# DISSERTAÇÃO DE MESTRADO


**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

**CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA**

**PROGRAMA DE PÓS-GRADUAÇÃO EM**

**CIÊNCIA DA COMPUTAÇÃO**


## “EXPANSÃO DE ONTOLOGIA ATRAVÉS DE LEITURA DE MÁQUINA CONTÍNUA”


**ALUNO: Paulo Henrique Barchi**
**ORIENTADOR:** Prof. Dr. Estevam Rafael Hruschka Júnior


**São Carlos**
**Outubro/2014**


**CAIXA POSTAL 676**
**FONE/FAX: (16) 3351-8233**
**13565-905 - SÃO CARLOS - SP**
**BRASIL**

**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

# EXPANSÃO DE ONTOLOGIA ATRAVÉS DE LEITURA DE MÁQUINA CONTÍNUA

**PAULO HENRIQUE BARCHI**

**ORIENTADOR: PROF. DR. ESTEVAM RAFAEL HRUSCHKA JÚNIOR**

São Carlos – SP

Outubro/2014

**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

# EXPANSÃO DE ONTOLOGIA ATRAVÉS DE LEITURA DE MÁQUINA CONTÍNUA

**PAULO HENRIQUE BARCHI**

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação, área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Estevam Rafael Hruschka Júnior

São Carlos – SP

Outubro/2014

Ficha catalográfica elaborada pelo DePT da Biblioteca Comunitária UFSCar
Processamento Técnico
com os dados fornecidos pelo(a) autor(a)

Barchi, Paulo Henrique
B243e Expansão de ontologia através de leitura de máquina contínua / Paulo Henrique Barchi. -- São Carlos : UFSCar, 2015.
96 p.

Dissertação (Mestrado) -- Universidade Federal de São Carlos, 2015.

1. Extração de conhecimento. 2. Descoberta de conhecimento. 3. Extensão de ontologia. 4. Auto-supervisão. I. Título.

**UNIVERSIDADE FEDERAL DE SÃO CARLOS**

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

## Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Dissertação de Mestrado do candidato Paulo Henrique Barchi, realizada em 31/03/2015:

Prof. Dr. Estevam Rafael Hruschka Junior
UFSCar

Profa. Dra. Heloisa de Arruda Camargo
UFSCar

Prof. Dr. Carlos Manuel Milheiro de Oliveira Pinto Soares
UP

# AGRADECIMENTOS

Aos meus pais, Antonio Claudio Barchi e Maria Lucia Madeira, e irmãos, Claudia Maria Barchi e André Paulo Barchi, pelo constante apoio e zelo em minha vida.

Ao meu orientador Estevam Rafael Hruschka Júnior pelo entusiasmo, dedicação, apoio, paciência e competência em todo o decorrer do curso e projeto.

Aos meus amigos de república e demais que estiveram próximos neste tempo de curso pela companhia, motivação e companheirismo.

Aos amigos do laboratório Machine Learning Lab (MaLL) pela companhia e colaboração no trabalho, e também aos integrantes dos laboratórios CIG e LaLiC, pelos conselhos e discussões proveitosos.

Ao Programa de Pós-Graduação em Ciência da Computação (PPG-CC) do Departamento de Computação (DC) da Universidade Federal de São Carlos (UFSCar) pela oportunidade de realização deste trabalho e pelas instalações.

À *Carnegie Mellon University (CMU)*, especificamente à equipe do projeto *Read the Web*, responsável pelo *NELL*, e especialmente ao Bryan Kisiel por todas as explicações e ajudas acerca do projeto.

Ao órgão de fomento à pesquisa CNPq, que incentivou financeiramente este projeto.

A todos que contribuíram para a concretização deste trabalho, direta e indiretamente.

*“Ninguém ignora tudo. Ninguém sabe tudo. Todos nós sabemos alguma coisa. Todos nós ignoramos alguma coisa. Por isso aprendemos sempre.”*

PAULO FREIRE

# RESUMO

NELL (Never Ending Language Learning system) (CARLSON et al., 2010) é o primeiro sistema a praticar as técnicas do paradigma de Aprendizado Sem-Fim (ASF). Ele possui um subsistema componente inativo para continuamente expandir a Base de Conhecimento (BC): OntExt, que tem como ideia principal identificar e adicionar à BC novas relações que são frequentemente afirmadas em grandes bases de texto. Para isso, matrizes de co-ocorrência são utilizadas para estruturar os valores normalizados de co-ocorrência entre as frases verbais para cada par de categorias a fim de identificar padrões de contexto que interligam estas categorias. O agrupamento de cada uma destas matrizes é feito com o algoritmo K-médias do Weka: uma possível relação nova a partir de cada agrupamento. Este trabalho apresenta newOntExt: uma abordagem atualizada com novos recursos para tornar a extensão de ontologia uma tarefa mais palpável. Além desta metodologia tradicional, newOntExt pode validar e nomear relações encontradas pelo Prophet, outro subsistema componente do NELL. As relações geradas são classificadas por humanos como válidas ou inválidas; para cada experimento é calculada a precisão e os resultados são comparados aos de OntExt. Resultados iniciais mostram que a extensão de ontologia com newOntExt pode ajudar sistemas de ASF a expandir o volume de crenças e manter alta precisão ao atuar na auto-supervisão e auto-reflexão.

**Palavras-chave**: Extração de Conhecimento, Descoberta de conhecimento, Extensão de Ontologia, Auto-Supervisão

# ABSTRACT

NELL (Never Ending Language Learning system) (CARLSON et al., 2010) is the first system to practice the Never-Ending Machine Learning paradigm techniques. It has an inactive component to continually extend its KB: OntExt (MOHAMED; Hruschka Jr.; MITCHELL, 2011). Its main idea is to identify and add to the KB new relations which are frequently asserted in huge text data. Co-occurrence matrices are used to structure the normalized values of co-occurrence between the contexts for each category pair to identify those context patterns. The clustering of each matrix is done with Weka K-means algorithm (HALL et al., 2009): from each cluster, a new possible relation. This work present newOntExt: a new approach with new features to turn the ontology extension task feasible to NELL. This approach has also an alternative task of naming new relations found by another NELL component: Prophet. The relations are classified as valid or invalid by humans; the precision is calculated for each experiment and the results are compared to those relative to OntExt. Initial results show that ontology extension with newOntExt can help Never-Ending Learning systems to expand its volume of beliefs and to keep learning with high precision by acting in auto-supervision and auto-reflection.

**Keywords**: Knowledge Extraction, Knowledge Discovery, Ontology Extension, Auto-Supervision

# LISTA DE FIGURAS

# LISTA DE TABELAS

# SUMÁRIO

# Capítulo 1

# INTRODUÇÃO

## 1.1 Contextualização

Apesar de ser uma forma de expressão e registro antiga, textos continuam sendo um dos principais repositórios de conhecimento e entendimento humano. Com a evolução da tecnologia referente a compartilhamento de informação, os recursos disponíveis de informação textual vão além da capacidade humana de leitura. Este é um dos importantes fatores que motivam pesquisas no campo da Leitura de Máquina (LM), que têm como objetivo, basicamente, extrair e armazenar conhecimento a partir de fontes textuais (ETZIONI et al., 2011; SUCHANEK; WEIKUM, 2010).

A LM pode ser definida como um mecanismo de compreensão autônoma de texto por parte da máquina (BANKO; ETZIONI, 2007). A partir da reunião de, principalmente três áreas de interesse origina-se a LM. Estas áreas (as quais não são as únicas utilizadas como superte para LM) podem ser superficialmente apresentadas como:

- **Aprendizado de Máquina (AM)**: área de pesquisa que investiga algoritmos, métodos, abordagens e implementações que permitam que uma máquina possa ter a capacidade de aprender a partir de experiências anteriores (MITCHELL, 1997);
- **Processamento de Língua Natural (PLN)**: pode ser definida como uma área de pesquisa que investiga formas para que máquinas possam processar linguagem humana (BETTUZZI, 2009); e,
- **Extração de Informação (EI)**: área de pesquisa que investiga modelos, métodos e algoritmos projetados para coletar informação de uma fonte (normalmente não estruturada) para uma estrutura de armazenamento de conhecimento (como uma ontologia, por exem-

plo).

Assim, surge a LM com o objetivo de investigar o problema de acumular e sistematizar conhecimento disponível textualmente e representá-lo de uma forma estruturada e de fácil acesso para as máquinas.

A representação em uma forma estruturada dos conhecimentos extraídos a partir de grandes volumes de textos (informação não estruturada) pode ser vista como uma importante fonte de informação para sistemas de aprendizado, como por exemplo, o *NELL — Never-Ending Language Learning*, do projeto Read The Web (CARLSON et al., 2010) — descrito brevemente a seguir, e abordado com maiores detalhes na Subseção 3.4.1.

Nos últimos anos, a pesquisa em Aprendizado de Máquina está se expandindo e novas abordagens têm sido propostas. O projeto *Read The Web* visa construir um sistema de aprendizado de máquina com base em uma destas novas abordagens, o aprendizado sem-fim. No aprendizado sem-fim um sistema inteligente melhora constantemente sua habilidade de aprender, podendo por exemplo, converter dados não-estruturados em informação relacional estruturada de maneira contínua, autônoma e incremental. Este sistema do projeto *Read The Web* é chamado *NELL* (*Never-Ending Language Learning*): o primeiro sistema computacional descrito na literatura a implementar os princípios do aprendizado sem-fim (CARLSON et al., 2010). O NELL utiliza sua própria capacidade de aprendizado, bem como sua Base de Conhecimento (BC) continuamente crescente, para aprender melhor a cada dia. Os dados de entrada necessários para este processo são uma ontologia inicial. A ontologia inicial é formada por um conjunto de categorias (como "Pessoa" e "Cidade"), exemplos de instâncias dessas categorias (como "Obama" para a categoria "Pessoa", e, "Pittsburgh" e "Washington" para "Cidade"), e, exemplos de relações entre as instâncias de categorias (por exemplo, "Obama reside em Washington" e "Obama frequenta Pittsburgh"). O NELL obtém vantagem da combinação de diversas estratégias e algoritmos para continuamente induzir novo conhecimento a partir de milhões de páginas da Web. Para ser capaz de continuar aprendendo continuamente, este sistema conta com duas importantes propriedades: auto-supervisão e auto-reflexão. Fatos relacionais de baixa confiabilidade envolvem grande incerteza para o aprendizado. Já fatos relacionais de alta confiabilidade (denominados crenças) são a base para expansão do aprendizado, pois atuam na supervisão da BC e, assim, também na reflexão. Assim, o que foi aprendido corretamente pode ser utilizado para refletir o conhecimento em outras áreas de interesse, além de supervisionar novos aprendizados. Dessa forma, o sistema atua em sua própria supervisão e reflexão (auto-supervisão e auto-reflexão). Além disso, o NELL também faz uso de redes sociais (como *Twitter* e *Yahoo-Answers!*) e uma supervisão humana superficial para assegurar que está livre de ruídos graves,

a fim de evitar desvios de conceito.

Para realizar esta tarefa de aprendizado contínuo, o *NELL* tem diferentes componentes. Dentre estes, especificamente relacionados à tarefa de expansão da Base de Conhecimento (BC), existem dois (abordados com maiores detalhes na Subseção 3.4.1). O *Prophet* trabalha com as informações relacionais na forma de grafo (que representa a BC do *NELL*), prevê novas relações entre nós deste grafo (que representam instâncias de categorias já conhecidas), induz regras de inferência e identifica ligações (relações) incorretas entre os nós (fatos errados) a partir da mineração do grafo (APPEL; Hruschka Jr, 2011). O *OntExt* usa informação redundante da Web para aprender novas relações entre instâncias de categorias já conhecidas: *OntExt* procura por padrões de contexto desconhecidos pelo *NELL* que são semanticamente similares e frequentemente afirmados em um enorme volume de texto e considera-os (juntamente com o sujeito e o objeto da frase) como possíveis novas crenças para realizar a expansão de ontologia. A BC existente do *NELL* é a fonte de exemplos rotulados como categorias e instâncias de categorias já conhecidas.

Mesmo com a existência do *Prophet* e do *OntoExt*, há ainda carência no NELL de um processo mais efetivo de extensão da ontologia. Por isso, o **objetivo** deste trabalho é colaborar com o aprendizado do sistema *NELL* do projeto *Read The Web*, com a evolução do componente *OntExt* a partir de uma nova abordagem e uma nova implementação: ***newOntExt (nOE)***. Além de técnicas de Leitura de Máquina mais recentes do estado-da-arte e do próprio sistema *NELL* estar com a Base de Conhecimento (BC) maior e mais confiável, *nOE* conta com soluções projetadas para melhor desempenho em relação aos desafios de Leitura da Web em grande escala (melhores descritos no decorrer deste documento). Assim, este trabalho visa tornar praticável a expansão de ontologia contínua a partir da metodologia introduzida na subseção 1.3, e detalhada na Seção 4. O *nOE* também pode ser útil no auxílio à auto-supervisão do sistema quanto às crenças de instâncias de categorias (o Capítulo 5 apresenta exemplos e maiores explicações sobre essa atuação). Além desta tarefa tradicional que antes era realizada pelo *OntExt*, este novo componente contribui com o aprendizado do *NELL* também a partir de uma colaboração com o componente *Prophet* (maiores informações na seção 4.3).

## 1.2 Formalização do Problema

Com base na definição do problema dada em (WIJAYA; TALUKDAR; MITCHELL, 2013), a terminologia utilizada e a formalização do problema considerado neste trabalho de mestrado são apresentadas nesta seção. Para o desenvolvimento da metodologia do newOntExt, uma Base de

Conhecimento (BC) $B$ é definida como uma 4-tupla ($C$, $I_C$, $R$ e $I_R$), onde $C$ é um conjunto de categorias (por exemplo, um conjunto formado por categorias relacionadas a esportes: atleta, esporte, time, campeonato e liga), $I_C$ é o conjunto de pares instâncias-categorias (por exemplo, *(Neymar, atleta)*) para categorias em $C$, $R$ é o conjunto de relações (neste contexto dos esportes, *atleta_joga_em_time* é um exemplo de relação), e $I_R$ é o conjunto de triplas instância-relação-instância para relações presentes em $R$ (por exemplo, *(Neymar, atleta_joga_em_time, Barcelona)*).

Em (WIJAYA; TALUKDAR; MITCHELL, 2013), uma Base de Conhecimento é definida como uma 6-tupla (além de $C$, $I_C$, $R$ e $I_R$, considera também $O_C$ e $O_R$), onde $O_C$ é a ontologia de categorias que especifica relações hierárquicas de subconjunto/superconjunto entre as categorias (por exemplo, *atleta* pode ser um subconjunto de *pessoa*); e $O_R$ é a ontologia de relações que especifica a hierarquia de relações (por exemplo, *capitão_de(atleta, time)* é um caso especial da relação *joga_em(atleta, time)*). Tanto $O_C$ quanto $O_R$ podem ser nulos, isto é, a BC pode ter uma estrutura plana de categorias e relações. A estrutura utilizada neste trabalho é plana, por isso estas especificações hierárquicas foram desconsideradas.

Cada instância de uma relação $r \in R$ é uma 3-tupla $(e_1, r, e_2) \in I_R$, onde $(e_1, c_1) \in I_C$, e $(e_2, c_2) \in I_C$ para categorias $c_1$ e $c_2 \in C$. Cada instância de categoria pode ser referenciada por um ou mais Sintagmas Nominais (SN). Por exemplo, a instância *Neymar* pode ser referenciada tanto com o próprio SN *Neymar* quanto com o SN *Neymar Júnior*. A partir deste raciocínio, *N(i)* é definido como o conjunto de SNs correspondentes à instância de conhecimento $i$.

Além da BC, outro recurso de entrada necessário para a metodologia é um conjunto amplo de triplas no formato Sujeito-Verbo-Objeto (SVO) extraídas de *corpus* de textos de língua natural. (Por enquanto foram utilizados recursos já neste formato de conjunto de triplas. Caso um *corpus* de textos de língua natural seja o foco de um experimento, as extrações serão realizadas com sistemas de extração de informação aberta de estado-da-arte, ReVerb e R2A2 - este pré- processamento é melhor descrito na subseção 4.1.1). Seja $D$ este recurso com um grande conjunto de triplas no formato Sujeito-Verbo-Objeto (SVO) com tuplas no formato ($sn_1$, $v$, $sn_2$, $f$), onde $sn_1$ e $sn_2$ são os sintagmas nominais (SN) que correspondem ao sujeito e objeto da frase, respectivamente, $v$ é um verbo (ou uma frase verbal), e $f \in \mathbb{R}_+$ é a contagem/frequência normalizada desta tupla em um grande *corpus* de texto.

Para cada tripla de $D$, verifica-se se $sn_1$ e $sn_2$ constam na BC $B$, isto é, se $\{\exists\ ((e_1,\ c_1),\ (e_2,\ c_2) \in I_C;\ c_1,\ c_2 \in C) \mid e_1 \equiv sn_1;\ e_2 \equiv sn_2)\}$. Para os casos positivos, esta tripla ($sn_1$, $v$, $sn_2$, $f$) é armazenada, para, no próximo passo, ser considerada para a construção das matrizes de co-ocorrência.

Para cada par de categorias $(c_1, c_2) \in C$ é montada uma matriz de co-ocorrência. Seja $n_{v(c_1, c_2)}$ o número de verbos (ou frases verbais) com o qual instâncias de $c_1$ e $c_2$ co-ocorrem. A matriz de co-ocorrência para estas categorias tem dimensões $[n_{v(c_1, c_2)}][n_{v(c_1, c_2)}]$, isto é, mesma quantidade de linhas e colunas (uma de cada para cada verbo, ou frase verbal). Os elementos desta matriz são preenchidos com os valores normalizados de co- ocorrência. A normalização é feita a partir do maior número de co-ocorrências: todos os valores são divididos por este maior número.

O agrupamento *K-médias* é aplicado para todas as matrizes de co-ocorrências construídas, a fim de obter agrupamentos que representem as novas relações. Para cada matriz de co-ocorrência, os valores são agrupados em $k$ agrupamentos $(k \in \mathbb{N}^+)$. Isto implica que para cada par de categorias $(c_1, c_2) \in C$ é possível gerar $k$ relações.

