

Capítulo 4

MAPEAMENTO SEMÂNTICO ENTRE ONTOLOGIAS

C: Um título mais representativo do mapeador desenvolvido?

C. Existe
com mais
dureza

A denominação L-Match vem do inglês Learning Match e foi dada ao sistema desenvolvido neste trabalho, pois seu funcionamento é centrado em Classificação Supervisionada, uma subárea de Aprendizado de Máquina. Neste contexto, propomos uma abordagem para a identificação de mapeamentos semânticos ~~de~~ locais entre conceitos de ontologias distintas com base na mensuração da intersecção entre os conceitos estimados por classificadores.

A intersecção é estimada??

Investigamos características de instâncias de ontologias OWL para então descrevermos informações sobre conceitos e finalmente, utilizando a estrutura taxonómica como guia,

estabelecemos os mapeamentos. Instâncias são ricas em evidências que funcionam na prática como canalizadores da informação do esquema da ontologia, obtendo características como conceitos, propriedades, generalizações, além de tipos e valores de propriedades em cada instância. As instâncias são então convertidas em texto em função de suas características que são separadas em escopos textuais, estabelecendo com isso uma entrada genérica para diferentes classificadores supervisionados, cuja saída é utilizada no mapeamento.

O interesse de estimar a intersecção ou sobreposição entre conceitos surgiu principalmente pela seguinte razão: a atual eficácia dos métodos de classificação torna viável a implementação da relação de compatibilidade entre contextos locais definida pela

O Estes primeiros parágrafos devem ser reescritos. Parecem conter informações demais com poucas palavras. Coloque o que quer comentar em mais alto nível e deixe os detalhes para o corpo do capítulo.

→ Semântica de Modelos Locais (Ghidini, et al., 2001) (veja a Seção 1.3). Já o interesse em mapeamento semântico é relevante, uma vez que o único mapeador realmente semântico desenvolvido até hoje é o *S-Match* (Giunchiglia, et al., 2005), e antes dele todos os mapeadores eram sintáticos. O *L-Match* surge como abordagem alternativa ao *S-Match*, estimulando e continuando pesquisas sobre mapeamento semântico.

O *S-Match* e *L-Match* são estruturalmente restritos à taxonomia por não considerar ainda regras e restrições genéricas. Esta restrição pode não ser tão grave, pois *S-Match* e OWL são fundamentados em Lógica de Descição, por isso podemos futuramente generalizar a taxonomia, entendendo que regras e restrições criam superconceitos anônimos contendo conceitos nomeados. Como o *S-Match*, o *L-Match* também tem dependências: como todo sistema de aprendizado de máquina supervisionado, depende da quantidade de instâncias no conjunto de treinamento, atualmente escassas em ontologias OWL. Contudo, esta escassez se deve em muito ao fato de ontologias constituírem ainda uma tecnologia nova, cujo uso ainda não é tão difundido quanto o de diretórios Web e bancos de dados relacionais. Mas isso não é justificativa para ignorar as instâncias, afinal são evidências que representam conhecimento especialista local e, se for o caso, a escassez pode ser contornada por abordagens de instanciação automática que mineram a Web.

Ao todo, foram utilizadas três estratégias de comparação (Subseção Erro! Fonte de referência não encontrada.): comparação terminológica, extensional e da estrutura externa. Portanto, a arquitetura geral do *L-Match* foi estabelecida contendo três módulos: Extração, Similaridade e Mapeamento. Veja na figura a seguir:
cada um correspondendo a uma das estratégias de comparação

C. Esta arquitetura deve ser explicado e receber maior destaque - uma seção própria pode ser utilizada. É importante para o entendimento do mapeador.

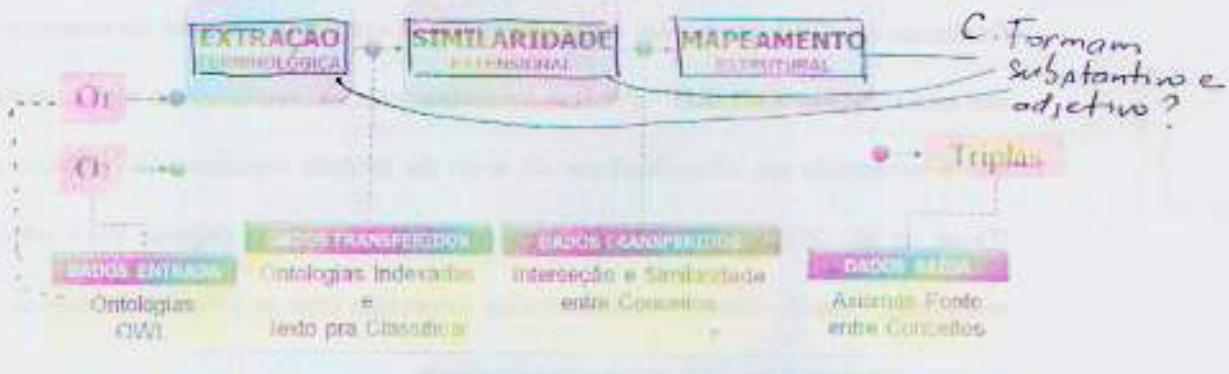


Figura 4: Arquitetura de trabalho proposta

Este capítulo descreve a abordagem proposta nesta Dissertação. Portanto, nas seções seguintes, os três módulos acima serão detalhados: será explicado como gerar e melhorar a qualidade do texto para cada instância, como manipular e combinar classificações para obter similaridade e interseção entre conceitos, como propagar estes valores numa taxonomia e, por fim, como obter os mapeamentos finais na forma axiomática.

1.1 EXTRAÇÃO

O Módulo de Extração é responsável por pré-processar as ontologias a mapear. Ontologias são formadas basicamente pelas mesmas entidades (conceitos, propriedades, instâncias e regras) e por isso nós as padronizamos num formato interno normal que independe do formato inicial (apenas OWL por enquanto), facilitando o mapeamento. As entidades são indexadas porque precisam ser tabeladas nos módulos seguintes, mas a principal funcionalidade deste módulo é gerar texto para classificar, o que iremos discutir nas subseções seguintes.

Indexação e explicada?

1.1.1 GERAÇÃO DE TEXTO PARA CLASSIFICAR

Instâncias em OWL (como em outros formatos de ontologia) não são documentos de texto e sim estruturas em XML/RDF. Por isso, primeiramente são renderizadas num conteúdo textual que é submetido aos classificadores de texto do módulo seguinte. Ao contrário do GLUE (Doan, et al., 2002), não consideramos conteúdo textual de páginas Web relacionadas a ontologias, consideramos apenas instâncias fornecidas com as próprias ontologias, o que nos dá certa autonomia ao gerar texto para classificar.

