

ALEJANDRO TARAFA GUZMÁN

**MODELO PARA SUMARIZAÇÃO
COMPUTACIONAL DE TEXTOS CIENTÍFICOS**

São Paulo
2017

ALEJANDRO TARAFA GUZMÁN

**MODELO PARA SUMARIZAÇÃO
COMPUTACIONAL DE TEXTOS CIENTÍFICOS**

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

São Paulo
2017

ALEJANDRO TARAFA GUZMÁN

**MODELO PARA SUMARIZAÇÃO
COMPUTACIONAL DE TEXTOS CIENTÍFICOS**

Dissertação apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Mestre em Ciências.

Área de Concentração:
Engenharia de Sistemas

Orientador:
Ademar Ferreira

São Paulo
2017

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuênciā de seu orientador.

São Paulo, _____ de _____ de _____

Assinatura do autor: _____

Assinatura do orientador: _____

Catalogação-na-publicação

Guzmán, Alejandro

Modelo para sumarização computacional de textos científicos / A. Guzmán -- versão corr. -- São Paulo, 2017.
101 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Telecomunicações e Controle.

1.Sumarização Computacional 2.Similaridade Semântica Textual
3.Processamento de Linguagem Natural I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Telecomunicações e Controle II.t.

Dedicatória

Dedico este trabalho à minha mãe, que tanto lutou e se sacrificou pela minha formação acadêmico-profissional.

AGRADECIMENTOS

Agradeço primeiramente a Deus, por ter-me acompanhado e dado forças no transcurso destes anos de árduo trabalho de pesquisa.

Ao meu orientador, Prof. Ademar Ferreira, pela imensurável paciência, dedicação e ensinamentos que tem me transferido nestes anos de pesquisa.

À minha mãe querida, Susel, por ter me apoiado sempre em todas as circunstâncias com seu amor, sacrifício, força e incondicionalidade.

Ao meu pai, Juan Enrique, por seu infinito apoio, ensinamentos e carinho.

À minha amada esposa Elizabeth por ter me acompanhado neste caminho árduo e difícil, sempre me brindando o seu apoio e amor nos momentos difíceis e de desânimo.

À minha filhinha Emily por ter-me dado ânimo e força para continuar sempre à frente.

Aos meus avós Teresa, Juan, Yolanda e Jose Miguel, por seu imensurável carinho.

Aos meus tios Daniel e Jose Miguel, por acreditar em mim, seu grande apoio e guia.

Às minhas tias Mayling, Susana e todos os meus primos pelo carinho.

Aos meus amigos por ter-me feito sentir como em casa, mesmo estando longe dela.

Em fim, para toda minha família e amigos pela, confiança e força que me foi dada por todos...

À todos meu eterno agradecimento!

*“Imagination will often carry us to
worlds that never were. But without it
we go nowhere. ”*

-- Carl Sagan

RESUMO

Neste trabalho, propõe-se um modelo para a sumarização computacional extrativa de textos de artigos técnico-científicos em inglês. A metodologia utilizada baseia-se em um módulo de avaliação de similaridade semântica textual entre sentenças, desenvolvido especialmente para integrar o modelo de sumarização. A aplicação deste módulo de similaridade à extração de sentenças é feita por intermédio do conceito de uma janela deslizante de comprimento variável, que facilita a detecção de equivalência semântica entre frases do artigo e aquelas de um léxico de frases típicas, atribuíveis a uma estrutura básica dos artigos. Os sumários obtidos em aplicações do modelo apresentam qualidade razoável e utilizável, para os efeitos de antecipar a informação contida nos artigos.

Palavras-Chave – Sumarização Computacional, Similaridade Semântica Textual, Processamento de Linguagem Natural, Artigo Científico.

ABSTRACT

In this work a model is proposed for the computational extractive summarization of scientific papers in English. Its methodology is based on a semantic textual similarity module, for the evaluation of equivalence between sentences, specially developed to integrate the summarization model. A variable width window facilitates the application of this module to detect semantic similarity between phrases in the article and those in a basic structure, assignable to the articles. Practical summaries obtained with the model show usable quality to anticipate the information found in the papers.

Keywords – Computational Summarization, Semantic Textual Similarity, Natural Language Processing, Scientific Paper.

LISTA DE FIGURAS

1	Incremento das publicações científicas segundo relatório da UNESCO [Hollanders e Soete(2010)]	15
2	Fases do processo de sumarização	17
3	Topologia/Taxonomia dos sumários	18
4	Estrutura base dos textos técnico-científicos(STT) segundo Lehman em [Lehmam(1999)]	21
5	Estrutura de tipo Taça que define os documentos científicos	26
6	Exemplo de algumas TTs, FTs e FTNs identificadas e utilizadas neste trabalho assim como a distribuição destes elementos na estrutura base dos documentos técnicos-científicos, IMRAD	29
7	Método de POS Tagging	33
8	Relações entre substantivos e entre verbos no WordNet [Miller <i>et al.</i> (1990)Miller, Beckwith, Fellbaum, Gross, e Miller]	36
9	Sistema híbrido de STS baseado no modelo proposto	41
10	Árvore sintáctica de S1	45
11	Árvore sintáctica de S2	45
12	Fases do modelo de extração de sentenças	51
13	Divisão do documento segundo sentenças extraídas.	70
14	Três primeiras sentenças da introdução	82
15	Quarta sentença da Introdução	83
16	Duas primeiras sentenças da Metodologia.	83
17	Terceira sentença da Metodologia.	84
18	Quarta sentença da Metodologia.	84
19	Quinta sentença da Metodologia.	85
20	Primeira parte da sexta sentença da Metodologia.	85

21	Segunda parte da sexta sentença da Metodologia.	85
22	Primeira sentença dos Resultados.	86
23	Segunda e terceira sentença dos Resultados.	86
24	Quarta sentença dos Resultados.	87
25	Primeira sentença das Conclusões.	87
26	Segunda sentença das Conclusões.	87

LISTA DE TABELAS

1	Tabela das categorias do Estado Retórico de Teufel e Moens tomada de [Teufel e Moens(2002)]	22
2	Exemplos de sentenças extraídas de um artigo técnico-científico, pré-processadas usando expressões regulares.	31
3	Tabela representando a relação de muitos a muitos entre synsets e conceitos	35
4	Coeficientes de Pearson resultantes do modelo de STS proposto sobre os dataset do SemEval2016	47
5	Exemplos de TTs na estrutura básica e como podem aparecer nos DTC . .	53
6	Limiares usados pelo sistema para a obtenção dos sumários curto e longo. .	58

LISTA DE ABREVIATURAS

- SAT** Sumarização Automática de Textos
- PLN** Processamento de Linguagem Natural
- RAFI** Automatic Summary Based on Indicative Fragments, em Francês o original
- SIFs** Sentence Indicator Fragments
- STT** Scientific and Technical Text Structure
- DTC** Documento Técnico e/ou Científico
- IMRAD** Introdução, Metodologia, Resultados e Discussão
- FTs** Frases típicas
- FTNs** Frases típicas negativas
- TTs** Títulos típicos
- NLTK** Natural Language Tool Kit
- POS Tagging** Part of speech tagging
- STS** Similaridade Semântica Textual
- TfIdf** Term frequency inverse document frequency
- NPs** Noun Phrases
- VPs** Verbal Phrases
- PPs** Prepositional Phrases
- PVs** Phrasal Verbs
- WUP** Wu and Palmer
- CKPD** Clustered Keywords Positional Distance
- ES** Extração de Sentenças
- VWM** Variable Window Model
- ROUGE** Recall-Oriented Understudy for Gisting Evaluation
- BLUE** Bilingual Evaluation Understudy

SUMÁRIO

1	Introdução	15
1.1	Importância dos sumários científicos	16
1.2	Definição de sumário	16
1.3	Trabalhos relacionados e estado da arte	19
1.4	Proposta e objetivos	23
2	Metodologia e elementos de teoria	25
2.1	Estrutura de artigos científicos	25
2.1.1	IMRAD	26
2.2	Critérios do Abstract	28
2.3	Sumarização por extração	30
2.4	Elementos teóricos de PLN	31
2.4.1	Pré-processamento de texto	31
2.4.2	Eliminação de StopWords	32
2.4.3	Segmentação de sentenças	32
2.4.4	Part-of-speech tagging(POS tagging)	33
2.4.5	Ontologias computacionais	34
2.4.6	WordNet	35
2.4.7	Open Multilingual WordNet	37
3	O MODELO DE ESTIMAÇÃO DE STS	38
3.1	Conceição do modelo	38
3.2	Estudos prévios	39
3.3	Visão geral do modelo de STS	40

3.4	O analisador sintático probabilístico (Chunker)	42
3.5	Módulo de similaridade semântica	42
3.6	Módulo de similaridade estatística	43
3.7	Exemplos trabalhados	43
3.8	Resultados experimentais	47
4	EXTRAÇÃO DE SENTENÇAS	48
4.1	Concepção do modelo	48
4.2	Antecipação de problemas	50
4.3	Visão general do modelo de Extração de Sentenças (ES)	50
4.4	Uso do modelo de STS	52
4.5	Alinhamento estrutural	52
4.6	O modelo de janela variável sobre STS	54
4.7	Eliminação de sentenças com FTNs	55
4.8	Extração	55
4.9	Exemplos trabalhados	56
5	LIMPEZA DO TEXTO	59
5.1	Reconhecimento e eliminação de paráfrases	59
5.2	Exemplos trabalhados	60
6	USO INTEGRADO DO MODELO DE SUMARIZAÇÃO	62
6.1	Exemplos Trabalhados	62
7	VALIDAÇÃO	68
7.1	Validação por Recall e Precision	69
7.2	Validação por Rouge	71
8	CONCLUSÕES	74

9 CONTRIBUIÇÕES	75
10 TRABALHOS FUTUROS	76
Referências	77
Apêndice A – ARTIGO SUMARIZADO PELO MODELO DE SAT PRO- POSTO	82
A.1 Sentenças da Introdução	82
A.2 Sentenças da Metodologia	83
A.3 Sentenças dos Resultados	86
A.4 Sentenças das Conclusões	87
Anexo A – Artigo usado nos exemplos trabalhados	88

1 INTRODUÇÃO

“The true sign of intelligence is not knowledge but imagination.”

-- Albert Einstein

O final da década de oitenta foi marcado pelo surgimento do computador pessoal e, acoplado com este acontecimento, nasceu, segundo o Relatório de Ciência 2010 da UNESCO [Hollanders e Soete(2010)], a chamada Era da Informação ou Era Digital, a qual é caracterizada pela dinamização dos fluxos informacionais pelo mundo. Estes termos são geralmente usados para designar os avanços tecnológicos advindos da Terceira Revolução Industrial e tiveram grande repercussão na difusão de um ciberespaço, um meio de comunicação instrumentalizado pela informática e pela Internet, assim como a introdução da WWW na mesma década.

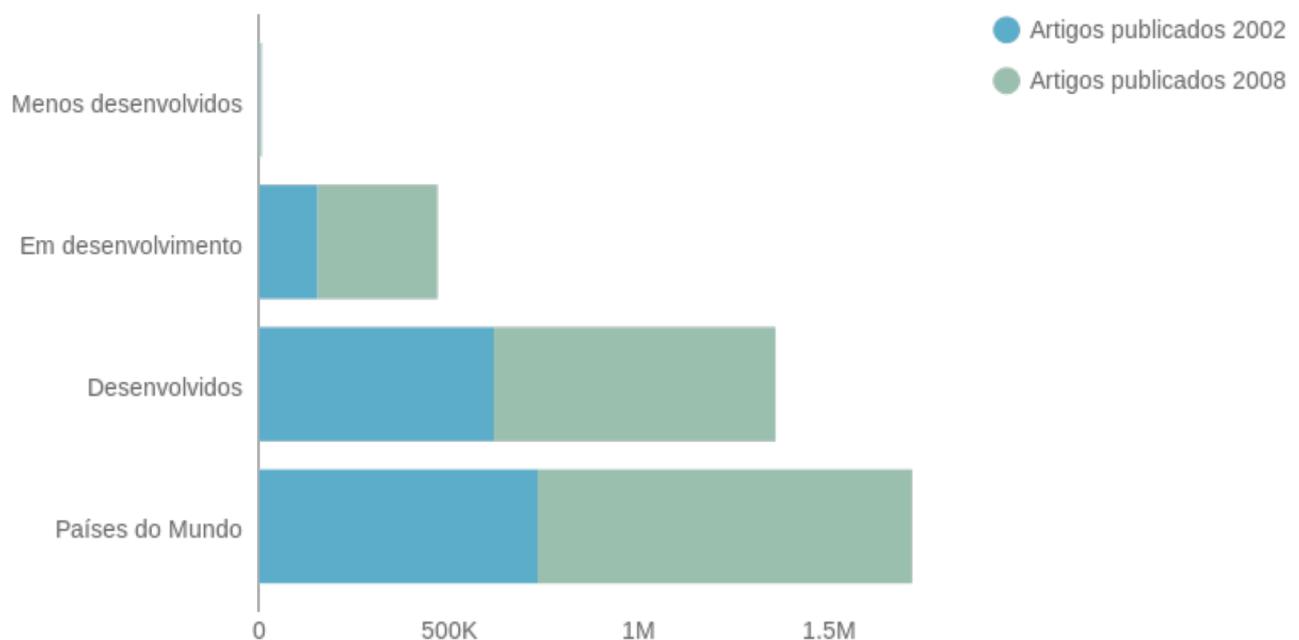


Figura 1: Incremento das publicações científicas segundo relatório da UNESCO [Hollanders e Soete(2010)]

A nomeada era digital tem sido a condução para um bem sucedido desenvolvimento econômico do mundo e, ao mesmo tempo, tem aumentado o interesse das nações do Terceiro Mundo para investir na geração de conhecimentos como um caminho para aumentar

o desenvolvimento econômico e social. Este interesse permitiu uma explosão na produção científica ilustrada na figura 1, que tem sido marcada pelo aumento da potência dos equipamentos de computação, portabilidade e acessibilidade destas tecnologias de informação, o que tem levado à criação de grandes bases de dados e de artigos científicos em particular, em formato eletrônico, como é o caso do IEEE e ACM, entre outros, que podem ser acessíveis a partir da web.

1.1 Importância dos sumários científicos

Um marco importante derivado desta revolução é o fenômeno da sobrecarga de informação científica disponível na Internet, o que dificulta o trabalho de pesquisadores devido ao acúmulo de tais informações. Assim, é preciso muito tempo na determinação da importância relativa entre um artigo e outro. Uma ferramenta amplamente utilizada de apoio para a solução deste problema tem sido a criação de sumários, os quais permitem ao leitor identificar as ideias essenciais através de uma visão sucinta do conteúdo do artigo.

No entanto a sumarização de artigos científicos pode ser um trabalho complexo e demorado para um sumarizador humano, esforços que resultam insuficientes para atingir a grande quantidade de informação científica que é gerada a cada momento no mundo. Por conta disso, torna-se essencial a existência de sistemas de sumarização automática que permitam custos mais baixos, em termos de tempo do pesquisador, assim como aumentar o grau de certeza no momento de decidir a relevância de um artigo para um determinado contexto ou investigação.

1.2 Definição de sumário

Segundo K.S Jones [Jones(1998)] um sumário é uma transformação redutiva de um texto fonte para um texto sumarizado, mediante a redução do conteúdo por seleção e/ou generalização a partir do que é importante no texto fonte. Partindo desta definição, Jones divide o processo de sumarização em três fases fundamentais como mostra a figura 2. Assume-se que estas fases são aplicadas também pelo modelo de sumarização proposto, as quais são descritas como:

Interpretação: Primeira fase do processo de sumarização, a qual realiza a interpretação da entrada (um ou mais documentos) para uma representação de base ou fonte.

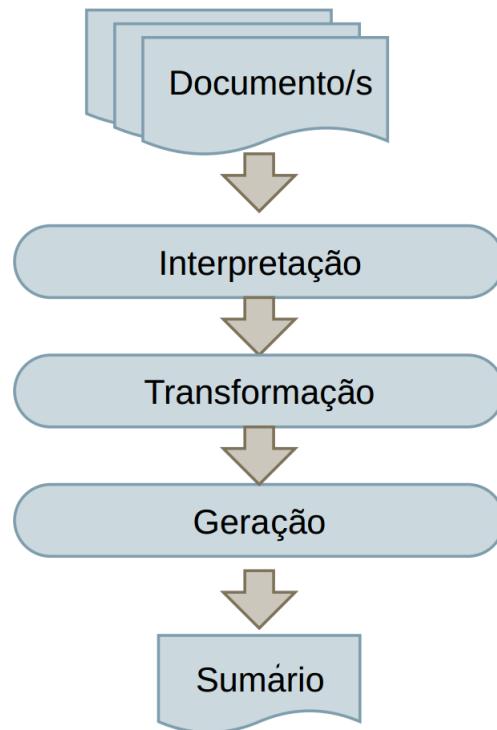


Figura 2: Fases do processo de sumarização

Transformação: A partir da representação de base da entrada, nesta fase se transforma dita representação da entrada em uma determinada representação inicial do sumário.

Geração: Nesta última fase, o texto do sumário é gerado a partir de sua representação inicial.

De igual forma, Jones em vários de seus trabalhos [Jones(1998), Jones(1993)] distingue três tipos de fatores que devem ser considerados na geração de sumários automáticos. Estes fatores, que definem a topologia ou taxonomia dos sumários representada na figura 3, são: Entrada, Propósito e Saída. Assim, aponta-se que definir estes fatores e suas variadas manifestações é uma tarefa difícil e identificá-los com suficiente precisão para guiar corretamente o processo de sumarização em cada caso particular é igualmente um desafio. Partindo desta questão, os fatores mencionados são definidos como:

Fator entrada: Reúne as propriedades relacionadas com a entrada ou fonte, estas respondem a forma ou assunto:

- Forma ou estrutura da entrada ou fonte, categorizada como explícita ou implícita.
- Escala ou tamanho define a porção do texto de entrada ou fonte, diga-se, livro, seção, parágrafo, entre outros.
- Assunto, propriedade categorizada como ordinário ou especializado, já que responde

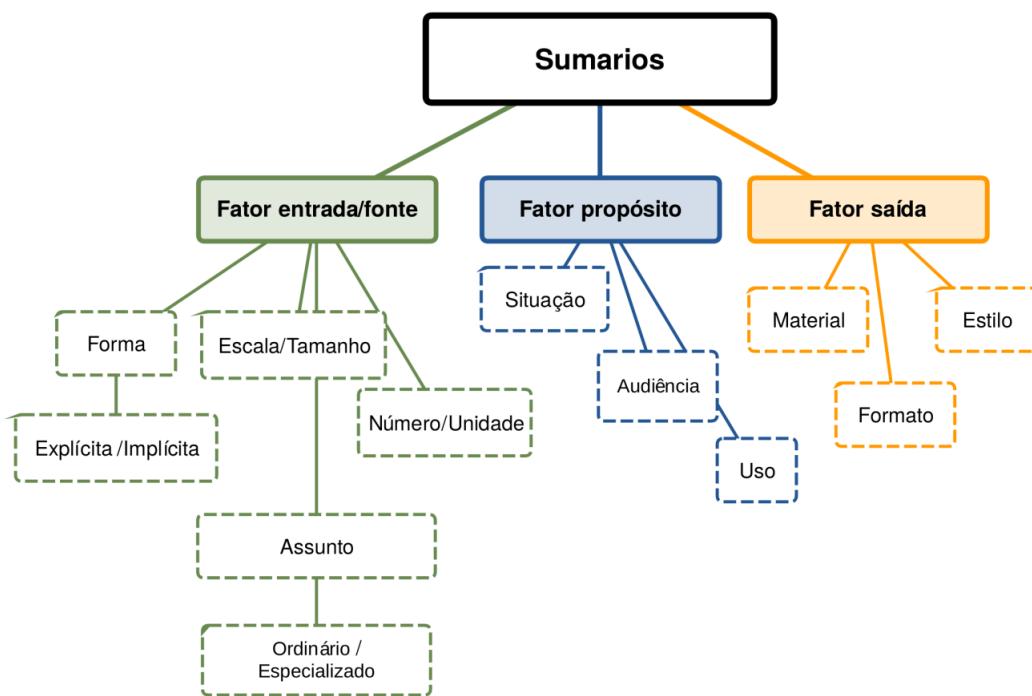


Figura 3: Topologia/Taxonomia dos sumários

ao alegado conhecimento do leitor sobre o assunto.

- Número ou Unidade, responde à quantidade de elementos de entrada no processo de sumarização.

Fator propósito: Este fator responde às seguintes perguntas: Qual é o contexto? Para quem e para quê está sendo produzido o sumário? Este fator está definido pelas propriedades abaixo:

- Contexto ou situação: define onde e sobre que contexto vai ser usado o sumário.
- Audiência: define para qual público ou leitores está sendo gerado o sumário.
- Uso: define qual será a finalidade do sumário gerado.

Fator saída: Corresponde aos critérios de formato e estilo do sumário. Este fator está definido pelas propriedades abaixo:

- Formato: define como vai ser estruturada a informação do sumário.
- Estilo: define, do ponto de vista literário, como vai ser o sumário, ou seja, indicativo, informativo, crítico, entre outros.

- Material: define os critérios de contenção ou inclusão de fonte, ou seja, quais partes ou aspectos conterá o sumário a partir da fonte.

A partir destes fatores, considera-se que o método de sumarização apresentado no presente trabalho tem como fator de entrada um documento científico explícito e altamente especializado, num contexto de ampla disponibilidade de informação científica. O propósito deste método se encontra no auxílio informacional da audiência, pesquisadores científicos, mediante um sumário informativo como saída, conformado por sentenças extraídas do texto original ordenadas de acordo com a ordem em que aparecem no texto fonte.

Além de sua topologia, os sumários são classificados como extrativos ou abstrativos. No presente trabalho, são adotados os sumários extrativos, os quais são obtidos através de uma combinação de frases selecionadas (ou extraídas) do documento fonte. No caso dos sumários abstrativos, utilizam-se palavras diferentes para representar o conteúdo do texto original.

Na Sumarização Automática de Textos (SAT), a grande maioria dos sumários são de tipo extrativo, devido a que os sumários abstrativos são de natureza muito mais complexa. Apesar disso, hoje em dia obter um bom sumário extrativo está muito longe de ser uma tarefa trivial. Existem várias questões não resolvidas atualmente para a produção de um texto final coerente, por exemplo, a limpeza final do texto do sumário para produzir coerência, além da solução do problema de anáfora.

1.3 Trabalhos relacionados e estado da arte

Originalmente, desde metade do século XX começou a se desenvolver um interesse por parte da comunidade científica pela SAT. Os primeiros trabalhos nesta área tiveram foco na elaboração de sumários indicativos mediante a extração de sentenças chave no texto. No início desta nova área do Processamento de Linguagem Natural (PLN), foi identificado que os sumários indicativos gerados certamente convergiam ao texto original.

Na mesma época Luhn [Luhn(1958)], um dos pais da SAT deu um enfoque estatístico para gerar o que ele chamou de auto-abstract. Para isto, foi usado o princípio de que frases que contêm maior número das palavras mais frequentes dentro das menos frequentes, possuem maior relevância que outras. Com base neste princípio, o modelo de Luhn, levando em conta só palavras de frequência média a baixa, ao omitir palavras de alta frequência (denominadas stop-words), caracteriza-se por realizar uma classificação com

base na frequência em que estas aparecem, no texto a sumarizar. Esta classificação servirá para definir pesos para as sentenças de acordo com o número de palavras melhor posicionadas no ranking e a proximidade entre estas (número de palavras de baixo peso que as separam) dentro da sentença. Depois, as sentenças de maior peso são extraídas para formar parte do sumário no qual sentenças de maior peso apareceram primeiro.

Posteriormente a Luhn, Edmundson em [Edmundson(1969)] definiu quatro métodos extrativos para a geração de sumários indicativos. Estes métodos, além de terem em conta a teoria de Luhn (palavras relevantes de maior frequência dentro das menos frequentes no texto ou keywords), introduz novos conceitos descritos como: Pragmatic Words (Cue Words), Title and Heading Words, e Structural Indicators (Sentence Location). Estes métodos, na definição de Edmundson, basicamente são caracterizados por atribuir pesos a determinadas características identificadas em cada sentença. Estes métodos diferem pelos tipos de características que são identificadas e a forma de ponderação das mesmas. No Cue Method [Edmundson(1969)], a relação de ponderação é associada à existência ou não nas sentenças de determinadas palavras (pragmatic words ou cue words). Estas palavras são previamente estruturadas em um dicionário de termos categorizadas em três classes: positivas, negativas ou não significativas. No Key Method, a relação de ponderação é associada com a ocorrência na sentença das palavras mais frequentes dentro das menos frequentes, que não estão definidas no dicionário de termos anteriormente mencionado. Por outro lado, no Title Method [Edmundson(1969)] a relação mencionada é dada pela existência de palavras na sentença e que além disso as mesmas apareçam no título e nos cabeçalhos do documento.

Mais tarde, Paice em [Paice(1981)] introduz um novo método de SAT, no qual faz uso de um componente definido como frase indicadora (Indicator Phrase no inglês). Estes componentes ou estruturas, segundo Paice, muito comuns na literatura técnica-científica, exemplo: **The principal aim of this paper is to investigate, In the present paper, a method is described for**, entre outros; apontam as sentenças com maior relevância no texto. Este novo método de sumarização definido por Paice consta de 4 fases, a saber:

- Identificação das sentenças indicadoras e atribuição de pesos apropriados às mesmas.
- Identificação e agregação das referências exofóricas de cada sentença indicadora para evitar incoerências no sumário.
- Extração de sentenças ou passagens com maior peso para formar o sumário
- Formatação da informação extraída mediante a remoção, substituição e transformação

de palavras em um tempo gramatical padrão ou um estilo determinado.

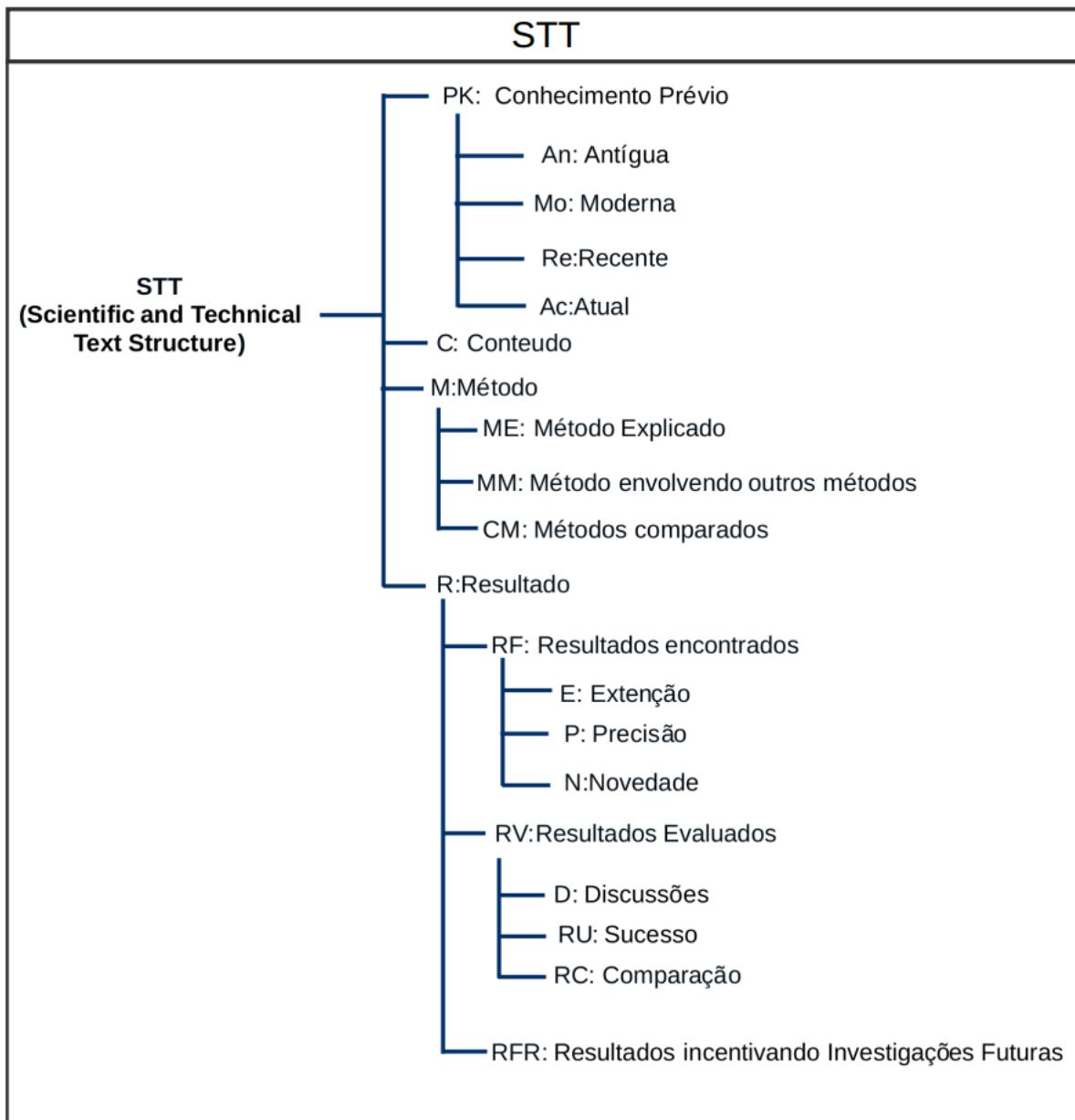


Figura 4: Estrutura base dos textos técnico-científicos(STT) segundo Lehman em [Lehmam(1999)]

Mais recentemente, Lehman definiu um modelo de SAT implementado no sistema RAFI (Automatic Summary Based on Indicative Fragments, em Francês o original), [Lehmam(1999)]. Dito modelo foi definido basicamente a partir de uma análise de discurso do ponto de vista linguístico, o que permitiu-lhe definir grupos denominados como índices linguísticos, onde encontram-se agrupados fragmentos identificadores de sentença SIFs (Sentence Indicator Fragments no inglês) tais como: **In this paper**, **This study presents**, **We used a new approach to**, entre outras; os quais segundo Lehman, identificam o conteúdo de maior relevância do texto.