Sendo $k$ o número de agrupamentos por matriz de co-ocorrência, considere $i \in \mathbb{N} \mid 0 \leq i \leq k$. Para cada agrupamento, o verbo (ou frase verbal) $v_{i(c_1, c_2)}$ mais próximo do centróide é considerado o melhor candidato para a criação da relação. Esta nova relação é dada como a relação entre as categorias $c_1$ e $c_2$ por meio do sentido do verbo $v_{i(c_1, c_2)}$, o que pode ser representado como a tripla $(c_1, v_{i(c_1, c_2)}, c_2)$.

Esta formalização apresentada até aqui são esforços para a expansão da ontologia por uma iteração. O sistema *NELL* viabiliza o Aprendizado Sem Fim (ASF) por iterações: dá continuidade ao processo de aprendizagem uma iteração por vez — os componentes atuam a cada iteração (maiores detalhes na subseção 3.4.1). Sendo assim, quando o newOntExt estiver incorporado ao *NELL*, este processo se repetirá a cada iteração do sistema, utilizando a BC mais recente possível.

**Definição do Problema**: Dada uma Base de Conhecimento (BC) $B$ ($C_1$, $I_{C_1}$, $R_1$ e $I_{R_1}$) e um *corpus* de texto $D$ com um conjunto de tuplas no formato SVO, a tarefa é buscar novas relações em $D$ ainda inexistentes em $B$ com instâncias de categorias já presentes em $B$. A tarefa pode ser resumida em encontrar novas relações para conceitos (instâncias de categorias) já conhecidos.

## 1.3 Metodologia de Trabalho

Para realizar esta tarefa, será feito uso de sistemas estado-da-arte de Extração de Informação Aberta (EIA — maiores detalhes na seção 2.3): *ReVerb* (FADER; SODERLAND; ETZIONI, 2011) e sua evolução *R2A2* (ETZIONI et al., 2011). A partir do corpus fonte do qual se deseja obter conhecimento, é feita uma adaptação para que o arquivo fique com o formato correto para o processamento de *ReVerb/R2A2*. Feito este pré-processamento, tem-se a saída com a infor-

mação relacional extraída do *corpus* inicial. Então, para cada combinação de instâncias das categorias da base de conhecimento do *NELL*, informações relacionais nas quais estas instâncias coincidem são procuradas na saída do processamento da etapa anterior, a fim de encontrar padrões relacionais (expressos por frases verbais) entre categorias.

A abordagem para a identificação das relações tem como base algoritmos de agrupamento aplicados a matrizes de co-ocorrência [*frases verbais X frases verbais*], como no componente *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011) — técnica descrita em 3.4.1.2. Com este agrupamento feito por padrões de contexto (frases verbais), pode-se estudar a possibilidade de inferir novos fatos contextuais sobre instâncias das categorias envolvidas.

As novas relações geradas são classificadas e avaliadas seguindo alguns critérios especificados na Seção 4.1.3. Com isso, os resultados são comparados aos obtidos por OntExt, através da contagem de relações válidas, inválidas, e da precisão dos experimentos.

Em trabalhos futuros, a expansão de conhecimento será contínua pois será utilizada a abordagem do ASF: para cada nova iteração, o sistema *NELL* possui uma base de conhecimento maior e mais confiável comparada à iteração anterior; então, para o mesmo *corpus* de entrada, há maior probabilidade da metodologia proposta obter novos resultados. Também serão utilizados diferentes *corpora* em experimentos futuros.

## 1.4 Organização do Trabalho

A seguir, no Capítulo 2, o trabalho é fundamentado teoricamente: a supervisão no aprendizado de máquina, o Aprendizado Sem Fim (ASF), e as áreas e técnicas que envolvem a LM são abordados. O Capítulo 3 referencia a Leitura de Web, apresenta alguns projetos (e, respectivos sistemas e subsistemas componentes) que fazem uso desta para diferentes finalidades. Posteriormente (no Capítulo 4), a abordagem proposta de leitura da Web para expansão da BC ontológica do *NELL* é apresentada. O Capítulo 5 apresenta os experimentos conduzidos com o newOntExt e as respectivas análises dos resultados. Finalmente, o Capítulo 6 aborda a continuidade deste trabalho e as conclusões do mesmo até então.

# Capítulo 2

## FUNDAMENTAÇÃO TEÓRICA

Para realizar a expansão de Base de Conhecimento (BC) proposta neste trabalho de mestrado é feito uso de técnicas de Extração de Informação Relacional (EIR). Estas técnicas se originam de três áreas de interesse que formam a base da LM. Estas quatro áreas são abordadas nas outras Seções que seguem, bem como as técnicas utilizadas para realizar a expansão de ontologia de fato.

## 2.1 Aprendizado de Máquina

O campo de Aprendizado de Máquina (AM) é governado pela questão central: "Como podemos construir computadores que podem melhorar seu desempenho automaticamente com experiência? E quais são as leis fundamentais que governam todo o processo de aprendizagem?". AM é originado naturalmente da intersecção de Ciência da Computação e Estatística. Pode-se dizer que a pergunta que define Ciência da Computação é "Como podemos construir máquinas que solucionam problemas, e quais problemas são inerentemente tratáveis/intratáveis?". A questão que define amplamante Estatística é "O que pode ser inferido a partir de dados somados a um conjunto de pressupostos modelos, com qual confiabilidade?". A questão de AM é construída com base em ambos, mas é uma questão distinta. Enquanto Ciência da Computação tem foco primário em como construir programas de computador manualmente, AM foca em como fazer computadores programarem a si mesmos (a partir de experiência e uma estrutura inicial). Enquanto Estatística tem foco primário em que conclusões podem ser inferidas a partir de dados, AM incorpora outras questões sobre quais arquiteturas computacionais e algoritmos podem ser usados para capturar, armazenar, indexar, recuperar e mesclar esses dados da forma mais efetiva, como múltiplas subtarefas de aprendizado podem ser governadas em um sistema grande (MITCHELL, 2006).

Segundo Mitchell (1997), uma máquina aprende com respeito a uma tarefa específica *T*, uma métrica de performance *P* e experiência *E*. A evolução deste aprendizado é notada se o sistema melhora confiavelmente sua performance *P* na tarefa *T*, seguindo a experiência E (MITCHELL, 1997)

Um exemplo de abordagem de AM é o estudo do problema de uma máquina ler textos:

- **Tarefa *T*:** ler e compreender textos (especificamente em inglês, por exemplo) utilizando técnicas de EIR;
- **Medida de performance *P*:** tempo de processamento em relação ao tamanho do texto de entrada, cobertura do conjunto de dados e precisão;
- **Experiência de treinamento *E***: instâncias de argumentos, relações e tuplas no formato (arg1, relação, arg2).

### 2.1.1 Supervisão no Aprendizado

O AM pode ser classificado em três tipos quanto a supervisão: Supervisionado, Não Supervisionado e Semissupervisionado (originado da reunião dos dois anteriores). Esta subseção aborda esta divisão do AM, contextualiza a supervisão abordada neste trabalho, e, assim, introduz o ASF, que é o assunto da próxima Subseção (2.1.2).

O Aprendizado Supervisionado é a forma de aprendizagem em que o processo é guiado por alguma forma de supervisão. Esta supervisão pode estar vinculada, por exemplo, a exemplos previamente rotulados; a partir destes, padrões podem ser identificados para classificar ou agrupar novos exemplos ainda não rotulados.

O pensamento *Bayesiano* fornece uma abordagem probabilística para aprendizagem: decisões ótimas podem ser tomadas com base em probabilidades conjuntamente com os dados e eventos observados, assim, fornece uma solução quantitativa ponderando a evidência e suportando hipóteses alternativas. Cada exemplo de treinamento pode decrementar ou incrementar a probabilidade de uma hipótese ser correta; conhecimento pode ser combinado com os dados observados para determinar a probabilidade de uma hipótese; métodos *Bayesianos* podem acomodar hipóteses que fazem predições probabilísticas (um exemplo fictício acerca da tarefa de aprendizado para ilustrar: a partir dos exemplos de instâncias de cidades — ou crenças a respeito de cidades — e dos fatos conhecidos acerca de Siena, Siena tem 95% de probabilidade de ser uma cidade); assim, novas instâncias podem ser classificadas combinando a probabilidade de múltiplas hipóteses ponderadas pelas suas probabilidades. O classificador *Naive Bayes* é

um método de aprendizado Bayesiano que pode ser aplicado em tarefas de Aprendizado Supervisionado. Um conjunto de exemplos de treinamento da função alvo é fornecido, e uma nova instância é apresentada, descrita pela tupla de valores de atributos considerados no problema $< a_1, a_2, ... a_n >$. O aprendiz deve prever o valor alvo, ou classificação, para esta nova instância (MITCHELL, 1997).

Como pode-se inferir a partir do nome, o Aprendizado Não Supervisionado se difere do Supervisionado quanto a supervisão, isto é, não há supervisão. Uma das tarefas mais comuns no aprendizado não supervisionado baseia-se em formar grupos dos exemplos não-rotulados de acordo com suas similaridades (ou dissimilaridades), por exemplo.

O algoritmo Maximização de Expectativa (ME) é uma abordagem amplamente utilizada para aprender na presença de variáveis não observáveis, isto é, no Aprendizado Não Supervisionado. Este algoritmo pode ser usado até para variáveis cujos valores nunca são diretamente observáveis, dado que a forma geral da distribuição de probabilidade que governa estas variáveis é conhecida. O algoritmo tem início com uma hipótese inicial arbitrária. Então, ele calcula repetidamente os valores esperados para as variáveis escondidas (assumindo que a hipótese atual é correta), e então recalcula a hipótese de máxima probabilidade (assumindo que as variáveis escondidas têm os valores esperados calculados no passo anterior). Esse procedimento converge para uma hipótese de máxima probabilidade local, juntamente com valores estimados para as variáveis escondidas (MITCHELL, 1997).

O aprendizado Semissupervisionado tem influências das duas abordagens anteriormente descritas, Supervisionada e Não Supervisionada, pois o processo faz uso de supervisão, mas, há também etapas nas quais há o aprendizado sem supervisão. Para o exemplo utilizado acima, no qual a supervisão se dá por meio de exemplos rotulados, no aprendizado semissupervisionado são utilizados exemplos previamente rotulados e a partir deles são classificados/agrupados novos exemplos sem rótulos. Este aprendizado é bastante indicado para a tarefa de classificar/agrupar grandes amostras de dados não rotuladas a partir de uma pequena amostra de dados rotulados (DUARTE, 2011). Aprendizado Semissupervisionado utiliza grande quantidade de dados não rotulados e dados rotulados, para construir melhores classificadores. Esta forma de aprendizado necessita de menos envolvimento humano e oferece maior precisão, e, assim, é de grande interesse tanto na teoria quanto na prática (ZHU, 2008).

Neste trabalho, o aprendizado tem uma abordagem Semissupervisionada: a BC existente de *NELL* fornece exemplos rotulados, relações, instâncias e categorias já conhecidas, com a finalidade de se obter novas relações que não existiam antes, o que aumenta a BC e a quantidade de exemplos rotulados. Desta maneira, esta nova BC com novas informações relacionais conhe-

cidas é uma entrada mais interessante que a antiga, usada na primeira iteração. Isso incita que o procedimento de aprendizado deveria ocorrer novamente, com os exemplos rotulados desta nova BC. A próxima subseção aborda a formalização deste aprendizado contínuo e iterativo — ASF.

### 2.1.2 Aprendizado Sem Fim

Um sistema é praticante das técnicas de Aprendizado Sem Fim (ASF) quando evolui continuamente, dia após dia. Um processo de ASF é incremental e contínuo: faz uso de conhecimento aprendido previamente para, continuamente, refinar sua capacidade de aprendizado ao longo do tempo (DUARTE, 2011; Hruschka Jr; DUARTE; NICOLETTI, 2013). É inspirado no aprendizado humano e tem como objetivo aprender algo que ajude em um aprendizado futuro.

A evolução do processo de aprendizado pode ser validada a partir de experimentos, ao comparar o estado do sistema em uma iteração com o correspondente de iterações anteriores. Assim, este método não espera ser melhor que métodos tradicionais na primeira iteração. A alta cobertura não é o foco inicial; importa mais a relevância e confiança do que foi aprendido, visto que a base de conhecimento expande-se continuamente a partir do que se aprende (relações inválidas/incorretas devem ser identificadas e removidas do processo para não propagarem o erro pela BC). Desta forma, espera-se que o desempenho do sistema melhore continuamente.

A longo prazo, um sistema com abordagem de ASF deve possuir uma BC completa e com alta precisão e, com isso, seu aprendizado deve estar otimizado com a experiência em aprender relações válidas e confiáveis e identificar informações relacionais inválidas ou incorretas. O sistema de ASF *NELL* do projeto *Read The Web* é o primeiro a colocar em prática as ideias deste novo paradigma. Maiores detalhes acerca do projeto e do sistema podem ser vistos na Seção 3.4 e na subseção 3.4.1, respectivamente.

## 2.2 Processamento de Língua Natural

O objetivo do campo de Processamento de Linguagem Natural (PLN) é fazer com que computadores realizem tarefas envolvendo a linguagem humana, como tornar viável comunicação humano-computador ou simplesmente fazer processamento útil de texto ou fala (BETTUZZI, 2009).

Em outras palavras, PLN é o nome que se dá à área de pesquisa que se dedica a investigar, propor e desenvolver formalismos, modelos, técnicas, métodos e sistemas computacionais que

têm a língua natural como objeto primário. De modo geral, em PLN buscam-se soluções para problemas computacionais, ou seja, tarefas, sistemas, aplicações ou programas, que requerem o tratamento computacional de uma língua natural, como português ou inglês, por exemplo (NUNES, 2008).

Nunes (2008) apresenta alguns exemplos de tarefas básicas de PLN:

- Pré-processamento de textos: subdividir o texto em unidades fonéticas, lexicais, gramaticais, semânticas ou discursivas, de acordo com o objetivo da tarefa em questão;
- Classificar (Etiquetar) automaticamente as unidades do texto, segundo classes pertinentes à tarefa: morfossintáticas (etiquetador *Part-of-Speech - POS*), sintáticas (*Parser*), semânticas (*Parser* Semântico ou Interpretador), discursivas (*Parser* discursivo). Em cada caso, é necessário definir linguagens de anotação, usadas para representar as classes: etiquetas, relações, estruturas (p.ex. árvore sintática).
- Mapear representações: da LN para uma representação sintática, semântica ou discursiva; e dessas para LN — Interpretação e Geração de LN.

Além disso, PLN apresenta métodos próprios para a tarefa de extração (ou identificação) de informação relacional. Taba e Caseli (2012) fazem uso de dados léxico-sintáticos para realizar a extração de relações a partir de um *corpus*, com relações pré-especificadas. Este viés linguístico é muito importante para esta tarefa de extrair e estruturar fatos relacionais de texto não estruturado. Mas, além de técnicas de PLN, nesta proposta de trabalho de mestrado será feito uso de uma abordagem de AM Semissupervisionada (com extensão para ASF, como abordado nas Seções anteriores) e também de técnicas de EI (com foco em padrões textuais), área de estudo abordada na próxima Seção (2.3).

## 2.3 Extração de Informação

A Extração de Informação (EI), também chamada de Recuperação de Informação (RI), tem como principal tarefa extrair dados factuais e manipuláveis para a máquina a partir de uma fonte textual, e armazená-los em uma estrutura de representação de conhecimento. Tradicionalmente tem dependido do envolvimento humano extensivo na forma de regras de extração artesanal ou rotulação manual de exemplos de treinamento. Além disso, as relações de interesse deveriam ser especificadas previamente. Enquanto EI tem-se tornado cada vez mais automatizada com o tempo, a enumeração de todas as potenciais relações de interesse para extração por um sis-

tema de EI é altamente problemática para corpora tão grande e variado quanto a Web (BANKO; CAFARELLA; SODERLAND, 2009; BANKO; ETZIONI, 2008).

Com a intenção de solucionar este problema, EI Aberta (EIA) propõe a identificação de frases relações (frases que denotam relações em sentenças em inglês (BANKO; ETZIONI, 2007)). A identificação automática de frases relações possibilita a extração de relações arbitrárias de sentenças, evitando a restrição de um vocabulário pré-especificado.

Sistemas de EIA evitam substantivos e verbos específicos. Os extratores são não-lexicáveis (formulados apenas em termos de *tokens*[1]) sintáticos e classes de palavras fechadas. Desse modo, extratores de EIA focam em modos genéricos (padrões) nos quais relações são expressas (naturalmente generalizando através dos domínios). Estes sistemas fazem um único (ou um número constante de) passo(s) sobre um *corpus* e extraem um grande número de tuplas relacionais *($arg_1$, relação, $arg_2$)* sem precisar de qualquer dado de treinamento de relação específica (não supervisionado). O principal ponto positivo de sistemas de EIA está no processamento eficiente assim como na habilidade de extrair um número ilimitado de relações (ETZIONI et al., 2011).