Ontologias fornecem informações precisas sobre suas instâncias: nome, conceitos, super conceitos e propriedades com valor e tipo definidos. Por exemplo, em uma de nossas ontologias, *WaterTemperature* instancia diretamente o conceito *TemperatureOf* e indiretamente os seus super-conceitos (*EcologicalData*, *Quantity* e *TemperatureQuantity*), relacionando-se com *celsius* (instância direta de *ScaleOfTemperature*) e *water* (instância direta de *EcologicalEntity*) por meio das propriedades *hasTemperatureScale* e *hasEntity*, respectivamente. Na configuração atual do L-Match, as características foram renderizadas e discriminadas em quatro blocos dependendo de sua origem, veja:

Lista de Todos os Conceitos	EcologicalData Quantity TemperatureQuantity TemperatureOf
Lista de Conceitos Diretos	TemperatureOf
Nome da Instância	waterTemperature
Propriedades de Objeto	hasTemperatureScale ← celsius : ScaleOfTemperature hasEntity ← water : EcologicalEntity

C Explicar notação

Figura 1-2) Texto básico gerado para a instância *WaterTemperature*

(Fornece
experiência para
onde na discussão
também se
discutido)

O texto gerado é pequeno e preciso já que é composto por palavras-chave. Observe que os conceitos diretos são discriminados, porém os super conceitos não: essa configuração

foi escolhida empiricamente porque propiciou os melhores valores de avaliação nos experimentos. Mas o texto precisa ser tratado: os nomes são convertidos para minúsculo, separados em tokens e remove-se as stopwords (*to, has, of, the, etc.*) deixando apenas as *keywords* (palavras-chave). O resultado é o seguinte:

C:
Justificar

Entidade	Tipo de Entidade	Keywords
	Entidade	ecological data
	Entidade	quantity
	Entidade	temperature quantity
	Entidade	temperature
	Entidade	WaterTemperature
Nome da Entidade		
Entidade	Propriedade da Entidade	temperature scale = celsius : scale temperature entity = water : ecological entity

Figura 1.3: Treinamento para a classificação de expressões

WaterTemperature está quase pronta para ser classificada, falta ainda ver estratégias mais elaboradas para melhorar a qualidade do texto na subseção seguinte.

1.4.2 QUALIDADE DO TEXTO

Deve-se lembrar que o resultado do Módulo de Extração será a entrada de classificadores de texto: por melhor que seja o algoritmo de classificação, ele não funcionará satisfatoriamente se os dados submetidos a ele forem de má qualidade. Portanto, o Módulo de Extração é muito importante, uma vez que influenciará diretamente a precisão dos classificadores e consequentemente do mapeamento final. Esta subseção descreve as estratégias que consideramos essenciais para qualidade: a utilização de escopos textuais

associada à identificação de sinônimos ajuda a obter boa qualidade textual. Primeiramente, observe a palavra *temperature* ocorrendo em todos os blocos principais do texto seguinte:

Lista de Todos os Conceitos	ecological data quantity temperature quantity temperature
Lista de Conceitos Diretos	temperature
Nome da Instância	water temperature
Propriedades de Objeto	temperature scale - celsius : scale temperature entity - water : ecological entity

Figura 1-4: Palavra repetindo-se em diferentes partes do texto prejudica a qualidade

Blocos têm semântica específica, mas isso não está explícito, causando um efeito negativo. Por exemplo, *temperature* ocorrendo como conceito que contém *WaterTemperature* não se diferencia de *temperature* ocorrendo como tipo de propriedade? esta instância pertence aos conceitos *TemperatureQuantity* e *TemperatureOf*, mas não a *ScaleOfTemperature*. Para esta situação não confundir os classificadores, adotamos uma solução simples e eficaz: como classificadores compararam *strings*, prefixamos as palavras originais, gerando novas palavras que discriminam **escopos textuais**, mesmo que originalmente as palavras sejam idênticas:

	CH	ClassName
Lista de Todos os Conceitos	CNecological CNdata CNquantity CNtemperature CNquantity CNtemperature	DC - DirectClass IN - InstanceName PN - PropertyName
Lista de Conceitos Diretos	DCtemperature	PV - PropertyValue PT - PropertyType
Nome da Instância	INwater INtemperature	
Propriedades de Objeto	PNtemperature PNscale - PVcelsius : PTscale PTtemperature PNentity - PVwater : PTecological PTentity	

Figura 1-5: Adição de prefixos de escopo no texto

Agora, toda palavra recebe um prefixo diferente dependendo do escopo textual em que aparece, explicitando a semântica do escopo. Para um classificador de texto, esta é uma forma indireta de passar significado. No exemplo, *temperature* não é mais a mesma palavra morfológicamente: CN*temperature*, DC*temperature*, IN*temperature*, PN*temperature* e PT*temperature*. Portanto, os escopos textuais foram utilizados para indicar o contexto de cada característica da instância, ou seja, de que parte da ontologia cada característica é ortodoxa, evitando com isso maiores ambiguidades.

Agora discutiremos sobre identificação de sinônimos e desambiguação de palavras. Acreditamos que reduzir palavras a uma única forma padrão é útil, pois ajuda a resolver o problema de nomear conceitos equivalentes com palavras diferentes nas ontologias.

Experimentos preliminares de hold out (reclassificação de instâncias na mesma ontologia) obtiveram altos valores de avaliação. Isso ocorre porque o vocabulário de uma ontologia é restrito e bem formado (específico, controlado, etc.), sem tanta "impureza" como, por exemplo, em hipertexto. Porem, vocabulários distintos apresentam muitas diferenças quando são comparados. Isto é interessante reduzir estas diferenças na medida do possível para, mais uma vez, facilitar a ação dos classificadores, os quais compararam *strings*.

A técnica terminológica aplicada durante a extração visa amenizar diferenças entre os vocabulários de um par de ontologias, estabelecendo um terceiro vocabulário comum que seja o maior possível. Palavras são reduzidas morfologicamente através de *stemming* e semanticamente através de *synsets* (grupos de sinônimos) para uma forma padrão e única. Descobrir *synsets* automaticamente seria ideal, porem é mais prático utilizar conhecimento externo via *thesaurus*. O algoritmo de *stemming* é o *Porter Stemmer*¹ (Porter, 1997) e o serviço de *thesaurus* é o *WordNet*² (Fellbaum, 1998). Veja exemplos de reduções:

C Explique o algoritmo brevemente.