Categoría	Descripción
AIM	Expressões sobre o objetivo principal da pesquisa no artigo em questão.
TEXTUAL	Expressões sobre a estrutura do artigo.
OWN	Expressões sobre o trabalho realizado no artigo em questão em relação à metodologia, resultados e conclusões.
BACKGROUND	Expressões sobre os trabalhos base da pesquisa realizada.
CONTRAST	Expressões que contrastam o trabalho em questão com outros trabalhos e destacam as debilidades dos outros.
BASIS	Expressões que concordam com outros trabalhos ou representam continuação de outros trabalhos.
OTHER	Expressões neutrais, comentários sobre outros trabalhos.

Tabela 1: Tabela das categorias do Estado Retórico de Teufel e Moens tomada de [Teufel e Moens(2002)]

De modo geral, Lehman utiliza um método de ponderação de sentenças com o objetivo de definir um critério de extração e assim selecionar as sentenças de maior peso como as mais importantes, extraindo estas para conformar um sumário indicativo. Para lograr isto, o sistema RAFI encontra-se constituído por dois métodos:

- Método locativo, onde é realizada a identificação das SIFs nas sentenças.
- Método indicativo, onde procuram-se as palavras de referência que formam parte dos SIFs, as quais são categorizadas e ponderadas em um dicionário de termos predefinido, de acordo com a distância semântica destas com o contexto onde aparece a SIF na sentença.

Lehman, através de sua análise, também identifica uma estrutura comum de índices linguísticos que define os textos técnicos-científicos denominada como STT (Scientific and Technical Text Structure em inglês). Esta estrutura consta de quatro índices fundamentais: Conhecimento Prévio, Conteúdo, Método, e Resultados, como mostra a figura 4. Segundo Lehman, o sumário extraído por este método, deve conter só sentenças com SIFs de pelo menos um dos índices definidos no STT.

Entre outros dos trabalhos relevantes mais recentes sobre a SAT encontra-se o realizado por Teufel e Moens em [Teufel e Moens(2002)], onde é apresentada uma estratégia de sumarização extrativa de artigos científicos, baseada na identificação do denominado por eles como estado retórico e das sentenças relevantes dentro deste. Segundo Teufel e Moens o estado retórico é determinado pelo contexto do artigo e encontra-se identificado por uma estrutura composta de 7 categorias descritas na tabela 1.3 as quais capturam segundo eles, importantes aspectos de argumentação e discurso, e respondem à 4 dimensões

fundamentais; estrutura do problema, atribuição intelectual, argumentação científicas, atitude sobre trabalhos de outras pessoas.

Como base de análise, foi constituído e anotado um corpus, a partir de 80 artigos multidisciplinares de Linguística Computacional em formato xml, onde determinados elementos internos (títulos, autores, abstracto, parágrafos, entre outros) encontram-se marcados e outros (equações, tabelas, imagens) foram eliminados e substituídos por um placeholder. O corpus foi anotado por 3 anotadores humanos experientes en ciências, 2 externos ao trabalho e um dos autores, procurando realizar duas tarefas; por um lado categorizar as sentenças nos artigos nas categorias que definem o estado retórico mostradas na tabela além de definir a relevância de cada sentença. A primeira tarefa foi realizada partindo de um esquema de anotação definido pelos autores, no caso da segunda, a relevância das sentenças, foi determinada partindo de uma abordagem semi-automática baseada em duas fases: a estimación do grau de similaridade entre cada sentença do corpo do artigo e cada sentença do abstract mediante una medida de similaridade de superfície e a determinación final da similaridade entre estas sentenças por um humano.

Este corpus foi usado como “gold standard”, precisamente o treinamento e teste de um classificador supervisionado para lograr identificar o estado retórico além da relevância das sentenças no texto. Dito classificador faz uso de 13 características fundamentais como: posição da sentença dentro em relação ao documento dividido em 10 seções identificadas como letras da A-J, posição relativa da sentença dentro de cada seção; posição relativa da sentença dentro de cada parágrafo, entre outras. Certamente a partir dos resultados alcançados pela comparação da aplicação deste classificador e o gold standard usando as métricas de Precision e Recall foram muito positivos.

1.4 Proposta e objetivos

A proposta deste trabalho é desenvolver uma metodologia para summarização automática extrativa de textos de artigos técnico-científicos, baseada no conceito de Similaridade Semântica Textual.

As contribuições deste trabalho são:

1. O desenvolvimento de um modelo de avaliação de similaridade semântica textual entre duas sentenças, que utiliza informação semântica e estatística por meio de um **Analisador Sintático Probabilístico(Chunker)**, de modo que a combinação seja integrada ao sistema com base em propriedades de linguagem introduzidas no

processamento de linguagem natural (PLN).

2. A aplicação de tal sistema à obtenção automática de sumários.

2 METODOLOGIA E ELEMENTOS DE TEORIA

Neste capítulo, é definida a metodologia aplicada para estabelecer as hipóteses dos modelos propostos neste trabalho, de modo a alcançar os objetivos delineados. Dita metodologia resume-se em três aspectos fundamentais, a serem detalhados nas seções à frente.

Estrutura dos artigos científicos: É feito um estudo sobre os artigos científicos com o objetivo de identificar quais características de tais documentos são fundamentais para determinar a relevância de suas partes constituintes.

Sumarização por extração: Nesta seção, são abordadas os conceitos fundamentais sobre as principais áreas da sumarização existentes (Sumarização Extrativa e Abstrativa), suas características e justificativa do tipo usado neste trabalho. Além disto, é feita uma análise sobre os principais modelos que aplicam a abordagem selecionada, permitindo assim estabelecer um caminho para definir as hipóteses sobre os modelos a serem propostos.

Elementos teóricos de PLN: Nesta seção, são localizados os problemas a serem resolvidos dentro da área de PLN. São analisados conjuntos de conceitos fundamentais que impactam diretamente nos modelos de PLN, com ênfase no pré-processamento de textos.

2.1 Estrutura de artigos científicos

Um DTC (Documento Técnico e/ou Científico) é um texto que descreve um determinado trabalho de pesquisa científico/técnico, teórico e/ou experimental, mediante a aplicação do universal método científico e onde o conteúdo do mesmo depende especificamente da área de ciência ou tecnologia a que se refere.

Existem vários tipos de DTC, entre os quais se encontram: Artigos Científicos, Relatórios Técnicos, Notas de Laboratório, Teses ou Dissertações, entre outros, sendo o primeiro tipo o mais abundante dentro da literatura científica na atualidade. Os Artigos

Científicos serão os únicos que abordaremos na presente análise.

Praticamente, os artigos científicos seguem um mesmo modelo organizativo, ou estrutura básica comum, definida pelo padrão IMRAD, acrônimo em inglês referente aos termos Introdução, Metodologia, Resultados e Discussão.

2.1.1 IMRAD

O padrão IMRAD foi constituído no ano de 1972, e posteriormente adotado como estandar no 1979 com a publicação do (American National Standard for the Preparation of Scientific Papers for Written or Oral Presentation, em inglês) [ANSI-Z39.16(1979)]. Tal publicação, de acordo com [Day(1989)], surgiu em resposta ao crescente aumento da produção de artigos científicos de baixa qualidade na segunda metade do século XX, e foi amplamente adotado por todas as áreas da ciência, mediante as principais organizações científicas e os principais jornais acadêmicos. Graças a isto, geralmente todos os artigos científicos na atualidade possuem uma estrutura.

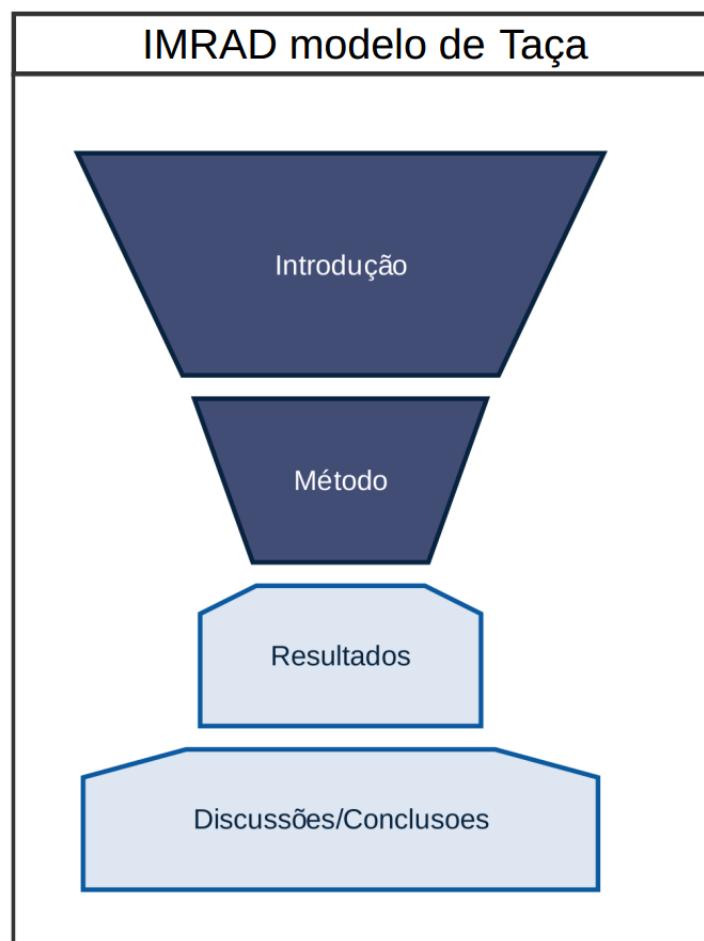


Figura 5: Estrutura de tipo Taça que define os documentos científicos

O IMRAD, mais que uma lista de cabeçalhos, é um formato que define como devem ser estruturadas e escritas cada uma das partes de um artigo científico onde de maneira geral, o formato está definido pelas seções. Existem grande variedade de livros e artigos sobre escrita e leitura de textos científicos os quais se baseiam no IMRAD como guia o estrutura básica [Burrough-Boenisch(1999), Glasman-Deal(2010)]. Esta estrutura básica se define em um modelo de taça como é mostrado na figura 5.

O modelo ou estrutura de taça mostrado na figura 5, representa a forma como pode ser abordada a escrita de um documento científico, onde mediante as estratégias top-down ou bottom-up, as mesmas ações realizadas no começo da escrita pela seção introdutória, podem ser realizadas no sentido inverso. Outra característica desta estrutura é a forma em que é organizada a informação indo de baixo para cima ou vice-versa, a informação está organizada de mais geral a mais específica, desta forma, nas seções Introdução e Discussões/Conclusões encontram-se as informações mais gerais, mientras que na seção de Método e Resultados as mais específicas.

De acordo com o princípio anterior, a forma ou o modelo que representa cada uma das seções no IMRAD estão caracterizados por:

Introdução: O modelo da introdução começa pela descrição da importância do tema da pesquisa. Posteriormente, são fornecidos os antecedentes gerais; o tipo de informação que está geralmente contida também pelas duas primeiras sentenças da introdução, mas de forma mais específica ou detalhada. Em seguida, descreve-se a área do problema ou são abordados alguns estudos atuais sobre esta. É realizada uma ponte entre a área geral do problema e a revisão da literatura. Posteriormente, é realizada uma breve descrição sobre os principais trabalhos sobre esta área, os problemas destes trabalhos e, de forma sucinta, é descrito o trabalho proposto, fornecendo-se informações sobre a metodologia aplicada, bem como sobre alguns resultados obtidos.

Metodologia: Esta seção do modelo começa com uma descrição geral de cada ponto a ser abordado; são fornecidas as bases, os trabalhos e uma justificativa da razão destas bases; descreve-se de forma geral o procedimento/método usado, os detalhes sobre o qual foram feitos os elementos usados e os cuidados que foram tomados; depois, são referidos na literatura os trabalhos feitos sobre este ponto; é mostrada a razão do uso deste método e por qual razão é uma boa escolha; finalmente, são mencionados os problemas e dificuldades do método.

Resultados: Inicia-se fazendo referência a conclusões e resultados e obtidos por outros; são aportadas mais informações sobre o método, onde são referidas figuras, tabelas,

esquemas, entre outros; são comparados os resultados com outros estudos; é oferecido um critério sobre os resultados obtidos; são comentados de forma específica determinados resultados; é descrito o método usado para analisar os resultados; são mencionados os problemas observados nos resultados tentando minimizar a importância destes; por último são mencionadas algumas implicações do uso do trabalho proposto.

Conclusões: Nesta última fase, primeiramente são abordados de forma abreviada trabalhos prévios; continua fazendo referência à introdução ressaltando problemas e debilidades de trabalhos prévios; é realizada uma revisão novamente da metodologia do trabalho; são sumarizados os resultados obtidos enquanto são ressaltados os de maior força ou mais destacados (inovadores, de grande valor acadêmico, entre outros); são indicadas algumas aplicações enquanto se definem as limitações das mesmas e são marcadas áreas do trabalho que podem constituir investigações futuras.

Outras características relevantes nos textos científicos, propiciadas pela adoção do IMRAD, são que, em geral, em todos estes se evidencia a existência de estilos de escrita comuns, no que respeita à definição de títulos típicos (TT) nas seções, assim como frases típicas (FT) usadas em todo o texto. Ditas características podem ser apreciadas também nos vocabulários que têm sido conformados pelos trabalhos [Burrough-Boenisch(1999), Glasman-Deal(2010)].

A partir do estudo destas características e da leitura de centenas de artigos nos foi possível constatar que ditas FTs encontram-se agrupadas na estrutura organizativa proposta pelo IMRAD(como é mostrado na figura 6), além de que geralmente tais FTs aparecem nas sentenças de maior relevância. Neste trabalho, estes dois últimos pontos mencionados representam os elementos chaves para identificar as sentenças serão extraídas para o sumário final, constituindo assim, a base de seleção e extração da informação do modelo proposto, ponto que será abordado no capítulo 4.

2.2 Critérios do Abstract

Como foi abordado na seção anterior, existe uma estrutura básica nos DTC, mas dita estrutura não contempla o abstract, elemento geralmente indispensável em todo trabalho científico, pois, além de ser uma exigência na maioria dos jornais científicos, este tipo de resumo é a principal ferramenta na hora de selecionar um texto científico para leitura. Na maioria dos guias ou livros acerca da escrita científica, define-se o abstract como um elemento independente do texto que tem que fazer sentido por si mesmo. Este

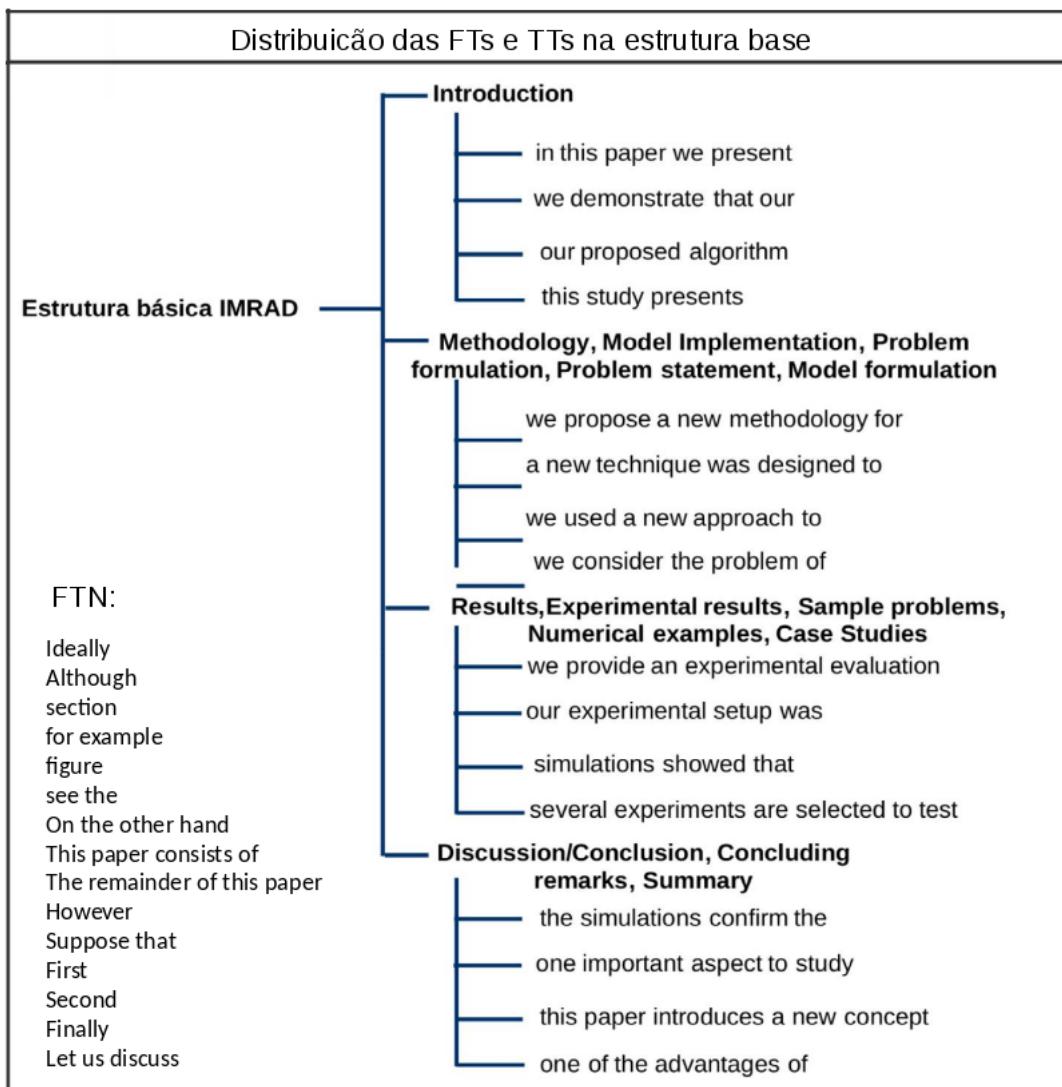


Figura 6: Exemplo de algumas TTs, FTs e FTNs identificadas e utilizadas neste trabalho assim como a distribuição destes elementos na estrutura base dos documentos técnicos-científicos, IMRAD

representa uma versão extremamente condensada do DTC, abordando os principais pontos do trabalho com o objetivo de que um leitor consiga entender a ideia deste, sem ter que olhar para o trabalho.

Na metodologia de escrita do abstract, são abordados os antecedentes da pesquisa (Introdução); posteriormente, é realizado um breve resumo da metodologia, resultando nos principais pontos, como motivação etc (Method); são abordados os principais resultados (Resultados) e se apontam as implicações destes resultados (conclusões/Discussões).

Como pode ser verificado, o abstract segue praticamente a mesma estrutura do IMRAD, mas em uma versão condensada, seguindo as características de independência e de compreensão. Embora os sumários gerados a partir do modelo de SAT sejam extrativos e

não abstrativos, como é a natureza dos abstract de artigos científicos, respondem igualmente a estes princípios.

Nesse sentido, com base no estudo de centenas de abstract de artigos científicos, foram identificadas determinadas frases típicas, que atentam negativamente contra estes princípio. Estas frases são denominadas neste trabalho FTN ou frases típicas negativas, correspondendo à natureza destas. Estas frases, à diferença das FT comuns, não estão associadas a seções específicas do IMRAD e sim a todo ele. Esta informação permitirá ao modelo proposto excluir este tipo de frases para corresponder aos princípios mencionados anteriormente, ponto que será abordado posteriormente.

2.3 Sumarização por extração

Em seções passadas, foram mencionados os tipos de SAT que existem (sumarização abstrativa e extrativa). No caso da sumarização abstrativa, a que maior grau de dificuldade apresenta, os modelos se baseiam principalmente em gerar abstrações do texto-fonte mediante um processo profundo de representação e modelação semântica, as quais permitem gerar um texto novo. Estes modelos, como [Zajic *et al.*(2007)Zajic, Dorr, Lin, e Schwartz, Knight e Marcu(2002)], caracterizam-se pela utilização de técnicas baseadas em grafos, combinadas com estratégias estatísticas e aprendizado de máquina para tentar compreender as sentenças. Ainda que este tipo de abordagem nos últimos anos tenha ganhado força, ainda está longe de ser sólida ou madura o suficiente, percepção que pode ser apreciada ao analisar a performance destes modelos.

A abordagem extrativa, que é usada neste trabalho, por outro lado, conta com um número elevados de trabalhos, alguns já abordados anteriormente, tais como [Luhn(1958), Edmundson(1969),Paice(1981),Lehmam(1999),Teufel e Moens(2002)]. Os trabalhos sobre este tipo de sumarização são focados na detecção de unidades textuais relevantes (em geral, sentenças ou frase) mediante diversos análisis léxico ou estatísticos. O principal objetivo destes modelos é extrair as unidades textuais do texto e posteriormente conformar o sumário concatenando-as. Segundo [Hahn e Mani(2000)] e como é mostrado neste trabalho, a maioria destes sistemas se caracteriza por aplicar um tipo de ponderação linear no processo de análise.

Neste processo, são atribuídos pesos a estas unidades textuais de acordo com as seguintes características: posição, aparição nestas de frases chaves ou cue phrases e outras métricas estatísticas de relevância. O peso final das unidades textuais está dado pela soma destes pesos individuais por característica, geralmente regulado por algum parâmetro adi-

cional.

Neste trabalho, são usadas também estas características no módulo de extração de sentenças a ser analisado no capítulo 4.

2.4 Elementos teóricos de PLN

2.4.1 Pré-processamento de texto

Atualmente a tarefa de pré-processamento de documentos técnico-científicos ou preparação destes tipos de documentos para PLN é pouco explorada pelos pesquisadores, com base nessa dificuldade que constitui um problema dada a grande variedade de formatos e padrões que existem nas publicações, assim como a existência de elementos não necessários para o sumário, o que representa um peso morto ou ruído para o processamento. Este problema carece de uma solução trivial, pois é preciso unificar esta informação num formato comum para logo ser processado, eliminando os elementos não necessários e, desta maneira, obter maior clareza nos sumários gerados.

Antes	Después
There are many fuzzy clustering methods proposed in the literature [6-13]	There are many fuzzy clustering methods proposed in the literature
Similar to [Alshawi and Carter, 1994; Grishman and Sterling, 1994; Ruge, 1992], we use a parser to extract dependency triples from the text corpus.	we use a parser to extract dependency triples from the text corpus.

Tabela 2: Exemplos de sentenças extraídas de um artigo técnico-científico, pré-processadas usando expressões regulares.

Hoje em dia, não existem muitas investigações relacionadas com este problema. Normalmente, para dar solução ao mesmo são adotadas técnicas como a limpeza ou formatação do texto usando grupos de expressões regulares (sequência de caracteres que definem um padrão de pesquisa) [Mitkov(2003)]. Temos um exemplo na tabela 2, onde as sentenças extraídas de um artigo técnico-científico são processadas usando a expressão regular:

$$sentence = re.sub(r', '\[[0-9]+\]\s+-\s+\[[0-9]+\]\|\[[0-9]+\]\', sentence)$$

O processo de limpeza e formatação do texto tem uma considerável importância para o modelo proposto neste trabalho, pois os documentos científicos contêm muitos elementos

que são descartáveis para efeito de sumário, tais como: referências, fórmulas matemáticas, figuras, etc. Além disso, o formato destes varia de acordo com os meios de publicação, daí a relevância deste pré-processamento. Foi implementado um mecanismo que usa um grupo de expressões regulares especializadas no reconhecimento dos padrões que caracterizam estes elementos insignificantes da informação a processar.

2.4.2 Eliminação de StopWords

Todas as linguagens humanas contêm um grupo de palavras que possuem uma alta taxa de ocorrência nos textos ou uma frequência elevada, devido a isto, segundo Luhn [Luhn(1958)], em toda a extensão de um texto somente é significativo um pequeno grupo de palavras. Para o idioma inglês as palavras como this, the, a, in aportam um valor insignificante de informação para determinadas tarefas de PLN. Palavras como estas são denominadas stop-words, ou dicionário negativo. Elas são geralmente ignoradas ou eliminadas do texto original em algumas instâncias específicas de determinados processos intrínsecos dos sistemas de PLN, pois constituem um elemento de ruído ou um peso morto para o processamento da informação.

A maioria dos sistemas de PLN, assim como neste trabalho, tem concebida a implementação de um mecanismo que elimina estas palavras quando são reconhecidas no texto. Tal mecanismo usa como base um dicionário de stop-words predefinido existente como parte da livraria NLTK (Natural Language Tool Kit) usada na implementação do sistema feito para testar o modelo proposto.

2.4.3 Segmentação de sentenças

A segmentação do texto em núcleos menores, como parágrafos, sentenças ou palavras (também conhecido como tokenização) é outro dos problemas existentes dado o grau de ambiguidade ao determinar os limites de cada um destes elementos. Especificamente no caso da segmentação de sentenças se experimenta um alto grau de ambiguidade em relação ao ponto final, devido à existência de abreviações, por exemplo U.S.A.

Existem vários estudos na literatura científica para tratar este problema, alguns dos quais definem modelos de aprendizagem supervisionados e não supervisionados. Uma destas abordagens é definida por [Ratnaparkhi(1996)] onde é proposto um modelo baseado na Entropia Máxima, o mesmo tem uma percentagem de precisão máxima de 98.8, o qual aponta para resultados muito satisfatórios.

2.4.4 Part-of-speech tagging(POS tagging)

Parte do discurso (POS ou Part of Speech) é um termo empregado na linguística computacional moderna que define categorias de palavras (word class, lexical class). As palavras são atribuídas ao mesmo POS em dependência do papel sintático que têm dentro da estrutura gramatical da sentença. As classes de palavras são divididas em dois tipos fundamentais, classes abertas ou fechadas. As classes abertas correspondem às categorias de palavras (substantivos, verbos e adjetivos) que não contêm um grupo finito de elementos dada a aparição de novos elementos constantemente. Por outro lado, as fechadas (pronomes, conjunções, etc) possuem um número finito de elementos.