## 2.4 Leitura de Máquina

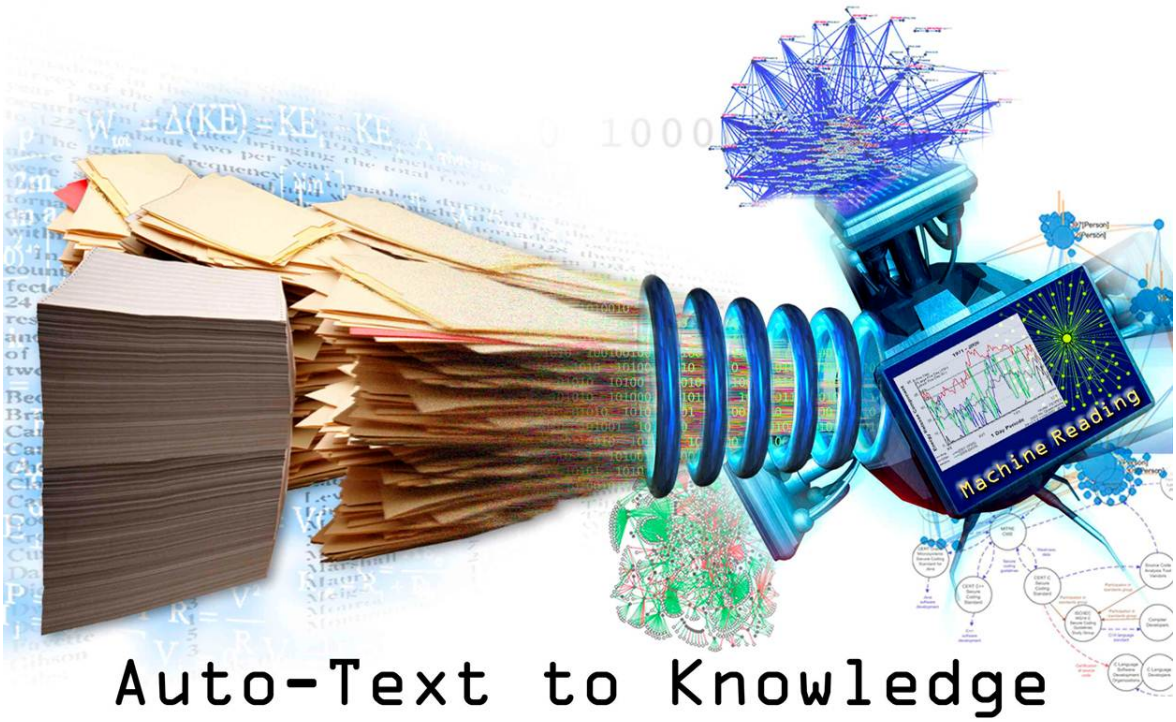


**Figura 2.1: Leitura de Máquina**

[1] Unidades de texto que fazem sentido.

A LM investiga o desafio da interpretação textual, de entender o texto escrito (NORVIG, 2007). Em resumo, LM visa estruturar informação textual para o aprendizado. Como introduzido no Capítulo 1, LM envolve técnicas de PLN, EI e AM (Hruschka Jr, 2012): técnicas de EI a partir de fontes textuais podem ser combinadas com técnicas de etiquetação morfossintática (ou outra forma de etiquetação mais conveniente de PLN), em um ambiente de AM, para aprender cada vez melhor a partir das experiências, como exemplificado na Seção 2.1.

Uma abordagem importante para LM é extrair fatos de texto e armazená-los em uma forma estruturada. Neste contexto, fatos podem ser vistos como instâncias de categorias de conhecimento ligadas por contextos relacionais.

A tarefa de LM pode ser subdivida como:

- Extração de instâncias de categorias de conhecimento: identificar os argumentos (sujeito e objeto) referenciados na frase relacional, a partir das instâncias de categorias de conhecimento e fatos já conhecidos.
- Extração de Relação: identificar a relação que interliga estes dois argumentos.
- Armazenamento dos fatos extraidos de forma estruturada: um dos modelos de dados mais comuns para representar os fatos extraídos é a ontologia. Uma ontologia constitui-se basicamente de instâncias de categorias de conhecimento (representados por nós) e relações que interligam estas instâncias. Esta é a forma utilizada para armazenar a base de conhecimento de *NELL*, e, desta forma, é a representação tratada neste trabalho.

Para exemplificar, considere a frase

*"O time A joga no campeonato B".*

Identificam-se inicialmente as entidades *"time A"* e *"campeonato B"*. Na sequência, identifica-se uma relação *"jogaEm"* entre as entidades previamente identificadas *"time A"* e *"campeonato B"* (também chamados de argumentos), que pode ser representada como a tupla

*"(time A, jogaEm, campeonato B)".*

Desta forma, a informação previamente na forma textual pode ser armazenada em uma representação ontológica, na qual sistemas computacionais podem aplicar métodos para manipular estes dados e adquirir conhecimento, isto é, expandir a ontologia.

## 2.5 Expansão de Ontologia

### 2.5.1 Métodos não supervisionados para expandir a ontologia

Com a finalidade de expandir uma ontologia que representa uma base de conhecimento, Hasegawa, Sekine e Grishman (2004) propõem uma abordagem de agrupamento não supervisionado. Um vetor de características é construído para cada par de instâncias de categorias co-ocorrentes baseado nas palavras de contexto com as quais estas instâncias de categorias co-ocorrem. Uma métrica de similaridade de cosseno é aplicada a cada par de vetores de características para gerar uma matriz de co-ocorrência *[par de instância de categoria X par de instância de categoria]*. Então, essa matriz é agrupada e cada agrupamento de pares de instâncias de categorias corresponde a um predicado relacional.

O trabalho de Zhang et al. (2005) gera uma árvore de análise superficial para cada sentença contendo um par de instâncias de categorias para gerar as instâncias de relação. Uma árvore de métrica de similaridade e utilizada para agrupar as instâncias de relação. Este método utiliza um etiquetador de instâncias de categorias especializado que identifica instâncias de categorias específicas predifinidas.

Ambos estes métodos (HASEGAWA; SEKINE; GRISHMAN, 2004; ZHANG et al., 2005) agrupam pares de instâncias de categorias primeiramente baseado na similaridade léxica das palavras de contexto conectando as instâncias de categorias. Desta forma, pares de instâncias de categorias conectados por padrões de contexto[2] diferentes lexicamente mas semanticamente similares (por exemplo, "rio 'no centro da' cidade" e "rio 'passa no meio da' cidade") provavelmente não seriam agrupados juntos. Os dados da Web são muito ruidosos e têm um número de categorias de conhecimento muito grande, assim, outra questão importante é que para tratar dados em escala Web, matrizes de similaridade de *[pares de instância de categorias X pares de instância de categorias]* são provavelmente não escaláveis para muitos milhares de pares de instâncias de categorias.

Para superar estes desafios característicos de dados de escala Web, o OntExt realiza o agrupamento de matrizes *[padrões de contexto X padrões de contexto]*. Isso torna a tarefa muito mais escalável já que os padrões de contexto são muito menos numerosos quando comparados aos pares de instâncias de categorias. Além deste foco diferenciado, o método proposto por Mohamed, Hruschka Jr. e Mitchell (2011) aplica diversos critérios para podar padrões irrele-

[2] O que é referenciado aqui como "padrão de contexto", termo adotado pelos autores do OntExt — Mohamed, Hruschka Jr. e Mitchell (2011), pode ser entendido como "frase verbal", termo utilizado na Seção 1.2 - Formalização do Problema.

vantes.

Esta formalização para expansão de ontologia com foco nas co-ocorrências de sujeitos e objetos (argumentos), com matrizes estruturadas pelos padrões de contexto (ou frases verbais) é a base para a abordagem de expansão de ontologia proposta neste trabalho. Maiores detalhes técnicos acerca da metodologia do OntExt e da nova implementação de newOntExt são descritos na Subsubseção 3.4.1.2 e no Capítulo 4, respectivamente.

# Capítulo 3

## LEITURA DA WEB — TRABALHOS RELACIONADOS

A tarefa da Leitura de Máquina de extrair fatos relacionais naturalmente se estendeu para a Web, pela abundante quantidade de informação textual disponível. Para contextualização deste trabalho no meio científico, projetos que visam atingir diferentes objetivos por meio de abordagens de Leitura da Web variadas e relevantes são introduzidos a seguir:

- *YAGO-NAGA*: utiliza recuperação de informação textual semi-estruturada, por meio de padrões e *templates*, com o objetivo de ter uma base de conhecimento da Web;
- *Google Research a Relation Extraction Corpus*: disponibiliza *corpus* com extrações relacionais limitado a duas relações, julgado por humanos, para colaborar com o treinamento e/ou avaliação de sistemas de Leitura de Máquina;
- *KnowItAll*: com técnicas de Extração de Informação Aberta, este projeto é um dos principais envolvidos com Leitura da Web e sugere que este é o foco do novo paradigma da busca Web;
- *Read the Web*: com a combinação de diferentes componentes, alguns focados plenamente na Leitura de Web, outros com estratégias fora do escopo da Leitura de Máquina aplicada a Web, *Read The Web* tem como objetivo a viabilização do aprendizado sem fim.

Estes projetos e os sistemas envolvidos nas tarefas de Leitura da Web são apresentados nas seções e subseções a seguir.

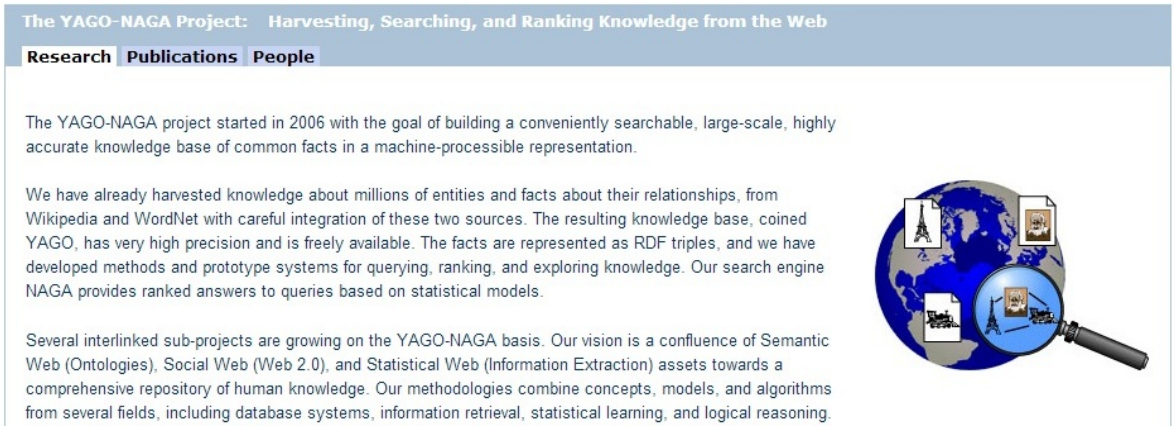


**Figura 3.1: Um trecho da página Web do projeto YAGO-NAGA (http://www.mpi-inf.mpg.de/yago-naga/).**

## 3.1 YAGO-NAGA

*YAGO-NAGA* (WEIKUM et al., 2009) tem como principal objetivo construir uma base de conhecimento de fatos comuns de alta precisão, de larga-escala, de busca conveniente e em uma representação processável por máquina. Resumidamente, o projeto visa transformar a Web em base de conhecimento.

Para este fim, vários sub-projetos formam a base do projeto *YAGO-NAGA*. De acordo com o foco deste trabalho, são apresentados aqueles com maior envolvimento em EI Relacional nas subseções seguintes (principalmente 3.1.1 - *YAGO*, sua evolução 3.1.2 - *YAGO2* e, por fim, 3.1.3 - *PATTY*).

Maiores informações sobre o projeto podem ser obtidas em *http://www.mpi-inf.mpg.de/yago-naga/*.

### 3.1.1 YAGO

*YAGO (Yet Another Great Ontology)* visa construir uma base de conhecimento compreensível a partir de conhecimento humano na forma textual. Para isso, simples técnicas de extração de informação semi-estruturada (como *infoboxes* e categorias de artigos de *Wikipedia*) são aplicadas a recursos de fontes de informação amplos relativamente confiáveis como *Wikipedia* (WEIKUM et al., 2009). Estas técnicas envolvem muito mais EI que PLN e AM.

A base de conhecimento é construída automaticamente, cada artigo da *Wikipedia* se torna uma entidade na base conhecimento (por exemplo, já que existe um artigo sobre Leonardo da Vinci em *Wikipedia*, *LeonardoDaVinci* se torna uma entidade em *YAGO*). Certas categorias são

exploradas para dar certo tipo de informação (por exemplo, o artigo sobre Leonardo da Vinci está na categoria pintores da Itália, então ele se torna um pintor da Itália na ontologia).

Como os dados são semi-estruturados, a tarefa de extração é simplificada; comparada a tarefa de obter frutos de uma árvore, é como se o foco fosse apenas nos frutos mais fáceis de serem colhidos, os que se localizam nos galhos mais baixos no caso (o que pode ser expresso em inglês como: "*Low-Hanging Fruit*"). Um bom exemplo destes "galhos mais baixos", para o *YAGO*, são os *infoboxes* e categorias de *Wikipedia*, que são mapeados para modelos de fatos através de padrões definidos manualmente, como por exemplo, o atributo *born* do infobox do artigo de Leonardo da Vinci é Anchiano, então pode ser feita a extração da tupla *wasBornIn(LeonardoDaVinci, Anchiano)*.

A consistência da ontologia é verificada por esta mesma automaticamente a partir de algumas regras heurísticas criadas por humanos que desconsideram aspectos temporais e espaciais:

- Checar a unicidade de argumentos funcionais, como a restrição de uma pessoa não se casar com ela própria, por exemplo:

  $$spouse(x,y) \wedge diff(y,z) \Rightarrow \neg spouse(x,z)$$

- Checar domínios e alcances das relações, como a restrição de um casal ser formado apenas por um homem e uma mulher, por exemplo:

  $$spouse(x,y) \Rightarrow female(x)$$

  $$spouse(x,y) \Rightarrow male(y)$$

  $$spouse(x,y) \Rightarrow (f(x) \wedge m(y)) \vee (m(x) \wedge f(y))$$

- Restrição rígida. Exemplo:

  $$hasAdvisor(x,y) \wedge graduatedInYear(x,t) \wedge graduatedInYear(y,s) \Rightarrow s < t$$

- Restrição leve. Exemplo:

  $$firstPaper(x,p) \wedge firstPaper(y,q) \wedge author(p,x) \wedge author(p,y) \wedge inYear(q) > inYear(q) + 5years \Rightarrow hasAdvisor(x,y)[0.6]$$

A representação de ontologia da base de conhecimento engloba entidades e relações de interesse público. *YAGO* aprende instâncias e padrões da *Wikipedia*, taxonomia (concepção, nomeação e classificação dos grupos) de *WordNet* e informações de *Geotagging* (procesaaso de adicionar metadados com identificação geográfica) de *Geonames*.

Tanto a extração de entidade nomeada quanto a extração de relação são feitas baseadas em regras e padrões extraídos da *Wikipedia*. No processo de extração/resolução de entidade nomeada, a disambiguação é uma questão relevante para o trabalho (WEIKUM; THEOBALD, 2010).

Cerca de 100 relações (como *wasBornOnDate*, *locatedIn* e *hasPopulation*) foram definidas manualmente e as extrações baseadas em padrões resultaram em 2 milhões de entidades extraídas e 20 milhões de fatos. Diferentemente dos projetos *KnowItAll* e *Read the Web* (Seções 3.3 e 3.4, respectivamente), está fora do foco de *YAGO* aprender novas relações além das definidas manualmente.

### 3.1.2 YAGO2

A continuidade do projeto se dá pelo *YAGO2* que tem como objetivo principal explorar e consultar conhecimento mundial considerando tempo, espaço, contexto e muitas línguas. Para tal fim, novas relações estão no plano especificamente para cobrir as questões de tempo, espaço e contexto. O foco continua em conhecimento da *Wikipedia*: páginas traduzidas são fontes para outras línguas.

*YAGO2* é uma BC imensa, com conhecimento de mais de 10 milhões de entidades e contém mais de 120 milhões de fatos sobre estas entidades.

Avaliação manual baseada em amostragem apresenta que *YAGO2* possui uma precisão (isto é, a ausência de falsos positivos) de mais do que 95%. Um total de 26 juízes avaliaram um número total de 7465 fatos. Isto fornece um valor de precisão para cada amostra. A precisão sobre a amostra é generalizada para o conjunto de dados com ajuda do intervalo de confiança de Wilson e o valor central obtido é de 95% em um comprimento de intervalo menor que $\pm 5\%$. Isto assegura a significância estatística de *YAGO2* (HOFFART et al., 2013).

Destaca-se também por ser uma ontologia ancorada em tempo e espaço (atribui dimensão temporal e espacial a muitos dos fatos e entidades); e, por sua BC possuir domínios temáticos como "música" (*music*) ou "ciência" (*science*).

### 3.1.3 PATTY

*PATTY* é uma coleção de padrões relacionais semanticamente tipados, obtidos a partir da mineração de grandes *corpora*. Em outras palavras, é um grande recurso para padrões textuais que denota relações binárias entre entidades (NAKASHOLE; WEIKUM; SUCHANEK, 2012). Sua taxonomia é derivada da *Wikipedia*. Os padrões são organizados entre sinônimos e subsunções,

como premissas de categoria — "*ÉUm*" ("*IsA*"). Este sistema é baseado em algoritmos para mineração de conjunto de itens frequentes e processa *corpora* de escala Web.

A partir de uma amostra aleatória, uma avaliação apresenta uma precisão de padrão de aproxidamente 85%. *PATTY* tem 8.162 subsunções, com uma precisão de 75%, também baseado em amostra aleatória. O recurso PATTY está disponível gratuitamente para acesso interativo e download no site *http://www.mpi-inf.mpg.de/yago-naga/patty/*.

## 3.2 Google Research — a Relation Extraction Corpus

Para contribuir com esta tarefa de Leitura da Web, o grupo de pesquisa sobre extração relacional do *Google* disponibiliza um conjunto de dados julgado por humanos de duas relações sobre figuras públicas na *Wikipedia*: aproximadamente 10 mil exemplos de "local de nascimento", e mais de 40 mil exemplos de "participou ou graduou em uma instituição". Cada um destes foi julgado por ao menos 5 avaliadores, e pode ser usado para treinar ou avaliar sistemas de extração de relação. Os pesquisadores envolvidos planejam disponibilizar mais relações de novos tipos em breve.

O projeto também aborda a forma relacional binária, isto é, cada extração é formada por uma tripla: a relação em questão, também chamado de contexto ou predicado; o sujeito da relação; e, o objeto da relação. No exemplo apresentado na página Web do projeto[1], "*Stephen Hawking graduated from Oxford*", Stephen Hawking é sujeito, "*graduated from*" ("graduado em") é a relação e *Oxford University* é o objeto.

Para colaborar com o treinamento e avaliação de sistemas de extração de relação, além das triplas também é disponibilizado a evidência da relação, na forma de uma URL e um trecho da página Web que os avaliadores julgaram. Também foram incluídos exemplos onde a evidência não suporta a relação, que servem como exemplos negativos para treinar sistemas de extração.

A intenção deste projeto é de que este *corpus* seja um pequeno avanço para a compreensão computacional da riqueza de relações a serem encontradas em todos os lugares.