¹ <http://tartarus.org/~martin/PorterStemmer/>

² <http://wordnet.princeton.edu/>

- C. *Explicar notações*
1. $\text{Base} \leftarrow \{\text{Base}, \text{Basic}, \text{Primary}, \text{Fundamental}\}$
 2. $\text{Dimens} \leftarrow \{\text{Dimension}, \text{Dimensions}\}$
 3. $\text{BaseDimens} \leftarrow \{\text{PrimaryDimension}, \text{FundamentalDimensions}\}$

C. *Explicar composições* Por que outras composições possíveis

Porém, os *synsets* do WordNet são insuficientes porque não incluem todas as palavras ex. *BasicDimension*, *PrimaryDimensions*, etc.) nelas?

que podem ser utilizadas como sinônimos: *Primary* e *Fundamental* são sinônimos óbvios, mas não estão num mesmo *synset* do WordNet. Felizmente, o WordNet mantém *links* entre palavras similares que não foram consideradas sinônimos. Explorando estes *links*, é possível descobrir o caminho *Fundamental* → *Important* → *Primary* e vice-versa, permitindo gerar um novo *synset* que inclui *Fundamental* e *Primary*. Mas outro problema surge: caminhando por *links* de similaridade, pode-se ir muito longe e gerar *synsets* grandes que criam generalizações em vez de equivalências? o que é indesejável. Veja:

1. $C \leftarrow \{C, IV, Carbon, Coulomb, Celsius, century\}$

Para explorar *links* de similaridade e controlar a abrangência do caminhamento,

procuramos por **cliques** no grafo de similaridade, gerando *synsets* mais equivalentes a partir

do WordNet: entendemos que palavras formam um *synset* se existir caminho de similaridade entre todo par de palavras, ou seja, as palavras concordam que são sinônimas quando formam

um clique entre si. Com isso introduzimos a idéia de **Unidade de Significado ou Clique**

(Venant, 2006) como a menor unidade semântica possível, o que tem sido aplicado em

trabalhos sobre modelos geométricos de palavras organizadas em coordenadas semânticas, como é o caso do *Atlas Semântico*³, um dos melhores *thesauri* disponíveis na Web.

³ <http://dico.isc.curs.fr/en/index.html>

Sendo assim, dadas duas ontologias a mapear, o vocabulário de ambas é extraído e transformado em um grafo direcionado $G = (V, E)$: as palavras são vértices em V e os links de similaridade em E são importados do WordNet. O passo seguinte é encontrar todos os cliques máximos no grafo, quando então nos deparamos com o problema *NP-Difícil* da **enumeração dos cliques máximos** (Seção Erro! Fonte de referência não encontrada.).

Apesar desta显而易见的 dificuldade, utilizamos um algoritmo exato como em (Bron, et al., 1973), pois o espaço de busca da nossa aplicação é pequeno por natureza: observe que $|V|$ é restrito ao tamanho do vocabulário extraído das ontologias, além do que *synsets* nunca são grandes e as palavras similares formam subgrafos pequenos que não se comunicam. Nos experimentos, o maior clique gerado ~~tem~~ ^{teve} tamanho 5 e não houve impacto no desempenho.

Veja o exemplo:

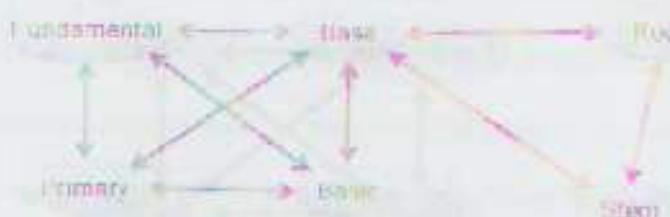


Figura 1-6: Exemplo de significados ambíguos de uma palavra.

C. Néonto?

No Figura 1-6 temos um exemplo de **polissemia** ou **ambigüidade**: a palavra *base* aparece em dois *synsets* distintos, logo possui dois significados. Isso contrasta severamente com a ideia de reduzir todas as palavras a uma única forma padrão, pois neste caso existem duas possibilidades de redução para *base*. A solução é eliminar polissemia garantindo que cada palavra esteja em um único *synset*, isto é, gerar um *thesaurus monossêmico*.

O problema de escolher o mais apropriado *synset*, significado ou sentido para uma palavra dentro de um contexto é chamado de **Desambiguação de Sentido de Palavra** ou *Word Sense Disambiguation* (WSD). Trabalhos sobre **Desambiguação Suave (Soft Disambiguation)** permitem que palavras continuem tendo mais de um significado no mesmo contexto e criticam a atribuição de apenas um *synset* por palavra. Contudo, as necessidades nesta Dissertação são diferentes, o que nos leva a concordar com (Buitelaar, et al., 2001) sobre que no contexto de um domínio restrito (como é o caso das ontologias) muitos termos ambíguos certamente terão forte preferência por apenas um de seus *synsets*, o qual é específico e maximiza a similaridade entre os domínios das ontologias a mapear. Por exemplo, nos nossos experimentos gostaríamos que os *synsets* da Figura 1-6: **Unidades de Significado**: os cliques seriam os novos *synsets* fossem desambiguados como segue:

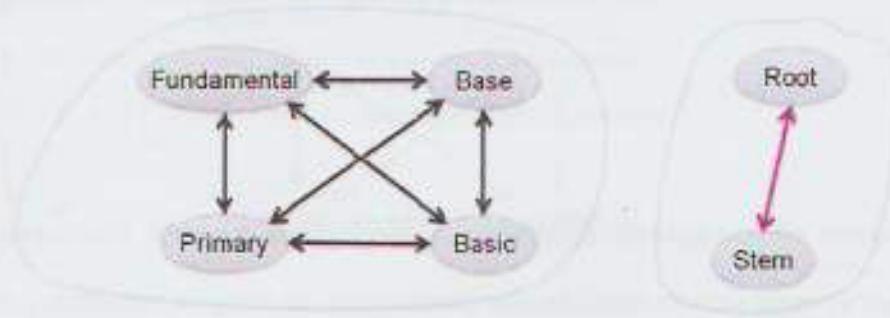


Figura 1-7: Apenas *synsets* mais informativos são mantidos, eliminando a polissemia.

Com base em Teoria da Informação, elaboramos uma técnica simples e eficaz para desambiguação, visando eliminar polissemia por completo. Diferente do tradicional, esta desambiguação não depende do contexto taxonômico e sim do contexto do vocabulário do domínio, isto é, apenas os ***synsets* mais importantes e informativos** para o domínio continuarão existindo. Os *synsets* são ranqueados do menos para o mais importante, quando então se aplica uma estratégia gulosa: percorrer o *rank* do inicio ao fim, penalizando cada

C Como?

Re-escr-
ver.

C. Re-
busca
nar an-
tes da
barreira
fo ante-
rior

- synset* removendo dele as palavras ainda polissêmicas. Ao atingir o final do *rank*, todas as palavras serão monossêmicas. → muitos *synsets* do íncio do *rank* deixarão de existir e as palavras tenderão a se concentrar nos *synsets* mais importantes do final do *rank*.

Agora precisamos criar nossa **definição de importância**: um *synset* importante possui informação valiosa para o domínio, já um *synset* sobreposto a muitos outros é polissêmico e não traz informação. Intuitivamente, utilizamos a ideia de **entropia** (Seção Erro! Fonte de referência não encontrada.) para mensurar a importância de um *synset* dada pela quantidade de informação que ele contém: quando mais polissêmico o *synset* possuir, menos informação ele terá, constituindo um forte candidato à eliminação. Para isso, a importância de um *synset* é definida em função (a) do total de palavras polissêmicas no *synset* e (b) do total de significados que cada palavra possui, isto é, total de *synsets* que se sobrepõem ao *synset* original.