Vários grupos de etiquetas (tagset) de diferentes tamanhos têm sido definidos para identificar as categorias de palavras em idioma inglês, associadas ao POS. Estes tagsets geralmente são versões reduzidas do tagset de 87 elementos definido por [Francis *et al.*(1982) Francis, Kucera, e Mackie] e usado para etiquetar o Brown Corpus [Francis e Kucera(1979)]. Um dos tagset que surgiram como versão reduzida do tagset antes mencionado é o Penn Treebank [Marcus *et al.*(1993) Marcus, Marcinkiewicz, e Santorini], que é composto por um grupo de 45 etiquetas e tem sido amplamente usado pela comunidade científica em sistemas de PLN. Tendo em conta tudo isto, foi definido o uso deste tagset para realizar a implementação do modelo proposto.

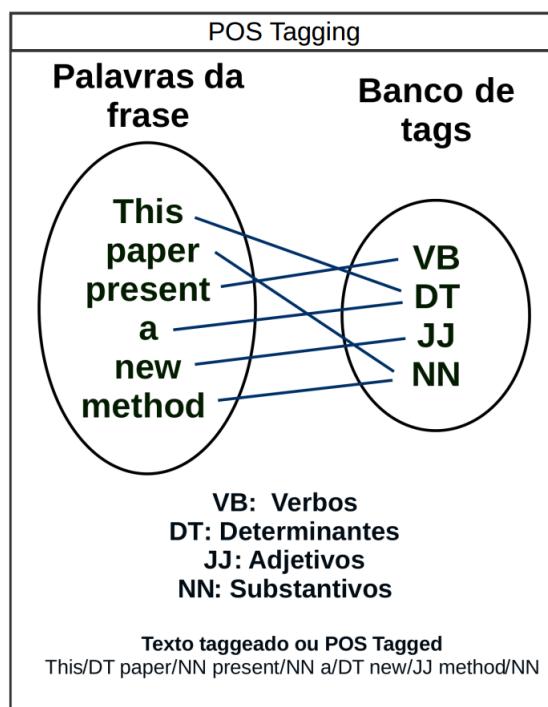


Figura 7: Método de POS Tagging

Na linguística computacional e na área de PLN, o processo de identificar e definir os

tags ou etiquetas anteriormente mencionadas associadas a cada palavra em um corpus é conhecido como POS tagging ou simplesmente tagging. É geralmente precedido por um processo de tokenização das palavras (mencionado na seção anterior), excluindo elementos não importantes, tais como sinais de pontuação. Na figura 7, é possível ver um exemplo disso.

O processo de tagging não é trivial devido a presença de ambiguidades. Tal fenômeno é apresentado quando uma palavra pertence tanto a uma classe quanto a outra ao mesmo tempo, o que configura um problema de resolução complexa, inclusive para os seres humanos. Na literatura existem vários algoritmos definidos para o POS Tagging, sendo categorizados da seguinte forma: algoritmos baseados em regras (rule-based taggers), algoritmos estocásticos (stochastic taggers) e algoritmos baseados em transformação (combina ambos, rule-based e stochastic). Para modelos baseados em regras, foi identificado o EngCG tagger. Dentre os estocásticos, existem os Modelos Ocultos de Markov (Hidden Markov Models) [Brants(2000)] e os Modelos de Máxima Entropia (Maximum Entropy Model) [Ratnaparkhi(1996)]. Por fim, dentre os modelos baseados em transformação, destaca-se o definido em [Brill(1994)]. Estes modelos têm uma precisão bem alta e similar, dado que geralmente são treinados usando os mesmos dados. Os modelos baseados em regras, como o mencionado anteriormente, possuem uma precisão maior segundo demonstrado por [Samuelsson e Voutilainen(1997)].

2.4.5 Ontologias computacionais

Ontologias computacionais definem a forma na qual é modelada a estrutura de um sistema, as entidades relevantes e as relações entre estas entidades. Consistem fundamentalmente na generalização ou especificação hierárquica de conceitos [Guarino *et al.*(2009)Guarino, Oberle, e Staab]. Segundo Gruber em [Gruber(1993)], uma ontologia é uma especificação de um conceito, sendo dito conceito uma abstrata, mas simplificada visão do domínio a ser representado por algum propósito. Ontologias são como os esquemas conceituais dos sistemas de base de dados, prevendo assim uma descrição lógica dos dados.

No âmbito do PLN as ontologias têm um papel fundamental em sistemas baseados em conhecimento, como os empregados na resolução de ambiguidades, tradução automática de textos, similaridade semântica, entre outros. Dentre as ontologias mais usadas e difundidas no campo de PLN, encontra-se o WordNet e o Open Multilingual WordNet, que serão descritos posteriormente, dada sua importância para este trabalho.

Synsets	Conceitos			
S1	C1	C2	C3	C4
S2	C5	C6	C7	C8
S3	C9	C6	C10	C11
S4	C12	C13	C14	C15

Tabela 3: Tabela representando a relação de muitos a muitos entre synsets e conceitos

2.4.6 WordNet

O WordNet é um sistema léxico de referência online, também denominado como base de dados léxica, ou ontologia linguística, desenvolvido pelo Cognitive Science Laboratory da Universidade de Princeton [Miller *et al.*(1990)Miller, Beckwith, Fellbaum, Gross, e Miller]. É inspirado em teorias psicolinguísticas sobre a memória lexical humana. Esta ontologia linguística está composta por conceitos (ou palavras) categorizados como substantivos, verbos, adjetivos e advérbios, segundo a categoria sintática associada, totalizando em sua última versão 155.287 conceitos, agrupados em 117.659 nós, denominados synsets, ou grupos de significados.

Tais synsets se encontram conectados entre si através de distintas relações. Tal agrupamento pode ser observado na tabela 3 [Miller *et al.*(1990)Miller, Beckwith, Fellbaum, Gross, e Miller], onde as colunas representam synsets, e as linhas, palavras ou conceitos. Nessa matriz é possível apreciar que o mapeamento entre synsets e conceitos representa uma relação de muitos a muitos, ou seja, um conceito pode ter vários significados, sendo portanto polissêmico(exemplo tabela 3 conceito C6), ou o inverso, um synset pode estar associado a vários conceitos, neste caso, os conceitos compartem o significado e são sinônimos(exemplo tabela 3 conceito C1,C2,C3,C4), sendo que o intercâmbio entre dois conceitos sinônimos nunca afetará o sentido da sentença onde estão sendo inter-substituídos [Miller *et al.*(1990)Miller, Beckwith, Fellbaum, Gross, e Miller], princípio que representa uma das chaves do modelo de similaridade semântica textual proposto neste trabalho.

As relações semânticas entre synsets estão diferenciadas de acordo com as categorias sintáticas dos conceitos que representam. Isto está demonstrado pela primeira relação de sinônimos descrita anteriormente, e evidencia que sistematicamente é impossível que duas palavras iguais mas de categorias sintáticas diferentes possam representar sinônimos.

Dentre as principais relações semânticas destacam-se as associadas aos substantivos e verbos entre verbos no WordNet descritas a seguir:

Substantivos figura 8:

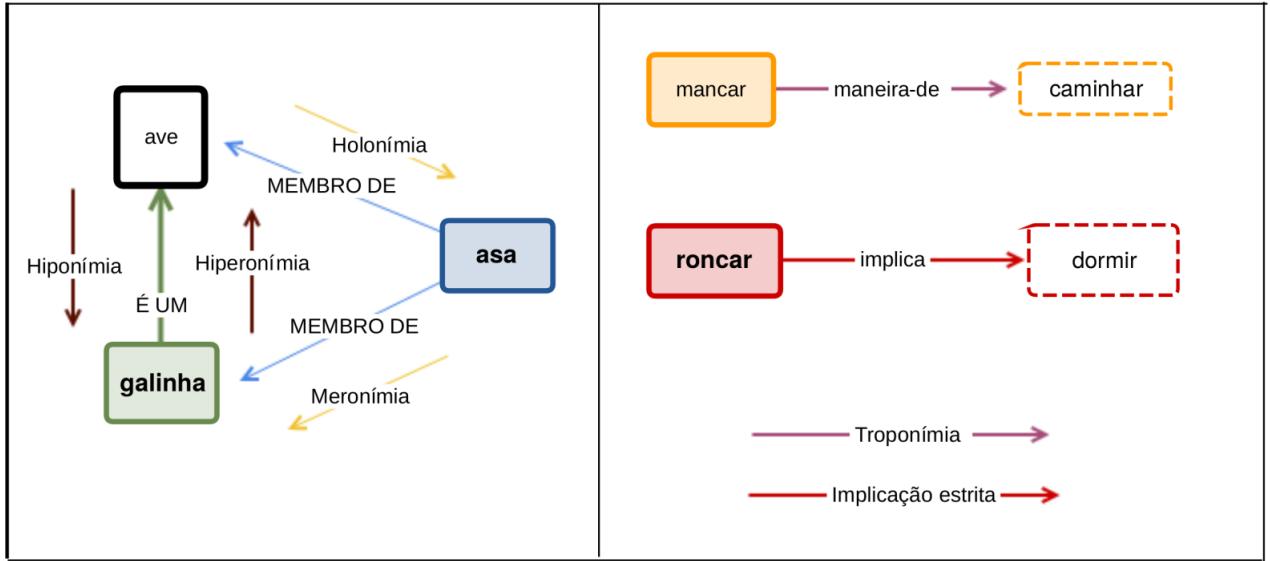


Figura 8: Relações entre substantivos e entre verbos no WordNet [Miller *et al.*(1990)Miller, Beckwith, Fellbaum, Gross, e Miller]

Hiperonímia: Relação hierárquica semântica de caráter ascendente, caracterizada pela relação é-um (is-a). X é um hiperônimo de Y se Y é um X. Exemplo da figura 8: ave é um hiperônimo de galinha.

Hiponímia: Relação hierárquica semântica de caráter descendente, caracterizada pela relação é-um (is-a). Y é um hipônimo de X se X é um Y. Exemplo da figura 8: galinha é um hipônimo de ave.

Holonímia: Relação hierárquica semântica de caráter ascendente, caracterizada pela relação membro-de (member-of). X é um holônimo de Y se Y é membro de X. Exemplo da figura 8: ave é um holônimo de asa.

Meronímia: Relação hierárquica semântica de caráter descendente, caracterizada pela relação membro-de (member-of). Y é um merônimo de X se X é membro de Y. Exemplo da figura 8: asa é um merônimo de galinha.

Verbos figura 8:

Troponímia: Relação semântica que representa maneira-de ou forma-de. O verbo X é tropônimo de Y se X é Y de certo modo. Exemplo da figura 8: engatinhar é um tropônimo de movimentar.

Implicação estrita: O verbo X implica Y se e somente se não existe o caso em que ocorra X e não ocorra Y. Exemplo da figura 8: roncar implica estritamente dormir.

2.4.7 Open Multilingual WordNet

Seguindo a ontologia WordNet constituída, foram criadas muitas extensões da mesma para distintas linguagens, as quais foram centralizadas em uma só base de dados léxica denominada Open Multilingual WordNet [Bond e Foster(2013)]. Esta base de dados multi-linguagens integra os WordNets de diversas línguas, o que permite extrapolar o poder do WordNet para a análise em outros idiomas sem necessidade de investir esforço na aprendizagem de uma nova ontologia.

3 O MODELO DE ESTIMAÇÃO DE STS

3.1 Conceição do modelo

Na área de PLN, o termo Medidas de Similaridade Semântica representa os métodos computacionais que estimam qual é o grau de similaridade semântica entre duas informações, geralmente constituintes linguísticos, diga-se sentenças, frases, palavras, entre outros. Correspondentemente, esta tarefa refere-se à Similaridade Semântica Textual (STS) [Agirre *et al.*(2012)Agirre, Diab, Cer, e Gonzalez-Agirre], sendo que os métodos associados a esta tentam alcançar um desempenho similar a um humano. Os modelos de STS são de extrema importância na área de PLN, devido a seu impacto em outras áreas, como recuperação de informação, sumarização entre outras.

Geralmente os métodos desenvolvidos para solucionar esta tarefa fazem uso de informação semântica mediante a consulta a ontologias ou bases de conhecimento, como WordNet [Wu e Palmer(1994)], assim como conhecimento estatístico com base em corpos linguísticos, como os modelos baseados em N-Grams [Buscaldi *et al.*(2010)Buscaldi, Rosso, Gómez-Soriano, e Sanchis, Buscaldi *et al.*(2012)Buscaldi, Tournier, Aussenac-Gilles, e Mothe]. Estes métodos na prática obtêm resultados razoáveis, mas muito longe ainda de serem adequados, em comparação com humanos.

Alguns autores como Hulth [Hulth(2003)] e [Hulth e Megyesi(2006)] têm afirmado acertadamente que a linguística computacional ganharia muito ao introduzir o uso de características e propriedades da linguagem. Tendo como base este princípio, características como conhecimento linguístico e análises sintáticas do inglês mediante a técnica de chunking são usadas neste trabalho na proposta de uma metodologia a qual integra conhecimento semântico mediante o uso de ontologias e conhecimento estatístico, de forma tal que a combinação destes é inherentemente interna a todo o sistema. O objetivo desta metodologia é propiciar uma melhor medida para o modelo de estimativa da similaridade semântica entre sentenças.

Uma importante motivação deste modelo encontra-se em Abney [Abney(1992)], para

quem os humanos fazem uso da técnica de chunking no processo de uso da linguagem. Ao ler uma passagem em voz alta, uma pessoa mostra-se indecisa para encontrar a entonação correta, e a hesitação significa a procura da divisão correta em padrões prosódicos. Este fato pode significar a execução no cérebro de alguma espécie de parsing da sentença em partes apropriadas, ou chunks no inglês. Esta intuição é incluída no modelo proposto de extração de estruturas sintáticas parciais, Tratando a sentença como uma estrutura, esta é associada a sub estruturas da sentença, diga-se, frases substantivas, verbais, etc., as quais são as unidades intermediárias da análise semântica realizada pelo modelo.

3.2 Estudos prévios

Alguns estudos prévios na área de STS concentraram-se principalmente sobre palavras individuais, ou então textos longos. Motivada por aplicações como question answering systems, motores de pesquisa sobre a web e extração de informação em geral, a tendência mudou para o estudo da similaridade entre textos curtos, passagens ou frases. Nestas aplicações precisamente a similaridade a nível semântico é chave. Para resolver este problema, a estrutura dos textos deve ser considerada [Mihalcea *et al.*(2006)Mihalcea, Corley, e Strapparava]. A forma de considerar a similaridade em medidas de STS evoluiu da similaridade semântica entre palavras para similaridade semântica entre textos, diga-se sentenças, frases, passagens, entre outros. Correspondentemente a isto lograram-se resultados promissores, mas longe ainda de serem satisfatórios.

Muito poucas publicações sobre STS para textos muito curtos foram realizadas até 2006 [Li *et al.*(2006)Li, McLean, Bandar, O'Shea, e Crockett]. No entanto, vários métodos para a estimativa de similaridade semântica entre palavras foram propostos. Para medir a STS entre fragmentos de textos deve-se atuar em nível semântico, o que é uma tarefa difícil, necessitando levar em conta as estruturas dos textos, além da similaridade entre palavras [Mihalcea *et al.*(2006)Mihalcea, Corley, e Strapparava].

Alguns autores têm optado por técnicas combinatórias de modelos de correspondência lexical, ou lexical matching no inglês, com outros modelos. Dita combinação na prática toma a forma da transformação de métricas de similaridade semântica word-to-word para text-to-text, como o modelo proposto por [Mihalcea *et al.*(2006)Mihalcea, Corley, e Strapparava]. Neste modelo é combinada uma métrica de similaridade semântica baseada em conhecimento a qual utiliza o WordNet, com outra baseada em corpus, a qual utiliza a distribuição estatística de um amplo corpus, para determinar a frequência inversa de documento, associada a palavra, ou tfIdf.

Este método, apesar de trazer algumas melhorias sobre métricas com base apenas em casamento lexical, continua sendo do tipo bag-of-words, sendo este tipo de modelo pouco acertado devido que não tem em conta nem a ordem das palavras, nem informação estrutural ou sintática.

Seguindo esta mesma tendência, [Buscaldi *et al.*(2012)Buscaldi, Tournier, Aussenac-Gilles, e Mothe] combina, mediante uma estratégia numérica, uma métrica de estatística baseada em NGrams com outra denominada Similaridade Conceitual, a qual é baseada em WordNet. Devido ao uso de NGrams, o que traz certa similaridade estrutural embutida, este método oferece alguma melhora na estimativa de STS, como foi indicado no ranking na competição SemEval 2012 [Agirre *et al.*(2012)Agirre, Diab, Cer, e Gonzalez-Agirre].

Com o surgimento da competição SemEval, as tarefas de STS foram impulsionadas enormemente. No SemEval 2012 [Agirre *et al.*(2012)Agirre, Diab, Cer, e Gonzalez-Agirre] e nos seguintes anos até chegar ao SemEval 2016, foi proposto uma forma nova para avaliar sistemas de STS, onde, na Tarefa 1 deste concurso, os participantes devem estimar o nível de similaridade semântica entre duas sentenças de diferentes fontes (notícias, perguntas, política, etc). Muitos sistemas foram propostos e avaliados sobre determinados datasets, usando o chamado Pearson correlation coefficient em relação com um padrão dourado ou estimativa ideal definido por humanos. Aqui, os sistemas participantes têm aplicado diversas abordagens para alcançar se igual a este padrão, desde a combinação de diversos modelos de STS [Agirre *et al.*(2016)Agirre, Banea, Cer, Diab, Gonzalez-Agirre, Mihalcea, Rigau, e Wiebe] até o uso de técnicas de Deep Learning.

3.3 Visão geral do modelo de STS

O sistema usado para estimar a similaridade semântica entre duas sentenças ou frases pode ser observado na figura 9. Este sistema está composto pelos subsistemas semântico e estatístico. Estes subsistemas encontram-se interconectados mediante o módulo de aproximação de sentenças, o qual é o centro do modelo proposto e é o responsável pela inovação introduzida por este método, como será tratado posteriormente.

Na áreas de PLN, como sumarização automática de textos e extração de informação, as frases substantivas (noun phrases, NPs, em inglês) são um conceito fundamental. Se estas são corretamente reconhecidas e extraídas, permitem um primeiro entendimento de uma sentença ou texto [hua Chen e Chen(1994)], de forma similar à leitura humana. Neste trabalho é usado um analisador parcial probabilístico denominado como chunker em inglês

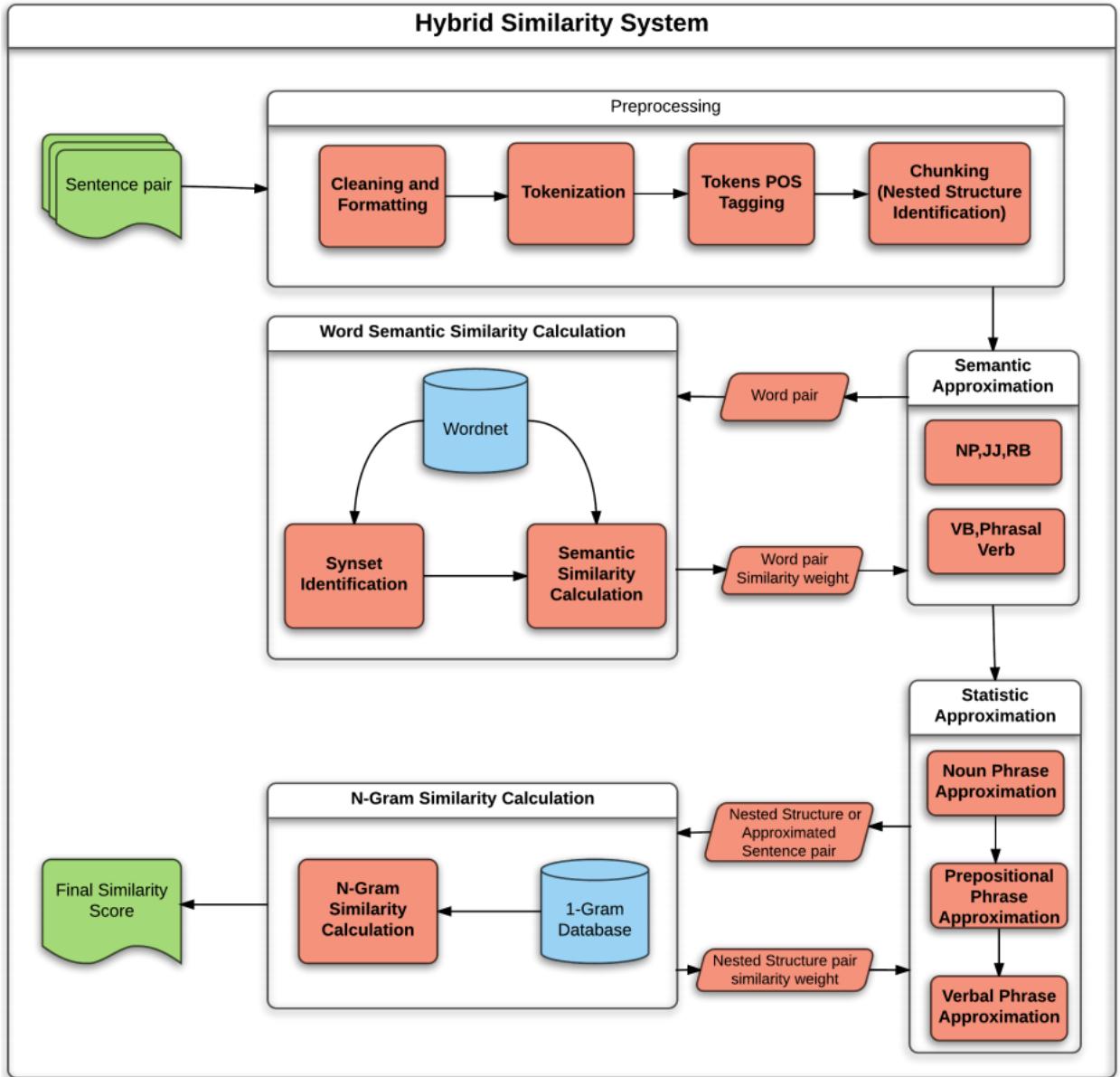


Figura 9: Sistema híbrido de STS baseado no modelo proposto

(figura 9), para identificar as estruturas linguísticas na sequência de tokens categorizados na fase de etiquetagem.

Este chunker permite a extração de determinadas partes de uma sentença taggeada, de acordo com uma gramática simples definida, como será analisado em mais detalhes posteriormente. Ditas partes reconhecidas e extraídas constituem as frases substantivas, NPs, as frases verbais, VPs, assim como os verbos frasais, PVs, os quais constituem expressões idiomáticas. Os conceitos (palavras) dentro de cada uma destas estruturas são avaliados semanticamente mediante o uso de uma medida de similaridade semântica com base no WordNet. Se alguma das palavras de cada sentença é semanticamente similar, uma destas é aproximada, isto é a operação de substituição de uma palavra por outra ao

serem similares, o que resulta em palavras iguais em ambas as sentenças. As sentenças modificadas em seguida, mostram um aumento do número de correspondências exatas favorecendo assim uma melhor estimativa por parte da similaridade estatística em base de NGrams. Este é em resumo o princípio novo usado pelo modelo de STS proposto.

3.4 O analisador sintático probabilístico (Chunker)

A entrada do sistema é um par de sentenças, as quais serão pré-processadas até serem divididas por palavras ou tokens (Word Tokenization) e depois etiquetadas (POS Tagging). Posteriormente o chunker analisa a sequência de tokens e identifica as estruturas sintáticas definidas pelas regras na gramática. Os correspondentes pares de substantivos, adjetivos, advérbios, verbos e verbos frasais nas duas sentenças são processadas pelo modelo de similaridade semântica.

3.5 Módulo de similaridade semântica

A similaridade semântica entre dois conceitos (palavras) está relacionada à medida que avalia quão perto em significado eles estão na hierarquia dos synsets no WordNet. Neste módulo dois synsets S1 e S2 são obtidos do WordNet, na associação com duas palavras w1 e w2, semanticamente categorizadas, recebidas do módulo de aproximação. A similaridade semântica entre w1 e w2 é o máximo valor de similaridade calculada por um algoritmo usando os synset S1 e S2. Neste trabalho foi usado o algoritmo, Path Similarity (WUP) [Wu e Palmer(1994)].

Se as correspondentes frases NPs, PPs, VPs ou PVs têm só uma palavra, então a similaridade semântica é usada no propósito da aproximação. Isto significa que, se a similaridade obtida entre w1 e w2 é maior que um limiar determinado, neste caso 0,8, então w1 substitui w2 ou vice versa. A propriedade de transitividade é usada: se a palavra w1 é similar à palavra w2 e w2 é similar a w3, então w1 é similar a w3 para efeitos de substituição.

É preciso apontar que, enquanto o WordNet só considera palavras de um ponto de vista léxical individualmente, nosso uso deste alcança maior qualidade de estimativa da similaridade semântica, uma vez que palavras altamente similares passam a pertencer às frases correspondentes em ambas sentenças, preservando a pertinência do fragmento a la consideração de similitude

3.6 Módulo de similaridade estatística

A similaridade de NGrams encontra-se baseada no princípio de que duas sentenças são similares se estas compartem sequências de elementos não vazios e, quanto maior seja a sequência, maior a similaridade entre estas. A similaridade de NGrams, denominada Clustered Keywords Positional Distance (CKPD) [Buscaldi *et al.*(2012)Buscaldi, Tournier, Aussenac-Gilles, e Mothe] em diferentes tipos de frases são avaliadas no modelo de similaridade estatística, começando com as NPs, PPs, PVs, VPs. Caso a similaridade seja maior que 0.6 (valor inicial tentativo), as frases correspondentes são aproximadas, com a maior substituindo a menor. A propriedade de transitividade é aplicada também.

Se as frases extraídas pelo chunker têm dois ou mais termos em qualquer sentença, então os correspondentes 2-Grams, 3-Grams, e assim por diante, serão avaliadas pelo módulo de similaridade estatística para o propósito da aproximação. Depois que todos os NGrams forem avaliados, e todas as frases possíveis, aproximadas, então as duas sentenças originais, agora aproximadas de forma máxima, são finalmente avaliadas pelo módulo de similaridade estatística para obter assim o valor de estimativa de similaridade final (figura 9).

3.7 Exemplos trabalhados

Para demonstrar o funcionamento do modelo de STS foi implementado um sistema em Python, usando as bibliotecas NLTK para a fase de pré-processamento do sistema e acesso ao wrapper do POS tagger e tokenizador de Stanford [Toutanova *et al.*(2003)Toutanova, Klein, Manning, e Singer].

Dentro deste sistema também foram implementados o modelo de similaridade semântica WUP [Wu e Palmer(1994)], mencionado anteriormente, e o modelo de similaridade estatística CKPD [Buscaldi *et al.*(2012)Buscaldi, Tournier, Aussenac-Gilles, e Mothe].

Uma vez que não são necessárias regras gramaticais muito complexas para descrever as subestruturas a identificar pelo chunker, a gramática usada é simples e encontra-se representada por um conjunto de regras gramaticais extraídas do capítulo 7 do livro de NLP with Python [Bird *et al.*(2009)Bird, Klein, e Loper]. Todavia, para ter uma cobertura maior de subestruturas deverá ser usada uma gramática mais sofisticada. Para os efeitos deste trabalho a gramática usada descrita abaixo apresenta-se adequada, como pode ser evidenciado pelos resultados obtidos a partir do uso da mesma no modelo em questão.

Descrição da gramática usada:

NPs: Um predeterminante (PDT) que pode ou não existir, seguido por um determinante (DT) ou pronome (PP\$) que pode ou não existir, seguido por nenhum, um, ou mais adjetivos (JJ) e, por último, um ou mais substantivos (NN).

NP: {<PDT>?<DT|PP\$>?<JJ>*<NN.*>+}

PPs: Uma preposição ou conjunção subordinada (IN) seguida por uma NP.

PP: {<IN><NP>}

VPs: Qualquer verbo(VB) seguido por uma ou mais NP, PP ou VP até o final da sentença.

