed scalable metaheuristics. **Neurocomputing**, Elsevier, v. 246, p. 45–57, 2017. Citado na página 17.
- O’NEILL, M.; RYAN, C. Grammatical evolution. **IEEE Transactions on Evolutionary Computation**, New York City, v. 5, n. 4, p. 349–358, 2001. Citado nas páginas 17 e 49.
- PAINTER, T. **Grammatical Evolution in Python**. Bachelor’s Thesis — University of Bath - Department of Computer Science, Bath, 2006. Citado nas páginas 48 e 49.

PRASETYO, N. D.; HAUFF, C. Twitter-based election prediction in the developing world. In: **ACM. Proceedings of the 26th ACM Conference on Hypertext & Social Media**. New York, 2015. p. 149–158. Citado na página 15.

PRICE, K.; STORN, R. M.; LAMPINEN, J. A. **Differential evolution: a practical approach to global optimization**. Berlim: Springer Science & Business Media, 2006. Citado na página 40.

RAJARAJESWARI, S.; NAIK, S.; SRIKANT, S.; PRAKASH, M. S.; UDAY, P. Movie recommendation system. In: **Emerging Research in Computing, Information, Communication and Applications**. Berlim: Springer, 2019. p. 329–340. Citado na página 15.

ROSSI, R. G. **Classificação Automática de Textos por meio de Aprendizado de Máquina baseado em Redes**. 125-141 p. Tese (Doutorado) — Universidade de São Paulo - Instituto de Ciências Matemáticas e de Computação, São Carlos, 2015. Citado nas páginas 22, 24, 25, 27, 29, 34, 35, 55 e 62.

ROSSI, R. G.; FALEIROS, T. P.; LOPES, A. A.; REZENDE, S. O. Inductive model generation for text categorization using a bipartite heterogeneous network. In: **IEEE. 12th International Conference on Data Mining**. Bruxelas, 2012. p. 1086–1091. Citado na página 62.

ROSSI, R. G.; LOPES, A. A.; REZENDE, S. O. A parameter-free label propagation algorithm using bipartite heterogeneous networks for text classification. In: **ACM. Proceedings of the 29th Annual ACM Symposium on Applied Computing**. New York, 2014. Citado na página 16.

ROSSI, R. G.; MARCACINI, R. M.; REZENDE, S. O. **Benchmarking text collections for classification and clustering tasks**. São Carlos, 2013. Disponível em: <http://conteudo.icmc.usp.br/CMS/Arquivos/arquivos_enviados/BIBLIOTECA_113_RT_395.pdf>. Citado na página 61.

ROSSI, R. G.; REZENDE, S. O.; LOPES, A. de A. Term network approach for transductive classification. In: **Computational Linguistics and Intelligent Text Processing**. Berlim: Springer, 2015. Citado nas páginas 16, 17, 22, 23, 27, 28, 34, 36, 53, 54, 55, 58, 60, 61, 62, 69 e 70.

RYAN, C.; AZAD, R. M. A. Sensible initialisation in grammatical evolution. In: **GECCO. Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference**. Chicago, 2003. p. 142–145. Citado nas páginas 42 e 58.

RYAN, C.; O'NEILL, M.; COLLINS, J. Grammatical evolution: solving trigonometric identities. In: **4th International Conference on Genetic Algorithms, Optimization Problems, Fuzzy Logic, Neural Networks and Rough Sets**. Brno, Czech Republic: Technical University of Brno, Faculty of Mechanical Engineering, 1998. p. 111–119. Citado nas páginas 17, 37, 40, 53 e 54.

SALTON, G. **Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer**. Boston, MA, USA: Addison-Wesley Longman Publishing, 1989. Citado na página 62.

SEBASTIANI, F. Machine learning in automated text categorization. **ACM Computing Surveys**, ACM, New York, NY, USA, v. 34, n. 1, p. 1–47, mar. 2002. Citado na página 35.

_____. Machine learning in automated text categorization. **Computing Surveys**, ACM, v. 34, n. 1, p. 1–47, 2002. Citado na página 59.

SIMON, D. **Evolutionary optimization algorithms**. Nova Jersey: John Wiley & Sons, 2013. Citado na página 45.

SINOARA, R. A.; CAMACHO-COLLADOS, J.; ROSSI, R. G.; NAVIGLI, R.; REZENDE, S. O. Knowledge-enhanced document embeddings for text classification. **Knowledge-Based Systems**, v. 163, p. 955–971, 2019. Citado na página 28.

SIPSER, M. **Introduction to the Theory of Computation**. Boston: Course Technology, 2006. v. 2. Citado nas páginas 48 e 56.

SOUSA, A. D. M.; LORENA, A. C.; BASGALUPP, M. P. Geek: Grammatical evolution for automatically evolving kernel functions. In: IEEE. **Trustcom/BigDataSE/ICSS**. Sydney, 2017. p. 941–948. Citado nas páginas 37, 48, 53 e 54.

SUBRAMANYA, A.; TALUKDAR, P. P. Graph-based semi-supervised learning. **Synthesis Lectures on Artificial Intelligence and Machine Learning**, Morgan & Claypool Publishers, v. 8, n. 4, p. 1–125, 2014. Citado nas páginas 16, 21, 22 e 28.

TAN, P.-N.; KUMAR, V.; SRIVASTAVA, J. Selecting the right interestingness measure for association patterns. In: ACM. **Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, 2002. p. 32–41. Citado nas páginas 16, 17 e 36.

TSOULOS, I.; GAVRILIS, D.; GLAVAS, E. Neural network construction and training using grammatical evolution. **Neurocomputing**, Elsevier, v. 72, n. 1-3, p. 269–277, 2008. Citado nas páginas 53 e 54.