Sejam dois vetores *A* e *B* criados para um *synset* *S* com *n* palavras, onde *A* é um vetor de inteiros que contabiliza a quantidade de *synsets* para cada palavra de *S*, e, onde *B* é um vetor booleano que marca cada palavra de *S* como polissêmica (*true*) ou monossêmica (*false*). A quantidade de informação em *S* é dada pela combinação *NoisyOr* (Seção Erro! Fonte de referência não encontrada.) da informação contida nos dois vetores *A* e *B*:

$$\text{Equação 5: } \text{Informação}(A, B) = \text{NoisyOr}(\text{Entropia}(A), \text{Entropia}(B))$$

Perceba que quantificar informação como grandeza oposta à polissemia é intuitivo e ajuda a eliminar *synsets* genéricos que pouco ou ~~mais~~ ajudarão. Análogo à definição de entropia, pode-se mensurar a importância do mesmo *synset* *S* com *n* palavras como segue:

Equação 2: $FatorA(S) = 1 - \text{contagemDePalavrasPolissemicas}(S)/n$

Equação 3: $FatorB(S) = n / \text{contagemDeSynsetsSobrepostos}(S)$

Equação 4: $Fator(S) = \text{NoisyOr}(\text{fatorA}(S), \text{fatorB}(S))$

- C - Nas equações, mas os fatores nas equações.
- Estas equações alcançam o maior valor de importância "1" quando não existe palavra polissêmica em S . A Equação 2 atinge o menor valor de importância "0" quanto toda palavra em S é polissêmica, a Equação 3 tende a zero quanto mais significados cada palavra em S possuir e a Equação 4 combina as duas Equações anteriores. Finalmente, a importância de um synset ficou definida nesta Dissertação como segue:

Equação 5: $Importancia(S) = Informacao(S) * Fator(S)$

C. Deixe comentários -
nos como esse para
o capítulo
de experimentos e
resultados

Durante os experimentos, a importância de S definida apenas pelo cálculo de informação (Equação 1) funcionou muito melhor do que quando definida apenas pelo cálculo do fator (Equação 4). Portanto, a definição de entropia é satisfatoriamente suficiente. Porém, as duas equações funcionam levemente melhor quando combinadas na Equação 5.

Todos os aspectos abordados nesta subseção contribuíram com a precisão da solução deste trabalho. Escopos textuais são essenciais para boa qualidade do mapeamento, enquanto a identificação de sinônimos potencializa sistemas que processam texto, como é o caso do

nosso L-Match, um mapeador baseado em algoritmos de classificação de texto.

C. Esta seção 11.2 está muito longa. Pode ser quebrada em seções menores. As várias técnicas de melhoria da qualidade do texto foram apresentadas principalmente através de texto.)

1.2 COMPUTANDO SOBREPOSIÇÃO E SIMILARIDADE ENTRE CONCEITOS

O Módulo de Similaridade do L-Match estabelece analogias iniciais entre conceitos de ontologias distintas através de algoritmos de classificação supervisionada, inserindo ? nessa corrida. Seria melhor usar alguma forma mais esquemática, talvez diagramas, para a apresentação ficar mais clara.

Dissertação abordagens de Aprendizado de Máquina e de Raciocínio Probabilístico. Nesta seção, discutiremos como manipular classificadores para descobrir dois tipos de analogias entre conceitos: **sobreposição** e **similaridade**, que em seguida sofrerão propagação *bottom-up* na taxonomia. Finalmente, a combinação de classificadores também será abordada.

C: Você parte
do pressuposto
que as ontolo-
gias repre-
sentam conhecimento
de domínios
relacionados,
ou não?

Inicialmente, nada se conhece sobre o que há em comum entre ontologias distintas.

Queremos começar a eliminar esta ignorância classificando instâncias de uma ontologia nos conceitos de outra ontologia e vice-versa, ou seja, fazer **mapeamentos de pertinência** *instância-conceito* (\approx), como esquematizado a seguir:



Figura 1.6: Elaborando relações de pertinência entre ontologias.

Nosso interesse em classificação automática decorre da sua habilidade de relacionar instâncias desconhecidas (conjunto de teste) com categorias conhecidas (conjunto de treino), com certa precisão e utilizando apenas informação heterogênea. Entendemos que ao descobrir relações de pertinência desconhecidas (com ajuda de classificadores) será possível comparar conceitos de ontologias, descobrindo mapeamentos semânticos também desconhecidos até então. Para isso, definimos a arquitetura do Módulo de Similaridade do *1. Match* em duas

partes (C: Isso é uma arquitetura, não um processo)
 etapas que revezam as ontologias hora como conjunto de treino e hora como conjunto de teste, de forma que as instâncias de uma sejam classificadas nos conceitos da outra:



Figura 1.9: Classificação automática alternada das instâncias de um par de ontologias

Quando uma instância é submetida a um classificador, o resultado é um *rank* de conceitos ordenados por similaridade decrescente. Por exemplo, na Figura 1-10 as instâncias do conceito *BaseUnit* foram comparadas por classificadores a conceitos de outra ontologia onde *BaseUnit* não foi declarado:

(Vide próxima página)

BaseUnit	TESTE				TREINO			
	1*	2*	3*	4*	1*	2*	3*	4*
dimensãounit	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000
idion	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
comida	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
local	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
holme	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
amper	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
atencion	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
motor	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
biogram	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000
acento	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000	0.0200000000

SIMILARIDADE	0.000	0.010	0.020	0.030	0.040	0.050	0.060	0.070
SUPERPOSIÇÃO	0.000	0.010	0.020	0.030	0.040	0.050	0.060	0.070

C. Pode-
cem ser
conce-
tos da
Eco-
língua.

Bem um
conceito
da Eco-
língua.

C. Similaridade
de entre ins-
tância e con-
ceito ?
faz sentido
ou trata-se
de classifi-
cações ?

A similaridade é computada entre toda instância e todo conceito de treino e depende do algoritmo de classificação. Mas a classificação só ocorre de fato ao atribuir pertinência entre cada instância com alguns conceitos. Para simplificar, adotamos um limiar *Top-k* com $k=1$ que classifica instâncias apenas na primeira classe de *rank* correspondente. Esclarecer:

Pertinência $x \in C$ e similaridade $Sim(x, C)$ são relações *instância-conceito* inuteis quando sozinhas, contudo podemos combinar as para descobrir relações *conceito-conceito*.

descrivendo conceitos em função de suas instâncias. Sejam então dois conceitos, $A =$

→ $[a_0, a_1, \dots, a_n] \in B = [b_0, b_1, \dots, b_n]$, queremos descobrir possíveis relações $a_i \in A$ e $b_j \in B$ para (a) comutizar as instâncias comuns e descobrir a sobreposição absoluta $|A \cap B|$ e (b) combinar as similaridades *instância-conceito* e descobrir a similaridade *conceito-conceito*.