VP: {<VB.*><NP|PP>+\$} {<VB.*><NP><VP>+\$}

PVs: Qualquer verbo (VB) seguido por uma NP que pode não existir seguido por advérbio (RB) que pode ou não existir.

PV: {<VB.*><NP>?<RB>?}

Gramática completa:

NP: {<PDT>?<DT|PP\$>?<JJ>*<NN.*>+}

PP: {<IN><NP>}

VP: {<VB.*><NP|PP>+\$} {<VB.*><NP><VP>+\$}

PV {<VB.*><NP>?<RB>?}

Para melhor visualização da operação de todo o processo, é apresentada a aplicação do sistema sobre duas sentenças simples, S1 e S2 tomadas do dataset HDL No. 729 do SemEval 2013, com um limiar de disparo semântico de 0.8 e estatístico de 0.6. Definidos a partir dos limiares associados aos melhores resultados da aplicação do sistema sobre o dataset HDL antes mencionado.

Sejam duas sentenças S1 e S2:

S1: Newark mayor rescues neighbor from burning house.

S2: Newark mayor saves neighbor from fire.

Saída da fase de tokenização e taggeado das sentenças:

S1: [(u'newark', u'NN'), (u'mayor', u'NN'), (u'rescues', u'VBZ'), (u'neighbor', u'NN'), (u'from', u'IN'), (u'burning', u'NN'), (u'house', u'NN')]

S2: [(u'newark', u'NN'), (u'mayor', u'NN'), (u'saves', u'VBZ'), (u'neighbor', u'NN'),

(u'from', u'IN'), (u'fire', u'NN')]

Saída da fase de Chunking, árvores sintáticas de S1 figura 10 e S2 figura 11:

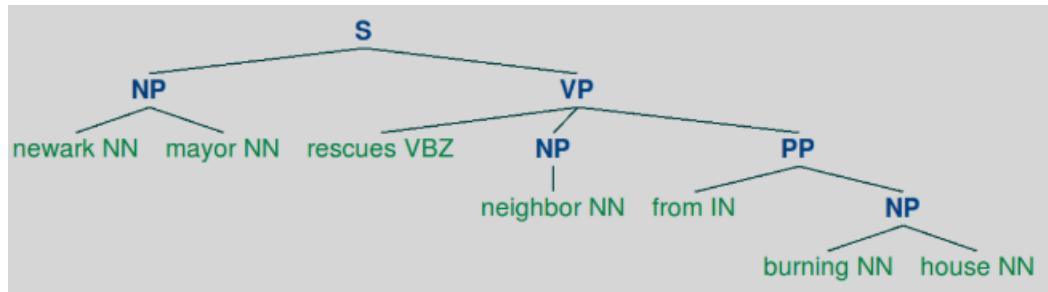


Figura 10: Árvore sintáctica de S1

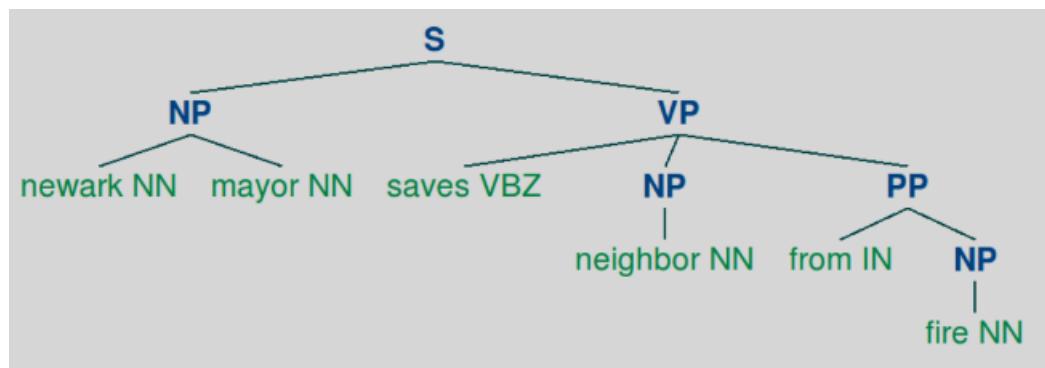


Figura 11: Árvore sintáctica de S2

Correspondentes NPs, VPs, PPs:

S1

NPs • newark NN mayor NN

• neighbor NN

• burning NN house NN

PPs • from IN burning NN house NN

VPs • rescues VBZ neighbor NN from IN burning NN house NN

S2

NPs • newark NN mayor NN

• neighbor NN

- fire NN
- PPs • from IN fire NN
- VPs • saves VBZ neighbor NN from IN fire NN

Aproximação semântica:

- NPs • Substitui noun, fire NN por burning NN, valor da similaridade semântica 0.94
- NP: fire NN -torna-se->burning NN
 - PP: from IN fire NN -torna-se->from burning NN
- VPs • Substitui verbo, save VBZ por rescues VBZ, valor da similaridade semântica 0.8
- saves VBZ neighbor NN from IN burning NN -torna-se->rescues VBZ neighbor NN from IN burning NN

Aproximação estatística:

- PPs • Substitui PP, from IN burning NN house NN por from IN burning NN, valor da similaridade estatística 0.67
- VPs • Substitui VP, rescues VBZ neighbor NN from IN burning NN by rescues VBZ neighbor NN from IN burning NN(note-se que depois da substituição do PP, os VPs são idênticos), valor da similaridade estatística 1.0

Sentenças aproximadas semântica e estatisticamente:

S1: newark mayor rescues neighbor from burning.

S2: newark mayor rescues neighbor from burning.

Similaridade final: 1.0

Vale apontar que este exemplo é para fins ilustrativos apenas. Os resultados do modelo de STS proposto são significantes sómente se aplicados sobre um número de sentenças considerável, como será o caso na seguinte seção de resultados experimentais.

ALL	Ans.-Ans.	HDL	Plagiarism	Postediting	Ques.-Ques.
0,60999	0.54755	0.61691	0.73621	0.82277	0.32652

Tabela 4: Coeficientes de Pearson resultantes do modelo de STS proposto sobre os dataset do SemEval2016

3.8 Resultados experimentais

Para testar o modelo de STS proposto, o sistema implementado foi rodado fazendo uso dos datasets do SemEval 2016 [Agirre *et al.*(2016) Agirre, Banea, Cer, Diab, Gonzalez-Agirre, Mihalcea, Rigau, e Wiebe] especificamente a tarefa 1 a qual corresponde a STS) e os resultados obtidos foram comparados com alguns modelos competidores. Como o objetivo não foi competir e sim avaliar as possibilidades do modelo proposto em comparação com modelos do estado da arte, sómente são comentados os resultados mais favoráveis ao modelo proposto.

Observando a tabela 4 é possível perceber que o resultado do modelo proposto no dataset postediting, 0.82277, é melhor que os resultados do segundo e quinto lugar respectivamente 0.82085 para a equipe UWB, e, 0.80939 para o UMB[3]. Além disto, os resultados obtidos pelo modelo de STS proposto no dataset answer-answer foram melhores que o obtido pelo equipe Ricoh, melhor colocada no dataset postediting. De fato todas as equipes foram classificadas de diversas formas em diferentes conjuntos de dados, o que mostra que há muito a aprender antes que um bom sistema geral seja obtido.

4 EXTRAÇÃO DE SENTENÇAS

4.1 Concepção do modelo

Como foi abordado em capítulos anteriores, a sumarização automática de textos é de extrema importância na área de PLN. Dentro das principais abordagens de sumarização automática, as relacionadas com a sumarização extrativa têm sido as mais estudadas pelos pesquisadores da área. Por outro lado, sumarização abstrativa ainda representa um grande problema em PLN. Embora tais estudos tenham avançado na última década, ainda não são obtidos resultados adequados em comparação com os humanos. Entretanto, a sumarização por extração ainda não pode ser considerada um problema fácil de resolver.

Uma das dificuldades ocorre na questão básica de qual informação extrair, ou que informação excluir da extração para a conformação do sumário. Este problema é intrínseco da fase de extração de informação, típica destes modelos.

De forma geral, os modelos desenvolvidos para solucionar o problema de extração de informação ou sentenças no âmbito da SAT caracterizam-se pela aplicação de uma série de processos que têm como base o uso de determinadas características do texto original com o objetivo de identificar que partes do mesmo devem ser extraídas.

Alguns autores como [Luhn(1958),Edmundson(1969),Paice(1981),Lehmam(1999),Teufel e Moens(2002)], tem apontado a detecção de determinadas características estruturais internas aos textos e a aplicação deste conhecimento na tarefa da SAT, com ênfase na sumarização por extração. Especificamente no domínio da SAT de textos científicos, Lehmam [Lehmam(1999)] fazendo uso do conhecimento estrutural dos artigos científicos, propôs um modelo de SAT no qual a extração da sentenças está condicionada à existência de uma serie de frases identificadoras ou sentenças (sentence identifier fragments, SIFs no inglês), predefinidas após estudo sobre um conjunto de artigos científicos. A pesar da abordagem de Lehman brindar resultados promissores, apresenta uma série de problemas herdados do uso de modelos definidos por (Paice [Paice(1981)], Edmundson [Edmundson(1969)]). Por exemplo, a necessidade de um banco de SIFs muito grande para

aumentar a generalização do modelo, a omissão do uso de outras características importantes como frases típicas negativas (FTNs, neste trabalho), para minimizar o trabalho computacional, e ainda, melhorar a qualidade do sumário final. O modelo de Lehman proposto no artigo [Lehmam(1999)] carece de avaliação sobre artigos completos. A avaliação apresentada por Lehman é restrita a porções de textos, o que não permite verificar o comportamento deste modelo sobre volumes relativamente grandes de sentenças.

Em correspondência a isto é possível dizer que uns dos problemas principais existentes na maioria dos modelos baseados em conhecimento, já mencionados [Luhn(1958), Edmundson(1969), Paice(1981), Lehmam(1999)] é que a identificação e generalização destas características apoia-se em determinados conjuntos de regras estáticas, dadas por gramáticas. Devido à contradição entre a natureza estática destas e a linguagem natural, que não é estática, geralmente estes modelos servem mais para domínios muito específicos.

As bases do modelo de extração de sentenças proposto neste trabalho não diferem muito daquelas de outros modelos existentes. No entanto, é possível dizer que o mesmo possui um caráter inovador, devido a basear-se totalmente no modelo de avaliação de similaridade semântica textual (ver capítulo 3 Modelo de Estimação de STS), o que lhe confere alto grau de generalização, em decorrência de usar o conceito de similaridade semântica, em vez de regras estáticas, mantendo compatibilidade com a linguagem natural.

Em nossa metodologia de sumarização, a avaliação da similaridade semântica é realizada através do modelo de STS definido no capítulo 3. Dito processo utiliza determinadas frases típicas (ver seção 2.1), obtidas da leitura de vários artigos.

De acordo estas bases, este modelo se caracteriza pela identificação e seleção de sentenças relevantes, enquanto ignora ou exclui outras. As sentenças selecionadas constituirão um sumário bruto, o qual será depois pós-processado mediante a aplicação do modelo de eliminação de paráfrases (ver capítulo 5.1). Para isto, o modelo em questão faz uso da informação obtida da leitura e análise estrutural de muitos artigos, que permite a identificação de quatro características fundamentais (ver seção 2.1):

1. Uma estrutura básica comum entre os artigos técnico-científicos.
2. As seções na estrutura básica possuem títulos característicos que permitem sua identificação nos artigos.
3. Cada seção possui uma série de frases típicas, as quais permitem identificar as sentenças mais importantes (na realidade, algumas frases podem aparecer em mais de uma seção).

4. Existe um conjunto frases típicas que impactam negativamente no sumário.

4.2 Antecipaçāo de problemas

Partindo da estrutura básica dos artigos científicos já comentada, o seu uso no modelo proposto traz uma série de problemas derivados da natureza da mesma. Para identificar estes problemas partimos de uma estratégia bottom-up estabelecendo os seguintes objetivos:

1. Todas as sentenças extraídas devem fazer parte de alguma das seções da estrutura básica.
2. Todas as sentenças extraídas devem conter FTs ou frases equivalentes a estas.
3. Não podem ser extraídas sentenças que contenham algum tipo de FTNs.

Analizando estes objetivos, é possível observar que os 3 compartilham um problema básico, a identificação de TTs, FTs, e FTNs (ver seção 2.1).

Um outro problema é que muitas FTs aparecem em mais de uma seção no artigo, conforme as diferentes escritas dos autores. Se cada FT corresponder a apenas uma seção, então sentenças de outras seções poderão deixar de ser extraídas, se contiverem tais FTs. Para evitar esse problema, repetimos algumas FTs em mais de uma seção. Pode ser o caso das seções de Discussão e Conclusão, ou mesmo, Resultados Experimentais.

Por outro lado, uma vez que estas FTs são específicas para cada seção, o modelo deverá processar cada seção em separado para identificar e extrair corretamente as sentenças que contenham FTs associadas a cada seção especificamente. Isto mostra outro problema a ser resolvido pelo modelo, o alinhamento entre a estrutura básica e o artigo.

4.3 Visão general do modelo de Extração de Sentenças (ES)

Para dar solução aos problemas previamente mencionados, como é mostrado na figura 12, o modelo proposto conta com três fases: Eliminação das Sentenças que contenham FTNs, Alinhamento Estrutural e por último Extração das Sentenças que contenham FTs.

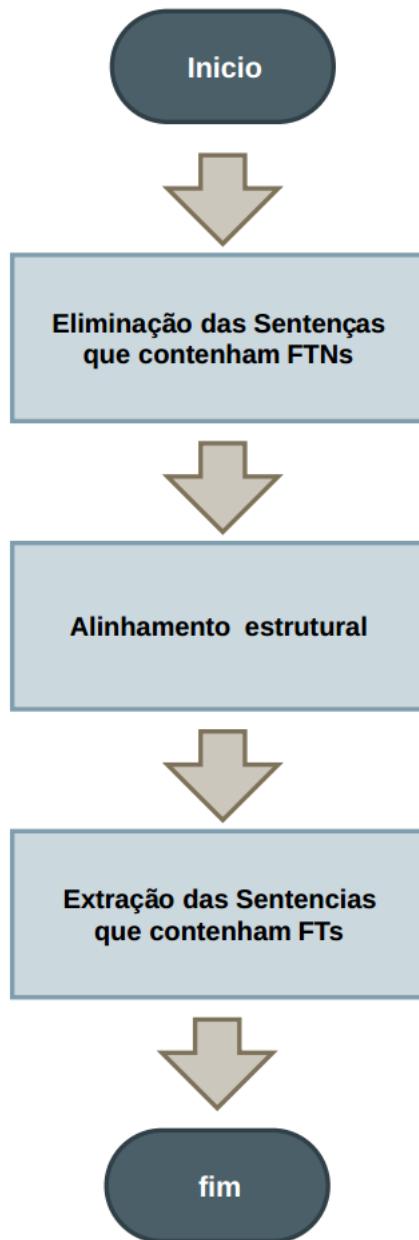


Figura 12: Fases do modelo de extração de sentenças

A finalidade de dito modelo é, partindo da informação obtida da estrutura básica de artigos científicos (estrutura comum, TTs, repositório de FTs e FTNs) (ver seção 2.1) e mediante o uso de uma medida de similaridade semântica, permitir, mediante o alinhamento estrutural, a extração das sentenças mais importantes do artigo e a remoção de outras, para efeito de sumarização. Estas sentenças conformarão um sumário bruto, que será um primeiro passo para a formação de um sumário indicativo, que responderá às seguintes características:

1. O sumário deverá contemplar as ideias chaves do texto original, ao conter as sen-

tenças mais importantes.

2. O sumário deverá conter ao menos uma sentença de cada seção da estrutura básica identificável pelos títulos.
3. O sumário deverá ser legível e coerente.

4.4 Uso do modelo de STS

Como já foi tratado no começo do presente capítulo, pode-se observar a centralidade do modelo de STS no modelo de ES proposto, o qual é usado praticamente em todas as fases identificadas.

Com o uso deste modelo de STS busca-se conseguir uma generalização de TTs, FTs e FTNs adequada. Este ponto encontra-se suportado no pressuposto de que, enquanto outros modelos baseiam-se no caráter estrutural das sentenças, o que limita a generalização ao número de estruturas a serem reconhecidas, o uso da semântica, como em nosso caso, permite uma generalização muito superior, já que o reconhecimento é condicionado pelo número de sentenças que tenham o sentido procurado. Isto posto, o repositório de FTs se converte automaticamente num repositório de sentidos, que vai ser usado para identificar e extrair as sentenças que contenham ao menos um dos sentidos no repositório, como será observado à frente.

4.5 Alinhamento estrutural

A partir da estrutura básica de um artigo científico, o objetivo desta fase é o reconhecimento de cada seção do artigo, tendo como referência os TTs das seções de tal estrutura. Como já foi mencionado, é preciso realizar esta tarefa para conseguir separar a fase de identificação de FTs por seção, uma vez que como foi indicado, as FTs estão associadas às seções da estrutura de base (ver seção 2.1).

Uma possível técnica para o reconhecimento é comparar o título da seção da estrutura de base com os correspondentes do artigo, reconhecendo a semelhança entre estes. Tal técnica, entretanto, só funcionaria para casos específicos, nos quais os títulos das seções fossem literalmente os mesmos da estrutura. Ou seja, muitas vezes os títulos variam, ainda que expressando o mesmo sentido.

É possível visualizar um exemplo disto nos seguintes títulos das seções encontradas

Estrutura básica	DTC
Conclusion	Conclusions and future research
Method	SVM Classifier Model

Tabela 5: Exemplos de TTs na estrutura básica e como podem aparecer nos DTC

nos artigos analisados neste trabalho:

Experimental results, e **Simulation examples**, seriam equivalentes ou estariam alinhados com os títulos **Experiments**, **Simulation results**, **Experimental evaluation**, entre outros.

A técnica aplicada pelo modelo proposto é justamente a aplicação do modelo de STS na comparação destes títulos. Desta forma é possível detectar a equivalência entre estes, quando o valor estimado de similaridade que determina o nível de equivalência exceder um determinado limiar.

Como foi observado na seção 2.1, outra característica dos TTs das seções da estrutura básica é o tamanho deles medido em palavras, o qual geralmente é pequeno. Por outro lado, os títulos dos artigos científicos podem ser extensos ou pouco maiores. Esta característica mostra um ponto interessante, em muitos casos, além destes diferirem em tamanho, continuam sendo equivalentes, para efeitos de alinhamento estrutural.

Existem diversas causas pelas quais aparece este fenômeno: o autor tenta oferecer mais detalhes, com o título, sobre o que vai dizer na seção; ou bem, o autor coloca informação específica sobre o trabalho, no título, entre outras causas. Exemplos destas práticas podem ser vistos na tabela 5

Este fenômeno representa um problema para a aplicação do modelo de STS na estimativa do nível de equivalência entre estes títulos já que qualquer processo de estimativa de similaridade semântica, inclusive o executado por um humano, é sensível à quantidade informacional dos textos comparados, devido a sua principal característica, comparação de sentidos. Ao comparar esses títulos diretamente, é de se esperar que o nível de similaridade entre eles seja penalizado até o ponto de não exceder o limiar de equivalência. Mas, existe uma relação de continência entre estes títulos como é possível observar. Títulos equivalentes mais extensos contêm TTs ou equivalentes dentro de seu significado.

Baseado neste princípio, a aplicação do modelo de STS ao modelo de extração de sentenças proposto é realizada mediante o uso de uma janela de comprimento variável, descrita à frente.

O princípio de operação do modelo de ES nesta fase se caracteriza por realizar comparações entre os títulos das seções da estrutura básica e os títulos do artigo mediante o modelo de STS e uso da janela mencionada. Uma vez que os títulos são equivalentes, então eles são alinhados. Este alinhamento será usado posteriormente na fase de extração.

Existem certos títulos do artigo que não guardam nenhuma relação com os TTs da estrutura base, e portanto não podem ser alinhados diretamente. Estes títulos geralmente são muito específicos ao tema do artigo, e o autor os utiliza para melhor orientar o leitor através da metodologia. Este princípio é usado nesta fase, para realizar o alinhamento destes títulos não reconhecidos com a seção Metodologia da estrutura base.

4.6 O modelo de janela variável sobre STS

Como foi mencionado na seção anterior, a aplicação do modelo de STS no contexto de ES está sujeita a uma série de problemas relacionados com o tamanho dos textos comparados. Para permitir a aplicação do modelo de STS sobre estas condições especiais, e de acordo com as características do problema, define-se um modelo de janela variável ou VWM (Variable Window Model, em inglês) para permitir a detecção de TTs, FTs e FTNs sobre qualquer texto.

A principal característica deste modelo é o emprego de uma janela de tamanho variável sobre o texto a ser comparado. O tamanho da janela é definido pelo tamanho, em número de palavras, das TTs, FTs ou FTNs que se desejam identificar. A essência de funcionamento deste modelo está no uso de uma estratégia de corte sobre a sentença mais longa, para conseguir realizar uma comparação sem penalização no grau de estimação de equivalência, como foi descrito na seção anterior.

Em correspondência ao dito anteriormente, neste modelo são usados NGrams como estratégia de corte. Primeiro são obtidos todos os NGrams do texto [Jurafsky e Martin(2009)] com tamanho igual à frase típica. Depois, compara-se cada n-gram com a frase mediante o modelo de STS. Se o valor da estimação exceder um limiar de equivalência determinado, então significa que a sentença ou título possui aquela determinada frase típica que se quer identificar.

Na fase de alinhamento estrutural, este modelo é usado para detectar a equivalência entre os TTs e os títulos do artigo. Já na fase de eliminação de sentenças com FTNs e na de extração, descritas a continuação, este modelo é usado para a identificação destes tipos de frases sobre as sentenças processadas.

4.7 Eliminação de sentenças com FTNs

Seguindo os princípios definidos no modelo de ES proposto neste trabalho, nesta fase, partindo da característica já abordada sobre a existência de FTNs (ver seção 2.1), são identificadas as sentenças contendo este tipo de frases, para serem excluídas do processamento.

A partir de um repositório de FTNs pré-definido e associado a estrutura básica, para cada FTN é aplicado o modelo de VWM para cada sentença no documento. Se o modelo indicar existência de uma relação de continência entre a FTN e a sentença processada, esta sentença será excluída do processamento.

Na área de sumarização por extração, este tipo de processamento é realizado geralmente por diversos modelos, uma vez que a maior parte da informação num texto qualquer, não é relevante para o sumário. Devido a isto, a detecção e eliminação do máximo de informação desnecessária, previamente à extração das sentenças relevantes, é um ponto de grande importância, já que permite a eliminação de ruído no processamento, assim como diminui o trabalho computacional.

4.8 Extração

Na fase final do processamento do modelo de extração, é realizada a identificação e extração das sentenças mais importantes do artigo científico. Dito processo encontra-se caracterizado pelo reconhecimento e extração de sentenças contendo FTs, para cada par de seções do alinhamento. Para isso, cada par de seções alinhadas é processado de forma independente, e posteriormente as sentenças extraídas são reunidas num estágio final.

A partir do repositório de FTs associadas à seção da estrutura básica alinhada, e para cada FT, é aplicado o modelo de VWM sobre cada sentença da seção do documento. Se o modelo indicar que existe uma relação de continência entre a FT e a sentença processada, esta sentença será extraída para o sumário bruto.

Uma vez processadas todas as seções, as sentenças são organizadas na mesma ordem em que aparecem no documento, formando assim um sumário bruto que corresponde às características apontadas no começo do capítulo.

4.9 Exemplos trabalhados

Para ilustrar a metodologia de extração de sentenças mediante o uso do modelo de estimação de STS, selecionamos as seções: *I. Introduction* e *II. Distributed Support Vector Machine* (esta última alinhada com a seção Metodologia, na estrutura básica) do artigo [Lu *et al.*(2008)Lu, Roychowdhury, e Vandenberghe](ver anexo A).

Desse texto, foram extraídas sentenças para compor um sumário longo, e um curto. A seguir, apresentamos as correspondentes sentenças extraídas(ver apêndices A.1 e A.2), conforme saída do programa.

Sumário Longo:

[SUMMARY SESSION] Introduction: INTRODUCTION

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields.

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN).

The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors.

We prove that our algorithm converges to a globally optimal classifier for arbitrarily distributed data over an SCN.

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity.

Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge.

[SUMMARY SESSION] Methodology : DISTRIBUTED SUPPORT VECTOR MACHINE

In an SVM training problem, we are seeking a hyperplane to separate a set of positively and negatively labeled training data.

SVs are a natural representation of the discriminant information of the underlying classification problem.



Sumário curto:

[SUMMARY SESSION] Introduction: INTRODUCTION

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields .

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN).

The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. We prove that our algorithm converges to a globally optimal classifier for arbitrarily distributed data over an SCN.

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation.

We show that the efficiency of DPSVM and the overall communication overhead can

be improved by controlling the network sparsity.

[SUMMARY SESSION] Methodology : DISTRIBUTED SUPPORT VECTOR MACHINE

Nenhuma sentença foi extraída

Inicialmente, observamos que, de acordo com a descrição anterior da metodologia, a quantidade de sentenças extraídas é regulada através do limiar de similaridade da janela, sendo que limiares maiores correspondem a sumários mais curtos como pode ser observado na table 6. Assim, verifica-se que a última sentença extraída da seção INTRODUCTION, para o sumário longo, não foi extraída para o sumário curto. Ainda em função desse limiar, enquanto para a seção DISTRIBUTED SUPPORT VECTOR MACHINE foram extraídas duas sentenças para o sumário longo, para o curto, nenhuma foi extraída.

Tipo de limiar	Curto	Longo
SEMANTIC_THRESHOLD	0.6	0.6
STATISTIC_CKPD_THRESHOLD	0.6	0.6
WINDOWS_SIM_THRESHOLD	0.55	0.4
TITLE_WINDOWS_SIM_THRESHOLD	0.8	0.8
TPFN_WINDOWS_SIM_THRESHOLD	0.95	0.95
PARAPHRASE_STATISTIC_CKPD_THRESHOLD	0.6	0.6

Tabela 6: Limiares usados pelo sistema para a obtenção dos sumários curto e longo.

Neste caso, a diferença nos tamanhos dos sumários integrados das duas seções, longo e curto, é de três sentenças. O exemplo consegue, assim, ilustrar o princípio da regulação da extensão dos sumários.

5 LIMPEZA DO TEXTO

Denominamos sumário bruto o conjunto de todas as sentenças extraídas, ordenadas segundo a sequência em que aparecem no DTC original. O sumário bruto extraído normalmente apresenta alguns tipos de problemas à leitura, tais como [Jurafsky e Martin(2009)] presença de frases não essenciais em algumas sentenças, problemas de coerência, resultantes, por exemplo, de anáforas, bem como a existência de paráfrases. De modo geral, a solução de tais problemas é complexa. Neste breve capítulo, vamos tratar apenas da resolução do problema de paráfrases.

5.1 Reconhecimento e eliminação de paráfrases

Normalmente, um artigo científico pode conter algumas centenas de sentenças. Algumas dessas sentenças, perfeitamente extraíveis, podem representar paráfrases, isto é, uma mesma ideia retrabalhada com outras palavras, para facilitar a descrição. Nessas condições, num sumário, que pode conter, tipicamente, cerca de dez sentenças, não seria conveniente inserir uma dessas sentenças repetidas.

Para o reconhecimento de paráfrase, mais uma vez vamos recorrer ao nosso modelo de avaliação de similaridade semântica textual. Neste caso, o modelo é usado para reconhecer similaridade semântica entre duas sentenças, tal como no capítulo 4, também com uso de uma janela deslizante, de comprimento variável, igual ao número de palavras da menor das sentenças do par comparado. Utilizamos o limiar de janela de 0.6.

Se detectada uma paráfrase, elimina-se a menor das sentenças, desde que não pertença à Conclusão. Se uma delas pertence à Conclusão, então esta é a que permanece, independentemente de sua extensão.