## 3.3 KnowItAll

Mudar o paradigma de busca é a maior motivação para o projeto *KnowItAll*: o futuro da pesquisa Web tem foco na leitura da Web em vez de recuperar páginas para realizar a busca

[1] *http://googleresearch.blogspot.com.br/2013/04/50000-lessons-on-how-to-read-relation.html*.

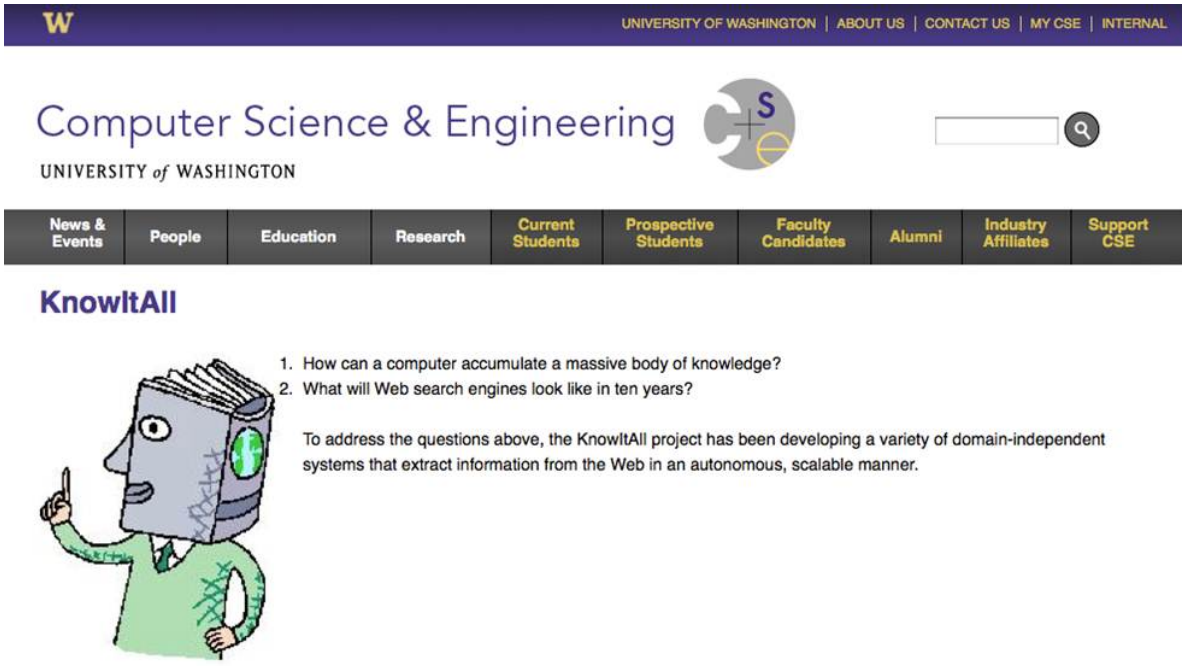


**Figura 3.2: Um trecho da página Web do Projeto KnowItAll (http://www.cs.washington.edu/research/knowitall/).**

Web (BANKO; ETZIONI, 2008). Para este fim, o foco de metodologia abordado primeiramente foi em EI tradicional, e, posteriormente, evoluiu para EIA.

Como abordado na Seção 2.3, a EI teve início focada em satisfazer requisições precisas e pré-especificadas de corpus pequeno e homogêneo (por exemplo, extrair a localização e o horário de seminários de um conjunto de anúncios) (BANKO; ETZIONI, 2007). Tradicionalmente, esses sistemas contavam com envolvimento humano extensivo na forma de regras de extração artesanal ou rotulação manual de exemplos de treinamento. Com isso, um sistema de EI era fixo a tais restrições.

Como o objetivo deste projeto envolve leitura da Web, técnicas de EI tradicional são limitadas e insuficientes para o abrangente conteúdo de informação em questão. Consequentemente, teorias de abordagem de EIA apareceram e TextRunner foi desenvolvido para consolidar a viabilidade na práticas destas teorias.

### 3.3.1 TextRunner

*TextRunner* é um sistema da primeira geração do paradigma de EIA, independente de domínio, desenvolvido com a intenção de superar algumas dificuldades da EI tradicional. Este realiza um único passo orientado a dados sobre o *corpus* e extrai um grande conjunto de tuplas relacionais (saída do sistema) sem precisar de entrada humana; sua única entrada é um *corpus*.

| **Frequência Relativa** | **Categoria** | **Padrão léxico-sintático simplificado (em inglês)** |
|---|---|---|
| 37.8 | Verbo | $E_1$ Verb $E_2$<br>*X estabilished Y* |
| 22.8 | Substantivo+Prep | $E_1$ NP Prep $E_2$<br>*X settlement with Y* |
| 16.0 | Verbo+Prep | $E_1$ Verb Prep $E_2$<br>*X moved to Y* |
| 9.4 | Infinitivo | $E_1$ to Verb $E_2$<br>*X plans to acquire Y* |
| 5.2 | Modificador | $E_1$ Verb $E_2$ Noun<br>*X is Y winner* |
| 1.8 | $\text{Coordenados}_n$ | $E_1$ (and \|, \|:) $E_2$ NP<br>*X-Y deal* |
| 1.0 | $\text{Coordenados}_v$ | $E_1$ (and \|,) $E_2$ Verb<br>*X, Y merge* |
| 0.8 | Aposto | $E_1$ NP (: \|,)? $E_2$<br>*X hometown: Y* |

**Tabela 3.1: Taxonomia de relações binárias extraídas por *TextRunner* (BANKO; ETZIONI, 2008).**

Banko e Etzioni (2007) relatam que, comparado com o *KnowItAll* — sistema de EI da Web de estado-da-arte até então — *TextRunner* atingiu uma redução de erro de 33% em um conjunto de extrações comparável. Além disso, enquanto o *KnowItAll* extrai de acordo com relações pré-especificadas, *TextRunner* extrai um conjunto muito mais amplo de fatos, pois trabalha com um escopo aberto de relações em seu processamento.

A tabela 3.1 apresenta os padrões léxico-sintáticos simplificados mais extraídos por este sistema: a primeira coluna apresenta a frequência relativa; a segunda, a categoria; e, a terceira, o padrão léxico-sintático simplificado, em inglês (língua na qual foram feitas as extrações).

Como apresentado em (FADER; SODERLAND; ETZIONI, 2011), apesar das melhoras notáveis, sistemas de EIA da primeira geração como *TextRunner* apresentam alguns problemas em relação a saída do processamento: conjunto de tuplas relacionais repleto de extrações não-informativas e incoerentes.

Extrações incoerentes são casos onde a frase relacional extraída não tem interpretação com sentido completo (ver tabela 3.2 para exemplos). Extrações incoerentes surgem porque o extrator aprendido faz uma sequência de decisões sobre incluir ou não cada palavra na frase relacional, comumente resultando em predições incompreensíveis. Estas extrações compõem aproximadamente 13% da saída de *TextRunner*. Para solucionar este problema, uma restrição sintática foi introduzida: toda frase relacional multi-palavras deve iniciar com um verbo, terminar com uma preposição, e ser uma sequência contínua de palavras na sentença. Assim, a identificação

| **Sentença** | **Relação Incoerente** |
|---|---|
| The guide *contains* dead links and *omits* sites | contains omits |
| The Mark 14 *was central* to the *torpedo* scandal of the fleet | was central torpedo |
| They *recalled* that Nungesser *began* his career as a precinct leader | recalled began |

**Tabela 3.2: Exemplos de extrações incoerentes (FADER; SODERLAND; ETZIONI, 2011).**

| | |
|---|---|
| is | is an album by, is the author of, is a city in |
| has | has a population of, has a Ph.D. in, has a cameo in |
| made | made a deal with, made a promise to |
| took | took place in, took control over, took advatage of |
| gave | gave birth to, gave a talk at, gave new meaning to |
| got | got tickets to, got a deal on, got funding from |

**Tabela 3.3: Exemplos de extrações não-informativas (esquerda) e seus complementos (direita) (FADER; SODERLAND; ETZIONI, 2011).**

de uma frase relacional é feita de uma só vez (in one fell swoop) em vez de na base de decisões múltiplas, palavra-a-palavra.

Extrações não-informativas são extrações que omitem informação crítica. Por exemplo, considere a sentença "*Faust made a deal with the devil*"("Faust fez um acordo com o diabo"). Sistemas de IE aberta anteriores retornam de forma não-informativa *(Faust, made, a deal)* — (Faust, fez, um acordo) — em vez de *(Faust, made a deal with, the devil)*. Relações não-informativas ocorrem em aproximadamente 7% da saida de *TextRunner*.

Este tipo de erro é causado por manuseio impróprio de frases relacionais que são expressas por uma combinação de um verbo com um substantivo, sendo este que carrega o conteúdo semântico do predicado. A tabela 3.3 ilustra a grande amplitude de relações expressas nessa forma, que não são capturadas por extratores abertos existentes.

### 3.3.2 ReVerb

A fim de superar estes problemas, o sistema da segunda geração de EIA *ReVerb* tem em sua implementação duas restrições simples (sintática e léxica) em relações binárias expressas por verbos. Com isso, o sistema mais que dobra a área sob a curva de precisão-cobertura em comparação a sistemas de EIA anteriores como *TextRunner*. Além disso, mais de 30% das extrações do ReVerb estão com precisão 0.8 ou maior — comparado a virtualmente nada dos sistemas anteriores (FADER; SODERLAND; ETZIONI, 2011).

| V \| V P \| VW * P |
|---|
| V = partícula de verbo? adv? |
| W = (subst \| adj \| adv \| pron \| det) |
| P = (prep \| partícula \| marcador de inf.) |

**Tabela 3.4: Restrição sintática baseada em padrões de etiquetas morfossintáticas (FADER; SODERLAND; ETZIONI, 2011).**

A restrição sintática serve a dois propósitos: elimina extrações incoerentes e reduz extrações não-informativas ao capturar frases relacionais expressas por uma combinação verbo-substantivo, incluindo Construções de Verbos Leves (CVL).

A restrição sintática requer que a frase relacional corresponda ao padrão de rotulação morfossintático apresentado no quadro 3.4. O padrão limita frases relacionais a ser um verbo (por exemplo, *invented*), um verbo seguido imediatamente por uma preposição (por exemplo, *located in*), ou um verbo seguido por substantivos, adjetivos, ou advérbios terminando em uma preposição (por exemplo, *has atomic weight of* ). Se há múltiplas possibilidades de combinação em uma sentença para um único verbo, a maior combinação possível é escolhida. Esse refinamento habilita o modelo a prontamente lidar com frases relacionais contendo múltiplos verbos. Uma consequência desse padrão é que a frase relacional deve ser um pequeno pedaço contíguo de palavras na sentença.

Esta restrição reduz extrações não-informativas, pois extrai frases relações expressas por CVL. Uma CVL é um predicado composto de um verbo e um substantivo, sendo este último o responsável pelo conteúdo semântico do predicado. Para o mesmo exemplo utilizado previamente, *"Faust made a deal with the Devil*, *ReVerb* pode extrair a frase relacional *"made a deal with"*, em vez da relação não-informativa *"made"* ("fez") (ETZIONI et al., 2011; FADER; SODERLAND; ETZIONI, 2011).

Enquanto a restrição sintática reduz bruscamente extrações não-informativas, ela pode algumas vezes combinar frases relacionais que são tão específicas que elas têm apenas algumas poucas instâncias possíveis, até em um corpus de escala de Web. Considere a sentença:

*The Obama administration is offering only modest greenhouse gas reduction targets at the conference.*

O padrão morfossintático vai combinar a frase:

*Is offering only modest greenhouse gas reduction targets at* (1)

Desse modo, existem algumas frases que satisfazem a restrição sintática, mas não são relacionais.

Para superar essa limitação, uma restrição lexical foi introduzida para separar frases relacionais válidas de frases relacionais excessivamente especificadas, como no exemplo (1). A restrição é baseada na intuição que uma frase relacional válida deveria ter muitas instâncias de extrações com muitos argumentos distintos em um corpus grande. A frase em (1) é específica ao par de argumentos *(Obama administration, conference)*, e, assim, improvável de representar uma relação confiável.

*ReVerb* é um extrator aberto novo baseado nas restrições definidas acima. Primeiro identifica frases relacionais que satisfazem as restrições sintáticas e léxicas, e então encontra um par de argumentos para cada frase relacional identificada. São atribuídas às extrações resultantes, então, uma pontuação de confiança usando um classificador de regressão logístico.

O algoritmo de extração de *ReVerb* difere de três formas importantes dos métodos anteriores. Primeiro, a frase relacional é identificada holisticamente em vez de palavra-a-palavra. Segundo, frases potenciais são filtradas baseadas em estatísticas sobre um *corpus* grande (restrição lexical). E, *ReVerb* tem foco em extrair a relação antes dos argumentos, o que permite evitar um erro comum feito pelos métodos anteriores — confundir um substantivo na frase relacional com um argumento.

Dada uma sentença $s$ de entrada, *ReVerb* usa o seguinte algoritmo de extração:

1. Extração de relação: Para cada verbo $v$ em $s$, encontrar a maior sequência de palavras $r_v$ tal que (1) $r_v$ começa em $v$, (2) $r_v$ satisfaz a restrição sintática, e (3) $r_v$ satisfaz a restrição léxica.

2. Extração de argumento: Para cada frase relacional $r$ identificada no Passo 1, encontrar o sintagma nominal $x$ mais próximo à esquerda de $r$ em $s$ tal que $x$ não é um pronome relativo, advérbio iniciado em *who*, ou *there* existencial. Encontre o sintagma nominal $y$ mais próximo à direita de $r$ em $s$. Se tal par *(x, y)* puder ser encontrado, retorne *(x, r, y)* como uma extração.

Para verificar se uma frase relacional candidata $r_v$ satisfaz a restrição sintática, a frase relacional deve corresponder a expressão regular no Quadro 3.4.

Para determinar se $r_v$ satisfaz ou não a restrição lexical, um grande dicionário D de frases relacionais que são conhecidas por terem intâncias com vários argumentos distintos é utilizado. D é construído ao encontrar todas as combinações do padrão morfossintático em um corpus de 500 milhões de sentenças da Web. Para cada frase relacional, seus argumentos são identificados heuristicamente (como no Passo 2 acima). D é o conjunto de todas as frases relações que

tem no mínimo 20 pares de argumentos distintos no conjunto de extrações (bom número para filtrar relações especificadas excessivamente, baseado nos experimentos). A fim de permitir variações mínimas em frases relacionais, cada frase relacional foi normalizada, isto é, a inflexão, verbos auxiliares, adjetivos e advérbios foram removidos. Isso resulta em um conjunto de aproximadamente 1.7 milhões frases relacionais normalizadas distintas, que são armazenadas em memória em tempo de extração (ETZIONI et al., 2011).

Como um exemplo do algoritmo de extração em ação, considere a seguinte sentença de entrada:

*Hudson was born in Hampstead, which is a suburb of London.*

O passo 1 do algoritmo identifica três frases relacionais que satisfazem as restrições sintáticas e léxicas: *was*, *born in*, e *is a suburb of*. As duas primeiras frases são adjacentes na sentença, então elas são fundidas em uma única frase relacional *was born in*. O passo 2, então, encontra um par de argumentos para cada frase relacional. Para *was born in*, os SNs mais próximos são *(Hudson, Hampstead)*. Para *is a suburb of*, o extrator pula o SN *which* e escolhe o par de argumentos *(Hampstead, London)*. A saída final é

$e_1$*: (Hudson, was born in, Hampstead)*

$e_2$*: (Hampstead, is a suburb of, London)*

Este algoritmo tem alta cobertura, mas, baixa precisão. Assim como os extratores abertos anteriores, um caminho para trocar cobertura por precisão por meio do ajuste de um limite de confiança é procurado. Um classificador de regressão logística foi usado para atribuir uma pontuação de confiança para cada extração, que usa as características mostradas na Tabela 3.5. Todas essas características são eficientemente computáveis e independentes de relação. A função de confiança foi treinada manualmente rotulando as extrações de um conjunto de 1.000 sentenças da Web e do Wikipedia como correto ou incorreto.

Extratores abertos anteriores requerem dados de treinamento rotulados para aprender um modelo de relações, que é então usado para extrair frases relacionais do texto. Em contraste, *ReVerb* usa um modelo específico de relações para extração, e requer dados rotulados apenas para atribuir a pontuação de confiança para suas extrações. Aprender uma função de confiança é uma tarefa muito mais simples do que um modelo completo de relações, usando duas ordens de magnitude menos exemplos de treinamento do que *TextRunner*.

Fader, Soderland e Etzioni (2011), Etzioni et al. (2011) desenvolveram uma análise detalhada dos erros produzidos por *ReVerb* para compreender suas limitações em precisão (extrações incorretas retornadas pelo sistema) e em cobertura (extrações corretas que *ReVerb* per-

| Peso | Recurso |
|---|---|
| 1.16 | *(x, r, y)* cobre todas as palavras em *s* |
| 0.50 | A última preposição em *r* é *for* |
| 0.49 | A última preposição em *r* é *on* |
| 0.46 | A última preposição em *r* é *of* |
| 0.43 | *tamanho(s)* $\leq$ 10 palavras |
| 0.43 | Existe uma palavra inciada em WH à esquerda de *r* |
| 0.42 | *r* corresponde ao padrão VW*P da figura 3.4 |
| 0.39 | A última preposição em *r* é *to* |
| 0.25 | A última preposição em *r* é *in* |
| 0.23 | 10 palavras $<$ *tamanho(s)* $\leq$ 20 palavras |
| 0.21 | *s* começa com *x* |
| 0.16 | *y* é um nome próprio |
| 0.01 | *x* é um nome próprio |
| -0.30 | Existe um SN à esquerda de *x* em *s* |
| -0.43 | 20 palavras $<$ *tamanho(s)* |
| -0.61 | *r* corresponde ao padrão V da figura 3.4 |
| -0.65 | Existe uma preposição à esquerda de *x* em *s* |
| -0.81 | Existe um SN à direita de *y* em *s* |
| -0.93 | Conjunção coord. à esquerda de *r* em *s* |

**Tabela 3.5: *ReVerb* utiliza estes recursos para atribuir uma pontuação de confiança a uma extração *(x, r, y)* de uma sentença *s* utilizando um classificador de regressão logística (FADER; SODERLAND; ETZIONI, 2011).**

deu/errou). Com uma investigação cuidadosa dos erros de saída do *ReVerb*, notou-se que a maior deficiência deste sistema de EIA estava na identificação dos argumentos da relação, os conceitos. Conforme análise de padrões sintáticos feita, a maioria dos argumentos se encaixa em um pequeno número de categorias sintáticas. Similarmente, existem delimitadores comuns que poderiam auxiliar na detecção de limites de argumentos.