→ $Sim(A, B)$. As combinações são implementadas com *Noisy* e entropia, a veremos na subseção seguinte.

C. Número de
elementos?

Pela regra teste → treino, $|A \cap B|$ e $\text{Sim}(A, B)$ são estimados nos sentidos

$A \rightarrow B$ (contexto da ontologia de A) e $B \rightarrow A$ (contexto da ontologia de B), sendo que

$A \leftrightarrow B$ é uma combinação que elimina a noção de sentido: trabalhos sobre mapeamento sintático (como o GLUE) perdem informação uma vez que mantêm apenas valores $A \leftrightarrow B$.

Por esta e outras razões (Seção 1.3), $|A \cap B|$ e $\text{Sim}(A, B)$ são mantidos nos três sentidos.

→ 1.2.1 PROPAGAÇÃO BOTTOM-UP NA TAXONOMIA

C. Conceitos
específicos x
genéricos?

Instâncias
direta x
indireta?

Freqüentemente, $|A \cap B|$ e $\text{Sim}(A, B)$ são estimados apenas em conceitos específicos: quanto mais específico, maior a chance de um conceito ser diretamente instanciado e, na prática, a maioria das instâncias está nas folhas, impedindo que classificadores comuns processem conceitos genéricos instaciados indiretamente. Mas a semântica taxonômica permite generalizar/propagar $|A \cap B|$ e $\text{Sim}(A, B)$: um conceito é a união (\sqcup) de suas subsunções (instâncias e subconceitos).

C. Definir explicitamente.

Como a união equivale à operação OR (\vee) e $\text{Sim}(A, B)$ é uma probabilidade, utilizamos o Modelo NoisyOr de Redes Bayesianas para propagação bottom-up de $\text{Sim}(A, B)$, desde instâncias e conceitos específicos até o topo genérico de uma taxonomia: as similaridades instância-conceito $\text{Sim}(x, C)$, obtidas por meio de classificadores, são combinadas na similaridade conceito-conceito $\text{Sim}(A, B)$ que então é propagada para conceitos mais genéricos. A definição recursiva de $\text{Sim}(A, B)$ no sentido $A \rightarrow B$ é:

$$\text{Equação 6: } \text{Sim}(A, B) = 1 - \left(\prod_{x \in A} (1 - \text{Sim}(x, B)) \right) * \left(\prod_{y \in A} (1 - \text{Sim}(y, B)) \right)$$

C. x é instância e y conceito?

Definir x e y

Explicar sentido real da equação.

A Equação 6 estima o NoisyOr: o primeiro produtório combina as similaridades entre B e apenas instâncias diretas de A e o segundo produtório combina as similaridades entre B e os subconceitos de A. Essa configuração evita recombinar as similaridades entre B e as instâncias dos subconceitos de A, que também são instâncias de A.

Por exemplo, a Rede Bayesiana da Figura 4.11 é utilizada para propagação de similaridade numa taxonomia. Nela, conceitos são representados por ~~estrelas~~^{4.11} maiores e instâncias por ~~estrelas~~^{círculos} menores. Deseja-se então calcular a similaridade entre um conceito qualquer Q com todos os conceitos de uma taxonomia, dado que tudo que se sabe é a similaridade entre Q e as instâncias da taxonomia:

(i)

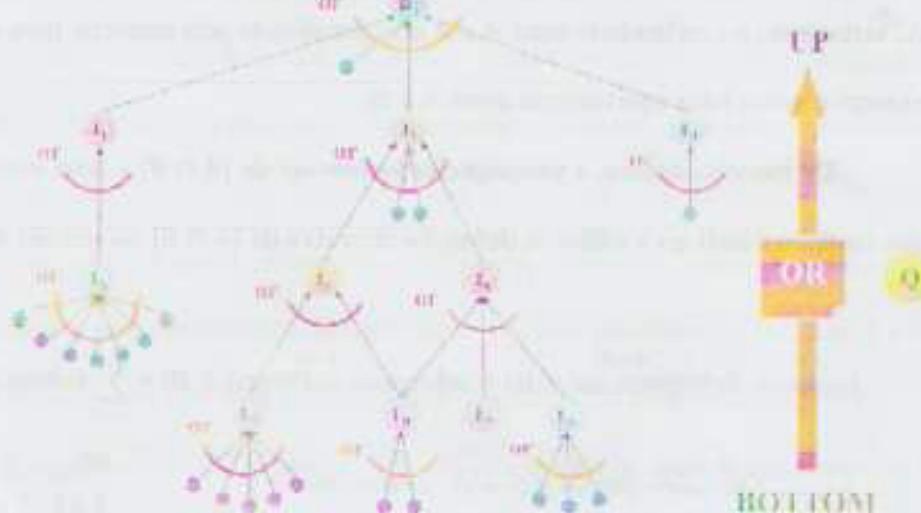


Figura 4.11: Taxonomia modelada para propagação de similaridade.

C A equação propaga ou combina similaridades?

Comendo, a propagação de $\text{Sim}(A, B)$ da Equação 6 não está completa, pois precisa

agregar conceitos de Teoria da Informação. Propagar $\text{Sint}(A, B)$ é equivalente a sintetizar

distribuições de probabilidade, conforme a síntese ocorre através de NoisyOr, a probabilidade

aumenta (maior disponibilidade de instâncias) mas a quantidade de informação diminui

perda de propriedades específicas portanto, quanto mais genérico um conceito, menor a

C Nos
ilustrarmos
vou também a
Eq 6 no
exemplo
Nós vimos
claro
como a pro-
pagação ef-
tivamente
ocorre.

quantidade de informação (Resnik, 1990). Recorremos novamente à definição de entropia para alterar a definição recursiva de $\text{Sim}(A, B)$ no sentido $A \rightarrow B$:

C. Se este é o equação utilizada, a anterior não precisa ser apresentado.

$$\text{Equação 7: } \text{Sim}(A, B) = \text{NoisyOr}(V) * \text{Entropia}(V)$$

- Onde V é a enumeração de $\text{Sim}(x, B)$, para todo $x \in A$ ou $x \subset A$.
- A entropia penaliza a similaridade conforme diminui a informação em V .

→ A perda de informação também ocorre devido a erros de classificação. Por exemplo, mesmo que uma instância de um conceito A erroneamente receba grande similaridade com um conceito disjunto B , espera-se que o mesmo não ocorra com a maioria das instâncias de (i) A . Certamente, a similaridade entre A e B será penalizada pela entropia, para não implicar, por exemplo, numa falsa equivalência entre A e B .