5.2 Exemplos trabalhados

Para ilustrar o funcionamento do Módulo de Reconhecimento e Eliminação de Paráfrase, vamos utilizar o mesmo sumário curto já apresentado no capítulo 4, mas desta vez na sua versão completa, e com as duas sentenças que representam paráfrase assinaladas.

A seguir, apresenta-se a saída do programa com o processamento referente ao reconhecimento e à eliminação da paráfrase, tal como mostrado no sumário resultante, neste caso, com a exclusão da sentença que não pertence à Conclusão (ver apêndice A.1 e A.4) usando os limiares mostrados na tabela 6.

No sumário logo abaixo, num trecho de saída do modelo de sumarização, mostram-se, em negrito, as duas sentenças que constituem uma paráfrase. Após, indica-se o processamento para reconhecimento e eliminação da paráfrase, e finalmente, o sumário com a sentença que restou.

[RAW SUMMARY]

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields. In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. ***We prove that our algorithm converges to a globally optimal classifier for arbitrarily distributed data over an SCN.*** The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Networks of sites 7 and 15 were tested for the two types of random generated networks. ***The algorithm has been proved to converge to the global optimal classifier in finite steps.*** Experiments over a real world database show that this algorithm is scalable and robust.

[REMOVE PARAFRASE]

[PARAPHRASE IDENTIFIED SENTENCES A and B]

[PARAPHRASE IDENTIFIED SENTENCES A]: We prove that our algorithm converges to a globally optimal classifier (at every site) for arbitrarily distributed data over an SCN.

[PARAPHRASE IDENTIFIED SENTENCES B]: The algorithm has been proved to converge to the global optimal classifier in finite steps.

[SIMILARITY VALUE]: 0.608920

[REMOVE SENTECE PARAPHRASE A]: We prove that our algorithm converges to a globally optimal classifier (at every site) for arbitrarily distributed data over an SCN.



[SUMMARY WITHOUT PARAPHRASE]

The support vector machine (SVM), which implements the principle of structural error minimization, is one of the most popular classification algorithms in many fields. In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Networks of sites 7 and 15 were tested for the two types of random generated networks. *The algorithm has been proved to converge to the global optimal classifier in finite steps.* Experiments over a real world database show that this algorithm is scalable and robust.

6 USO INTEGRADO DO MODELO DE SUMARIZAÇÃO

Neste capítulo, apresentamos o resultado global, detalhado do processamento do artigo trabalhado (ver apêndice A e anexo A) para efeito de obtenção automática de um sumário, nas versões curto e longo.

O sumário curto corresponderia ao Abstract do artigo, feito pelos autores. Já, o sumário longo não encontra um paralelo na prática dos periódicos científicos. Poderia, assim, preencher uma função de apresentar ao possível leitor do artigo, uma prévia mais completa, com informações adicionais, não encontráveis no abstract. As saídas do programa abaixo mostram as sentenças extraídas das diferentes secções, o sumário longo resultante, a eliminação da paráfrase, e finalmente o sumário final.

Observar que no sumário consta, em negrito, uma sentença, que não deveria ter sido extraída, isto porque contém números, que não interessam ao sumário. Tal sentença poderia ter sido excluída, como depois o foi, com o uso de uma expressão regular.

6.1 Exemplos Trabalhados

[RAW SUMMARY]

[SUMMARY SESSION] Introduction: INTRODUCTION

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields.

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly

connected network (SCN).

The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors.

We prove that our algorithm converges to a globally optimal classifier for arbitrarily distributed data over an SCN.

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation.

We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity.

Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge.

[SUMMARY SESSION] Methodology : DISTRIBUTED SUPPORT VECTOR MACHINE

In an SVM training problem, we are seeking a hyperplane to separate a set of positively and negatively labeled training data. SVs are a natural representation of the discriminant information of the underlying classification problem.

[SUMMARY SESSION] Methodology : PROOF OF CONVERGENCE

Nenhuma sentença extraída

[SUMMARY SESSION] Methodology : ALGORITHM IMPLEMENTATION

In this paper, we propose an intuitive method to generate a parameter that has good

performance to the global optimization problem by using locally available partial training data and statistics of the global training data.

One may observe that the objective value is monotonously increasing while the test accuracy is improving, which may not be monotonic.

Since the DPSVM constructs a local problem by adding critical data, local SVM training problems in subsequent iterations are essentially problem adjustments, which append variables and constraints based on the problem of the last iteration.

[SUMMARY SESSION] Experimental results : PERFORMANCE STUDIES

We test our algorithm over a realword data set: MNIST database of handwritten digits.

We test our algorithm over an array of network configurations.

Networks of sites 7 and 15 were tested for the two types of random generated networks.

The results show that our algorithm scales very well for all network topologies except for the ring network, given the size of the network is limited.

We compare the test accuracy of DPSVM to the minimum, maximum, and average values of the test accuracy of normal SVM among all sites.

[SUMMARY SESSION] Conclusion : CONCLUSION AND FUTURE RESEARCH

The algorithm has been proved to converge to the global optimal classifier in finite steps.

Experiments over a realworld database show that this algorithm is scalable and robust.

The DPSVM algorithm is able to work on multiple arbitrarily partitioned working sets and achieve close to linear scalability if the size of the network is not too large.

[SUMMARY WITH PARAPHRASES]

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields. In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. We prove that our algorithm converges to a globally optimal classifier for arbitrarily distributed data over an SCN. The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge. In an SVM training problem, we are seeking a hyper-plane to separate a set of positively and negatively labeled training data. SVs are a natural representation of the discriminant information of the underlying classification problem. In this paper, we propose an intuitive method to generate a parameter that has good performance to the global optimization problem by using locally available partial training data and statistics of the global training data. One may observe that the objective value is monotonously increasing while the test accuracy is improving, which may not be monotonic. Since the DPSVM constructs a local problem by adding critical data, local SVM training problems in subsequent iterations are essentially problem adjustments, which append variables and constraints based on the problem of the last iteration. We test our algorithm over a real word data set: MNIST database of handwritten digits. We test our algorithm over an array of network configurations. ***Networks of sites 7 and 15 were tested for the two types of random generated networks.*** The results show that our algorithm scales very well for all network topologies except for the ring network, given the size of the network is limited. We compare the test accuracy of DPSVM to the minimum, maximum, and average values of the test accuracy of normal SVM among all sites. The algorithm has been proved to converge to the global optimal classifier in finite steps. Experiments over a real world database show that this algorithm is scalable and robust. The DPSVM algorithm is able to work on multiple arbitrarily partitioned working sets and achieve

close to linear scalability if the size of the network is not too large.

[REMOVE PARAFRASE]

[PARAPHRASE IDENTIFIED SENTENCES A and B]

[PARAPHRASE IDENTIFIED SENTENCES A]: We prove that our algorithm converges to a globally optimal classifier (at every site) for arbitrarily distributed data over an SCN.

[PARAPHRASE IDENTIFIED SENTENCES B]: The algorithm has been proved to converge to the global optimal classifier in finite steps.

[SIMILARITY VALUE]: 0.608920

[REMOVE SENTCE PARAPHRASE A]: We prove that our algorithm converges to a globally optimal classifier (at every site) for arbitrarily distributed data over an SCN.



[SUMMARY WITHOUT PARAPHRASE]

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields. In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge. In an

SVM training problem, we are seeking a hyperplane to separate a set of positively and negatively labeled training data. SVs are a natural representation of the discriminant information of the underlying classification problem. In this paper, we propose an intuitive method to generate a parameter that has good performance to the global optimization problem by using locally available partial training data and statistics of the global training data. One may observe that the objective value is monotonously increasing while the test accuracy is improving, which may not be monotonic. Since the DPSVM constructs a local problem by adding critical data, local SVM training problems in subsequent iterations are essentially problem adjustments, which append variables and constraints based on the problem of the last iteration. We test our algorithm over a real word data set: MNIST database of handwritten digits .We test our algorithm over an array of network configurations. ***Networks of sites 7 and 15 were tested for the two types of random generated networks.*** The results show that our algorithm scales very well for all network topologies except for the ring network, given the size of the network is limited. We compare the test accuracy of DPSVM to the minimum, maximum, and average values of the test accuracy of normal SVM among all sites. The algorithm has been proved to converge to the global optimal classifier in finite steps. Experiments over a real world database show that this algorithm is scalable and robust. The DPSVM algorithm is able to work on multiple arbitrarily partitioned working sets and achieve close to linear scalability if the size of the network is not too large.

7 VALIDAÇÃO

A questão da validação é muito importante nas aplicações de PLN. Segundo [Manning e Schütze(1999)], o sucesso de uma aplicação é, em última análise, o bom desempenho na tarefa.

O sumário obtido com o modelo proposto neste trabalho será comparado com um sumário de referência do artigo considerado, obtido por aplicação manual das regras do sistema proposto.

Este sumário padrão seria o seguinte, inicialmente em sentenças isoladas:

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields .

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN).

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation.

We test our algorithm over a real-world data set: MNIST database of handwritten digits.

The results show that evenly distributed data may result in a fast convergence (less iteration and shorter CPU time) while the special portional distribution (eight sites of data and seven empty sites) obviously achieves less communication overhead.

The algorithm has been proved to converge to the global optimal classifier in finite

steps.

Experiments over a real world database show that this algorithm is scalable and robust.

E, na forma de texto corrido:

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields . In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We test our algorithm over a real-world data set: MNIST database of handwritten digits. The results show that evenly distributed data may result in a fast convergence (less iteration and shorter CPU time) while the special portional distribution (eight sites of data and seven empty sites) obviously achieves less communication overhead. The algorithm has been proved to converge to the global optimal classifier in finite steps. Experiments over a real world database show that this algorithm is scalable and robust.

7.1 Validação por Recall e Precision

Para aplicar este tipo de validação, vamos considerar as sentenças do artigo como pertencentes a dois grupos, relevantes para serem extraídas, e não relevantes. As sentenças do sumário obtido pelo modelo, e que também tenham sido extraídas para o sumário do expert (ou seja, extraídas relevantes), pertencem ao grupo true positive, e são designadas por tp. Aquelas sentenças extraídas pelo modelo, e que não constam no sumário expert (ou seja, extraídas não-relevantes), são designadas por false positive, representadas por fp. Finalmente, aquelas que deveriam ter sido extraídas e não o foram, são designadas por fn.

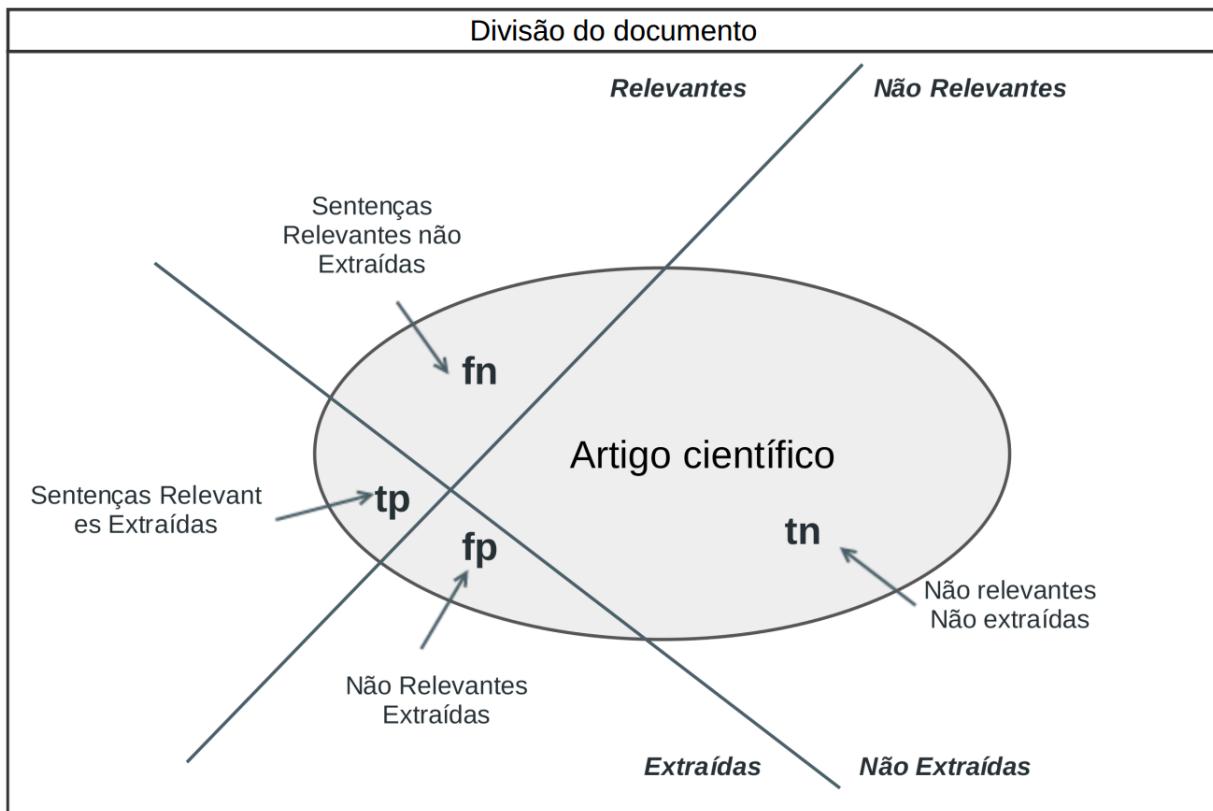


Figura 13: Divisão do documento segundo sentenças extraídas.

A figura 13, adaptada de [Zervanou(1998)], indica a divisão do documento (artigo) quanto aos diferentes tipos de sentenças para efeito de extração.

Repetimos aqui o sumário curto obtido, para efeito de obtenção dos valores de tp, fp, e fn:

The support vector machine (SVM), which implements the principle of structural error minimization , is one of the most popular classification algorithms in many fields.

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN).

The basic idea of DPSVM is to exchange SVs over an SCN and update the local solutions iteratively, based on the SVs that a particular site receives from its neighbors.

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc.,

which have dramatic impact on the performance in terms of convergence speed and data accumulation.

We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity.

The algorithm has been proved to converge to the global optimal classifier in finite steps.

Experiments over a real world database show that this algorithm is scalable and robust.

Calculando, resultam:

$$\mathbf{tp = 4}$$

$$\mathbf{fn = 3}$$

$$\mathbf{fp = 2}$$

Aplicando as fórmulas de precision e recall, vem:

$$\mathbf{precision = tp/(tp + fp) = 4/(4 + 2) = 0.66}$$

$$\mathbf{recall = tp/(tp + fn) = 4/(4 + 3) = 0.57}$$

O número de sentenças não relevantes e não extraídas, tn, é muito grande, e não foram contadas. Assim, não calculamos os índices de acurácia (accuracy), e erro (error), que resultariam em valores muito pequenos. Entretanto, nesses casos, os índices de precision e recall são mais adequados para avaliar a qualidade do sumário [Manning e Schütze(1999)].

Por outro lado, podemos dizer que o valor da avaliação realizada pode ser relativizado em parte, pois se baseia em gold standard feito por um expert humano, que tem algum grau de subjetividade. Teríamos um padrão mais significativo no caso de obtenção por média entre vários sumários de experts.

7.2 Validação por Rouge

Uma das métricas mais usadas no universo da SAT para avaliar a qualidade de sumários automáticos se denomina ROUGE (Recall-Oriented Understudy for Gisting Evaluation, em inglês) [Lin e Hovy(2003), Lin(2004)], a qual é inspirada na métrica BLEU (Bilingual Evaluation Understudy, em inglês), embora a BLEU seja usada na área de

tradução automática de textos e esteja baseado na métrica estatística Precision, no entanto o Rouge encontra-se baseado no Recall [Jurafsky e Martin(2009)].

Existem diversas variantes de ROUGE, sendo uma das mais comuns, a ROUGE-N [Lin(2004)], onde, para um documento D, um sumário automático (gerado por um sistema de SAT) X deste documento e sumários de D produzidos por humanos (sumários de referência) R_n, o ROUGE-N determina que percentual de n-grams do X que estão presentes em R_n, se o valor deste percentual é elevado então pode-se afirmar que a qualidade do sumário gerado pelo sistema é boa.

Neste trabalho são aplicadas as variantes de ROUGE-N: ROUGE-1, ROUGE-2, ROUGE-3, ou seja Rouge usando unigrams, bigrams e trigrams. Dita aplicação foi realizada utilizando uma implementação do Rouge-N em Java, localizada na plataforma de desenvolvimento colaborativo GitHub, onde, o ROUGE-N encontra-se definido pela formula clássica [Lin(2004)] 7.1 seguinte:

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{RefSumaries}\}} \sum_{gram_i \in S} \text{Count}_{match}(gram_i)}{\sum_{S \in \{\text{RefSumaries}\}} \sum_{gram_i \in S} \text{Count}(gram_i)} \quad (7.1)$$

onde:

$$\text{Count}_{match}(gram_i) = min(count(i, X), count(i, S)) \quad (7.2)$$

e:

$$\text{Count}(gram_i) = count(i, S) \quad (7.3)$$

Na formula 7.1 temos que o denominador esta definido pela contagem de todos os n-grams de todas as sentenças de todos os sumários de referencia, e o numerador, para todos os n-grams de todas a sentenças de todos os sumários de referencia, pela contagem mínima dos n-grams entre o sumário produzido pelo sistema e o sumário de referencia, ou seja, a contagem dos n-grams que ocorrem, tanto no sumário obtido pelo sistema como nos sumários de referencia. Finalmente o valor da aplicação do Rouge-N encontra-se determinado pela divisão destas das duas contagens mencionadas.

Para aplicar o Rouge-N na avaliação do modelo de SAT proposto neste trabalho, damos continuidade ao uso do artigo, **Distributed Parallel Support Vector Machines**

in Strongly Connected Networks, além de o uso, tanto do sumário curto gerado pelo sistema como do sumário de referência mostrados na seção anterior (ver seção 7.1). Partindo deste ponto, os parâmetros de aplicação do ROUGE-N são:

D = Distributed Parallel Support Vector Machines in Strongly Connected Networks

X = Sumário curto gerado pelo sistema (ver seção 7.1)

R1 = Sumário gerado pelo expert (ver seção 7.1)

Os resultado da aplicação do Rouge-N nas 3 variantes mencionadas, são:

Rouge-1 = 0.77647

Rouge-2 = 0.70659

Rouge-3 = 0.68293

Os resultados obtidos pelo ROUGE mostram que o sumário gerado pelo sistema proposto apresentam qualidade aceitável para a tarefa a que se propõe, dada a existência de percentagem relativamente alta de n-grams comuns entre os sumários comparados.

8 CONCLUSÕES

Com base nos resultados obtidos, podemos tirar as seguintes breves conclusões deste trabalho:

1. O modelo de avaliação de similaridade semântica textual desenvolvido representa adequada métrica para avaliar a similaridade semântica entre pares de sentenças.
2. A aplicação de tal modelo à sumarização automática de artigos científicos mostrou resultados promissores, permitindo a obtenção de sumários utilizáveis.
3. A aplicação do conceito de janela deslizante variável mostrou-se útil na extração de sentenças.
4. Os resultados, no entanto, podem e devem ser melhorados em trabalhos futuros.

9 CONTRIBUIÇÕES

Entre os principais aportes deste trabalho encontra-se:

1. Definição de um modelo original à tarefa de estimação de similaridade semântica textual, o qual foi apresentado no WorkShop (FETLT 2016).
2. A aplicação deste modelo como base de um sistema de sumarização automática de documentos técnicos-científicos igualmente resultante deste trabalho.

10 TRABALHOS FUTUROS

Propõe-se como trabalhos futuros:

1. Realizar uma experimentação mais exaustiva do modelo de sumarização automática.
2. Realizar uma experimentação do modelo de estimativa de similaridade semântica mudando os algoritmos de estimativa da similaridade semântica entre palavras e similaridade estatística entre frases.
3. Realizar uma experimentação de ambos modelos sobre outras linguagens, como o português.

REFERÊNCIAS

- [Abney(1992)] **Abney(1992)** Steven P. Abney. *Parsing By Chunks*, páginas 257–278. Springer Netherlands.
- [Agirre *et al.*(2012)] **Agirre et al.(2012)** Eneko Agirre, Diab, Cer, e Gonzalez-Agirre] **Agirre et al.(2012)** Eneko Agirre, Mona Diab, Daniel Cer e Aitor Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. Em *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, páginas 385–393. Association for Computational Linguistics. URL <http://dl.acm.org/citation.cfm?id=2387636.2387697>.
- [Agirre *et al.*(2016)] **Agirre et al.(2016)** Agirre, Banea, Cer, Diab, Gonzalez-Agirre, Mihalcea, Rigau, e Wiebe] **Agirre et al.(2016)** Eneko Agirre, Carmen Banea, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau e Janyce Wiebe. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. Em *SemEval@NAACL-HLT*, páginas 497–511. The Association for Computer Linguistics.
- [ANSI-Z39.16(1979)] **ANSI-Z39.16(1979)** ANSI-Z39.16. Preparation of Scientific Papers for Written or Oral Presentation. Standard, American National Standards Institute – ANSI.
- [Bird *et al.*(2009)] **Bird et al.(2009)** Steven Bird, Ewan Klein e Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc.
- [Bond e Foster(2013)] **Bond e Foster(2013)** Francis Bond e Ryan Foster. Linking and extending an open multilingual wordnet. Em *ACL (1)*, páginas 1352–1362. The Association for Computer Linguistics.
- [Brants(2000)] **Brants(2000)** T. Brants. TnT: A Statistical Part-of-Speech Tagger. Em *Proceedings of the 6th Conference on Applied Natural Language Processing*, páginas 224–231. ACL. URL <http://acl.ldc.upenn.edu/A/A00/A00-1031.pdf>.
- [Brill(1994)] **Brill(1994)** Eric Brill. Some advances in transformation-based part of speech tagging. Em *Proceedings of the Twelfth National Conference on Artificial Intelligence (Vol. 1)*, AAAI '94, páginas 722–727. American Association for Artificial Intelligence. URL <http://dl.acm.org/citation.cfm?id=199288.199378>.
- [Burrough-Boenisch(1999)] **Burrough-Boenisch(1999)** Joy Burrough-Boenisch. International reading strategies for imrd articles. *Written Communication*, 16(3):296–316. doi: 10.1177/0741088399016003002.

- [Buscaldi *et al.*(2010)] **Buscaldi et al.(2010)** Davide Buscaldi, Paolo Rosso, José Manuel Gómez-Soriano e Emilio Sanchis. Answering questions with an n-gram based passage retrieval engine. *J. Intell. Inf. Syst.*, 34(2):113–134. doi: 10.1007/s10844-009-0082-y.
- [Buscaldi *et al.*(2012)] **Buscaldi et al.(2012)** Davide Buscaldi, Ronan Tournier, Nathalie Aussenac-Gilles e Josiane Mothe. Irit: Textual similarity combining conceptual similarity with an n-gram comparison method. Em *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, páginas 552–556. Association for Computational Linguistics.
- [Day(1989)] **Day(1989)** R. A. Day. The origins of the scientific paper: The imrad format. *American Medical Writers Association*, 4(2):16–18.
- [Edmundson(1969)] **Edmundson(1969)** H. P. Edmundson. New methods in automatic extracting. *J. ACM*, 16(2):264–285. doi: 10.1145/321510.321519. URL <http://doi.acm.org/10.1145/321510.321519>.
- [Francis e Kucera(1979)] **Francis e Kucera(1979)** W. Nelson Francis e Henry Kucera. The Brown Corpus: A Standard Corpus of Present-Day Edited American English, 1979.
- [Francis *et al.*(1982)] **Francis et al.(1982)** W. Nelson Francis, Henry Kucera e A.W. Mackie. *Frequency analysis of English usage: lexicon and grammar*. Houghton Mifflin.
- [Glasman-Deal(2010)] **Glasman-Deal(2010)** H. Glasman-Deal. *Science Research Writing for Non-native Speakers of English*. Imperial College Press. URL <https://books.google.com.br/books?id=nu9LZ1x818oC>.
- [Gruber(1993)] **Gruber(1993)** T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220.
- [Guarino *et al.*(2009)] **Guarino et al.(2009)** Nicola Guarino, Daniel Oberle e Steffen Staab. What is an ontology? Em *Handbook on ontologies*, páginas 1–17. Springer.
- [Hahn e Mani(2000)] **Hahn e Mani(2000)** Udo Hahn e Inderjeet Mani. The challenges of automatic summarization. *IEEE Computer*, 33(11):29–36.
- [Hollanders e Soete(2010)] **Hollanders e Soete(2010)** Hugo Hollanders e Luc Soete. The growing role of knowledge in the global economy. *UNESCO Science Report 2010*, páginas 5–29.
- [hua Chen e Chen(1994)] **hua Chen e Chen(1994)** Kuang hua Chen e Hsin-Hsi Chen. Extracting noun phrases from large-scale texts: A hybrid approach and its automatic evaluation. Em *Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics*, ACL '94, páginas 234–241. Association for Computational Linguistics. doi: 10.3115/981732.981764.

- [Hulth(2003)] **Hulth(2003)** Anette Hulth. Improved automatic keyword extraction given more linguistic knowledge. Em *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, EMNLP '03, páginas 216–223. Association for Computational Linguistics. doi: 10.3115/1119355.1119383. URL <http://dx.doi.org/10.3115/1119355.1119383>.
- [Hulth e Megyesi(2006)] **Hulth e Megyesi(2006)** Anette Hulth e Beáta B. Megyesi. A study on automatically extracted keywords in text categorization. Em *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, páginas 537–544. Association for Computational Linguistics. doi: 10.3115/1220175.1220243. URL <http://dx.doi.org/10.3115/1220175.1220243>.
- [Jones(1993)] **Jones(1993)** Karen Sparck Jones. What Might Be in a Summary? Em *Information Retrieval 93: Von der Modellierung zur Anwendung*.
- [Jones(1998)] **Jones(1998)** Karen Sparck Jones. Automatic summarising: factors and directions. *CoRR*, cmp-lg/9805011. URL <http://arxiv.org/abs/cmp-lg/9805011>.
- [Jurafsky e Martin(2009)] **Jurafsky e Martin(2009)** Daniel Jurafsky e James H. Martin. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc.
- [Knight e Marcu(2002)] **Knight e Marcu(2002)** Kevin Knight e Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artif. Intell.*, 139(1):91–107. doi: 10.1016/S0004-3702(02)00222-9. URL [http://dx.doi.org/10.1016/S0004-3702\(02\)00222-9](http://dx.doi.org/10.1016/S0004-3702(02)00222-9).
- [Lehmam(1999)] **Lehmam(1999)** Abderrafih Lehman. Text structuration leading to an automatic summary system: Rafi. *Inf. Process. Manage.*, 35(2):181–191. doi: 10.1016/S0306-4573(98)00043-0. URL [http://dx.doi.org/10.1016/S0306-4573\(98\)00043-0](http://dx.doi.org/10.1016/S0306-4573(98)00043-0).
- [Li *et al.*(2006)] **Li *et al.*(2006)** Yuhua Li, David McLean, Bandar O'Shea, e Crockett. Sentence similarity based on semantic nets and corpus statistics. *IEEE Trans. Knowl. Data Eng.*, 18(8): 1138–1150.
- [Lin(2004)] **Lin(2004)** Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. Em *Proc. ACL workshop on Text Summarization Branches Out*, página 10. URL <http://research.microsoft.com/cyl/download/papers/WAS2004.pdf>.
- [Lin e Hovy(2003)] **Lin e Hovy(2003)** Chin-Yew Lin e Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. Em *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, páginas 71–78. Association for Computational Linguistics. doi: 10.3115/1073445.1073465. URL <http://dx.doi.org/10.3115/1073445.1073465>.
- [Lu *et al.*(2008)] **Lu *et al.*(2008)** Yumao Lu, Vwani P. Roychowdhury e Lieven Vandenberghe. Distributed parallel support vector machines in strongly connected networks. *IEEE Transactions on Neural Networks*, 19:1167–1178.