Tabela 3.6 sintetiza os tipos de extrações incorretas presentes na saída de *ReVerb* — 65% de extrações incorretas retornadas pelo ReVerb foram casos onde a frase relacional foi identificada corretamente, mas a heurística de encontrar argumento falhou. Os erros restantes foram casos onde o sistema extraiu uma frase relacional incorreta. Um erro comum foi extrair uma frase

| Extrações incorretas de *ReVerb* | |
|---|---|
| 65% | frase relacional correta, argumentos incorretos |
| 16% | Relação n-ária |
| 8% | frase relacional não contígua |
| 2% | Verbo imperativo |
| 2% | frase relacional excessivamente especificada |
| 7% | Outros, incluindo erros de conflito ou etiquetação morfossintática |

**Tabela 3.6: Estatísticas de extrações incorretas de *ReVerb* (FADER; SODERLAND; ETZIONI, 2011).**

| Extrações perdidas por *ReVerb* | |
|---|---|
| 52% | Não conseguiu identificar argumentos corretos |
| 23% | Relação filtrada pela restrição léxica |
| 17% | Identificou uma relação mais específica |
| 8% | Erros de conflito ou etiquetação morfossintática |

**Tabela 3.7: Estatísticas de extrações perdidas por *ReVerb* (FADER; SODERLAND; ETZIONI, 2011).**

relacional que expressa uma relação n-ária por meio de um verbo que tem um sujeito e dois objetos. Por exemplo, dada a sentença "*I gave him 15 photographs*", *ReVerb* extrai *(I, gave, him)*. Estes erros ocorrem porque este sistema só modela relações binárias.

Tabela 3.7 sumariza as extrações corretas que foram extraídas por outros sistemas, mas não pelo *ReVerb*. Assim como com as extrações positivas falsas, a maioria das negativas falsas (52%) foram devido à escolha errada da heurística de encontrar argumentos, ou falha ao extrair todos os argumentos possíveis. Outras fontes de falha foram por conta da restrição lexical: tanto falhando para filtrar uma frase relacional excessivamente especificada quanto removendo uma frase relacional válida.

Desta forma, a evolução do projeto se consolidou em uma nova versão do sistema de extração de informação com um novo componente para extração de argumentos.

### 3.3.3 R2A2

Na continuação da evolução dos sistemas de EIA, tem origem o *R2A2* (também da segunda geração), que adiciona à implementação do *ReVerb* um identificador de argumentos para melhor extraí-los, *ArgLearner*. *R2A2* e *ArgLearner* serão melhor explicados no decorrer desta subseção.

Etzioni et al. (2011) apresentam que simples sintagmas nominais somam apenas 65% de $arg_1$s (sujeitos) e por volta de 60% de $arg_2$s (objetos). Isto naturalmente determina um limite superior de cobertura para sistemas que não manipulam argumentos mais complexos. Felizmente, existem apenas algumas outras categorias proeminentes — para $arg_1$: frases preposicionais e listas, e para $arg_2$: frases preposicionais, listas, $arg_2$s com orações independentes e orações relativas. Essas categorias cobrem mais de 90% das extrações, o que implica em um esforço para manipular isso de forma inteligente para aumentar a precisão significativamente.

Outro ponto importante é a posição dos argumentos na frase toda. Tem-se que 85% de $arg_1$s são adjacentes a frase relacional. Aproximadamente todos os casos restantes são devido a verbos compostos (10%) ou orações relativas interventivas (5%). Esses três casos somam 99%

das relações da amostra de (ETZIONI et al., 2011).

Um exemplo de verbos compostos é da sentença "*Mozart was born in Salzburg, but moved to Vienna in 1781*", que resulta em uma extração com $arg_1$ não-adjacente:

*(Mozart, moved to, Vienna)*

Um exemplo de uma oração relativa interventiva é a da sentença "*Starbucks, which was founded in Seattle, has a new logo*". Também resulta em uma extração com $arg_1$ não-adjacente:

*(Starbucks, has, a new logo)*

$Arg_2$s quase sempre seguem imediatamente a frase relacional. No entanto, seus delimitadores de fim são mais complicados, pois existem vários. Em 58% das extrações, $arg_2$ extende para o fim da sentença. Em 17% dos casos, $arg_2$ é seguido de uma conjunção ou uma palavra de função como "*if*", "*while*"ou "*although*"e então seguida de uma oração independente ou frase verbal. Mais difícil de detectar são os 9% onde $arg_2$ é seguido diretamente por uma oração independente ou uma frase verbal. Mais difícil que tudo isso são os 11% onde $arg_2$ é seguido por uma preposição, visto que frases preposicionais poderiam também ser parte de $arg_2$. Eis o difícil problema de fixação de frase preposicional. Em (ETZIONI et al., 2011), uma evidência sintática limitada (rotulação morfossintática) foi utilizada para identificar argumentos, embora mais conhecimento semântico poderia vir a calhar para desambiguar frases preposicionais.

A arquitetura do sistema *ArgLearner* é apresentada na Figura 3.3: à esquerda tem-se os dados de treinamento como entrada para o construtor de dados de treinamento; abaixo, as sentenças que são processadas pelo extrator de relação (ou padrão de contexto); à direita, tem-se o extrator de argumentos formado pelos classificadores de limite: para *arg1*, um a direita e outro a esquerda, e, para *arg2* apenas um à direita (já que o limite a esquerda é a relação (ou, mais especificamente, o término desta)"; feito a extração de argumentos, as extrações passam por um componente que atribui novos valores à confiança destas (*Reranker*), e, então, tem-se o resultado final das extrações.

Desta forma, com a metodologia de extração de relações de *ReVerb* e com a extração de argumentos de *ArgLearner*, *R2A2* quase dobra a curva de precisão-cobertura comparado ao *ReVerb* por si só, como ilustrado na Figura 3.4.

## 3.4 Read the Web - RTW

O objetivo do projeto *Read the Web* é construir um sistema de aprendizado de máquina sem fim que melhore constantemente sua habilidade (aprenda) de extrair informação estruturada de

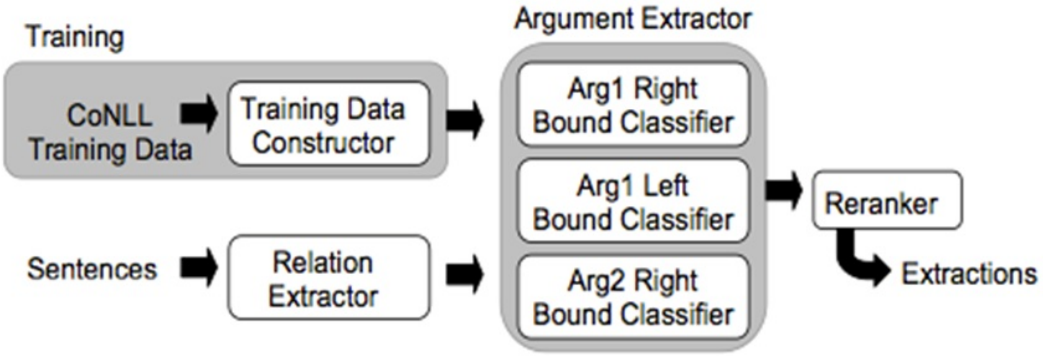


**Figura 3.3: Arquitetura do sistema *ArgLearner* (ETZIONI et al., 2011)**

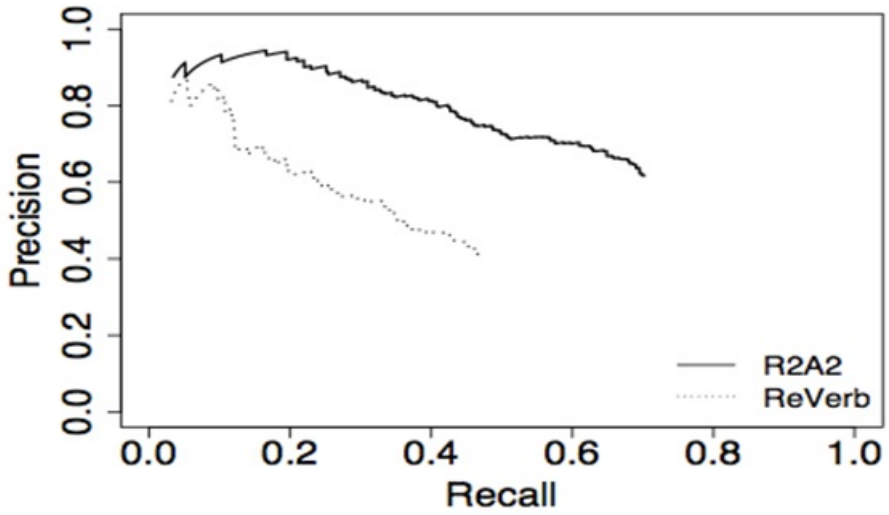


**Figura 3.4: *R2A2* tem cobertura e precisão substancialmente mais altos que o *ReVerb* (ETZIONI et al., 2011).**

páginas Web não estruturadas. Caso tenha sucesso, resultará em uma base de conhecimento (ou seja, uma base de dados relacional) de informação estruturada que reflete o conteúdo da Web, que pode ser a estrutura principal de diversas aplicações para uso da humanidade, como por exemplo um assistente pessoal com o qual pode-se conversar.

Este projeto de pesquisa tem a intenção de definir formalmente e comprovar que o recém-originado paradigma de Aprendizado Sem Fim (ASF, detalhado na subseção 2.1.2) é eficiente e viável na teoria e na prática (DUARTE, 2011; Hruschka Jr; DUARTE; NICOLETTI, 2013). Neste contexto, surgiu o *NELL - Never-Ending Language Learner*, um sistema que opera 24 horas por dia e continuamente melhora sua habilidade de extrair fatos da web (CARLSON et al., 2010). Uma descrição mais profunda deste sistema é apresentada na subseção que segue (3.4.1).

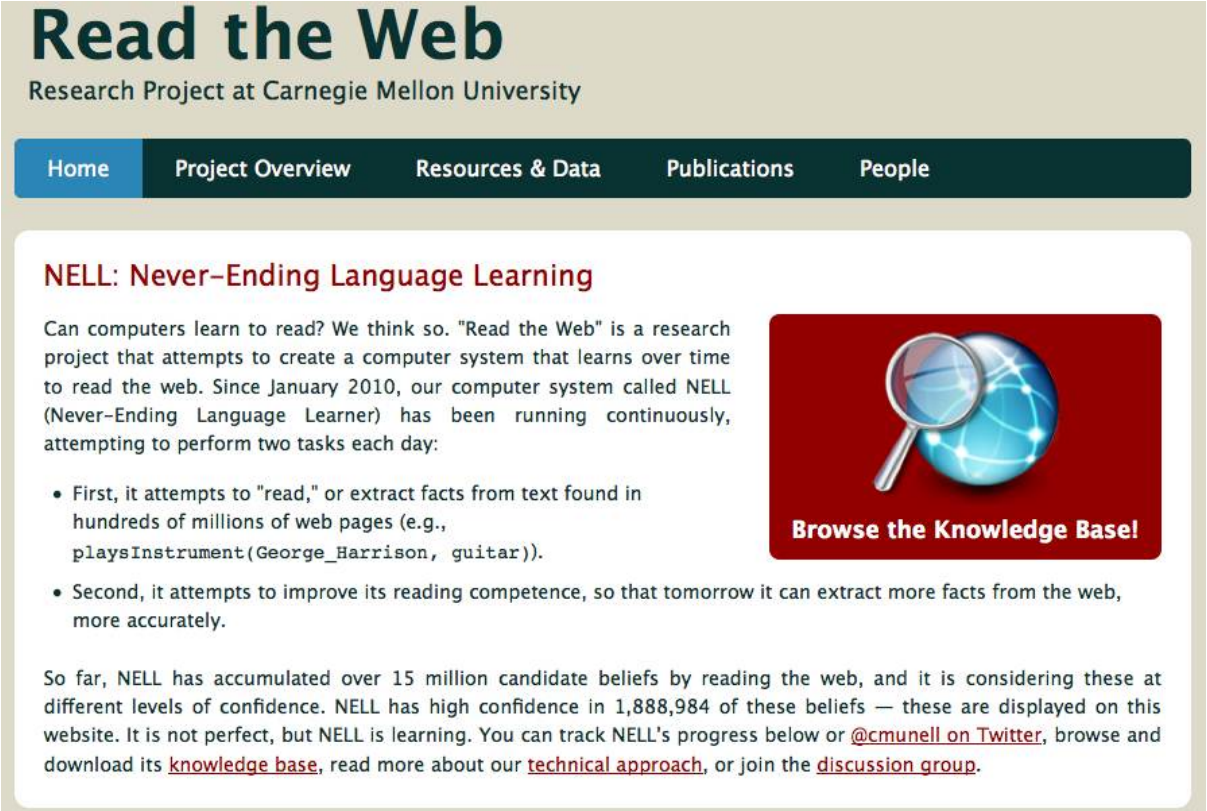


**Figura 3.5: Uma introdução sobre *NELL* na página do projeto *Read The Web* (http://rtw.ml.cmu.edu/rtw/).**

### 3.4.1 NELL

O *NELL* possui uma base de conhecimento que contém vários milhões de crenças extraídas por ele da web com grau confiança variável. Dessas crenças, o sistema tem confiança bastante elevada em aproximadamente dois milhões, as quais são denominadas crenças candidatas. A Figura 3.5 ilustra a página principal do projeto Read the Web e uma introdução sobre o NELL.

O sistema tem como entrada uma ontologia que define centenas de categorias (por exemplo, pessoa, bebida, atleta, esporte) e relações *tipadas* entre essas categorias. Também é fornecido um conjunto de 10 a 20 exemplos positivos sementes para cada categoria e relação, juntamente com uma coleção de 1 bilhão de páginas da web do corpus *ClueWeb09* (melhor detalhado no apêndice A) como dados não rotulados, e acesso a 100.000 pesquisas diárias no mecanismo de busca do Google. *NELL* tem duas tarefas por dia: (1) extrair novas crenças da web para popular sua base de conhecimento (KB) crescente com as instâncias das categorias e relações desta ontologia, e (2) aprender a realizar a tarefa 1 melhor hoje do que podia ontem. Sua competência de aprendizado pode ser mensurada ao permitir que este considere hoje os mesmos documentos de texto que considerou ontem; então, verifica-se se ele extrai mais crenças, com maior precisão hoje do que fazia ontem (LAO; MITCHELL; COHEN, 2011).


**Figura 3.6: Fatos aprendidos por NELL (http://rtw.ml.cmu.edu/rtw/).**

O *NELL* usa um algoritmo de aprendizado multi-tarefa semi-supervisionado em larga-escala que acopla o conjunto de treinamento de mais de 2500 diferentes classificadores e métodos de extração. O sistema é composto por diversos componentes que aprendem de diferentes perspectivas dos dados: por exemplo, uma visão usa características ortográficas de um nome de entidade potencial, e outra usa contextos livres de texto nos quais o sintagma nominal é encontrado. Outra característica relevante é a de ser um sistema que treina a si mesmo com base em sua coleção crescente de crenças confiáveis (CARLSON et al., 2010).

Este sistema de aprendizado contínuo possui um Integrador de Conhecimento (IC) para examinar os fatos candidatos propostos e promove os mais fortemente amparados destes ao status de crença. Em cada iteração, cada subsistema componente realiza seu processamento utilizando como entrada a Base de Conhecimento (BC) atual, e então o IC decide quais fatos candidatos propostos são promovidos. A BC cresce iteração a iteração, provendo mais e mais crenças que são utilizadas por cada subsistema componente para retreinar a si mesmos a aprender a ler melhor na próxima iteração. Neste sentido, esta abordagem pode ser vista como uma implementação de um método de aprendizado semissupervisionado acoplado no qual múltiplos componentes aprendem e compartilham tipos complementares de conhecimento, supervisionados pelo IC. Esta abordagem também pode ser vista como uma aproximação para um algoritmo de Maximização de Expectativa (ME) no qual o passo da Expectativa envolve estimar iterativamente os valores verdadeiros para um conjunto muito grande de crenças candidatas virtuais da BC compartilhada, e o passo de Maximização envolve retreinar os vários métodos de extração dos subsistemas componentes (CARLSON et al., 2010). Assim, iterativamente, os próprios componentes juntos com o IC concretizam duas importantes propriedades do NELL:

auto-supervisão e auto-reflexão (mais detalhes na Seção 1).

O *NELL* está funcionando desde Janeiro de 2010. Como resultado, possui uma base de conhecimento continuamente crescente com mais de 70 milhões de fatos extraídos (com diferentes graus de certeza), sendo mais de 2 com alto grau de certeza (e alta precisão) considerados crenças. A figura 3.6 apresenta fatos (aleatoriamente amostrados) aprendidos pelo sistema (instâncias — *instance*), o número da iteração na qual foram aprendidos (*iteration*), a data (*date learned*) e um valor para a confiança do fato aprendido, de 0 a 100 (*confidence*); além de dois botões em formas de mão (um positivo e um negativo, assinalados com o polegar) para aprovação ou reprovação do fato aprendido.

A base de conhecimento e maiores informações sobre este sistema do projeto *Read the Web* podem ser encontradas em *http://rtw.ml.cmu.edu*. Os componentes base de NELL são descritos nas subseções a seguir.

#### 3.4.1.1 Coupled Pattern Learner - CPL

Coupled Pattern Learner (CPL) é um sistema de aprendizado semi-supervisionado que tem como entrada uma ontologia e suas regras (como regras de exclusão mútua entre predicados relacionais). O sistema iterativamente extrai padrões de contexto e instâncias para categorias e para as relações de um corpus de cerca de 500 milhões de páginas web. CPL é um dos sistemas integrantes do NELL (CARLSON et al., 2010).