De maneira similar, a propagação *bottom-up* de $|A \cap B|$ é feita através da soma (+), que também é análoga à união. A definição recursiva de $|A \cap B|$ no sentido $A \rightarrow B$ é:

$$\text{Equação 8: } \text{Sobreposição}(A, B) = \text{sobreposição Direta}(A, B) + \sum_{y \in A}^{\overset{A \rightarrow B}{\text{A}}} \text{Sobreposição}(y, B)$$

C. Explicar sentido do equação. C. Mencionar sentidos

Por fim, falta ainda definir as combinações dos valores de similaridade e sobreposição. Para a similaridade, a combinação que teve a melhor avaliação foi a média harmônica, em comparação com *NoisyOr*, média aritmética e *Máx*. Veja:

C. Comentário para a Equação 10?

$$\text{Equação 9: } \text{Sim}(A, B) = \frac{2 * \text{Sim}(A, B) * \text{Sim}(B, A)}{\text{Sim}(A, B) + \text{Sim}(B, A)}$$

$$\text{Equação 10: } |A \cap B| = |A \cap B| + |A \cap B|$$

L.2.2 COMBINAÇÃO DE CLASSIFICADORES

C. Referência
para seção de
experiments

Diferentes algoritmos de classificação foram investigados e comparados em busca da melhor forma de classificar para mapas, incluindo INN, Naive Bayes, SVM, Maxima Entropia, NB-Shrinkage e TF-IDF que foram testados individualmente e combinados.

C. Referência
para discussão -
pôs sobre re-
zões para tal

Nossa motivação inicial era o SVM, pois esperávamos que sobrepuasse os outros algoritmos, como comumente ocorre. Mas os experimentos nos surpreenderam com o pessimo desempenho do SVM, enquanto os melhores resultados vieram de onde menos esperávamos: do NB-Shrinkage, que inicialmente nem ao menos havia sido cogitado.

Ensemble ou meta learner é a denominação de um classificador resultante da combinação de outros. Combina-se as decisões dos algoritmos e não os algoritmos propriamente ditos (embutindo um no outro). Apesar de não haver garantias, espera-se que a eficácia melhore se entendermos a combinação como um processo democrático: se um classificador individual erra, espera-se que a maioria dos classificadores não o faça, discordando do primeiro. Contudo, o processo demora mais a média que mais classificadores são combinados. O ideal seria que um classificador individual produzisse a melhor eficiácia, mas isso nem sempre ocorre, sendo válido investigar *ensembles*.

A combinação convencional de classificadores é feita com *NaiveOr*. Como na Equação 7, preferimos combinar *NaiveOr* com entropia para penalizar respostas controversas quando classificadores discordam entre si. Em \mathcal{S} :ja é uma instância de teste, \mathcal{C} é um conceito de treino e R um vetor de respostas dadas por n classificadores atribuindo diferentes similaridades entre \mathcal{S} e \mathcal{C} , temos que:

$$(1)$$

C. Now paraíba

Equação 7: $\text{combinacaoDoClassificacao}(R) = \text{NaiveOr}(R)^T \text{Entropia}(R)$

1.3 COMPUTANDO MAPEAMENTOS SEMÂNTICOS ENTRE CONCEITOS

Os módulos de Extração e Similaridade fornecem satisfatoriamente valores de similaridade e sobreposição entre conceitos de ontologias distintas, valores que na sequência são utilizados para alimentar o Módulo de Mapeamento, onde finalmente os mapeamentos são estabelecidos em sua forma axiomática e que por isso são semânticos.

1.3.1 REGRAS DE COMPATIBILIDADE ENTRE CONCEITOS

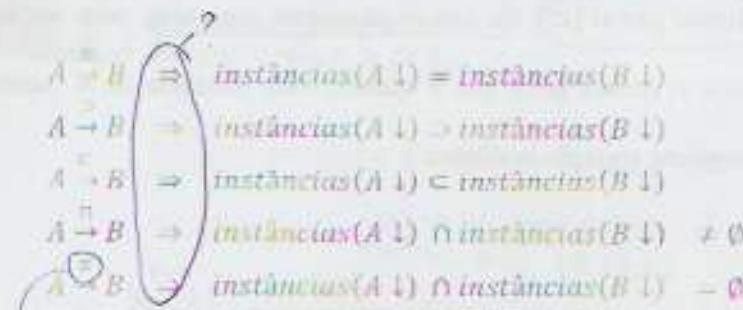
- Apesar de obtermos valores de similaridade e sobreposição, a similaridade apenas diferencia os conceitos.
- ajuda saber o quanto equivalentes/diferentes dois conceitos são. Por isso, o que realmente importa para o mapeamento semântico é a sobreposição, pois conceitos são delimitados pela enumeração de suas instâncias e/ou pela definição de suas propriedades.

Podemos mais uma vez diferenciar L-Match e S-Match, pois o primeiro raciocina sobre a enumeração (estatisticamente e localmente) enquanto o segundo raciocina sobre propriedades taxonômicas (simbolicamente e globalmente) para chegar num objetivo comum: relacionar conceitos através de equivalência (\equiv), mais geral (\supset), menos geral (\subset), disjunto (\sqcap) e diferença (\neq). Teoricamente, concordamos que este problema remete à relação de compatibilidade entre contextos locais definida pela **Semântica de Modelos Locais** (Ghidini, et al., 2001) e que inspirou o S-Match. Na prática, isso é Teoria de Conjuntos utilizada para regras de interpretação de contextos locais, tornando-os compatíveis.

Sendo assim, a semântica formal dos axiomas \equiv , \supset , \subset , \sqcap e \neq é dada pela compatibilidade entre a classificação de documentos (Bouquet, et al., 2003): uma função de $\text{inference}(\cdot)$

O mapeamento será extensionalmente correta se as condições abaixo se aplicarem às instâncias de dois conceitos arbitrários A e B oriundos de hierarquias taxonômicas:

- C. Note que
 $A \sqsupseteq B \rightarrow A \sqsupset B$ e
que
 $A \sqsubseteq B \rightarrow A \sqsupset B$.



C. Será a negação de equivalência que significa
 $\neg(\text{instances}(A↓) = \text{instances}(B↓))$ e não que $\text{instances}(A↓) \cap \text{instances}(B↓) = \emptyset$,
onde o símbolo \downarrow indica que as instâncias indiretas também são consideradas. Ou seja,

Podemos traduzir as regras acima para um algoritmo:

01. SE as instâncias de A e B são as mesmas ENTÃO $A = B$
02. SENÃO SE toda instância de A pertence a B ENTÃO $A \sqsubset B$
03. SENÃO SE toda instância de B pertence a A ENTÃO $A \sqsupset B$
04. SENÃO SE existe alguma instância entre A e B ENTÃO $A \sqcap B$
05. SENÃO $A \neq B$

C. E a de similaridade?

Esta ideia é simples, porém chave, pois através dos valores de sobreposição, obtidos com a ajuda de classificadores supervisionados, implementaremos as regras acima. Contudo, estas regras são muito fortes dado que ocorrem erros de classificação não permitindo esperar,

por exemplo, que dois conceitos compartilhem 100% de suas instâncias para concluirmos que são equivalentes. Portanto, as regras de compatibilidade serão relaxadas através de limites.