- [Luhn(1958)] **Luhn(1958)** H. P. Luhn. The automatic creation of literature abstracts. *IBM J. Res. Dev.*, 2(2):159–165. doi: 10.1147/rd.22.0159. URL <http://dx.doi.org/10.1147/rd.22.0159>.
- [Manning e Schütze(1999)] **Manning e Schütze(1999)** Christopher D. Manning e Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press.
- [Marcus *et al.*(1993) Marcus, Marcinkiewicz, e Santorini] **Marcus *et al.*(1993)** Mitchell P. Marcus, Mary Ann Marcinkiewicz e Beatrice Santorini. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19(2):313–330. URL <http://dl.acm.org/citation.cfm?id=972470.972475>.
- [Mihalcea *et al.*(2006) Mihalcea, Corley, e Strapparava] **Mihalcea *et al.*(2006)** Rada Mihalcea, Courtney Corley e Carlo Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. Em *AAAI*, páginas 775–780. AAAI Press.
- [Miller *et al.*(1990) Miller, Beckwith, Fellbaum, Gross, e Miller] **Miller *et al.*(1990)** George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross e Katherine Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244.
- [Mitkov(2003)] **Mitkov(2003)** Ruslan Mitkov, editor. *The Oxford Handbook of Computational Linguistics*. Oxford Handbooks in Linguistics.
- [Paice(1981)] **Paice(1981)** C. D. Paice. The automatic generation of literature abstracts: An approach based on the identification of self-indicating phrases. Em *Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval*, SIGIR '80, páginas 172–191. Butterworth & Co. URL <http://dl.acm.org/citation.cfm?id=636669.636680>.
- [Ratnaparkhi(1996)] **Ratnaparkhi(1996)** Adwait Ratnaparkhi. A maximum entropy model for part-of-speech tagging. Em *In Proceedings of the Conference on Empirical Methods in Natural Language Processing*, páginas 133–142. URL <http://www.aclweb.org/anthology-new/W/W96/W96-0213.pdf>.
- [Samuelsson e Voutilainen(1997)] **Samuelsson e Voutilainen(1997)** Christer Samuelsson e Atro Voutilainen. Comparing a linguistic and a stochastic tagger. *CoRR*, cmp-lg/9706005.
- [Teufel e Moens(2002)] **Teufel e Moens(2002)** Simone Teufel e Marc Moens. Summarizing scientific articles: Experiments with relevance and rhetorical status. *Comput. Linguist.*, 28(4):409–445. doi: 10.1162/089120102762671936. URL <http://dx.doi.org/10.1162/089120102762671936>.
- [Toutanova *et al.*(2003) Toutanova, Klein, Manning, e Singer] **Toutanova *et al.*(2003)** Kristina Toutanova, Dan Klein, Christopher D. Manning e Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. Em *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, páginas 173–180. Association for Computational Linguistics. doi: 10.3115/1073445.1073478. URL <http://dx.doi.org/10.3115/1073445.1073478>.

- [Wu e Palmer(1994)] **Wu e Palmer(1994)** Z. Wu e M. Palmer. Verb semantic and lexical selection. Em *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics (ACL-1994)*, páginas 133–138.
- [Zajic et al.(2007)Zajic, Dorr, Lin, e Schwartz] **Zajic et al.(2007)** David Zajic, Bonnie J. Dorr, Jimmy Lin e Richard Schwartz. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing & Management*, 43(6):1549–1570.
- [Zervanou(1998)] **Zervanou(1998)** K. Zervanou. A corpus based approach to text summarization. Dissertação de Mestrado, University of Manchester, England.

APÊNDICE A – ARTIGO SUMARIZADO PELO MODELO DE SAT PROPOSTO

Neste apêndice são mostrados em amarelo as sentenças extraídas pela aplicação do modelo proposto por cada seção da estrutura base sobre o artigo [Lu *et al.*(2008)Lu, Roychowdhury, e Vandenberghe].

A.1 Sentenças da Introdução

mining usually include robustness to changes in the network topology [1], efficient representation for high-dimensional and massive data sets, least synchronization and communication, least duplication, better load balancing, and good decision precision and certainty.

The classification problem is an important problem in data mining. The support vector machine (SVM), which implements the principle of structural error minimization [2], [3] is one of the most popular classification algorithms in many fields [4]–[6].

SVMs are ideally suited for a framework where different sites can potentially exchange only a small number of training vectors. For a data set at a particular site, the support vectors (SVs) are the natural representatives of discriminant information of the database *and* the optimal solution to a local classifier. Distributed support vector machines (DSVMs) and parallel support

however, is suboptimal depending on the definition of R . In order to accelerate DSVM, Graf *et al.* had an algorithm that implemented distributed processors into cascade top-down network topology, namely, cascade SVM [13]. The bottom node of the network is the central processing center. To the best of our knowledge, the cascade SVM is the fastest DSVM that is globally optimal. This is the first time that network topology was taken into consideration to accelerate the distributed training process. However, there is no comparison to other network topologies. It is not clear either that their distributed SVM may converge in a more general network.

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The objective of the distributed classification problem is to classify distributed data, i.e., determine a single global classifier, by judiciously sampling and distributing subsets of data among the various sites. Ideally, a single site or server should not end up storing the complete data set; instead, the different sites should exchange a minimum number of data samples, and together converge to a single global solution in an iterative fashion.

The basic idea of DPSVM is to exchange SVs over an SCN and update (instead of recompute) the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. We prove that our algorithm converges to a globally

Manuscript received January 21, 2007; revised September 5, 2007 and October 30, 2007; accepted November 1, 2007. First published March 12, 2008; last published July 7, 2008 (projected).

Y. Lu is with the Yahoo! Inc., Sunnyvale, CA 94089 USA (e-mail: yu-mao@yahoo-inc.com).

V. Roychowdhury and L. Vandenberghe are with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095 USA.

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TNN.2007.2000061

Figura 14: Três primeiras sentenças da introdução

The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge.

This paper is organized as follows. We present our distributed classification algorithm in the next section followed by the proof of the global convergence in Section III. We introduce the detailed algorithm implementation in Section IV. The performance study is given in Section V. We conclude in Section VI.

II. DISTRIBUTED SUPPORT VECTOR MACHINE

A. Problem

We consider the following problem: the training data are arbitrarily distributed in L sites. Each site is a node within an SCN, defined as follows.

Definition 1. An SCN is a directed network in which it is

introduced. The SVM training problem for the nonseparable problem is defined as follows:

$$\begin{aligned} & \text{minimize} \quad (1/2)g^T g + \gamma \mathbf{1}^T \xi \\ & \text{subject to} \quad y_i g^T \phi(g_i) \geq 1 - \xi_i, \quad i = 1, \dots, N \\ & \quad \xi \geq 0 \end{aligned} \quad (2)$$

where $\mathbf{1}$ is a vector of ones, ϕ is a lifting function, and the scalar γ is called regularization parameter that is usually empirically selected to reduce the testing error rate.

The corresponding dual of the problem (2) is shown as follows:

$$\begin{aligned} & \text{maximize} \quad -(1/2)\alpha^T Q\alpha + \mathbf{1}^T \alpha \\ & \text{subject to} \quad 0 \leq \alpha \leq \gamma \mathbf{1} \end{aligned} \quad (3)$$

where the Gram matrix Q has the component $Q_{ij} = y_i y_j \phi(x_i)^T \phi(x_j)$.

It is common to replace $\phi(x)^T \phi(\hat{x})$ by a kernel function $F(x, \hat{x})$ such that $Q_{ij} = K_{ij} y_i y_j$ and $K_{ij} = F(x_i, x_j)$, where $K \in \mathbb{R}^{N \times N}$ is the so-called kernel matrix. The nonlinear kernel may lift the dimension of training vectors to a higher dimension so that they can be separated linearly. If the kernel matrix K is positive definite, problem (3) is guaranteed to be

Figura 15: Quarta sentença da Introdução

A.2 Sentenças da Metodologia

There are N_l training vectors in site l and N training vectors in all sites, where $N_l, \forall l$ can be an arbitrary integer, such that $\sum_{l=1}^L N_l = N$. Each training vector is denoted by z_i $i = 1, \dots, N$ where $z_i \in \mathbb{R}^n$, and $y_i \in \{+1, -1\}$ is its label.

The global SVM problem, i.e., the traditional centralized problem is briefly summarized in the next section.

B. Support Vector Machine

In an SVM training problem, we are seeking a hyperplane to separate a set of positively and negatively labeled training data. The hyperplane is defined by $w^T z + b = 0$, where the parameter $w \in \mathbb{R}^n$ is a vector orthogonal to the hyperplane and $b \in \mathbb{R}$ is the bias. The decision function is the hyperplane classifier

$$H(x) = \text{sign}(w^T z + b).$$

Correspondingly, we have the decision function of the form

$$H(x) = \text{sign} \left(\sum_{i=1}^N y_i \alpha_i F(x_i, x) \right).$$

For the general SVM optimization problem, the complementary slackness condition has the form

$$\alpha_i [y_i (g^T x_i) - 1 + \xi_i] = 0 \quad \text{for all } i = 1, \dots, N \quad (4)$$

in the optimum. Therefore, SVs are the vectors that lie in between the margin boundary (including those on the boundary) and that are misclassified.

Figura 16: Duas primeiras sentenças da Metodologia.

C. Efficient Information Carrier: SVs

SVs, namely, the training data that have nonzero α values, lie physically on the margin, or are misclassified. Those vectors carry all the classification information of the local data set. Exchanging SVs, instead of moving all the data, is a natural way to fuse information among distributed sites.

SVs are a natural representation of the discriminant information of the underlying classification problem. The basic idea of the DSVMs is to exchange SVs over an SCN and update the local solutions for each site iteratively. We exploit the idea of exchanging SVs in a deterministic way instead of a randomized way [14] in our distributed learning algorithms. Our algorithm is based on the observation that the number of SVs may be very limited for the local classification problems. We prove that our algorithm converges to a global optimal classifier for an arbitrarily distributed database across an SCN.

D. Algorithm

2) For any site $l, l = 1, \dots, L$, once it receives SVs from its ancestor sites, we repeat the following steps.

- Merge training vectors $X_l^{\text{add}} = \{x_i : x_i \in V^{t,m}, \forall m \in \text{UPS}_l, x_i \notin S^{t-1,l}\}$, where UPS_l is the set of all immediate ancestor sites of site l , to the current training sets $S^{t,l}$ of the current site l .
- Solve (3). Record the optimal objective value $h^{t,l}$ and the solution $\alpha^{t,l}$.
- Find the set $V^{t,l} = \{x_i : \alpha_i^{t,l} > 0\}$ and pass them to all immediate descendant sites.

3) If $h^{t,l} = h^{t-1,l}$ for all l , stop; otherwise, go to step 2).

Every site starts to work once it receives SVs from its ancestor sites. In step 2b), we start solving the newly formed problem from the best solution currently available and update this solution locally. We use SVM^{light} [15] as our local solver. The numerical results show that the computing speed can be dramatically increased by applying the available best solution α^{t-1} as the initial starting point in step 1b). We will discuss this issue later in this paper.

To demonstrate the convergence, we randomly generate 200

Figura 17: Terceira sentença da Metodologia.

problem is α^* . Since “ α^* ” is unique by uniqueness of optimal solution for strictly convex problem, we have $\alpha^* = \alpha_1^* = 0$, $\alpha^* = \alpha_1^* = \alpha_2^*$, and $\alpha^* = \alpha_2^* = 0$. Since α^* is nonzero, we have

$$\{\alpha_1^*\} = \emptyset.$$

This already shows that the SVs sets $V^{t-1,l}$, $V^{t,t-1}$, and $V^{t,l}$ are identical. Then, we prove that $[0; \dots; 0; \alpha]$ is the optimal solution for the SVM problem with union of the training vector from site l and m . The Karush–Kuhn–Tucker (KKT) conditions for the problem (5) can be stated as follows:

$$\begin{aligned} Q_i \alpha &\geq 1, & \text{if } \alpha_i = 0 \\ Q_i \alpha &\leq 1, & \text{if } \alpha_i = \gamma \\ Q_i \alpha &= 1, & \text{if } 0 < \alpha_i < \gamma \end{aligned}$$

where Q_i is the i th row of the Q corresponding to α_i . Since $[0; 0; \dots; 0; \alpha_1]$, α_2 and α satisfy the KKT condition of the problem $P^{t,m}$, $P^{t-1,l}$, and $P^{t,l}$, where $P^{t,l}$ denotes the SVM problem at iteration t for site l . By simple algebra, $[0; \dots; 0; \alpha_b; 0; \dots; 0]$ satisfies the KKT condition of the SVM problem with union training vectors. Therefore, the $[0; \dots; 0; \alpha_b; 0; \dots; 0]$ is the optimal solution for the SVM problem with union of the training vector from site l and m . By

margin, such that the final converged solution is guaranteed to be the global optimum. In a centralized method, such parameters are usually tuned empirically through cross validations. In distributed data mining, such tuning may not be feasible given partially available training data.

In this paper, we propose an intuitive method to generate a parameter γ that has good performance to the global optimization problem by using locally available partial training data and statistics of the global training data. The training parameter γ may be estimated as follows:

$$\gamma = \gamma_l \frac{N_l \sigma_l^2}{N_l \sigma^2} \quad (6)$$

where γ_l is a good parameter selected based on data in site l , and σ_l and σ are defined as follows:

$$\sigma_l^2 = \frac{1}{N_l} \sum_{i \in l} K(x_i, x_i) - 2K(x_i, v_l) + K(v_l, v_l)$$

and

$$\sigma^2 = \frac{1}{N} \sum_{\forall i} K(x_i, x_i) - 2K(x_i, v) + K(v, v)$$

where v is the center of all training vectors and v_l is the center

Figura 18: Quarta sentença da Metodologia.

To justify the estimated parameter γ , we enumerate several possible value of global parameter γ , train a model based on all training vectors, and obtain the test error summarized in Table II. One may observe the best $\gamma^* = 2^{-3} = 0.125$. Using this error, in this specific application, $\hat{\gamma}$ is a good estimation of the true γ^* .

With the estimated γ^* , we plot the test accuracy and objective values of site 1 in each iteration of our DPSVM algorithm in Fig. 2. The network we choose has seven sites and a random sparse topology. One may observe that the objective value (of the primal problem) is monotonously increasing while the test accuracy is improving, which may not be monotonic. The power of DPSVM is to get global optimal classifier in a local site without transmitting all data into one site.

B. Network Configuration

Network configuration has a dramatic impact on performance of our DPSVM algorithm. Size and topology of a network determine the number of sites that may work concurrently and the frequency at which a site receives new training data from other

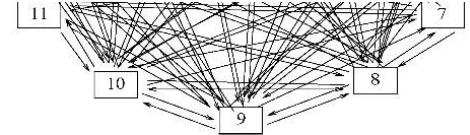


Fig. 4. Diagram: a fully connected network.

meaningful to have more than N/N_{SV} sites while N_{SV} denotes the number of SVs for the global problems. Actually, Lu and Roychowdhury showed that the number of training vectors must be bounded below in order, where a randomized distributed algorithm has a provably fast convergence rate [14]. Our experiments show that the training speed is, in general, faster in a larger size of a network given the network size is limited.

Network topology configuration is a key issue in implementing the distributed algorithms. Let us first consider two extreme cases: a ring network (see Fig. 3) and a fully connected

Figura 19: Quinta sentença da Metodologia.

A former strategy is called asynchronous implementation. In this strategy, each site processes its data upon receiving new data from any of its immediate ancestor sites. For example, in network defined in Fig. 6, in second and later iterations, site 1 will start processing once having received data from either site 7, 9, or 15, given it is currently idle. Asynchronous implementation guarantees that all servers are utilized as much as possible.

The advantage and disadvantage of both implementation strategies are empirically compared in Section V.

D. Online Implementation

Since the DPSVM constructs a local problem by adding critical data, local SVM training problems in subsequent iterations are essentially problem adjustments, which append variables

among all sites among all iterations.

Elapsed central processing unit (CPU) seconds of running DPSVM.

Standard deviation of initial training data distribution.

Throughout this section, we use Pentium 4 3.2 GHz with 1 GB RAM to solve local SVM problem.

A. Effect of Initial Data Distribution

In real-world applications, data may be distributed arbitrarily around the world. In certain applications, the distributed data may not be balanced or unbiased. For example, search engine

Figura 20: Primeira parte da sexta sentença da Metodologia.

dashed line of the fully connected network in Fig. 7. Specifically, an RSCN in Fig. 6 has its connectivity matrix

$$C_{ra} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (16)$$

and constraints based on the problem of the last iteration. Online algorithm is a popular solution for incremental learning problems. In this paper, we compare a simple online algorithm with an offline algorithm.

In the simple online implementation, at each iteration, every site always uses current dual solution (α values) as its initial α values. The newly added training data will also bring a vector of nonzero α 's. We use a slightly modified SVM^{light} [15] as our local SVM solver, which may take an arbitrary α vector as input. To avoid possible infinite loops due to numerical inaccuracies in the core QP-solver, a somewhat random working set is selected in the first iteration and we repeat it for every 100 iterations.

Our experiments in Section V show that this simple online implementation greatly improves the distributed training speed

Figura 21: Segunda parte da sexta sentença da Metodologia.

A.3 Sentenças dos Resultados

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

This RSCN has 77 directed edges and 15 nodes. Its network density

$$d_{ru} = \frac{77}{210} = 0.37.$$

V. PERFORMANCE STUDIES

We test our algorithm over a real-word data set: MNIST database of handwritten digits [17]. The digits have been size-normalized and centered in a fixed-size image. This database is a standard database online for researchers to compare their training method because MNIST data is clean and does not need any preprocessing. All digit images are centered in a 28×28 image. We vectorize each image to a 784 dimension

Figura 22: Primeira sentença dos Resultados.

TABLE III
PERFORMANCE OVER THE INITIAL DATA DISTRIBUTION

s	T	δ	Δ	$\max_{l,t} \{N_{lt}\}$	e
Random Dense Network					
0	8	126	15191	1836	9.00
305.7	8	115	13897	2086	8.93
610.7	8	130	15363	2247	12.32
766.9*	6	102	12247	2097	16.92
Binary Cascade Network					
0	16	37	8842	1008	15.51
305.7	17	35	8977	1569	15.63
610.7	18	30	8120	1653	17.20
766.9*	18	26	6906	1731	16.43

* Training data are evenly distributed in eight sites. The other seven sites are empty before receiving any SV from other sites.

TABLE IV
TESTED NETWORK CONFIGURATIONS

L	Topology	E	d
3	ring	3	0.50
3	binary-cascade	4	0.67
3	fully connected	6	1.00
7	ring	7	0.17
7	binary-cascade	10	0.24
7	random sparse	14	0.33
7	random dense	33	0.79
7	fully connected	42	1.00
15	ring	15	0.07
15	binary-cascade	22	0.10
15	random sparse	54	0.26
15	random dense	159	0.76
15	fully connected	210	1.00

B. Scalability on Network Topologies

We test our algorithm over an array of network configurations. Five network topologies are tested including ring networks, binary cascade networks, fully connected networks, random sparse networks, and random dense networks. For the two types of random networks, we impose a constraint over the network density such that

$$d \leq 0.33$$

for random sparse networks and

$$d \geq 0.75$$

for random dense networks. Networks of sites 3, 7, and 15 are tested for ring, binary cascade, and fully connected networks. Networks of sites 7 and 15 were tested for the two types of random generated networks. The number of sites, number of edges, and the corresponding network densities of all tested networks are summarized in Table IV.

In this section, we apply online and synchronized implementation for each tested networks. The offline and asynchronous implementation will be discussed later. We record the total training time in terms of elapsed CPU seconds, the number of iterations, and the number of transmitted training vectors per site. The results are plotted in Fig. 8. The results show that our algorithm scales very well for all network topologies except for the ring network, given the size of the network is limited. In ring networks, however, one site's information reaches all other sites only after at least $L - 1$ iterations. The communication is

Figura 23: Segunda e terceira sentença dos Resultados.

L. Performance Comparison between SVM and DPSVM

When merging all training data is infeasible due to physical or political reasons, one may apply the DPSVM that is able to reach a global optimal solution or the normal SVM that works on local data only. We demonstrate that our DPSVM has the real advantage in terms of test accuracy over normal SVMs that are trained based on local training data.

We compare DPSVM and SVM when MNIST data are randomly distributed into sites 3, 7, and 15 assuming that each site has the same number of training vectors. The training parameter is estimated following the approach introduced in Section IV-A. **We compare the test accuracy of DPSVM to the minimum, maximum, and average values of the test accuracy of normal SVM among all sites.** The results are summarized in Table V. The table shows expected phenomena: when less training is used, the achieved test accuracy is worse. This is one of the motivations of the DPSVM: to reach global optimum and better test performance without aggregating all the data.

- [7] L. M. Adams and H. F. Jordan, "Is SOR color-blind?", *SIAM J. Sci. Statist. Comput.*, vol. 7, no. 2, pp. 490–506, 1986.
- [8] U. Block, A. Frommer, and G. Mayer, "Block coloring schemes for the SOR method on local memory parallel computers," *Parallel Comput.*, vol. 14, no. 11, pp. 61–75, 1990.
- [9] G. Zanghirati and L. Zanni, "A parallel solver for large quadratic programs in training support vector machines," *Parallel Comput.*, vol. 29, pp. 535–551, Nov. 2003.
- [10] N. Syed, H. Liu, and K. Sung, "Incremental learning with support vector machines," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, San Diego, CA, 1999.
- [11] C. Caragea, D. Caragea, and V. Honavar, "Learning support vector machine classifiers from distributed data sources," in *Proc. 20th Nat. Conf. Artif. Intell. Student Abstract Poster Program*, Pittsburgh, PA, 2005, pp. 1602–1603.
- [12] A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J. Navarro-Abellan, "Distributed support vector machines," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 1091–1097, Jul. 2006.
- [13] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik, "Parallel support vector machines: The cascade SVM," in *Proc. 18th Annu. Conf. Neural Inf. Process. Systems*, Vancouver, BC, Canada, 2004, pp. 521–528.
- [14] Y. Lu and V. Roychowdhury, "Parallel randomized support vector machine," in *Proc. 10th Pacific-Asia Conf. Knowl. Disc. Data Minin*

Figura 24: Quarta sentença das Resultados.

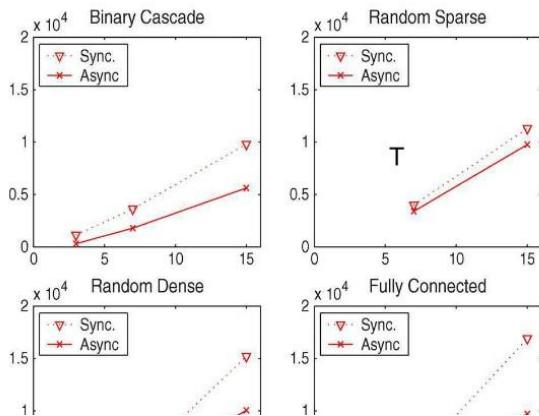
A.4 Sentenças das Conclusões

VI. CONCLUSION AND FUTURE RESEARCH

The proposed DPSVM training algorithm exploits a simple idea of partitioning training data and exchanging SVs over an SCN. **The algorithm has been proved to converge to the global optimal classifier in finite steps. Experiments over a real-world database show that this algorithm is scalable and robust.** The properties of this algorithm can be summarized as follows.

- 2006, pp. 205–214.
- [15] T. Joachims, "Making large-scale SVM learning practical," in *Advances Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1998, pp. 41–56.
- [16] T. Joachims, "SVM-light support vector machine," 1998 [Online]. Available: <http://svmlight.joachims.org/>
- [17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

Figura 25: Primeira sentença das Conclusões.



- **The DPSVM algorithm is able to work on multiple arbitrarily partitioned working sets and achieve close to linear scalability if the size of the network is not too large.**
- Data accumulation during SVs exchange is limited if the overall SVs are limited. Communication cost is proportional to the number of SVs. Asynchronous implementation over a sparse network achieves the minimum data accumulation.
- The DPSVM algorithm is robust in terms of computing time and communication overhead to the initial distribution of the database. It is suitable for classification over arbitrary distributed databases as long as a network is denser than the ring network.
- In general, denser networks achieve less computing time while sparser networks achieve less data accumulation.

Figura 26: Segunda sentença das Conclusões.

ANEXO A – ARTIGO USADO NOS EXEMPLOS TRABALHADOS

Distributed Parallel Support Vector Machines in Strongly Connected Networks

Yumao Lu, Vwani Roychowdhury, and Lieven Vandenberghe, *Member, IEEE*

Abstract—In this paper, we propose a distributed parallel support vector machine (DPSVM) training mechanism in a configurable network environment for distributed data mining. The basic idea is to exchange support vectors among a strongly connected network (SCN) so that multiple servers may work concurrently on distributed data set with limited communication cost and fast training speed. The percentage of servers that can work in parallel and the communication overhead may be adjusted through network configuration. The proposed algorithm further speeds up through online implementation and synchronization. We prove that the global optimal classifier can be achieved iteratively over an SCN. Experiments on a real-world data set show that the computing time scales well with the size of the training data for most networks. Numerical results show that a randomly generated SCN may achieve better performance than the state of the art method, cascade SVM, in terms of total training time.

Index Terms—Convergence, distributed data mining, parallel computing, strongly connected networks (SCNs), support vector machine (SVM).

I. INTRODUCTION

DISTRIBUTED classification is necessary if a centralized system is infeasible due to geographical, physical, and computational constraints. The objectives of distributed data mining usually include robustness to changes in the network topology [1], efficient representation for high-dimensional and massive data sets, least synchronization and communication, least duplication, better load balancing, and good decision precision and certainty.

The classification problem is an important problem in data mining. The support vector machine (SVM), which implements the principle of structural error minimization [2], [3] is one of the most popular classification algorithms in many fields [4]–[6].

SVMs are ideally suited for a framework where different sites can potentially exchange only a small number of training vectors. For a data set at a particular site, the support vectors (SVs) are the natural representatives of discriminant information of the database *and* the optimal solution to a local classifier. Distributed support vector machines (DSVMs) and parallel support

Manuscript received January 21, 2007; revised September 5, 2007 and October 30, 2007; accepted November 1, 2007. First published March 12, 2008; last published July 7, 2008 (projected).

Y. Lu is with the Yahoo! Inc., Sunnyvale, CA 94089 USA (e-mail: yu-mao@ yahoo-inc.com).

V. Roychowdhury and L. Vandenberghe are with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095 USA.

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TNN.2007.2000061

vector machines (PSVMs), which emphasize global optimality, draw more and more attentions in recent years.

Current PSVMs include matrix multicoloring successive overrelaxation (SOR) method [7], [8] and variable projection method (VPM) in sequential minimal optimization (SMO) [9]. These methods are excellent in terms of speed. However, they all need centralized access to the training data, and therefore, cannot be used in distributed classification applications. In summary, the current popular parallel methods are not distributed.

On the other hand, current DSVMs are not taking enough advantage of parallel computing. The main obstacle is that the more servers work concurrently, the more data are transmitted, causing excessive data accumulation that slows down the training process. Syed *et al.* proposed the first DSVM algorithm that finds SVs locally and processes them altogether in a central processing center [10]. Their solution, however, is not global optimal. Caragea *et al.* improved this algorithm by allowing the data processing center to send SVs back to the distributed data source and iteratively achieve the global optimum [11]. This model is slow due to extensive data accumulation in each site [12]. Navia-Vazquez *et al.* proposed distributed semiparametric SVM to reduce the communication cost by transmitting a function of subset of SVs R . Their algorithm, however, is suboptimal depending on the definition of R . In order to accelerate DSVM, Graf *et al.* had an algorithm that implemented distributed processors into cascade top-down network topology, namely, cascade SVM [13]. The bottom node of the network is the central processing center. To the best of our knowledge, the cascade SVM is the fastest DSVM that is globally optimal. This is the first time that network topology was taken into consideration to accelerate the distributed training process. However, there is no comparison to other network topologies. It is not clear either that their distributed SVM may converge in a more general network.

In this paper, we propose a distributed parallel support vector machine (DPSVM) for distributed data classification in a general network configuration, namely, strongly connected network (SCN). The objective of the distributed classification problem is to classify distributed data, i.e., determine a single global classifier, by judiciously sampling and distributing subsets of data among the various sites. Ideally, a single site or server should not end up storing the complete data set; instead, the different sites should exchange a minimum number of data samples, and together converge to a single global solution in an iterative fashion.