VANNESCHI, L.; PIROLA, Y.; MAURI, G.; TOMASSINI, M.; COLLARD, P.; VEREL, S. A study of the neutrality of boolean function landscapes in genetic programming. **Theoretical Computer Science**, Elsevier, v. 425, p. 34–57, 2012. Citado na página 48.

WAN, X. Co-training for cross-lingual sentiment classification. In: ACL. **Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1**. Stroudsburg, 2009. p. 235–243. ISBN 978-1-932432-45-9. Citado na página 24.

WONG, M. L.; LEUNG, K. S. **Data mining using grammar based genetic programming and applications**. Berlim: Springer Science & Business Media, 2006. v. 3. Citado na página 41.

XING, E. P.; JORDAN, M. I.; RUSSELL, S. J.; NG, A. Y. Distance metric learning with application to clustering with side-information. In: NIPS. **Advances in neural information processing systems**. Cambridge: MIT Press, 2003. p. 521–528. Citado nas páginas 17, 30 e 31.

YANG, L.; JIN, R. Distance metric learning: A comprehensive survey. **Michigan State University**, v. 2, n. 2, p. 4, 2006. Citado na página 30.

YAROWSKY, D. Unsupervised word sense disambiguation rivaling supervised methods. In: ACL. **33rd annual meeting of the association for computational linguistics**. Cambridge: MIT Press, 1995. p. 189–196. Citado nas páginas 22, 23 e 24.

YU, X.; GEN, M. **Introduction to Evolutionary Algorithms**. Berlim: Springer Science & Business Media, 2010. Citado na página [44](#).

ZHOU, D.; BOUSQUET, O.; LAL, T. N.; WESTON, J.; SCHÖLKOPF, B. Learning with local and global consistency. In: NIPS. **Advances in neural information processing systems**. Cambridge: MIT Press, 2004. p. 321–328. Citado nas páginas [16](#), [22](#), [28](#), [33](#), [34](#) e [35](#).

ZHU, X.; GHAHRAMANI, Z.; LAFFERTY, J. D. Semi-supervised learning using gaussian fields and harmonic functions. In: ACM. **Proceedings of the 20th International conference on Machine learning**. Washington D.C., 2003. p. 912–919. Citado nas páginas [16](#), [22](#), [28](#), [32](#) e [33](#).

ZHU, X.; GOLDBERG, A. B. Introduction to semi-supervised learning. **Synthesis lectures on artificial intelligence and machine learning**, Morgan & Claypool Publishers, v. 3, n. 1, p. 1–130, 2009. Citado nas páginas [23](#) e [24](#).

ZHU, X.; LAFFERTY, J.; ROSENFELD, R. **Semi-supervised learning with graphs**. 18 p. Tese (Doutorado) — Carnegie Mellon University, Language Technologies Institute, 2005. Citado nas páginas [33](#) e [58](#).

TABELAS DE RESULTADOS

Neste apêndice, uma tabela com os desvios-padrão dos resultados de classificação discutidos no Capítulo 5 é apresentada. Todos os algoritmos considerados nessa tese foram executados 10 vezes considerando 10% dos documentos rotulados de maneira aleatória e a mesma quantidade de exemplos por classe.

Tabela 10 – Desvios-padrão dos resultados de classificação do Grid-TCTN, EM, LLGC e GE-TCTN considerando avaliações realizadas com a medida F^1 -ponderada.

Coleções	Grid-TCTN	EM	LLGC	GE-TCTN
20ng	0.0170	0.0037	0.0018	0.0143
Classic4	0.0193	0.0020	0.0355	0.0157
CSTR	0.0336	0.0258	0.0442	0.0362
Dmoz-Computers	0.0071	0.0083	0.0019	0.0074
Dmoz-Health	0.0077	0.0053	0.0031	0.0078
Dmoz-Science	0.0076	0.0103	0.0007	0.0085
Dmoz-Sports	0.0041	0.0071	0.0011	0.0033
FBIS	0.0535	0.0339	0.0136	0.0430
Hitech	0.0327	0.0173	0.0107	0.0269
IrishEconomicSentiment	0.0344	0.0164	0.0144	0.0199
La1s	0.0174	0.0043	0.0169	0.0166
La2s	0.0181	0.0033	0.0206	0.0135
LATimes	0.0071	0.0043	0.0257	0.0064
New3s	0.1279	0.0256	0.0061	0.0816
NFS	0.0054	0.0023	0.0047	0.0049
Oh0	0.0074	0.0104	0.0121	0.0073
Oh10	0.0106	0.0096	0.0118	0.0107

Oh15	0.0149	0.0142	0.0034	0.0190
Oh5	0.0106	0.0099	0.0092	0.0086
Ohscal	0.0024	0.0138	0.0044	0.0025
Ohsumed-400	0.0066	0.0038	0.0007	0.0051
Opinosis	0.0130	0.0083	0.0012	0.0129
Re0	0.0464	0.0229	0.0204	0.0458
Re1	0.0689	0.0354	0.0068	0.0674
Re8	0.0093	0.0046	0.0113	0.0089
Reuters-21578	0.0073	0.0089	0.0105	0.0089
Reviews	0.0217	0.0043	0.0292	0.0257
Syskillwebert	0.0664	0.0198	0.0161	0.1417
Tr11	0.1164	0.1042	0.0411	0.1157
Tr12	0.2373	0.1202	0.0165	0.1866
Tr21	0.0574	0.1987	0.1261	0.0195
Tr23	0.1108	0.1991	0.0613	0.0961
Tr31	0.0847	0.1991	0.0405	0.0478
Tr45	0.2272	0.0466	0.0283	0.0000
Wap	0.0567	0.0089	0.0158	0.0363
WebACE	0.0100	0.0193	0.0034	0.0106
WebKB	0.0357	0.0121	0.0150	0.0210