O algoritmo CPL aprende a extrair instâncias de categoria e de relação a partir de texto não estruturado e pode ser resumido no Algoritmo 1. CPL aprende padrões de contexto que são extratores de alta precisão para cada predicado (por exemplo, "*arg1* e outras firmas de software"e "*arg1* marcou um gol para *arg2*") e usa-os para compor um conjunto de instâncias de predicado de alta precisão. Instâncias de categoria de conhecimento que aparecem nas posições de *arg1* e *arg2* em sentenças do texto não estruturado *co-ocorrem* com estes padrões de contexto.

CPL inicia os conjuntos de instâncias e padrões promovidos com instâncias sementes e padrões providos como entrada. Em cada iteração, CPL expande estes conjuntos para cada predicado obedecendo regras de exclusão mútua e de verificação de tipo. Isso é realizado ao filtrar candidatos que co-ocorrem com instâncias e padrões de classes mutualmente exclusivas e ao verificar se os argumentos das relações candidatas são candidatos das categorias de interesse.

**Algoritmo 1:** ***Coupled Pattern Learner (CPL)***

**Entrada**: Uma ontologia $O$, e um corpus de texto $C$

**Saída**: Instâncias (padrões de contexto) confiáveis para cada predicado (relação)

Para i = 1, 2, ..., ∞ faça

  Para cada *predicado* $p \in O$ faça

    Extrair novas instâncias (padrões de contexto) candidatos utilizando instâncias recentemente promovidas;

    Filtrar candidatos que violem acoplamento;

    Pontuar as instâncias candidatas;

    Promover melhores candidatos;

  Fim-Para

Fim-Para

#### 3.4.1.2 OntExt

O *NELL* possui atualmente um componente que utiliza técnicas de EIR da Web para gerar novas relações: *OntExt*. Este sistema combina características de Extração Relacional tradicional e Extração Relacional Aberta, para descobrir novas relações entre categorias que já estão presentes na ontologia, e para as quais muitas instâncias já foram extraídas (MOHAMED; Hruschka Jr.; MITCHELL, 2011).

O foco da abordagem utilizada no modelo do *OntExt* é utilizar redundância de informação da Web, ou seja, o mesmo fato relacional ser frequentemente afirmado em um *corpus* de texto muito grande, com diferentes padrões de contexto. Com isso, padrões de contexto semanticamente similares são agrupados embora exista a possibilidade de serem lexicamente não-similares.

Possui três etapas. (1) Começa explorando um grande corpus da web e (2) instâncias de categoria extraídas pelo CPL para gerar novas relações. Depois que as relações são geradas, (3) um classificador é utilizado para classificar semanticamente relações válidas.

O *OntExt* tem o objetivo de descobrir novas relações afirmadas frequentemente entre categorias da ontologia. Para isso, para cada par de categorias, ele agrupa pares de instâncias conhecidas e a relação de contexto que interliga tais instâncias. O algoritmo para esta tarefa é apresentado a seguir:

**Algoritmo 2: Geração de Relações**

**Entrada**: Um par de categorias (C1, C2) e conjunto de sentenças, cada sentença contendo um par de instâncias conhecidas pertencentes a C1 e C2 (a frase que conecta as instâncias nesta sentença é o contexto).

**Saída**: Relações e suas instâncias sementes.

Passos

1. Das sentenças de entrada, construir um Contexto pela matriz de co-ocorrência de Contexto. A matriz é, então, normalizada.
2. Aplicar agrupamento K-means na matriz para agrupar os contextos relacionados. Cada grupo corresponde a uma possivel nova relação entre as duas categorias de entrada.
3. Atribuir pontuação aos pares de instâncias conhecidos (pertencentes a C1, C2) para grupo e pegar as 50 instâncias semetes com mais pontos para a relação.

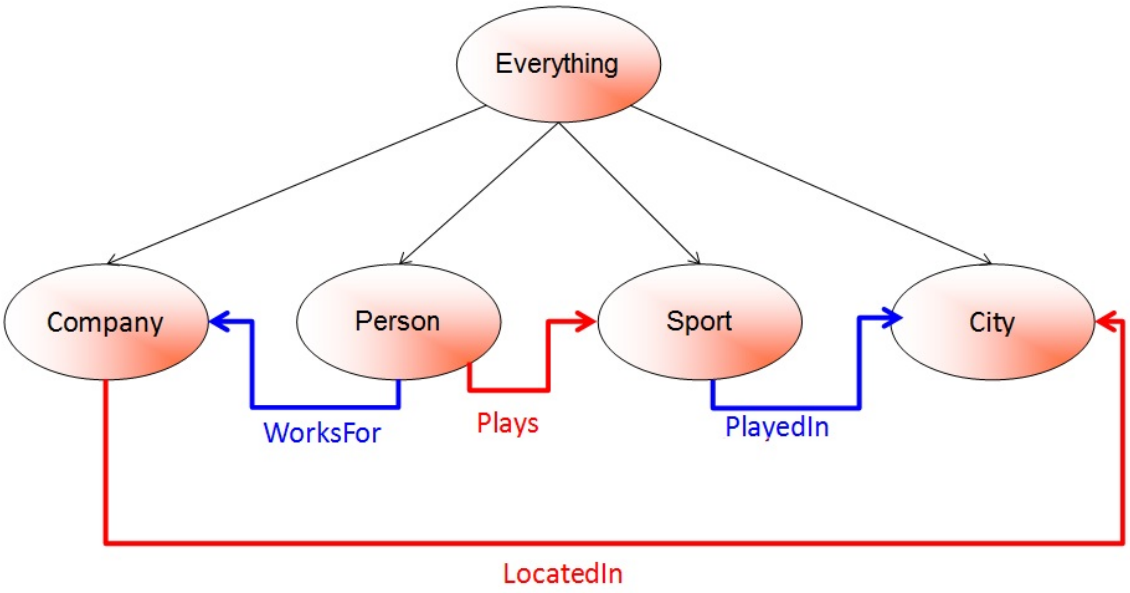


**Figura 3.7: Representação das relações criadas por OntExt (Hruschka Jr, 2012).**

A figura 3.7 esboça um exemplo possível a partir da execução do *OntExt*: (1) se é conhecido que pessoas trabalham para empresas (*(Person, WorksFor, Company)*) e que esportes são jogados em cidades (*(Sport, PlayedIn, City)*), (2) e são conhecidos vários exemplos de pessoas e vários exemplos de esportes, (3) se são identificadas várias sentenças textuais conectando pessoas a esportes, (4) então, pode-se inferir que pessoas praticam esportes (*(Person, Plays, Sport)*). (5) E são conhecidos vários exemplos de empresas e vários exemplos de cidades, (6) se são identificadas várias sentenças textuais conectando empresas a cidades no sentido de localização, (7) então, o agrupamento da matriz de co-ocorrência que envolve empresas e cidades

resulta na frase verbal ”localiza-se em” (*LocatedIn*) como sendo a mais próxima do centróide, portanto pode-se inferir que empresas localizam-se em cidades (*(Company, LocatedIn, City)*).

No *OntExt*, é realizada uma tarefa de agrupamento (aprendizado não supervisionado) com base numa matriz [*padrões de contexto X padrões de contexto*], por ser muito mais escalável que uma matriz [*pares de sintagmas nominais X pares de sintagmas nominais*], já que os padrões de contexto são menos numerosos e o sistema aplica diversos critérios para remover padrões irrelevantes.

A matriz de co-ocorrência de contextos para cada par de categorias é a principal estrutura de dados usada pelo *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011); cada célula corresponde ao número de pares de instâncias de categoria no qual ambos os contextos co-ocorrem (por exemplo, nas sentenças "*Vioxx can cure Arthritis*" e "*Vioxx is a treatment for Arthritis*", os contextos "*can cure*" e "*is a treatment for*" co-ocorrem com o par de instâncias [*Vioxx, Arthritis*]). Inicialmente, o valor de $Matriz(i,\ j)$ é o número de pares de instâncias de categorias que ocorrem com ambos os contextos ($i$ e $j$). Então, toda célula da matriz é normalizada (dividida pela contagem total para sua linha) desta forma:

$$Matriz(i,j) = \frac{Matriz(i,j)}{\sum_{j=0}^{N} Matriz(i,j)} \tag{3.1}$$

A fim de promover contextos menos genéricos, maior peso foi dado para contextos que co-ocorrem com apenas alguns contextos, como na fórmula a seguir:

$$Matriz(i,j) = Matriz(i,j) * \frac{N}{\mid \{Contexto(j)\ :\ Matriz(i,\ j)\ >\ 0\} \mid} \tag{3.2}$$

Onde $N$ é o número total de contextos, e $|\{Contexto(j) : Matriz(i,j) > 0\}|$ refere-se ao número de células na linha $Matriz(i)$ que são maiores que zero.

Mohamed, Hruschka Jr. e Mitchell (2011) exemplificam que para o par de categorias $<$*drug, disease*$>$, contextos como "*to treat*", "*for treatment of*", "*medication*" tem valores altos de co-ocorrência, pois indicam a mesma relação ("*drug -to treat- disease*"). Similarmente, "*can cause*", "*may cause*", "*can lead to*" (que indicam a relação "*drug -can cause- disease*") também têm altos valores de ocorrência.

*OntExt* agrupa os contextos com alto número de co-ocorrências juntos na matriz de co-ocorrência. Cada grupo é usado, então, para propor uma nova relação. Assim, o peso atribuído a cada instância semente é inversamente proporcional ao desvio padrão do contexto em relação

ao centróide[2] do agrupamento de contextos, e diretamente proporcional ao número de vezes que esta co-ocorre com o contexto (MOHAMED; Hruschka Jr.; MITCHELL, 2011).

Em resumo, o peso de cada instância semente *s* (par de instâncias de categorias) é atribuído como segue:

$$\sum_{c \in AP} \frac{Oco(c,\ s)}{1 + dp(c)} \tag{3.3}$$

Onde *AP* é o agrupamento de contextos padrões para a relação em questão; $Oco(c,\ s)$ é o número de vezes que a instância *s* co-ocorre com o contexto padrão *c*; $dp(c)$ é o desvio padrão do contexto *c* em relação ao centróide do agrupamento padrão.

As instâncias foram pontuadas de acordo com esta métrica e as 50 melhores são colhidas como instâncias sementes iniciais para a relação proposta.

Mesmo escolhendo as melhores, mais da metade das relações geradas são inválidas. Os principais motivos são: erros em instâncias de categorias, ambiguidade semântica, relações incompletas semanticamente e relações não lógicas. Como a introdução de relações inválidas pode afetar adversamente o desempenho do *NELL*, fez-se necessário superar o problema desafiador de classificar relações válidas semanticamente.

Para esta tarefa, algumas características e recursos são utilizados: contador de frequência de cada instância de categoria normalizado; distribuição dos padrões de extração; características de relação que ajudam a identificar relações válidas; número de contextos padrões alcançados através do agrupamento de padrões para a relação; e, quão específico é o padrão de contexto para a relação em questão.

Extrações realizadas por um sistema de aprendizado semi-supervisionado componente do *NELL* | *Coupled Pattern Learner (CPL)* | são utilizadas para realizar a geração de relações. Este sistema tem como entrada uma ontologia e restrições (como regras de exclusão mútua entre predicados). *CPL* iterativamente extrai padrões e instâncias para predicados de categoria e relação de um *corpus* web de cerca de 500 milhões de páginas web (CARLSON et al., 2010).

Aproximadamente 22.000 instâncias pertencentes a 122 categorias extraídas por *CPL* e o *corpus* web foram utilizados como entrada para o *OntExt* nos experimentos de Mohamed, Hruschka Jr. e Mitchell (2011). O processo gerou 781 relações. Alguns exemplos de instâncias de relações geradas são apresentadas na Tabela 3.8, com o contexto textual e o par de categorias referente.

[2]Relação no centro do agrupamento, melhor situada, com maior pontuação.

| Par de categorias | Relação | Contextos textuais | Instâncias extraídas |
|---|---|---|---|
| MusicInstrument<br>Musician | Master | $ARG_1$ master $ARG_2$<br>$ARG_1$ legend $ARG_2$<br>$ARG_2$ plays $ARG_1$ | sitar, George Harrison<br>tenor sax, Stan Getz<br>trombone, Tommy Dorsey<br>vibes, Lionel Hampton |
| Disease<br>Disease | IsDueTo | $ARG_1$ is due to $ARG_2$<br>$ARG_1$ is caused by $ARG_2$ | pinched nerve, herniated disk<br>tennis elbow, tendonitis<br>blepharospasm, dystonia |
| CellType<br>Chemical | ThatRelease | $ARG_1$ that release $ARG_2$<br>$ARG_2$ releasing $ARG_1$ | epithelial cells, surfactant<br>neurons, serotonin<br>mast cells, histomine |
| Mammals<br>Plant | Eat | $ARG_1$ eat $ARG_2$<br>$ARG_1$ eating $ARG_2$ | koala bears, eucalyptus<br>sheep, grasses<br>goats, saplings |

**Tabela 3.8: Instâncias de relações geradas por *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011).**

| Categoria1 -relação- Categoria2 | Contextos relacionais | Instâncias sementes |
|---|---|---|
| SportsGame -beating- Country | "beating" | Tournament, Sri Lanka<br>Champions, France<br>Match, Canada |
| Animal -will eat- Condiment | "will eat"<br>"eating" | Wolf, Sheep<br>Fox, Rabbit<br>Lion, Lamb |

**Tabela 3.9: Instâncias de categorias incorretas geradas por *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011).**

As Tabelas 3.9, 3.10, 3.11 e 3.12 mostram relações inválidas para cada tipo de invalidade: "erro nas instâncias de categoria", "ambiguidade semântica", "relações incompletas semanticamente" e "relações que representam fatos não concretos" respectivamente. Mais especificamente, a Tabela 3.9 apresenta uma amostra de relações geradas incorretas por ser atribuído uma categoria incorreta a uma entidade (instâncias de categoria incorretas estão em itálico).

A Tabela 3.10 apresenta uma amostra de relações que foram geradas por causa de ambiguidade semântica (instâncias com ambiguidade estão em itálico); a Tabela 3.11 mostra algumas das relações geradas que são incompletas semanticamente; e, a Tabela 3.12 apresenta amostras de relações ilógicas que não estabelecem um fato concreto.

Nestes experimentos de Mohamed, Hruschka Jr. e Mitchell (2011), *OntExt* obteve 71,6% de precisão e 72,2% de cobertura para relações válidas; 76,5% de precisão e 75,9% de cobertura para inválidas; e, 74,2% de precisão e cobertura para a média ponderada.

Apesar dos resultados satisfatórios, esse processamento pode ser mais rápido e expandir ainda mais a ontologia do sistema de forma confiável, o que despertou motivação para desen-

| Relação | Contextos relacionais | Instâncias sementes |
|---|---|---|
| Bird -play- City | "play" | Carinals, Atlanta<br>Ravens, Miami<br>Eagles, Chicago |
| BakedGood -baking- Magazine | "baking" | Time, Cakes<br>People, Cookies |

**Tabela 3.10: Relações semanticamente ambíguas geradas por *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011).**

| Relação | Contextos relacionais | Instâncias sementes |
|---|---|---|
| Person -acknowledged- Date | "acknowledged"<br>"warned"<br>"met" | Mr Obama, Tuesday<br>George W. Bush, Tuesday<br>Al Gore, Thursday |
| NewsPaper -is reporting that- Company | "is reporting that"<br>"writes that"<br>"reported that" | Financial Times, Apple<br>Wall Street Journal, GM<br>Wall Street Journal, Yahoo |

**Tabela 3.11: Relações incompletas semanticamente geradas por *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011).**

| Relação | Contextos relacionais | Instâncias sementes |
|---|---|---|
| Emotion -of living in- StateOrProvince | "of living in" | Joy, California<br>Excitement, Colorado<br>Fear, Iowa |
| BodyPart -to keep- BodyPart | "to keep"<br>"guard" | Hand, Eye<br>Nose, Throat<br>Eye, Brain<br>Elbow, Hand |

**Tabela 3.12: Relações que representam fatos não concretos geradas por *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011).**

volver um novo componente que substitua-o com maior eficiência e precisão a curto prazo, e maior cobertura a longo prazo. Tal novo componente é o resultado esperado deste trabalho de mestrado.

#### 3.4.1.3 Prophet

Além de *OntExt*, o sistema *NELL* possui outros componentes para aumentar sua BC. O componente chamado *Prophet* (APPEL; Hruschka Jr, 2011) prevê novas relações a partir da mineração do grafo que representa a BC de *NELL* (único recurso de conhecimento utilizado) com três objetivos principais:

1. Extender a BC ao prever novas relações (arestas) que podem existir entre pares de nós;
2. Induzir regras de inferência;
3. Identificar arestas foras de lugar que podem ser utilizadas pelo NELL como dicas para identificar conexões erradas entre nós (fatos errados).

Com isso, o *NELL* pode ter como base os resultados do *Prophet* (em conjunto com o *OntExt*) para automaticamente estender sua BC ao prever novos fatos assim como novas relações com alta precisão. Para tanto, triângulos abertos são procurados no grafo, e, se conveniente, são fechados. Um exemplo de triângulo aberto é apresentado na Figura 3.8: os lados fechados do triângulo são representados pelas relações entre *Basketball* e *Milwaukee Bucks* e entre *Milwaukee Bucks* e *NBA*; o lado restante, entre *Basketball* e *NBA* é o lado aberto. Para fechá-lo, faz-se necessária uma nova relação entre *Basketball* e *NBA*.

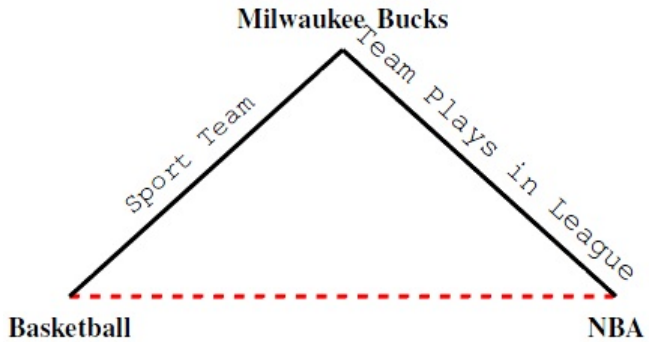


**Figura 3.8: Triângulo aberto A entre categorias na BC de *NELL* (APPEL; Hruschka Jr, 2011).**

As categorias e relações do *NELL* são mapeadas em um grafo. As Figuras 3.8 e 3.9 ilustram como duas relações que compartilham de um predicado podem ser vistas como um triângulo aberto, chamado $A$, e, seu agrupamento de categorias, $A_c$.