The basic idea of DPSVM is to exchange SVs over an SCN and update (instead of recompute) the local solutions iteratively, based on the SVs that a particular site receives from its neighbors. We prove that our algorithm converges to a globally

optimal classifier (at every site) for arbitrarily distributed data over an SCN. Recall that an SCN is a directed graph where there is a directed path between any pair of nodes; the strongly connected property makes sure that the critical constraints are shared across all the nodes/sites in the network, and each site converges to the globally optimal solution. Although this proof does not guarantee the convergence speed, Lu and Roychowdhury proved that a similar parallel SVM has the provably fastest average convergence rate among all decomposition algorithms if each site randomly selects training vectors that follows a carefully designed distribution [14].

Practically, the DPSVM achieves the fast convergence time. The proposed algorithm is analyzed under a variety of network topologies as well as other configurations such as network size, online and offline implementation, etc., which have dramatic impact on the performance in terms of convergence speed and data accumulation. We show that the efficiency of DPSVM and the overall communication overhead can be improved by controlling the network sparsity. Specifically, a randomly generated SCN with a sparsity constraint outperforms the cascade SVM, which is reported to be the fastest distributed SVM algorithm to the best of our knowledge.

This paper is organized as follows. We present our distributed classification algorithm in the next section followed by the proof of the global convergence in Section III. We introduce the detailed algorithm implementation in Section IV. The performance study is given in Section V. We conclude in Section VI.

II. DISTRIBUTED SUPPORT VECTOR MACHINE

A. Problem

We consider the following problem: the training data are arbitrarily distributed in L sites. Each site is a node within an SCN, defined as follows.

Definition 1: An SCN is a directed network in which it is possible to reach any node starting from any other node by traversing edges in the direction(s) in which they point.

Differing from a weakly connected network, which becomes a connected (undirected) graph if all of its directed edges are replaced with undirected edges, an SCN makes sure each node in the work can be accessed from any other nodes. This is an important property required by our convergence proof (see the proof of Theorem 1).

There are N_l training vectors in site l and N training vectors in all sites, where $N_l, \forall l$ can be an arbitrary integer, such that $\sum_{l=1}^L N_l = N$. Each training vector is denoted by z_i $i = 1, \dots, N$ where $z_i \in \mathbf{R}^n$, and $y_i \in \{+1, -1\}$ is its label.

The global SVM problem, i.e., the traditional centralized problem is briefly summarized in the next section.

B. Support Vector Machine

In an SVM training problem, we are seeking a hyperplane to separate a set of positively and negatively labeled training data. The hyperplane is defined by $w^T z + b = 0$, where the parameter $w \in \mathbf{R}^n$ is a vector orthogonal to the hyperplane and $b \in \mathbf{R}$ is the bias. The decision function is the hyperplane classifier

$$H(x) = \text{sign}(w^T z + b).$$

The hyperplane is designed such that $y_i(w^T z_i + b) \geq 1$. The margin is defined by the distance of the two parallel hyperplanes $w^T z + b = 1$ and $w^T z + b = -1$, i.e., $2/\|w\|_2$. The margin is related to the generalization of the classifier [2]. One may easily transform the decision function to

$$H(x) = \text{sign}(g^T x) \quad (1)$$

where vector $g = [w; b] \in \mathbf{R}^{n+1}$ and $x = [z; 1] \in \mathbf{R}^{n+1}$. The benefit of this transformation is to simplify the formulation so that the bias b does not need to be calculated separately. For general linear nonseparable problems, a set of slack variables ξ_i 's is introduced. The SVM training problem for the nonseparable problem is defined as follows:

$$\begin{aligned} & \text{minimize} && (1/2)g^T x + \gamma \mathbf{1}^T \xi \\ & \text{subject to} && y_i g^T \phi(g_i) \geq 1 - \xi_i, \quad i = 1, \dots, N \\ & && \xi \geq 0 \end{aligned} \quad (2)$$

where $\mathbf{1}$ is a vector of ones, ϕ is a lifting function, and the scalar γ is called regularization parameter that is usually empirically selected to reduce the testing error rate.

The corresponding dual of the problem (2) is shown as follows:

$$\begin{aligned} & \text{maximize} && -(1/2)\alpha^T Q \alpha + \mathbf{1}^T \alpha \\ & \text{subject to} && 0 \leq \alpha \leq \gamma \mathbf{1} \end{aligned} \quad (3)$$

where the Gram matrix Q has the component $Q_{ij} = y_i y_j \phi(x_i)^T \phi(x_j)$.

It is common to replace $\phi(x)^T \phi(\tilde{x})$ by a kernel function $F(x, \tilde{x})$ such that $Q_{ij} = K_{ij} y_i y_j$ and $K_{ij} = F(x_i, x_j)$, where $K \in \mathbf{R}^{N \times N}$ is the so-called kernel matrix. The nonlinear kernel may lift the dimension of training vectors to a higher dimension so that they can be separated linearly. If the kernel matrix K is positive definite, problem (3) is guaranteed to be strictly convex. There are several popular choices of nonlinear kernel functions such as inhomogeneous polynomial kernels

$$F(x, \tilde{x}) = x^T \tilde{x} + 1$$

and Gaussian kernels

$$F(x, \tilde{x}) = \exp\left(-\frac{\|x - \tilde{x}\|^2}{2\sigma^2}\right).$$

Correspondingly, we have the decision function of the form

$$H(x) = \text{sign}\left(\sum_{i=1}^N y_i \alpha_i F(x_i, x)\right).$$

For the general SVM optimization problem, the complementary slackness condition has the form

$$\alpha_i [y_i(g^T x_i) - 1 + \xi_i] = 0 \quad \text{for all } i = 1, \dots, N \quad (4)$$

in the optimum. Therefore, SVs are the vectors that lie in between the margin boundary (including those on the boundary) and that are misclassified.

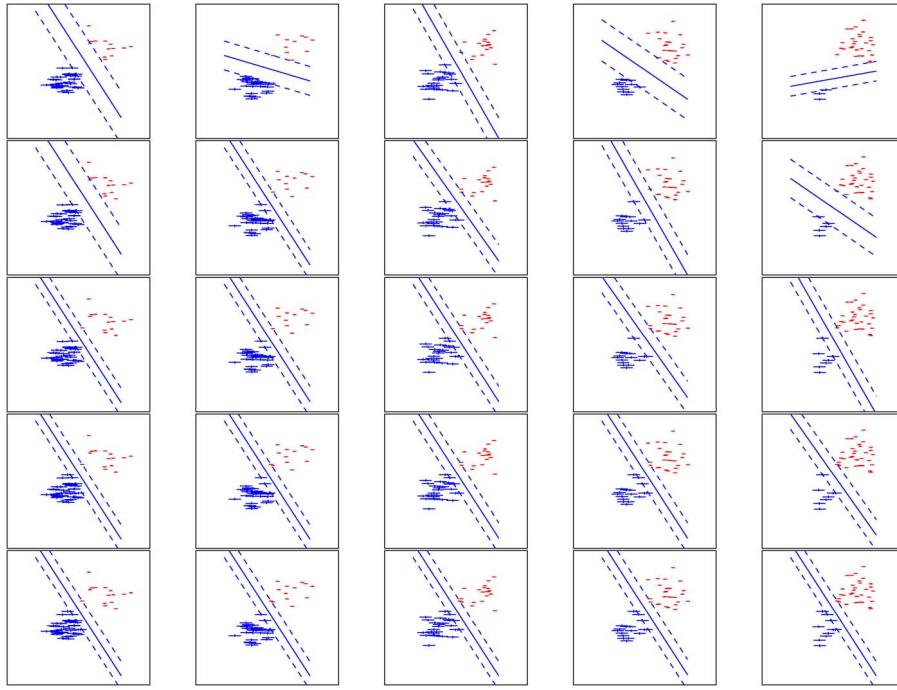


Fig. 1. Demonstration of data distribution in DPSVM iterations. Distribution of training data in five sites before iterations begins (the first row) and in iteration 1–4 (the second to the fifth rows). This figure shows how all local classifiers converge to the global optimum classifier.

C. Efficient Information Carrier: SVs

SVs, namely, the training data that have nonzero α values, lie physically on the margin, or are misclassified. Those vectors carry all the classification information of the local data set. Exchanging SVs, instead of moving all the data, is a natural way to fuse information among distributed sites.

SVs are a natural representation of the discriminant information of the underlying classification problem. The basic idea of the DSVMs is to exchange SVs over an SCN and update the local solutions for each site iteratively. We exploit the idea of exchanging SVs in a deterministic way instead of a randomized way [14] in our distributed learning algorithms. Our algorithm is based on the observation that the number of SVs may be very limited for the local classification problems. We prove that our algorithm converges to a global optimal classifier for an arbitrarily distributed database across an SCN.

D. Algorithm

The DPSVM algorithm works as follows. Each site within an SCN classifies subsets of training data locally via SVM, passes the calculated SVs to its descendant sites, receives SVs from its ancestor sites, recalculates the SVs, passes them to its descendant sites, and so on. The algorithm of DPSVM consists of the following steps.

Initialization: We initialize the algorithm with iteration $t = 0$ and local SV set in site l at iteration t , $V^{t,l} = \emptyset, \forall l, t = 0$. The training set at iteration t in site l is denoted by $S^{t,l}$. The $S^{0,l}$ is initialized arbitrarily such that $\bigcup_{l=1}^L S^{0,l} = S$, where S denotes the total sample space.

Iteration: Each iteration consists of the following steps.

- 1) $t := t + 1$.

2) For any site $l, l = 1, \dots, L$, once it receives SVs from its ancestor sites, we repeat the following steps.

- a) Merge training vectors $X_l^{\text{add}} = \{x_i : x_i \in V^{t,m}, \forall m \in \text{UPS}_l, x_i \notin S^{t-1,l}\}$, where UPS_l is the set of all immediate ancestor sites of site l , to the current training sets $S^{t,l}$ of the current site l .
- b) Solve (3). Record the optimal objective value $h^{t,l}$ and the solution $\alpha^{t,l}$.
- c) Find the set $V^{t,l} = \{x_i : \alpha_i^{t,l} > 0\}$ and pass them to all immediate descendant sites.

3) If $h^{t,l} = h^{t-1,l}$ for all l , stop; otherwise, go to step 2).

Every site starts to work once it receives SVs from its ancestor sites. In step 2b), we start solving the newly formed problem from the best solution currently available and update this solution locally. We use SVM^{light} [15] as our local solver. The numerical results show that the computing speed can be dramatically increased by applying the available best solution α^{t-1} as the initial starting point in step 1b). We will discuss this issue later in this paper.

To demonstrate the convergence, we randomly generate 200 independent 2-D data sampled from two independent Gaussian distributions. We randomly distributed all the data to five sites. We assume the five sites form a ring network. The SVM problem is solved locally, and pass the SVs to their descendant sites. The DPSVM converges in four iterations and the local results are shown in Fig. 1. One may observe the decreasing of the local margins and they converge to the global optimum classifier.

We prove the global convergence in Section III.

III. PROOF OF CONVERGENCE

We prove that our algorithm DPSVM converges to the global optimal classifier in finite steps in this section.

Lemma 1—Global Lower Bound: Let h^* be the global optimum value, which is the optimum value of the following problem:

$$\begin{aligned} & \text{maximize} \quad -(1/2)\alpha^T Q \alpha + \mathbf{1}^T \alpha \\ & \text{subject to} \quad 0 \leq \alpha \leq \gamma \mathbf{1} \end{aligned} \quad (5)$$

where Q is the Gram matrix for all training samples. Then

$$h^{t,l} \leq h^* \quad \forall t, l.$$

Proof: Define $\tilde{\alpha}^{t,l}$ as follows:

$$\tilde{\alpha}^{t,l} = \begin{cases} \alpha^{t,l}, & \text{if } i \in S^{t,l} \\ 0, & \text{otherwise.} \end{cases}$$

Then, the proof follows immediately by the fact that the solutions $\tilde{\alpha}^{t,l} \forall t, l$ are always a feasible solution to the global problem (5) with objective value $h^{t,l}$.

Lemma 1 shows that each solution of the subproblem serves as a lower bound for the global SVM problem.

Lemma 2—Nondecreasing: The objective value for any site, say site l , at iteration t , is always greater than or equal to the maximum of the objective of the same site in the last iteration and the maximal objective values of its immediate ancestor sites, site m , $\forall m \in \text{UPS}_l$, from which site l receives SVs in the current iteration. That is

$$h^{t,l} \geq \max \{h^{t-1,l}, \max_{m \in \text{UPS}_l} \{h^{t,m}\}\}.$$

Proof: Assume $t > 1$ without losing generality. Let us first assume site l receives SVs only from one of its ancestor site, say site m . The solutions corresponding to the objectives $h^{t,l}$, $h^{t-1,l}$, and $h^{t,m}$ are $\alpha^{t,l}$, $\alpha^{t-1,l}$, and $\alpha^{t,m}$. Denote the nonzero part of $\alpha^{t,m}$ by α_1 . The corresponding Gram matrix is denoted by Q_1 . Therefore

$$h^{t,m} = -\alpha_1^T Q_1 \alpha_1 + \mathbf{1}^T \alpha_1.$$

Define $\alpha_2 = \alpha^{t-1,l}$ and the corresponding Gram matrix to be Q_2 . We have

$$h^{t-1,l} = -\alpha_2^T Q_2 \alpha_2 + \mathbf{1}^T \alpha_2.$$

Since some SVs corresponding to α_1 may be the same as some of those corresponding to α_2 , we reorder and rewrite α_1 and α_2 as follows:

$$\begin{aligned} \alpha_1 &= [\alpha_1^a; \alpha_1^b] \\ \alpha_2 &= [\alpha_2^b; \alpha_2^c] \end{aligned}$$

so the newly formed problem for site l at iteration t has the Gram matrix

$$Q = \begin{bmatrix} Q_{aa} & Q_{ab} & Q_{ac} \\ Q_{ba} & Q_{bb} & Q_{bc} \\ Q_{ca} & Q_{cb} & Q_{cc} \end{bmatrix}$$

and we have

$$Q_1 = \begin{bmatrix} Q_{aa} & Q_{ab} \\ Q_{ba} & Q_{bb} \end{bmatrix}$$

and

$$Q_2 = \begin{bmatrix} Q_{bb} & Q_{bc} \\ Q_{cb} & Q_{cc} \end{bmatrix}$$

where Q_{aa} , Q_{bb} , and Q_{cc} are the Gram matrices corresponding to α_1^a , α_1^b (or α_2^b), and α_2^c . The newly formed problem can be written as

$$\begin{aligned} & \text{maximize} \quad -(1/2)\alpha^T Q \alpha + \mathbf{1}^T \alpha \\ & \text{subject to} \quad 0 \leq \alpha \leq \gamma \mathbf{1}. \end{aligned}$$

Note that $[\alpha_1; 0; 0; \dots; 0]$ and $[0; 0; \dots; \alpha_2]$ are both feasible solution of the previous maximization problem with the objective value $h^{t-1,l}$ and $h^{t,m}$, respectively. Therefore, the optimum value $h^{t,l}$ satisfies

$$h^{t,l} \geq \max \{h^{t-1,l}, h^{t,m}\}.$$

Now we remove the assumption that site l only receives SVs from site m . Since m is selected arbitrarily from l 's ancestor sites and adding more samples cannot decrease the optimal objective value (following the same argument in Lemma 1), we have that

$$h^{t,l} \geq \max \left\{ h^{t-1,l}, \max_{m \in \text{UPS}_l} \{h^{t,m}\} \right\}.$$

Lemma 2 shows that the objective value of the subproblem in our DPSVM algorithm monotonically increases in each iteration.

Corollary 1: The stopping criterion, $h^{t,l} = h^{t-1,l}$ for all l and $t > T$, can be satisfied in finite number of iterations.

Proof: By Lemma 2, adding SVs from the adjacent site with higher objective value results in increase of the objective value of the newly formed problem. Lemma 1 shows that every optimal objective value for each site at each iteration is bounded by h^* . Since the data points are limited and no duplicated data are allowed, $h^{t,l} = h^{t-1,l}$ for all l and $t > T$ can be satisfied in finite number of steps.

Before proving our theorem, we have to prove a key proposition first.

Proposition 1: If any Gram matrix formed by any training sample data sets are positive definite, and $h^{t,l} = h^{t-1,l} = h^{t,l-1} = h^{t,l-2} = \dots = h^{t,l-m_l}$ where $l-1, \dots, l-m_l$ are the ancestor sites from which site l receives SVs at the current iteration, then the SVs sets $V^{t-1,l}, V^{t,m}, \forall m \in \text{UPS}_l$ and $V^{t,l}$ are equal and the solution corresponding to the SVs from either of the previous site is the optimal solution for the SVM training problem for the union of the training vectors of site l and its immediate ancestor m , $\forall m \in \text{UPS}_l$ at iteration t .

Proof: Note that at time $t+1$, for site l , it is the same to update its data set by simultaneously adding SVs from site m , $\forall m \in \text{UPS}_l$, as to update the data set by sequentially adding SVs from those ancestor sites. That is to say, we only need to show the result in the case that site l only receives SVs from one such site, say site $l-1$. Recall the notation used in the proof of Lemma: $h^{t,l-1} = -\alpha_1^T Q_1 \alpha_1 + \mathbf{1}^T \alpha_1$, $h^{t-1,l} = -\alpha_2^T Q_2 \alpha_2 + \mathbf{1}^T \alpha_2$, $h^{t,l} = -\alpha^T Q \alpha + \mathbf{1}^T \alpha$, $\alpha_1 = [\alpha_1^a; \alpha_1^b]$, $\alpha_2 = [\alpha_2^b; \alpha_2^c]$, and $\alpha = [\alpha^a; \alpha^b; \alpha^c]$, where α_1 is the nonzero part of $\alpha^{t,l-1}$,

$\alpha_2 = \alpha^{t-1,l}$, and $\alpha = \alpha^{t,l}$. The newly formed problem for site l at iteration t has the Gram matrix

$$Q = \begin{bmatrix} Q_{aa} & Q_{ab} & Q_{ac} \\ Q_{ba} & Q_{bb} & Q_{bc} \\ Q_{ca} & Q_{cb} & Q_{cc} \end{bmatrix}$$

and we have

$$Q_1 = \begin{bmatrix} Q_{aa} & Q_{ab} \\ Q_{ba} & Q_{bb} \end{bmatrix}$$

and

$$Q_2 = \begin{bmatrix} Q_{bb} & Q_{bc} \\ Q_{cb} & Q_{cc} \end{bmatrix}.$$

Note that $[\alpha_1; 0; \dots; 0]$ and $[0; \dots; 0; \alpha_2]$ are both feasible for the global optimal problem (5) and the optimal solution of this problem is α . Since $h^{t-1,l} = h^{t,l-1} = h^{t,l}$ by uniqueness of optimal solution for strictly convex problem, we have $\alpha^a = \alpha_1^a = 0$, $\alpha^b = \alpha_1^b = \alpha_2^b$, and $\alpha^c = \alpha_2^c = 0$. Since α^a is nonzero, we have

$$\{\alpha_1^a\} = \emptyset.$$

This already shows that the SVs sets $V^{t-1,l}$, $V^{t,l-1}$, and $V^{t,l}$ are identical. Then, we prove that $[0; \dots; 0; \alpha]$ is the optimal solution for the SVM problem with union of the training vector from site l and m . The Karush–Kuhn–Tucker (KKT) conditions for the problem (5) can be stated as follows:

$$\begin{aligned} Q_i \alpha &\geq 1, & \text{if } \alpha_i = 0 \\ Q_i \alpha &\leq 1, & \text{if } \alpha_i = \gamma \\ Q_i \alpha &= 1, & \text{if } 0 < \alpha_i < \gamma \end{aligned}$$

where Q_i is the i th row of the Q corresponding to α_i . Since $[0; 0; \dots; 0; \alpha_1]$, α_2 and α satisfy the KKT condition of the problem $P^{t,m}$, $P^{t-1,l}$, and $P^{t,l}$, where $P^{t,l}$ denotes the SVM problem at iteration t for site l . By simple algebra, $[0; \dots; 0; \alpha_b; 0; \dots; 0]$ satisfies the KKT condition of the SVM problem with union training vectors. Therefore, the $[0; \dots; 0; \alpha_b; 0; \dots; 0]$ is the optimal solution for the SVM problem with union of the training vector from site l and m . By sequentially adding SVs from m , $\forall m \in \text{UPS}_l$, we may repeat the previous argument so that our proposition is true.

Theorem 1: Distributed SVM over an SCN converges to the global optimal classifier in finite steps.

Proof: The Corollary 1 shows that the DPSVM converges in finite steps. Proposition 1 shows that when the DPSVM converges, the SVs of each sites that are immediately adjacent are identical. Therefore, if site i can be assessed by site j , they have identical SVs upon convergence. Since the network is strongly connected, all sites can be accessed by all other sites. Therefore, the SVs of all sites over the network are identical. Let V^* denote the converged SV set. The solution defined by

$$\alpha_i^* = \begin{cases} \alpha^{t,l}, & \text{if } x_i \in V^* \\ 0, & \text{otherwise} \end{cases}$$

is always a feasible solution for the global SVM problem (5). By Proposition 1, V^* is also the SV set for the union of the training samples from one site and its ancestor sites and the

corresponding solution is optimal for the SVM with the union of the training samples from those sites. As each site is accessible by all other sites by SCNs, V^* is the SV set for the union of the training samples from all sites. Therefore, the solution α^* is global optimum.

IV. ALGORITHM IMPLEMENTATION

The proposed algorithm has an array of implementation options, including training parameter selection, network topology configuration, sequential/parallel and online/offline implementation. Those implementation options have dramatic impact on classification accuracy and performance in terms of training time and communication overhead.

A. Global Parameter Estimation

One may note that the algorithm requires that all sites use an identical training parameter γ , which balances training error and margin, such that the final converged solution is guaranteed to be the global optimum. In a centralized method, such parameters are usually tuned empirically through cross validations. In distributed data mining, such tuning may not be feasible given partially available training data.

In this paper, we propose an intuitive method to generate a parameter γ that has good performance to the global optimization problem by using locally available partial training data and statistics of the global training data. The training parameter γ may be estimated as follows:

$$\gamma = \gamma_l \frac{N\sigma_l^2}{N_l\sigma^2} \quad (6)$$

where γ_l is a good parameter selected based on data in site l , and σ_l and σ are defined as follows:

$$\sigma_l^2 = \frac{1}{N_l} \sum_{i \in l} K(x_i, x_i) - 2K(x_i, o_l) + K(o_l, o_l)$$

and

$$\sigma^2 = \frac{1}{N} \sum_{\forall i} K(x_i, x_i) - 2K(x_i, o) + K(o, o)$$

where o is the center of all training vectors and o_l is the center of the training vectors at site l .

Equation (6) can be justified qualitatively as follows. By (2), if γ is larger, the empirical error is weighted higher in the objective. That is equivalent to say, the corresponding prior is weaker for a larger γ . Some researchers choose N/σ^2 to be a default value of γ [16]. Therefore, after determining a local parameter γ_l at site l by cross validation or other methods, one may expect (6) to be a good parameter for the overall problem. Our experiment results confirm such expectation.

In the MNIST [17] digit image classification discussed in Section V, we classify digit 0 from digit 2. Suppose the 11 881 training vectors are distributed into 15 sites. One may estimate parameter γ from the global optimization problem with a local optimal parameter γ_i at site i by (6). We use 1000 vectors as a validation set for parameter tuning and another 1000 vectors as test set. Both are sampled from MNIST test set. We first search for a local optimal γ_1 at site 1 where there are 807 training vectors. Table I gives the test error over the validation set using different setting of γ . Therefore, the optimal local parameter

TABLE I
VALIDATION ERROR AT SITE 1

γ_1	2^{-9}	2^{-7}	2^{-5}	2^{-3}	2^{-1}	2	2^3	2^5
Validation Error	0.9851	0.9876	0.9846	0.9811	0.9811	0.9811	0.9811	0.9811

TABLE II
TEST ERROR FOR THE GLOBAL TRAINING PROBLEM

γ	2^{-9}	2^{-7}	2^{-5}	2^{-3}	2^{-1}	2	2^3	2^5
Test Error	0.9911	0.9911	0.9911	0.9920	0.9906	0.9886	0.9886	0.9841

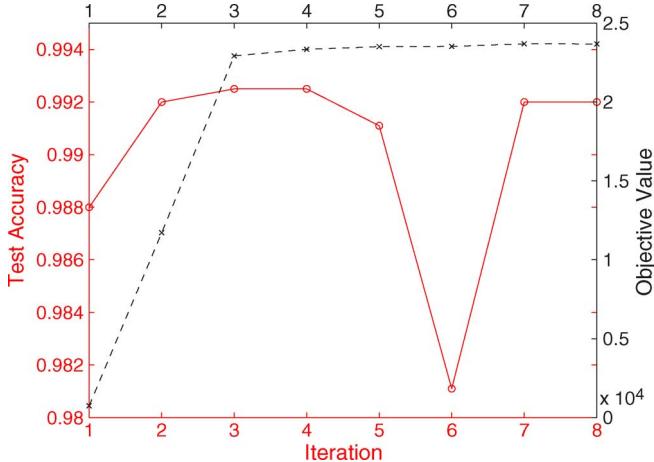


Fig. 2. Test accuracy and objective value in iterations.

$\gamma_1 = 2^{-7}$. Given the global statistics $\sigma^2 = 109.49$ and the local statistics $\sigma^2 = 110.36$, we could estimate $\hat{\gamma}$ as

$$\hat{\gamma} = \gamma_1 \frac{N\sigma_l^2}{N_l\sigma^2} = 2^{-7} \frac{11881 \times 110.36}{807 \times 109.49} = 0.117.$$

To justify the estimated parameter $\hat{\gamma}$, we enumerate several possible value of global parameter γ , train a model based on all training vectors, and obtain the test error summarized in Table II. One may observe the best $\gamma^* = 2^{-3} = 0.125$. Using this error, in this specific application, $\hat{\gamma}$ is a good estimation of the true γ^* .

With the estimated γ^* , we plot the test accuracy and objective values of site 1 in each iteration of our DPSVM algorithm in Fig. 2. The network we choose has seven sites and a random sparse topology. One may observe that the objective value (of the primal problem) is monotonously increasing while the test accuracy is improving, which may not be monotonic. The power of DPSVM is to get global optimal classifier in a local site without transmitting all data into one site.

B. Network Configuration

Network configuration has a dramatic impact on performance of our DPSVM algorithm. Size and topology of a network determine the number of sites that may work concurrently and the frequency at which a site receives new training data from other sites.

The size of a network is restricted by the number of servers available and number of data centers distributed. The larger a network is, the more servers there are that may work parallel; however, it causes more communication overhead as well.

One may note that, upon convergence, each site must at least contain the whole set of SVs, therefore, it is no longer very

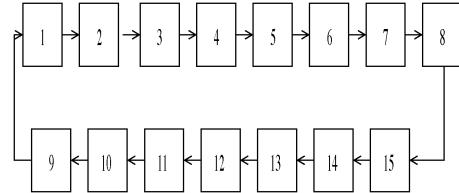


Fig. 3. Diagram: a ring network.