Não utilizamos a popular similaridade de Jaccard sobre os valores de sobreposição, pois perdemos informação valiosa, impossibilitando o mapeamento semântico os axiomas \sqsupset e \sqsubset são duacionados, assimétricos e inversos o que remete à direcionalidade do Fluxo da Informação (Bouquet, et al., 2003), área intimamente ligada à semântica da classificação de

*C. Ju ntu que
ou nos merece*

instâncias (Kent, 2000). Logo, deve-se guardar a **ontologia origem** e a **ontologia destino** ao estimar similaridade e sobreposição, mantendo-só nas direções $A \rightarrow B$, $B \rightarrow A$ e $B \leftrightarrow A$.

C. Explicar o sentido da equação.

C. Justificar

C. Este círculo é para que? Relação?

C. Quem?

C. Não é a única escolha possível para o certo desejado,

considera-se A menos geral que B quando toda instância de A também pertence a B, logo

pois temos a seguinte equação booleana:

$$\text{Equação 12: } \text{menosGeral}(A, B) = \frac{|A \cap B|}{|A|} \geq T_{\max_overlap}$$

há uma equivalência lógica entre 'Menos geral' e 'Mais geral' (comparação anterior)

- Onde $T_{\max_overlap}$ é um limiar maior que 0.5 e próximo de 1;
- $|A \cap B|$ é a sobreposição na direção $A \rightarrow B$, pois inclui apenas instâncias de A classificadas em B , independente das instâncias de B classificadas em A ;
- Utiliza-se \geq e não $>$ para podermos implementar a equivalência a seguir.

Equivalência (\equiv) e mais geral (\supseteq) são definidas em função de menos geral (\subseteq).

C. Justificar

pois \supseteq e \subseteq são inversos e $\not\models$ ocorre quando dois conceitos estão contidos um no outro:

$$\text{Equação 13: } \text{maisGeral}(A, B) = \text{menosGeral}(B, A)$$

$$\text{Equação 14: } \text{equivalente}(A, B) = \text{menosGeral}(A, B) \wedge \text{menosGeral}(B, A)$$

C. Explicar

A Equação 12 fica mais precisa conforme $T_{\max_overlap}$ aumenta; um valor na casa dos

80% funciona bem na prática. Contudo, ainda é possível melhorar esta equação, observando-

C. Justificar

se que quando um conceito A está contido em outro conceito B :

a) espera-se que $\text{Sim}(A, B)$ seja 1; todo instância de A ~~especificamente~~ está em B.

Podemos esperar que esta similaridade esteja próxima de 1, para o que estabeleceremos um limiar T_{\min} não próximo de 1:

$$\text{regrA}(A, B) = \text{Sim}(A, B) > T_{\min, \text{sim}}$$

C. Nós queremos menor do que 1? AEB ocorre mesmo que $|B \setminus A| = 1$

b) espera-se que $\text{Sim}(A, B)$ ~~deceba~~ a 0 nem toda instância de B ~~especificamente~~ está em A. Podemos esperar o mínimo de sobreposição ou pelo menos similaridade de B $\rightarrow A$ para o que estabeleceremos um limiar α próximo de 0:

$$\text{regrB}(A, B) = \left(\frac{|A \cap B|}{|B|} \geq \alpha \right) \vee \left(\frac{|B \cap A|}{|A|} \geq \alpha \right)$$

Considerando as observações nos itens a e b ~~alteramos~~ entendo a Equação 12 para:

utilizada, não é necessário apresentar os anteriores. Numa discussão normalmente, não se conta a hipótese de um desenvolvimento, mas apresenta-se e explica-se o resultado $\wedge \text{regrA}(A, B) \wedge \text{regrB}(A, B)$ deste.

$$\text{menosGen}(A, B) = \left(\frac{|A \cap B|}{|A|} \geq T_{\min, \text{sobre}} \right)$$

C. Sei V quando todas as 3 condições forem ~~satisfatias~~ satisfeitas. É isso mesmo?

A sobreposição (1) ~~não~~ mais ~~mais~~ geral \rightarrow ocorre quando ainda existe alguma interseção entre os conceitos, mas nenhuma relação mais forte se aplica. Na prática, de
assumimos um limiar $T_{\min, \text{sobre}}$ perto de 0 que difere a sobreposição e diferença.

$\begin{cases} \text{não -} \\ \text{geral (1)} \\ \text{mais -} \\ \text{geral (2)} \end{cases}$

$$\text{sobrepos}(A, B) = \left(\frac{|A \cap B|}{|A|} \geq T_{\min, \text{sobre}} \right) \wedge \left(\frac{|A \cap B|}{|B|} \geq T_{\min, \text{difer}} \right)$$

A diferença (\neq) é o que resta quando as relações anteriores falharem. Temos agora como consultar qual a relação mais apropriada para um par de conceitos A e B , iniciando da relação mais forte até a mais fraca, como a seguir no método *consultaRelacao*:

```
00 consultaRelacao(A,B) : Relacao
01 INICIO
02 : SE      equivalente(A,B)  RETORNE ≡
03 : SENOAO SE maisGeral(A,B)  RETORNE ⊃
04 : SENOAO SE menosGeral(A,B) RETORNE ⊂
05 : SENOAO SE sobreposto(A,B) RETORNE ∩
06 : SENOAO                               RETORNE ≠
07 FIM
```

C. que?

1.3.2 COMPARAÇÃO TOP-DOWN

De posse do método *consultaRelacao*, o passo seguinte poderia ser comparar todos os conceitos entre si ao custo de uma complexidade quadrática. Contudo, essa abordagem não é interessante uma vez que, devido a erros oriundos da classificação das instâncias, observamos no decorrer dos experimentos algumas relações fortes sendo estabelecidas entre conceitos que, além de diferentes, estavam localizados em regiões taxonômicas também diferentes.

*C. Referência
para os expe-
rimentos*

Na tentativa de contornar os erros que naturalmente ocorrem durante a classificação, exploramos a estrutura taxonômica das ontologias para evitar erros grosseiros de mapeamento. Basicamente, a idéia é percorrer as taxonomias de cima para baixo comparando seus conceitos, ou em outras palavras, utilizar uma **estratégia dedutiva top-down** para mapear primeiramente conceitos mais genéricos, trançando o contexto taxonômico (conjunção de ancestrais) de conceitos mais específicos para só então mapeá-los usando o método *consultaRelacao*. Esta estratégia tem caráter dedutivo porque, como toda dedução, envolve inferência partindo de princípios gerais, isto é, do universal para o particular.