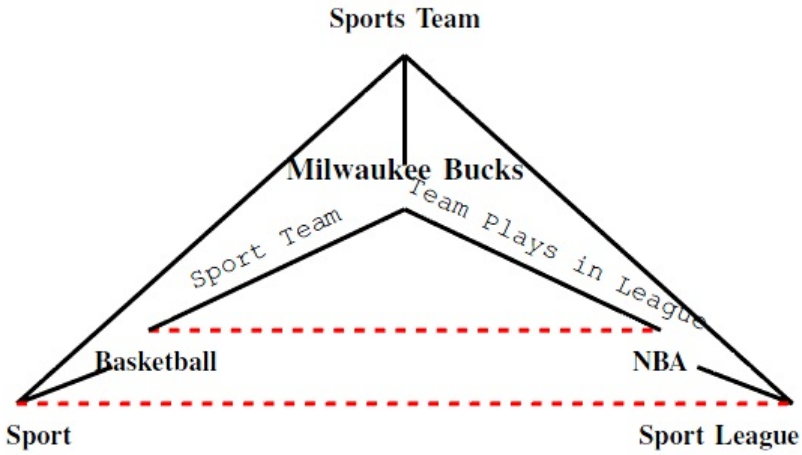


**Figura 3.9: Triângulo aberto *A(Basketball, NBA)* agrupado em $A_c$*(Sports, SportsLeague)* (APPEL; Hruschka Jr, 2011).**

A Figura 3.10 ilustra a conexão de *Basketball* e *NBA* por: *Milwaukee Bucks* (*sportsTeam*), *Michael Redd* (*athlete*) e *Madison Square Garden* (*StadiumOrEventVenue*), que compõem $A_c$*(sport, sportsTeam, sportLeague)*, $A_c$*(sport, athlete, sportLeague)* e $A_c$*(sport, StadiumOrEventVenue, sportLeague)*. Isso significa que *Sport* alcança *sportLeague* por três caminhos independentes do triângulo *A*.

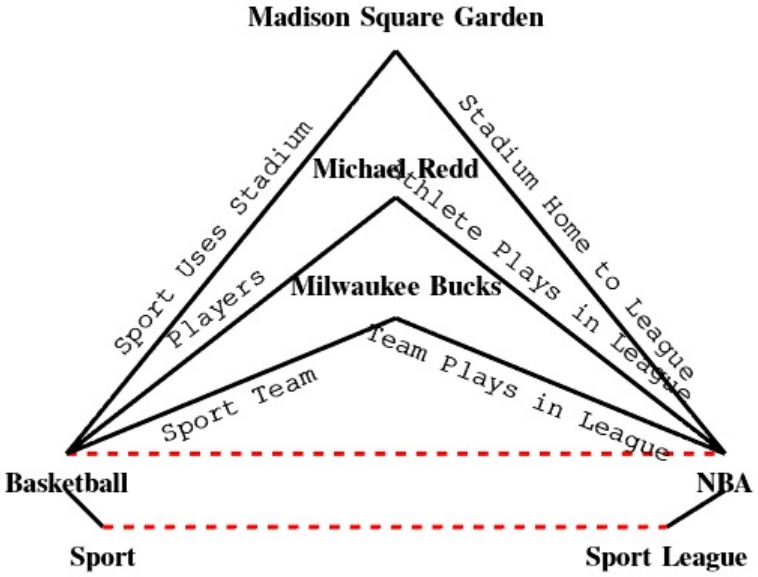


**Figura 3.10: Triângulo aberto *A* com seus três caminhos independentes (*Madison Squase Garden, Michael Redd, Milwaukee Bucks*) e $A_c$*(Sports, SportsLeague)* (APPEL; Hruschka Jr, 2011).**

Os resultados obtidos por (APPEL; Hruschka Jr, 2011), revelam que *Prophet* pode adicionar novo conhecimento a sistemas de ASF (Seção 2.1.2) como o *NELL*. Além de inferir novas relações e adicionar novos fatos a BC, colabora com a autossupervisão do *NELL*, o que permite que o sistema identifique automaticamente possíveis erros na BC (PEDRO; APPEL; Hruschka Jr., 2013). Essa é uma questão importante ao lidar com sistemas de ASF onde erros inseridos podem

ser propagados e gerar desvios de conceitos.

#### 3.4.1.4 Path Ranking Algorithm - PRA

Outro componente que visa expandir a BC de *NELL* é o *PRA (Path Ranking Algorithm)* (LAO; MITCHELL; COHEN, 2011). Como o nome sugere, é baseado em um algoritmo de pontuação de caminho. Tem foco em popular relações, gerar novas instâncias com pares de argumentos inéditos e relações já existentes.

O problema considerado por Lao, Mitchell e Cohen (2011) é de aprender e inferir em uma base de conhecimento de grande escala que possui conhecimento imperfeito e cobertura incompleta. O procedimento de inferência é baseado em uma combinação de passeios aleatórios pontuados (atribuição de pesos) por um grafo de BC — pode ser usado para inferir confiavelmente novas crenças para a base de conhecimento. Mais especificamente, o sistema pode aprender a inferir diferentes relações alvo ao ajustar os pesos associados a passeios aleatórios que seguem caminhos diferentes através do gráfico.

Esse componente (*PRA*) apresenta uma melhora significativa em relação ao método de inferência e aprendizado de cláusulas de Horn anterior do *NELL*: obtém aproximadamente o dobro de precisão para as 100 inferências de maior pontuação (*top 100*), e além disso, é aplicável a muitas outras tarefas de inferência. O método de inferência treinável (do *PRA*) que aprende a inferir relações combinado com os resultados de diferentes passeios aleatórios por este grafo consegue boas propriedades de escala e de inferência robusta em uma BC que contém mais de 500.000 triplas extraídas da Web pelo sistema *NELL* (BC de *NELL* até então).

A abordagem deste trabalho de mestrado é complementar ao *PRA* na tarefa de expansão da BC de *NELL*: enquanto Lao, Mitchell e Cohen (2011) visa popular relações já existentes com novos argumentos (instâncias de categorias), este trabalho tem foco (assim como o *Prophet* e o *OntExt*) em descobrir novas relações para as instâncias de categorias, como apresentado no capítulo seguinte (4).

#### 3.4.1.5 PIDGIN

NELL também tem um método robusto, escalável e flexível baseado em grafo para adquirir conhecimento (de categorias e relações) a partir do alinhamento de duas Bases de Conhecimento: PIDGIN.

Algoritmos para alinhar ontologias de estado-da-arte até então não conseguem alinhar Bases de Conhecimento especialmente caso estas não possuam dados em comum. Para superar

este problema de esparsidade de dados, PIDGIN emprega o uso de um corpus robusto de informações Sujeito-Verbo-Objeto (SVO) obtidas a partir de textos de linguagem natural. Neste algoritmo, o problema de alinhamento é encarado como um problema de classificação de grafos propriamente estruturados. O sistema consiste de duas etapas:

1. Construção do grafo: Dados duas Bases de Conhecimento e um corpus no formato Sujeito-Verbo-Objeto (SVO), PIDGIN estrutura estes dados como um grafo.

2. Alinhamento como classificação dos nós do grafo: Com o grafo construído na Etapa 1, PIDGIN alinha as ontologias ao classificar os nós do grafo.

Em (WIJAYA; TALUKDAR; MITCHELL, 2013) são apresentados resultados de experimentos e a superioridade de PIDGIN sobre PARIS — a abordagem de estado-da-arte para alinhamento de ontologias até então. PIDGIN é tipicamente capaz de melhor a cobertura sobre PARIS sem degradar a precisão, além de aprender automaticamente quais verbos estão associados com quais relações da ontologia. Estes verbos podem ser usados para extrair novas instâncias para popular a BC ou identificar relações entre as entidades nos documentos.

# Capítulo 4

## NEW ONTOLOGY EXTENSION

Este capítulo apresenta o *New Ontology Extension — newOntExt*, uma nova abordagem e com uma nova implementação, toda em Java. A inspiração para esta nova abordagem é o antigo subsistema componente do NELL para extensão da ontologia (OntExt) e nesta nova abordagem/implementação é possível aplicar a extensão da ontologia a partir de qualquer corpus de texto e da própria ontologia. A finalidade desta proposta é disponibilizar uma abordagem plausível para a geração contínua de novas relações para a BC de NELL. Essa contribuição para o aprendizado do sistema do projeto *Read The Web* utiliza técnicas de Leitura de Máquina e expansão de ontologia (ver Capítulo 2; especificamente Seções 2.4 e 2.5). O newOntExt apresenta as seguintes principais contribuições (em relação ao OntExt anterior): i) utiliza o processo de recuperação de informação (a partir do corpus textual) com base em algoritmos que são estado-da-arte; ii) possui um novo algoritmo, o qual foi implementado com vários novos recursos de pré-processamentos e otimização do código; iii) foi integrado ao *Prophet* permitindo que informação semântica (nome das relações) seja inserida no modelo baseado em grafos; e iv) foi adaptado para que resultados inconsistentes possam servir de alerta para o NELL realizar correções/revisões de sua base de conhecimento.

A proposta tem como objetivo o aprendizado de novas informações relacionais entre instâncias de categorias já conhecidas e segue a definição do problema dada ao final da Seção 1.2. Para cada frase do corpus de entrada, newOntExt procura tanto pelo sujeito como pelo objeto da frase na Base de Conhecimento (BC). Caso ambos os argumentos (sujeito e objeto) existam na ontologia, esta frase é considerada na contagem de frequência para a montagem das matrizes de co-ocorrência e posterior agrupamento das matrizes para gerar novas relações.

Realizar este processamento para todas as tuplas de um corpus consideravelmente grande é uma tarefa muito custosa e demorada. Ao mesmo tempo, como o método visa a contagem de sentenças frequentemente afirmadas, não faria sentido utilizar corpora muito menores e poucas

instâncias de conhecimento como guia para o processo. Desta forma, fez-se necessário construir estratégias para superar estes desafios. O Capítulo 5 apresenta maiores detalhes acerca dos experimentos feitos para superar as dificuldades desta tarefa.

A próxima Seção (4.1) apresenta os passos para realização da metodologia tradicional para expansão da ontologia. Na sequência, a Seção 4.2 apresenta os novos recursos em relação ao OntExt para otimizar esta tarefa; e, na Seção 4.3 é apresentada uma metodologia de colaboração entre newOntExt e outros subsistemas componentes do NELL — no caso dos experimentos apresentados, o subsistema componente Prophet encontra que existem possíveis relações entre certas categorias e, a partir destes dados, newOntExt se guia para a identificação, validação e nomeação destas relações.

Além dos novos recursos e novas estratégias para guiar o aprendizado, a nova implementação tem foco na otimização da execução. Quanto a isso, deve-se atentar para não construir um número muito grande de objetos desnecessariamente, pois a coleta de lixo (*garbage collection*) pode aumentar consideravelmente o tempo de processamento. A reutilização de objetos em laços de repetição (*loops*) melhora o desempenho de forma significativa em relação à construção de novos objetos, o que também reduz o uso da memória *Heap* e custo da coleta de lixo do Java dependendo do tempo de vida destes objetos. Para os experimentos mais recentes, os caminhos das categorias de interesse estão armazenados em um *Array*, que é mais rápido do que uma instanciação de *Collection* para se percorrer.

## 4.1 Metodologia tradicional

### 4.1.1 Pré-processamento

A abordagem aqui proposta, considera a *extração de informação aberta* da Web para a geração de novas relações para as instâncias de categorias já existentes na BC do *NELL*. Para esta tarefa de extração, os sistemas *ReVerb/R2A2* (apresentados nas Seções 3.3.2 e 3.3.3, respectivamente) são os protagonistas e devem ser utilizados para trazer ganhos à metodologia definida no componente *OntExt*.

Para tanto, fez-se necessário desenvolver um script que tenha como entrada um arquivo com o formato do corpus de entrada, e, que gere como saída um arquivo com o formato de entrada esperado por *ReVerb/R2A2*. Este script modifica cada sentença da seguinte forma: (1) insere-se um contador de sentenças no começo de cada uma (a primeira sentença terá início com o número 1, a segunda com o 2, e assim por diante); e (2) eliminam-se etiquetas morfossintáticas

(desnecessárias para o próximo processamento), o que resulta no seguinte formato aceito por *ReVerb/R2A2* como entrada:

```
<número da sentença><TAB><palavra | número | caracter de pontuação>
    [<SPACE><palavra | número | caracter de pontuação>]*
```

### 4.1.2 Extração e Geração de Relações

Terminada esta etapa de pré-processamento na qual os arquivos de entrada para o processamento de *ReVerb/R2A2* são gerados, pode-se então realizar a *extração de informação* do corpus com a execução do sistema. Feito isso, tem-se os arquivos de saída com as extrações relacionais realizadas pelo *ReVerb/R2A2*.

O conjunto de dados disponibilizado pelos pesquisadores do projeto *KnowItAll* (FADER; SODERLAND; ETZIONI, 2011) é uma das principais bases de entrada para este trabalho. Este corpus é composto de extrações realizadas pelo *ReVerb* a partir do conjunto de dados *ClueWeb09* (ambos os conjuntos de dados citados são melhores especificados nos apêndices B e A, respectivamente). Este conjunto de dados é a entrada para o processamento do gerador de relações. Isso torna desnecessárias as outras duas etapas previstas (descritas acima) para este experimento.

Para futuros experimentos, a fase de extração tem possibilidade de envolver o processamento de *R2A2* a partir de *ClueWeb09* ou algum outro corpus mais atualizado (com tamanho, cobertura e precisão suficientemente satisfatórios).

Então, com as extrações geradas já disponíveis, é necessário fazer um tipo de fusão das informações geradas por um dos sistemas do projeto *KnowItAll* com as da BC de *NELL*. Para isso, considerando apenas relações binárias, do tipo *(sujeito, frase_verbal, objeto)*, newOntExt procura por relações que ocorrem com os mesmos sujeito e objeto (arg1 e arg2).

Assim, o algoritmo busca por novas relações com base nas instâncias (considerando a categoria Cidade, por exemplo: *Pittsburgh*, *New York*, *Boston*, *Dallas*, entre outros). Com essa finalidade, usa-se todas as instâncias que o *NELL* já conhece para a categoria "*City*" (cidade), por exemplo, e utiliza-se destes sintagmas nominais como sendo sujeito (arg1) e testa-se todos estes sujeitos com instâncias de todas as outras categorias. O trecho do algoritmo referente a isso, em um nível alto, seria como:

```
para i = 1 até numeroDeCategoriasNaOntologiaDoNell
  para j =1 até numeroDeCategoriasNaOntologiaDoNell
    para k =1 até numeroDeInstânciasNaCategoria(i)
      para l =1 até numeroDeInstânciasNaCategoria(j)
        procurar nas extrações de ReVerb/R2A2(instância(k).categoria(i),
                              padrão de contexto, instância(l).categoria(j);
      fim-para
    fim-para
  fim-para
fim-para
```

Feito isso, a sequência do trabalho envolve uso de algum algoritmo de agrupamento para identificar grupos. A abordagem deste agrupamento é baseado nos relatos de Mohamed, Hruschka Jr. e Mitchell (2011): são construídas matrizes de co-ocorrência de relações [*frases verbais X frases verbais*] para cada par de categorias; as células são preenchidas com o número de vezes que as relações de tal linha e tal coluna co-ocorreram com as mesmas instâncias (argumentos). Então, as matrizes são normalizadas: divide-se o valor da célula pela contagem total de valores da linha (como apresentado na fórmula 3.1). Contextos que co-ocorrem com apenas alguns contextos têm maior peso, para que contextos menos genéricos sejam promovidos — Equação 3.2 (MOHAMED; Hruschka Jr.; MITCHELL, 2011).

A continuidade do trabalho envolve agrupar os contextos co-ocorrentes relacionados a partir das informações armazenadas nas matrizes. Para esta tarefa, é utilizado agrupamento pelo algoritmo *K-médias* da ferramenta *Weka* (HALL et al., 2009). Cada grupo identificado em cada matriz corresponde a uma nova possível relação entre as categorias em questão.

Como já mencionado, a geração de relações é baseada em *OntExt* (MOHAMED; Hruschka Jr.; MITCHELL, 2011), mas se utiliza dos algoritmos *ReVerb/R2A2* para a extração de relações candidatas. Instâncias sementes para as relações a serem propostas são geradas. Instâncias que se relacionam por contextos que correspondem ao centróide[1] do agrupamento ou próximos deste são melhores representantes da relação. A força da instância semente é inversamente proporcional ao desvio padrão do contexto em questão em relação ao centróide do agrupamento; e, diretamente proporcional ao número de vezes que esta co-ocorre com o contexto (como explicado na Seção 3.4.1.2 e formalizado na fórmula 3.3).

[1] Relação no centro do agrupamento, melhor situada, com maior pontuação.

### 4.1.3 Classificação de Relações Válidas e Avaliação de Resultados

Para classificar as relações geradas, a mesma abordagem utilizada para os resultados de OntExt (MOHAMED; Hruschka Jr.; MITCHELL, 2011) é empregada neste trabalho. Mais especificamente, uma relação gerada é considerada incorreta (por humano) se há:

1. Ambiguidade semântica: caso as instâncias pertencentes a uma ou ambas as categorias envolvidas na relação são ambíguas e não fazem sentido no contexto da relação. Por exemplo: *insect-such_as-animal* (inseto como animal), relação extraída pelo OntExt.

2. Erro de classificação de instância: caso uma ou ambas as instâncias envolvidas estão erroneamente associadas à(s) respectiva(s) categoria(s). Por exemplo, *animal-using-animal* (animal usando animal), relação extraída pelo OntExt.

3. Informação incompleta semanticamente: caso a relação necessite de maiores informações para fazer sentido semântico. Por exemplo *arthropod-can_be_use_instead_of-mollusk* (artrópode pode ser utilizado no lugar de molusco), relação extraída pelo newOntExt.