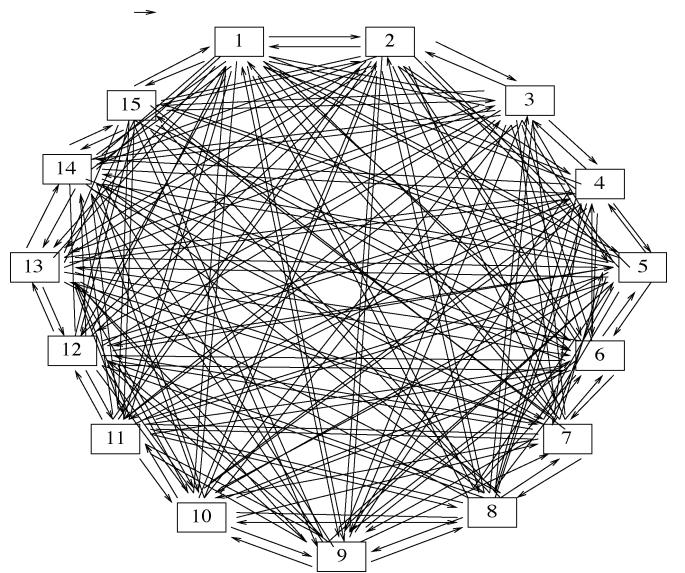


Fig. 4. Diagram: a fully connected network.

meaningful to have more than N/N_{SV} sites while N_{SV} denotes the number of SVs for the global problems. Actually, Lu and Roychowdhury showed that the number of training vectors must be bounded below in order, where a randomized distributed algorithm has a provably fast convergence rate [14]. Our experiments show that the training speed is, in general, faster in a larger size of a network given the network size is limited.

Network topology configuration is a key issue in implementing the distributed algorithms. Let us first consider two extreme cases: a ring network (see Fig. 3) and a fully connected network (see Fig. 4). A ring network is the sparsest SCN while the fully connected network is the densest one. The denser a network is, the more frequently the information exchange occurs. Graf *et al.* proposed a special network, namely, cascade SVM, and demonstrated that the training speed in such network outperform other methods [13]. The cascade SVM is a special case of SCN. Fig. 5 gives a typical example.

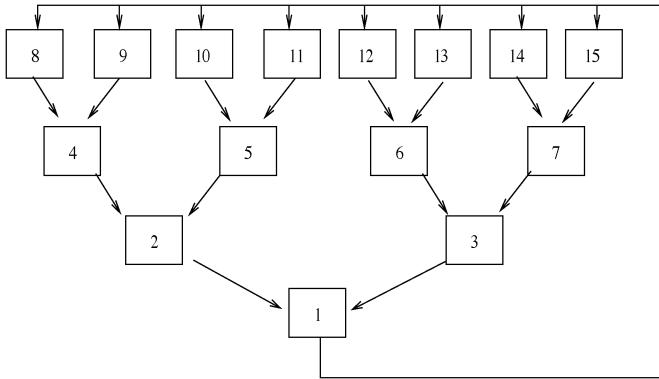


Fig. 5. Diagram: a cascade network.

We may define a connectivity matrix C such that

$$C(k, l) = \begin{cases} 1, & \text{if node } k \text{ has an edge pointing to node } l \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

Without losing generality, the connectivity matrix for a ring network has the following form:

$$C_{rr}(k, l) = \begin{cases} 1, & \text{if } l = k + 1, k < L \text{ or } k = L, l = 1 \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

A fully connected network has the corresponding connectivity matrix

$$C_f(k, l) = \begin{cases} 0, & \text{if } k = l \\ 1, & \text{otherwise.} \end{cases} \quad (9)$$

A typical cascade network in Fig. 5 has the corresponding connectivity matrix

$$C_c(k, l) = \begin{cases} 1, & \text{if } l = 2^p, 2^{p+1} \leq k \leq 2^{p+1} + 1 \\ 1, & \text{if } l = 2^p + 2^q, \\ & 2^{p+1} + 2^{q+1} \leq k \leq 2^{p+1} + 2^{q+1} + 1, \\ & \forall q < p \\ 1, & \text{if } k = 1, 2^p \leq l \leq 2^{p+1} - 1, p = \log_2(N+1) \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

We define a density of a directed network d as follows:

$$d = \frac{E}{L(L-1)} \quad (11)$$

where E is the number of directed edges and L is the number of nodes in the network. Since $c(k, k) = 0, \forall k$ always holds in our network configurations, we have

$$\frac{1}{L-1} \leq d \leq 1. \quad (12)$$

The lower and upper bounds of the density are achieved by the ring network and fully connected network, respectively. That is

$$d_{rr} = \frac{1}{L-1} \quad (13)$$

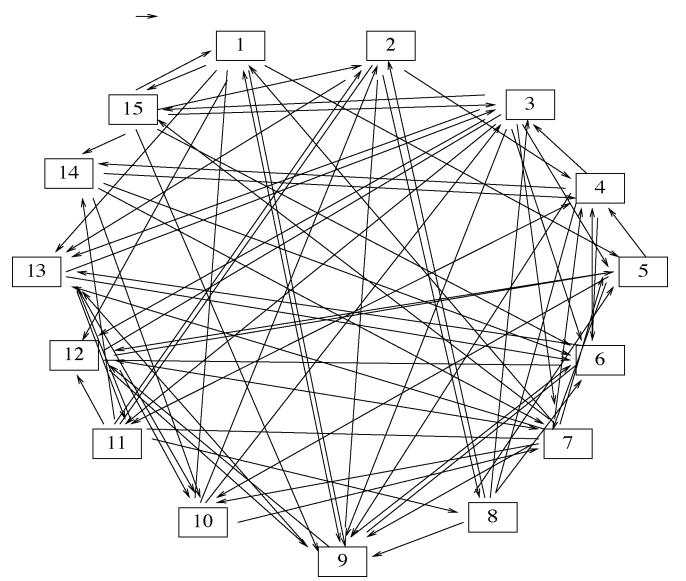


Fig. 6. Diagram: RSCN.

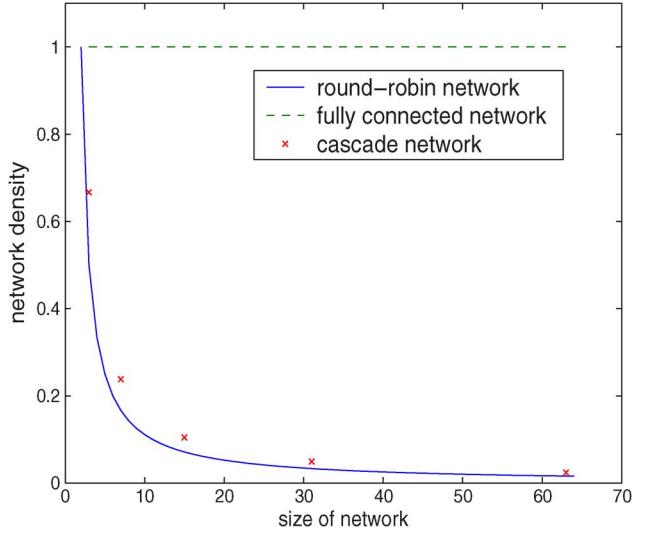


Fig. 7. Network density versus network size.

and

$$d_f = 1. \quad (14)$$

One may calculate the density of the cascade network as follows:

$$d_c = \frac{2^p + 2^{p-1} - 2}{(2^p - 1)(2^p - 2)} \quad \forall p \geq 2. \quad (15)$$

In Fig. 7, we plot the network density against network size for the previous three networks.

A random strongly connected network (RSCN) may have a topology shown in Fig. 6. The network density of an RSCN may be anywhere between the solid line of a ring network and the

dashed line of the fully connected network in Fig. 7. Specifically, an RSCN in Fig. 6 has its connectivity matrix

$$C_{ra} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (16)$$

This RSCN has 77 directed edges and 15 nodes. Its network density

$$d_{ra} = \frac{77}{210} = 0.37.$$

That being said, it is denser than the cascade network of the same size. In our experiments, we try a variety of network topologies. The results show that the cascade networks are not the “golden” density positions. A random network of the same size, such as the one in Fig. 6, may achieve better performance in terms of training speed.

C. Synchronization

Other than network topology, timing of local computing also plays an important role on the effect of data accumulation, and therefore, the training speed. In this paper, we discuss two timing strategies: synchronous and asynchronous implementations.

One strategy is to process local data upon receiving SVs from one site’s all immediate ancestor sites. For example, in network defined in Fig. 6, in second and later iteration, site 1 will start processing once having received data from sites 7, 9, and 15, given it is currently idle. One may also observe this from the first column of (16). We call this strategy synchronous implementation as each server needs to wait for all of its immediately ancestor sites to finish processing.

Another strategy is called asynchronous implementation. In this strategy, each site processes its data upon receiving new data from any of its immediate ancestor sites. For example, in network defined in Fig. 6, in second and later iterations, site 1 will start processing once having received data from either site 7, 9, or 15, given it is currently idle. Asynchronous implementation guarantees that all servers are utilized as much as possible.

The advantage and disadvantage of both implementation strategies are empirically compared in Section V.

D. Online Implementation

Since the DPSVM constructs a local problem by adding critical data, local SVM training problems in subsequent iterations are essentially problem adjustments, which append variables

and constraints based on the problem of the last iteration. Online algorithm is a popular solution for incremental learning problems. In this paper, we compare a simple online algorithm with an offline algorithm.

In the simple online implementation, at each iteration, every site always uses current dual solution (α values) as its initial α values. The newly added training data will also bring a vector of nonzero α ’s. We use a slightly modified SVM^{light} [15] as our local SVM solver, which may take an arbitrary α vector as input. To avoid possible infinite loops due to numerical inaccuracies in the core QP-solver, a somewhat random working set is selected in the first iteration and we repeat it for every 100 iterations.

Our experiments in Section V show that this simple online implementation greatly improves the distributed training speed over the offline implementation.

V. PERFORMANCE STUDIES

We test our algorithm over a real-word data set: MNIST database of handwritten digits [17]. The digits have been size-normalized and centered in a fixed-size image. This database is a standard database online for researchers to compare their training method because MNIST data is clean and does not need any preprocessing. All digit images are centered in a 28×28 image. We vectorize each image to a 784 dimension vector. The problem is to classify digit 0 from digit 2, which involves 11 811 training vectors. A linear kernel is applied in all experiments to simplify the parameter tuning process.

We first introduce our terminology used in this section.

N	Total number of training samples.
L	Total number of distributed sites.
E	Total number of directed edges in the network.
d	Density of network.
N_{sv}	Total number of global SVs.
T	Total number of DPSVM iterations.
δ	Total number of transmitted data per time.
Δ	Total number of transmitted vectors.
$N_{l,t}$	Number of training samples in site l at iteration t .
$\max\{N_{..}\}$	Maximum number of training samples among all sites among all iterations.
e	Elapsed central processing unit (CPU) seconds of running DPSVM.
ς	Standard deviation of initial training data distribution.

Throughout this section, we use Pentium 4 3.2 GHz with 1 GB RAM to solve local SVM problem.

A. Effect of Initial Data Distribution

In real-world applications, data may be distributed arbitrarily around the world. In certain applications, the distributed data may not be balanced or unbiased. For example, search engine

TABLE III
PERFORMANCE OVER THE INITIAL DATA DISTRIBUTION

ς	T	δ	Δ	$\max_{l,t}\{N_{lt}\}$	e
Random Dense Network					
0	8	126	15191	1836	9.00
305.7	8	115	13897	2086	8.93
610.7	8	130	15363	2247	12.32
766.9*	6	102	12247	2097	16.92
Binary Cascade Network					
0	16	37	8842	1008	15.51
305.7	17	35	8977	1569	15.63
610.7	18	30	8120	1653	17.20
766.9*	18	26	6906	1731	16.43

*: Training data are evenly distributed in eight sites. The other seven sites are empty before receiving any SV from other sites.

TABLE IV
TESTED NETWORK CONFIGURATIONS

L	Topology	E	d
3	ring	3	0.50
3	binary-cascade	4	0.67
3	fully connected	6	1.00
7	ring	7	0.17
7	binary-cascade	10	0.24
7	random sparse	14	0.33
7	random dense	33	0.79
7	fully connected	42	1.00
15	ring	15	0.07
15	binary-cascade	22	0.10
15	random sparse	54	0.26
15	random dense	159	0.76
15	fully connected	210	1.00

companies usually need to classify Web pages for different markets, where the document data are usually stored locally due to both legal and technical constraints. Those data are highly unbalanced and biased. For example, Chinese market may contain much less business pages than the United States market. In other applications, the data may be evenly distributed, intentionally or naturally. For example, the United States Citizen and Immigration Services (USCIS) often distributes cases to their geographically distributed service centers to balance workloads.

To analyze the effect of initial data distribution, we randomly distribute the 11811 training data to a fully connected network with 15 sites. We do this several times with different settings of standard deviation of data distribution. When the data are distributed evenly, the standard deviation of the initial data distribution ς is approximately 0. When ς is larger, the distributed data are more unbalanced. We run our DPSVM for $\varsigma = 0, 305.7, 610.7$, respectively. We also test a special portional distribution where all data are initially distributed into eight sites only and the rest seven sites are left empty. This particular distribution has $\varsigma = 766.9$. In a four-layer binary cascade network, the data are initially distributed into the first layer under this distribution. The total communication cost Δ and elapsed training time e against the initial data distribution are summarized in Table III. The results show that evenly distributed data may result in a fast convergence (less iteration and shorter CPU time) while the special portional distribution (eight sites of data and seven empty sites) obviously achieves less communication overhead.

B. Scalability on Network Topologies

We test our algorithm over an array of network configurations. Five network topologies are tested including ring networks, binary cascade networks, fully connected networks, random sparse networks, and random dense networks. For the two types of random networks, we impose a constraint over the network density such that

$$d \leq 0.33$$

for random sparse networks and

$$d \geq 0.75$$

for random dense networks. Networks of sites 3, 7, and 15 are tested for ring, binary cascade, and fully connected networks. Networks of sites 7 and 15 were tested for the two types of random generated networks. The number of sites, number of edges, and the corresponding network densities of all tested networks are summarized in Table IV.

In this section, we apply online and synchronized implementation for each tested networks. The offline and asynchronous implementation will be discussed later. We record the total training time in terms of elapsed CPU seconds, the number of iterations, and the number of transmitted training vectors per site. The results are plotted in Fig. 8. The results show that our algorithm scales very well for all network topologies except for the ring network, given the size of the network is limited. In ring networks, however, one site's information reaches all other sites only after at least $L - 1$ iterations. The communication is, therefore, not efficient and the convergence becomes slow. The convergence over a variety of network topologies can be observed from the number of iterations in Fig. 8(b). One may also observe that a random dense network may outperform binary cascade SVM in terms of training time when the network size is not too small. In a small network, a fully connected network has its advantage over other topologies. Fig 8(c) shows the fact that the communication cost per site is not flat when size of the network is increasing. Therefore, the total communication overhead increases faster than the network grows. The reason is because of the synchronization. That is, each site has to wait and accumulate data from all of its ancestor sites in each iteration. This issue will be further discussed in Section V-D.

C. Effect of Online Implementation

We implement a simple online algorithm described in Section IV-D. To show its effect, we plot the computing time in each iteration at site 1 in Fig. 9. One may observe that in offline implementation, the computing speed heavily depends on the absolute number of SVs. On the other side, the online computing time is more sensitive to the change of the number of SVs. The initial sharp increase of CPU seconds is due to the sharp increase of number of local SVs. The later decreasing of CPU seconds is because of the advantage of online implementation.

D. Effect of Server Synchronization

Recall that each site may start processing its local training vectors every time when it receives a batch of training data from

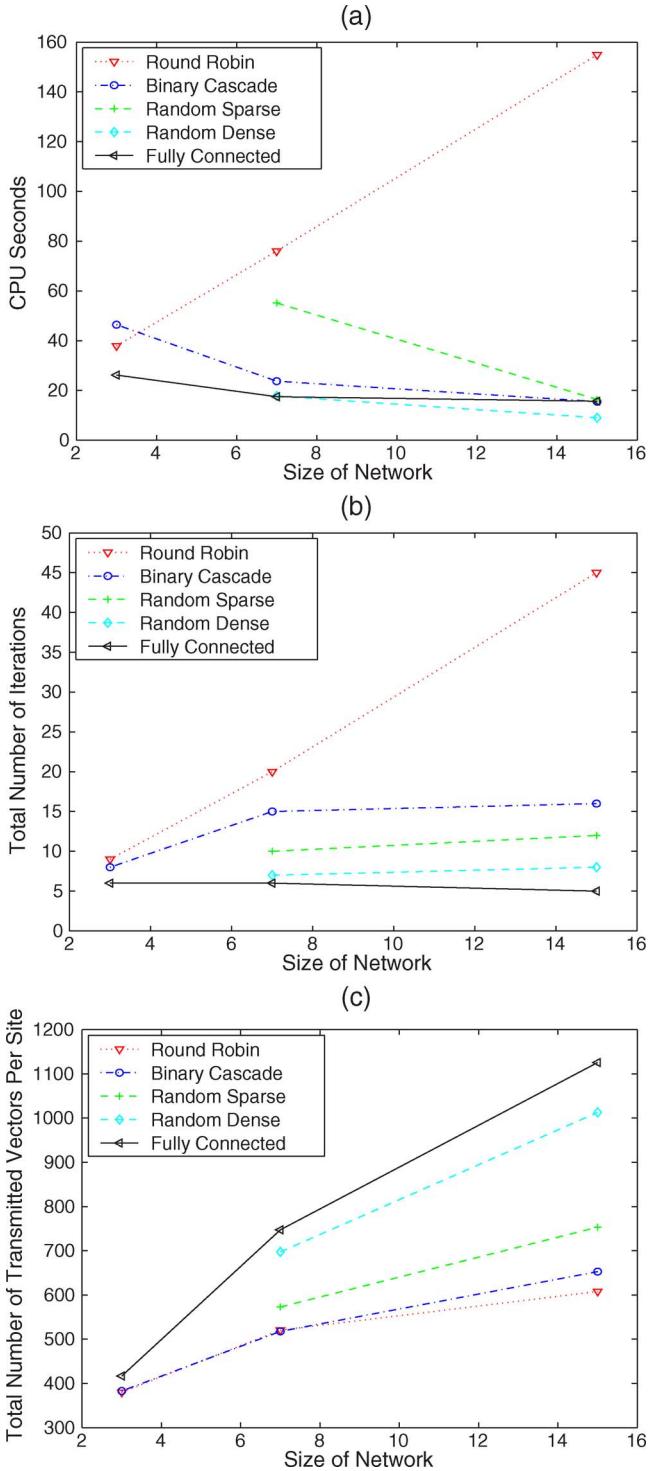


Fig. 8. Performance over network configurations. Five network topologies are compared. (a) Elapsed CPU seconds of training time over size of networks. (b) Number of iterations over size of networks. (c) Total number of transmitted training vectors per site upon convergence over size of networks.

one of its ancestor sites. We call it asynchronous implementation. Another implementation is to let each site wait until it receives new data from all of its ancestor sites. We call it synchronous implementation.

We implement the two strategies in a variety of networks. We compare their total training time and communication overhead for each of networks in Figs. 10 and 11, respectively. One may

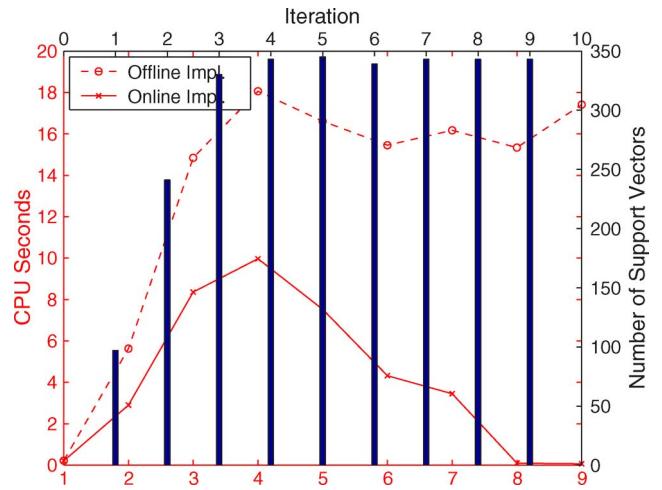


Fig. 9. Effect of the online implementation. The number of SVs and the computing time of site 1 per iteration DPSVM, implemented over a random sparse network with 15 sites. The solid line is the computing time per iteration for online implementation. The dashed line represents the computing time per iteration for offline implementation. The bars represent the number of SVs per iterations. The figure demonstrates that the computing time of offline implementation also depends on the number of SVs, while the online implementation does not.

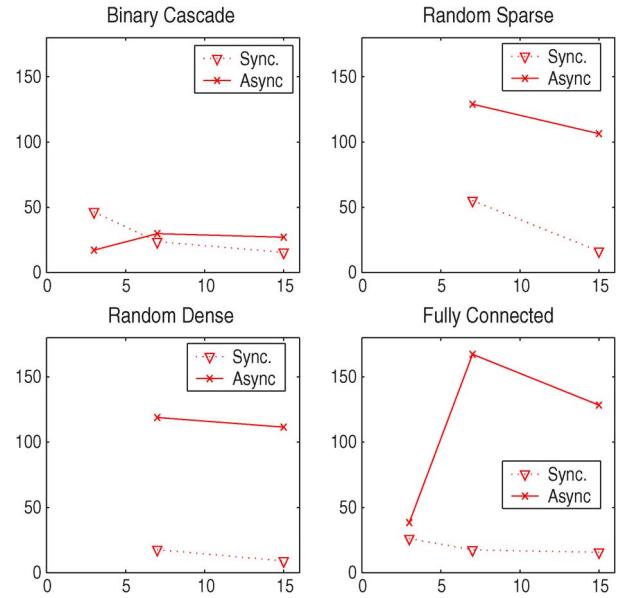


Fig. 10. Training time of asynchronous and synchronous implementation.

observe that, except for the cascade network, the synchronous implementation always dominates asynchronous one in terms of total training time. The reason that the difference between synchronous and asynchronous implementations is small in cascade networks is probably because the topology of the cascade network already ensures certain synchronization.

A more interesting result is shown in Fig. 11, where one may observe that the communication cost increases almost linearly with network size in asynchronous implementations and slower than in synchronous implementations. The scenario can be explained as follows. In asynchronous implementation, because there are always sites that process slower than other sites, the computing and communications are more frequent among those faster sites. The network that is effectively working is actually smaller because those slower sites contribute relatively less. We

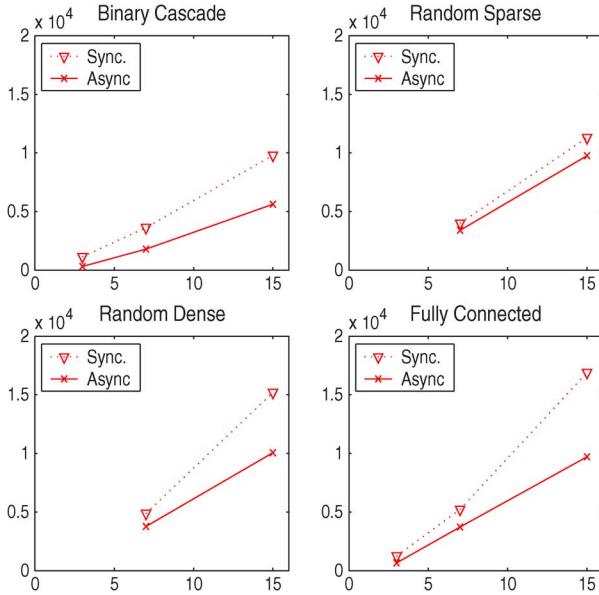


Fig. 11. Communication overhead of asynchronous and synchronous implementation.

TABLE V
TESTED ACCURACY OF DPSVM AND NORMAL SVMs

L	DPSVM	Min (SVM)	Max (SVM)	Mean (SVM)
1	0.9920	0.9920	0.9920	0.9920
3	0.9920	0.9881	0.9916	0.9903
7	0.9920	0.9871	0.9911	0.9878
15	0.9920	0.9806	0.9891	0.9854

know that a small network usually has longer processing time and less communication cost, and the asynchronous implementation is similar as when using a smaller network.

E. Performance Comparison Between SVM and DPSVM

When merging all training data is infeasible due to physical or political reasons, one may apply the DPSVM that is able to reach a global optimal solution or the normal SVM that works on local data only. We demonstrate that our DPSVM has the real advantage in terms of test accuracy over normal SVMs that are trained based on local training data.

We compare DPSVM and SVM when MNIST data are randomly distributed into sites 3, 7, and 15 assuming that each site has the same number of training vectors. The training parameter is estimated following the approach introduced in Section IV-A. We compare the test accuracy of DPSVM to the minimum, maximum, and average values of the test accuracy of normal SVM among all sites. The results are summarized in Table V. The table shows expected phenomena: when less training is used, the achieved test accuracy is worse. This is one of the motivations of the DPSVM: to reach global optimum and better test performance without aggregating all the data.

VI. CONCLUSION AND FUTURE RESEARCH

The proposed DPSVM training algorithm exploits a simple idea of partitioning training data and exchanging SVs over an SCN. The algorithm has been proved to converge to the global optimal classifier in finite steps. Experiments over a real-world database show that this algorithm is scalable and robust. The properties of this algorithm can be summarized as follows.

- The DPSVM algorithm is able to work on multiple arbitrarily partitioned working sets and achieve close to linear scalability if the size of the network is not too large.
- Data accumulation during SVs exchange is limited if the overall SVs are limited. Communication cost is proportional to the number of SVs. Asynchronous implementation over a sparse network achieves the minimum data accumulation.
- The DPSVM algorithm is robust in terms of computing time and communication overhead to the initial distribution of the database. It is suitable for classification over arbitrary distributed databases as long as a network is denser than the ring network.
- In general, denser networks achieve less computing time while sparser networks achieve less data accumulation.
- Online implementation is much faster. A fast online solver is critical for our DPSVM algorithm. This is also one of our future research directions.

REFERENCES

- [1] Y. Zhu and X. R. Li, "Unified fusion rules for multisensor multi-hypothesis network decision systems," *IEEE Trans. Syst. Man Cybern. A, Syst. Humans*, vol. 33, no. 4, pp. 502–513, Jul. 2003.
- [2] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.
- [3] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, pp. 273–297, 1995.
- [4] H. Drucker, D. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1048–1054, Sep. 1999.
- [5] L. Cao and F. Tay, "Support vector machine with adaptive parameters in financial time series forecasting," *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1506–1518, Nov. 2003.
- [6] S. Li, J. T. Kwok, I. W. Tsang, and Y. Wang, "Fusing images with different focuses using support vector machines," *IEEE Trans. Neural Netw.*, vol. 15, no. 6, pp. 1555–1561, Nov. 2004.
- [7] L. M. Adams and H. F. Jordan, "Is SOR color-blind?," *SIAM J. Sci. Statist. Comput.*, vol. 7, no. 2, pp. 490–506, 1986.
- [8] U. Block, A. Frommer, and G. Mayer, "Block coloring schemes for the SOR method on local memory parallel computers," *Parallel Comput.*, vol. 14, no. 11, pp. 61–75, 1990.
- [9] G. Zanghirati and L. Zanni, "A parallel solver for large quadratic programs in training support vector machines," *Parallel Comput.*, vol. 29, pp. 535–551, Nov. 2003.
- [10] N. Syed, H. Liu, and K. Sung, "Incremental learning with support vector machines," in *Proc. 5th ACM SIGKDD Int. Conf. Knowl. Disc. Data Mining*, San Diego, CA, 1999.
- [11] C. Caragea, D. Caragea, and V. Honavar, "Learning support vector machine classifiers from distributed data sources," in *Proc. 20th Nat. Conf. Artif. Intell. Student Abstract Poster Program*, Pittsburgh, PA, 2005, pp. 1602–1603.
- [12] A. Navia-Vazquez, D. Gutierrez-Gonzalez, E. Parrado-Hernandez, and J. Navarro-Abellan, "Distributed support vector machines," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 1091–1097, Jul. 2006.
- [13] H. P. Graf, E. Cosatto, L. Bottou, I. Durdanovic, and V. Vapnik, "Parallel support vector machines: The cascade SVM," in *Proc. 18th Annu. Conf. Neural Inf. Process. Systems*, Vancouver, BC, Canada, 2004, pp. 521–528.
- [14] Y. Lu and V. Roychowdhury, "Parallel randomized support vector machine," in *Proc. 10th Pacific-Asia Conf. Knowl. Disc. Data Mining*, 2006, pp. 205–214.
- [15] T. Joachims, "Making large-scale SVM learning practical," in *Advances Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. Cambridge, MA: MIT Press, 1998, pp. 41–56.
- [16] T. Joachims, "SVM-light support vector machine," 1998 [Online]. Available: <http://svmlight.joachims.org/>
- [17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.