Intuitivamente, como conceitos similares contêm subconceitos também similares, a comparação *top-down* usa taxonomias como guias, evitando comparações desnecessárias e reduzindo o espaço de busca. A hierarquia taxonómica melhora a estimativa dos parâmetros de enunciado (Kesnerk, 1990), no nosso caso similaridade e sobreposição; devido à propagação *bottom-up*, apoiada na semântica de herança, classes genéricas levantam problemas por falta de instâncias diretas e possam ter mais instâncias do que classes específicas. Logo, parâmetros estimados próximos ao topo taxonómico são mais confiáveis. Espera-se então que esta estratégia repare estimativas menos confiáveis nas classes específicas, para evitar impactamentos estatisticamente fortes entre conceitos distintos de regiões taxonómicas distintas.

A comparação *top-down* é simples, recursiva e foi implementada basicamente como no método *compareTopDown* a seguir:

```
00 compareTopDown(A, B) : void
01 INICIO
02   : relacao = consultaRelacao(A, B)
03   : ESCOLHA(relacao)
04   : INICIO
05   :   : CASO == : compareTopDown(subconceitos[A], subconceitos[B]) PARE
06   :   : CASO != : compareTopDown(subconceitos[A], B) PARE
07   :   : CASO < , compareTopDown(A, subconceitos[B]) PARE
08   :   : CASO > : compareTopDown(A, subconceitos[B])
09   :   : compareTopDown(subconceitos[A], B) PARE
10   :   : CASO == : compareTopDown(A, subconceitos[B])
11   :   : compareTopDown(subconceitos[A], B) PARE
12   : FIM
13 FIM
```

Diferente de *compareTopDown(A, B)*, as chamadas recursivas nas linhas 5 a 11 recebem vetores. Mas isso é apenas uma conveniência didática para assumir implicitamente as alterações que chamam novamente *compareTopDown(A, B)*, segundo a ordenação decrescente da similaridade dos pares *(A, B)*, o que chamamos de alinhamento horizontal.

Os argumentos da função de recursão são diferentes

C Intitular

A recursão é diferente dependendo da relação estimada. Ao estimar $A \equiv B$, devemos comparar apenas seus subconceitos, evitando comparar um conceito com os subconceitos do outro. Ao estimar $A \sqsupseteq B$, devemos comparar B aos subconceitos de A , sendo desnecessário comparar A com os subconceitos de B . Analogamente, o mesmo é feito ao estimar $A \sqsubset B$. Ao estimar $A \sqcap B$, significa que pode haver uma sobreposição mais específica entre subconceitos, ou mesmo diretamente entre A ou B com subconceito um do outro. Finalmente, ao estimar $A \not\equiv B$, podemos encontrar alguma relação que, em classes mais específicas, passe nos testes de limiar durante a relaxamento das regras de compatibilidade.

O método *compareTopDown* apresentado é básico por motivo de clareza, por isso precisa de adaptações. A mais óvia é evitar que a recursão continue ao estimar $A \not\equiv B$ quando a sobreposição entre A e B for exatamente nula. Um alinhamento vertical ao estimar $A \equiv B$ é outra adaptação importante. O problema é que quando detectamos uma equivalência no topo das taxonomias, não sabemos se a equivalência realmente ocorre no topo ou mais abaixo, pois devemos considerar a possibilidade de escassez e má distribuição de instâncias concentradas em poucos conceitos mais específicos. Veja:

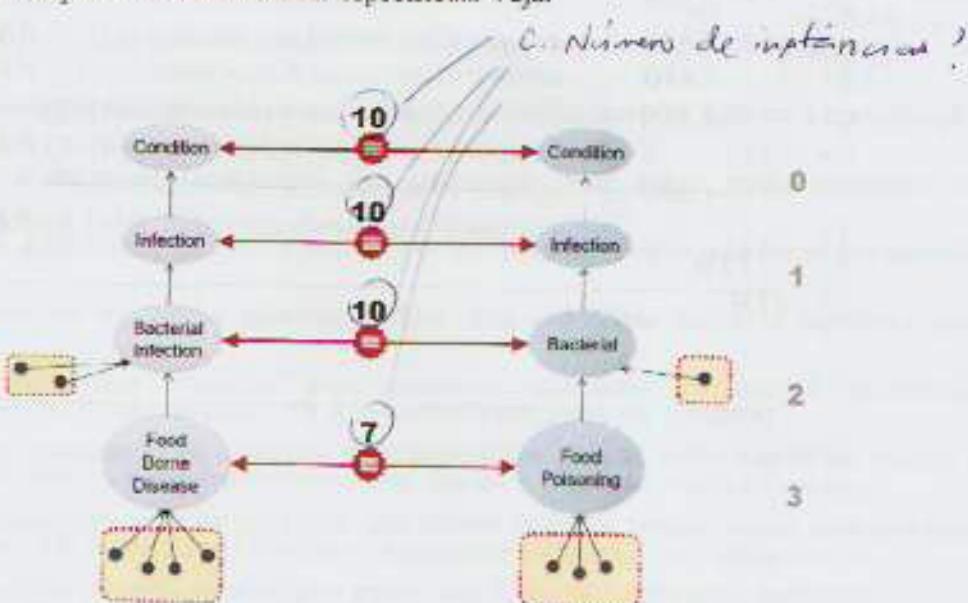


Figura 1-12: Má-distribuição de instâncias numa ramificação da taxonomia

No exemplo, todas as classes são equivalentes (o que nem sempre acontece). Contudo,

percebe que nos níveis 0 e 1 não existem instâncias senão aquelas criadas de Foodborne Disease e Food Poisoning.
BacterialInfection e Bacterial (11 instâncias no total); o ideal seria que Infection e Condition possuíssem outros subconceitos instanciados em mesma instâncias diretas. Portanto, apesar dos pares (Infection, Infection) e (Condition, Condition) serem equivalentes, a sobreposição estimada não é suficiente para concluir isso em casos como este em que faltam instâncias. Esta situação também pode ocorrer devido a uma regra ou restrição X (axioma de domínio) definida em um dos conceitos, tornando-o mais geral que o outro, apesar da enumeração de instâncias em ambos sugerir que sejam equivalentes.

Portanto, na dúvida se uma equivalência está no topo ou no fundo das taxonomias, o alinhamento vertical foi implementado dando preferência para o fundo, se o valor de sobreposição que indica a equivalência aumentar no próximo nível mais específico, a equivalência desce um nível na taxonomia, exceto se alguma outra condição se opuser a isso. Essa condição excepcional deve ser uma técnica palliativa não baseada nos valores de sobreposição sendo que, atualmente, o LMatch simplesmente compara os nomes dos conceitos, nomes em quais provavelmente serão parecidos, dado que a abordagem com os valores de sobreposição serviu só menos para indicar uma equivalência naquela ramificação da taxonomia.

C. O capítulo não deixa a impressão de um bom entendimento do trabalho. Há muita informação ausente/dida. O que lhe parececlaro pode não ser para quem não é autor do trabalho. Cabe que explanações mais explícitas favoreçam também o entendimento do todo do trabalho através de esquemas, diagramas, etc. de alto nível de abstração.

Fornecer informações que contextualizem as ontologias utilizadas nos exemplos.

MM
201