4. Lógica incorreta: caso simplesmente a relação não faça sentido lógico. Por exemplo, *animal-be_a_lovely_alternative_to-mollusk*, (animal ser uma amável alternativa para molusco), relação extraída por newOntExt.

Feita a classificação de relações válidas e inválidas por humanos, é feito o cálculo para obtenção da Precisão ($P$) através da fórmula:

$$P = \frac{Resultados\ Positivos\ Verdadeiros}{(Resultados\ Positivos\ Verdadeiros + Resultados\ Positivos\ Falsos)} \tag{4.1}$$

que para o caso deste trabalho equivale a

$$P = \frac{RV}{(RV + RI)} \tag{4.2}$$

onde $RV$ é o número de relações válidas e, $RI$, o número de relações inválidas.

Com este resultado e com a contagem do total de relações geradas, válidas e inválidas, pode ser feita, então, a comparação com o OntExt[2].

[2]O cálculo da cobertura não se aplica a este trabalho já que este envolve falsos negativos — newOntExt não gera relações com intenção prévia de que sejam inválidas ou negativas; tem foco apenas nas positivas, que possam agregar conhecimento à ontologia.

### 4.1.4 Geração Sem Fim de Relações

Quando suficientemente estável para a tarefa, espera-se integrar este sistema de geração de relação ao *NELL*, para iterativamente estender a ontologia do *NELL* em um fluxo contínuo de novas tarefas de aprendizagem. Depois de cada conjunto fixo de iterações, a BC crescente é um dos recursos para o sistema de geração de relação que, por sua vez, alimentaria o *NELL* com novos fatos relacionais. Com isso, a proposta apresenta-se como uma contribuição com grande potencial para o Aprendizado Sem Fim (ASF) com técnicas de Leitura de Máquina (LM) para expansão da ontologia que representa a Base de Conhecimento (BC).

## 4.2 Novos recursos

Conforme introduzido no início deste Capítulo (4), newOntExt possui novos recursos para melhorar o desempenho da tarefa de expansão de ontologia, cujos passos estão descritos na Seção anterior 4.1. A seguir, estes recursos são apresentados e descritos.

### 4.2.1 Pré-processamento da Base de Conhecimento (BC)

A fim de otimizar a tarefa de percorrer todas as instâncias de categorias da Base de Conhecimento (BC) do NELL, foi realizado um pré-processamento que percorre toda a ontologia que representa a BC separando as instâncias por categorias de conhecimento. Para cada categoria existe um arquivo que contém as instâncias pertencentes a esta categoria, uma instância por linha. Para ilustrar, eis um trecho com as 30 primeiras instâncias, uma por linha como no arquivo utilizado, que constam no arquivo referente à categoria pessoa, *person.txt*:
d__mehmet_do_an
czeslaw_rzepinski
michele_gregory
neil_mccallum
albentosa
john_fank_brown
mussorgsky
alexander_von_der_marwitz
of_monsters_and_men
james_sutorius
stephan_bender

katharina_tueschen
jaron_lowenstein
duke_of_marlborough
jerome_strohkirch
alexandre_d_arcy
suzanna_hamilton
alessio_perilli
kevin_sherrington
alex_vance
alex_vandi
anna_simpson
alysha_anderson
giuseppe_abbati
alfred_zacharias
kathy_boudin
arthur_hacker
jonathan_bogner
michael_d_rogers
lawrence_t__pileggi

Como se pode observar, as palavras referentes a cada Sintagma Nominal (SN) são separadas por *underline* (_), da forma como estão na BC do NELL. E, existem ruídos: por exemplo, *of_monsters_and_men* certamente não é uma pessoa.

A BC utilizada nos experimentos descritos no Capítulo 5 é a BC do NELL na iteração 656. Esta BC possui 241 categorias, e assim, com este pré-processamento da BC, foram criados 241 arquivos, todos no formato <*categoria*>.txt contendo as respectivas instâncias de tais categorias.

### 4.2.2 Organização Das Extrações

Originalmente, as extrações feitas pelo sistema de Extração de Informação Aberta (EIA) ReVerb utilizando como entrada o corpus de páginas web ClueWeb09 foram disponibilizadas em um único arquivo, contendo mais de 14 milhões de extrações. Nossa primeira implementação do gerador de relações percorria todas estas extrações de forma sequencial, procurando por relações com argumentos (sujeito e objeto) já conhecidos pelo NELL (já existentes na Base de Conhecimento). O processamento estava tomando dias e não dava indícios de que se encerraria

logo. Então, este arquivo foi dividido em diversos outros. Considere tuplas no formato *(sujeito, frase_verbal, objeto)*. Para cada sujeito é criado um arquivo com as relações deste. Assim, para cada sujeito já conhecido busca-se diretamente o arquivo referente com as relações de interesse. No entanto, sistemas operacionais não suportam que uma pasta tenha dezenas de milhares ou milhões de arquivos nela. Para resolver este problema, as pastas foram organizadas da seguinte forma:

- Pasta "raiz": ReverbExtractions/
- Subpastas, cada uma nomeada com uma letra do alfabeto (por exemplo, ReverbExtractions/a/, ReverbExtractions/b/, ReverbExtractions/c/ etc.)
- Sub-subpastas: cada subpasta contém sub-subpastas com a letra do alfabeto da subpasta acrescido das letras do alfabeto também (por exemplo, ReverbExtractions/a/a, ReverbExtractions/a/b, ReverbExtractions/a/c ( ... ), ReverbExtractions/b/a, ReverbExtractions/b/b, ReverbExtractions/b/c, etc.)
- Sub-sub-subpastas: a mesma subdivisão baseada nas letras do alfabeto é novamente utilizada (por exemplo, ReverbExtractions/a/a/a, ReverbExtractions/a/a/b ReverbExtractions/a/a/c, ( ... ), ReverbExtractions/b/a/a, etc.)
- Cada arquivo referente a cada arg1 existente nas extrações será criado no caminho referente às suas letras iniciais (por exemplo, para o arg1 "banana", o caminho seria: ReverbExtractions/b/a/n/banana.txt
- Caracteres especiais e espaços foram desconsiderados para efeito de nomes de diretórios (ou pastas). Assim, para os arg1's: "a cidade", "1 maior" e "#n2 casa" seriam gerados os seguinte arquivos, armazenados nos diretórios: a/c/i/a_cidade.txt, m/a/i/1_maior.txt e n/c/a/_n2_casa.txt.
- Para efeito de nomeação de arquivos, o caractere "/" causaria problemas; ele foi substituído por --" (por exemplo, o arg1 "a bar/bat mitzvah" está no arquivo "a bar–bat mitzvah.txt").

### 4.2.3 Divisão e Conquista

Encontrar todas as informações relacionais com argumentos já existentes na Base de Conhecimento (BC) requer muito tempo de processamento, visto que o corpus de entrada deve

ser muito grande, contendo milhões (ou talvez bilhões) de tuplas com informações relacionais. Esse é o grande motivo para a dificuldade encontrada para obtenção de resultados.

Este processamento necessita de um servidor muito robusto. Neste trabalho, um servidor do projeto Read the Web (RTW) foi utilizado remotamente, situado na Carnegie Mellon University, em Pittsburgh, PA - EUA. Mesmo com a disponibilidade deste servidor foi encontrada grande dificuldade para obtenção de resultados: os arquivos de entrada são realmente grandes e outros pesquisadores do projeto também utilizam este servidor para tarefas de semelhante custo computacional.

Considerando um total de 4.597.174 instâncias de 241 categorias da BC do NELL na iteração 656, para reconhecer sentenças com sujeito e objeto conhecidos (instâncias combinadas 2 a 2) e considerando também a fórmula matemática para combinação,

$$\binom{n}{k} = \frac{n!}{k!\ (n-k)!} \tag{4.3}$$

tem-se:

$$\binom{4597174}{2} = \frac{4597174!}{2! \times 45971722!} = 10567002094551 \approxeq 1 \times 10^{13}. \tag{4.4}$$

Então, para cada um dos principais corpora de entrada utilizados neste trabalho, tem-se os seguintes totais de comparação:

- RCE 1.1: total de 14.728.268 tuplas. Total de comparações:

$$10567002094551 \times 14728268 \approxeq 1,5 \times 10^{20} \tag{4.5}$$

- SVO: total de 604.934.719 tuplas.

$$10567002094551 \times 604934719 \approxeq 6,4 \times 10^{21} \tag{4.6}$$

Para superar este aspecto, uma tentativa de processamento paralelo foi inciada com métodos de MapReduce do Hadoop. Mas, esta implementação não foi finalizada, dada a dificuldade de acesso a grandes clusters realmente poderosos, eficientes e confiáveis; e, o tempo estimado para se gerar um código realmente plausível para tal processamento.

Então, soluções computacionalmente mais elegantes e eficientes para a terefa foram buscadas, como alguma metodologia que envolvesse os conceitos de "Divisão e Conquista", na qual se realizaria o processamento por partes, e, ao fim, o resultado final seria obtido da junção das saídas. A primeira tentativa envolve divisão das categorias em grupos, isto é, para cada

iteração de leitura da entrada de informação relacional são utilizados grupos de categorias para verificação de instâncias de categorias (argumentos) conhecidos.

Para os primeiros experimentos com subgrupos de categorias, o foco consiste nas categorias pessoa, cidade, esporte e relacionadas a esporte. Outra estratégia envolve as categorias mais populadas da BC do NELL, isto é, categorias que possuem maior número de exemplos (instâncias), assim, há uma maior chance numérica de reconhecer sentenças com sujeitos e objetos conhecidos.

Os experimentos têm foco em subgrupos de conhecimento, como descrito no trabalho de Dos Santos e Hruschka Jr (2014). Assim, os experimentos foram conduzidos com subconjuntos de categorias temáticos (itens no formato <*categoria-tema*>*:* <*lista-de-categorias-envolvidas*>):

- animal: animal, molusco, inseto, réptil, mamífero, artrópode.
- construção: recurso de construção, material de construção, atração turística, cidade.
- doença: bactéria, condição fisiológica, droga, emoção, planta, doença, pessoa, roupa.
- esporte: atleta, treinador, estádio, time esportivo, troféu de torneio, posição em time esportivo, liga esportiva, jogo esportivo.

Para exemplificar este processamento com subgrupos, considere o subconjunto que envolve as categorias com animais. A categoria animal possui 61023 instâncias; a categoria molusco, 7904 instâncias; a categoria inseto, 8106; a categoria réptil, 5212 instâncias; a categoria mamífero, 6373 instâncias; e a categoria artrópode, 14147 instâncias. Assim este subconjunto envolve um total de 102765 instâncias. A seguir, cálculos para este subgrupo considerando o processamento de todo o SVO de uma vez são apresentados: a equação 4.7 apresenta o total de combinações entre as instâncias deste subgrupo e a Equação 4.8 apresenta o total de comparações ().

$$\binom{102765}{2} = \frac{102765!}{2! \times 102763!} = 5280271230 \approxeq 5,3 \times 10^9. \tag{4.7}$$

$$5280271230 \times 604934719 \approxeq 3,2 \times 10^{18} \tag{4.8}$$

A partir desta estratégia de utilizar uma amostra do corpus de entrada, surge a estratégia de Divisão e Conquista. A partir de testes empíricos e estudo da capacidade e uso do servidor utilizado, o SVO foi dividido em 23 partes iguais, cada uma com 26 milhões de tuplas,

mais uma parte com o restante (6.934.719 tuplas). Para efeitos comparativos, eis o cálculo das combinações para cada uma destas partes:

$$10567002094551 \times 26000000 \approxeq 2,75 \times 10^{20}. \tag{4.9}$$

Os processos podem ser executados em paralelo, assim, ao comparar com o número total de comparações do SVO completo, é um custo computacional de aproximadamente 4,3% (em condições ideais) do processamento total do SVO de uma vez.

Se combinadas as estratégias descritas acima, isto é, com foco em um subconjunto de categorias, por exemplo categorias relacionadas a animal, e a separação do SVO por partes, tem-se um custo computacional de aproximadamente 0,02% [3] comparado ao processamento de toda a BC e todo o SVO de uma vez (em condições ideais). O valor absoluto de comparações para o processamento do subconjunto relacionado a animal e com o SVO particionado é apresentado na Equação 4.10.

$$5280271230 \times 26000000 \approxeq 1,37 \times 10^{17}. \tag{4.10}$$

Para maiores detalhes acerca dos experimentos, ver Capítulo 5.

## 4.3 Nomeação de relações candidatas com newOntExt

O subsistema newOntExt tem limitações significantes para expandir a ontologia inteira em uma iteração. No entanto, ao utilizar algumas estratégias para guiar o aprendizado, a metodologia para expansão da ontologia funciona. Uma das formas de reduzir o escopo de conhecimento a ser aprendido é com colaboração de outro subsistema componente do NELL — o Prophet.

Pelas características do grafo que representa a BC do NELL, Prophet reúne exemplos de possíveis novas relações, ainda não nomeadas, entre categorias de conhecimento. Por exemplo, no último experimento, Prophet identificou que há uma possível relação entre álbuns musicais e formas de arte visual pelas características do grafo que representa a BC do NELL.

Esta indicação de relacionamentos entre pares de categorias ainda inexistentes na ontologia serve como guia para newOntExt buscar por sentenças contendo instâncias destas respectivas categorias em grandes fontes de texto. Desta forma, newOntExt funciona como um validador deste processo. Caso não encontre dados suficientes para construção das matrizes

[3] Porcentagem exata: 0,02140625%.

de co-ocorrência, newOntExt invalida esta relação indicada pelo Prophet, isto é, esta possível relação entre estas categorias não é plausível o suficiente para ser uma crença na BC. Mas, caso newOntExt construa matrizes de co-ocorrência e faça o agrupamento destas com sucesso, novas relações são validadas e nomeadas apropriadamente. Para os experimentos descritos no próximo Capítulo, o número de agrupamentos utilizado foi 2 a fim de agrupar dois sentidos diferentes de relações entre as categorias. Por exemplo, para o par de categorias citado, álbuns musicais e formas de arte visual, as frases verbais escolhidas pelo newOntExt para representar as novas relações foram "tem" e "foca em", isto é, uma indica que alguns álbuns musicais possuem formas de artes visuais, mas não têm foco principal nisso; já a outra indica que o foco de alguns álbuns musicais têm foco em formas de artes visuais, com encartes bem trabalhados.

# Capítulo 5

## EXPERIMENTOS

Como mencionado no Capítulo 4, estratégias para extensão de ontologia baseadas em informação redundante requerem o processamento de uma grande quantidade de informação. Como o tempo necessário para obter resultados a respeito da metodologia tradicional como um todo utilizando toda a Base de Conhecimento (BC) se torna proibitivo, estratégias alternativas devem ser adotadas pelo NELL a fim de obter resultados práticos que possam contribuir efetivamente para a sequência do aprendizado sem-fim. Desta forma, os experimentos descritos neste capítulo também seguem estas estratégias alternativas (que se mostram viáveis e adequadas para a utilização prática no sistem NELL).

Para os experimentos descritos, a BC considerada é a BC do NELL até a iteração 656. Esta ontologia foi escolhida pois é robusta o suficiente para experimentos. Atualmente, a BC do NELL está disponível até a iteração 873. Quanto mais recente é a BC (isto é, quão maior o número da iteração desta), mais confiável e repleta de exemplos ela é — possui maior número de crenças; porém, mais demorado é o processamento.

Descrições a respeito de cada experimento realizado e respectivas análises de resultados são apresentados nas seções que seguem.

## 5.1 Experimentos com a metodologia tradicional e escopo reduzido

As subseções que seguem descrevem cada experimento realizado para expansão de ontologia. Os nomes indicam o subconjunto da ontologia utilizada (isto é, as categorias da BC do NELL na iteração 656) .

Os experimentos conduzidos (Subsubseções 5.1.1, 5.1.2 e 5.1.3) focam em subgrupos temá-

| | | |
|---|---|---|
| Possíveis crenças | 22 | 40,74% |
| Relações incorretas | 32 | 59,26% |
| Total de relações geradas | 54 | 100% |

**Tabela 5.1: Resumo das relações geradas com experimentos com subgrupos de categorias.**

ticos de conhecimento, como citado na Subseção 4.2.3. Estes experimentos tem como entrada o corpus RCE 1.1. A Tabela 5.1 apresenta um resumo geral dos experimentos com esta metodologia e estratégia: mais da metade dos resultados gerados são candidatos a crenças adicionáveis à BC. Os resultados em sua totalidade estão apresentados no apêndice, no Capítulo E.3.1 — todas as relações geradas e classificadas para estes experimentos.

Para cada experimento, uma tabela com um resumo geral é apresentado, bem como um exemplo positivo (possível crença — informação relacional válida) e um negativo; além da Precisão ($P_a$ para o subconjunto relacionado a animal, $P_c$ para o subconjunto relacionado a construção e $P_e$ para o subconjunto relacionado a esporte), calculada segundo a Equação 4.2 exposta na Subseção 4.1.3. Considerando os valores apresentados na Tabela 5.1, segue o cálculo da Precisão geral ($P_g$) destes experimentos:

$$P_g = \frac{RV}{RV + RI} = \frac{22}{22+32} \approx 41\%. \tag{5.1}$$

### 5.1.1 Com subconjunto de categorias relacionadas a animal

Neste experimento com foco no subconjunto de categorias de conhecimento relacionadas a animal são consideradas as categorias animal, molusco, inseto, réptil, mamífero e artrópode. Como resultados, tem-se um total de 30 relações geradas, sendo 13 consideradas corretas e 17 incorretas.

Um exemplo de relação correta gerada é: "artrópode pode ser muito irritante para mamífero" (ou, no formato que o newOntExt gera: "*arthropod-can_be_very_irritating_to-mammal*"), como é caso de pernilongos com humanos, que, apesar de serem muito menores no tamanho, podem perturbar períodos de descanso ou de concentração dos seres do reino animal com maior poder de mudar o planeta e seus esquemas e sistemas de funcionamento. Como exemplo real gerado por newOntExt, tem-se o par *{"flea", "dog"}*, que indica que a pulga pode ser irritante para um cachorro.

Por outro lado, um exemplo de conhecimento incorreto gerado é: "artrópode pode ser usado em vez de molusco" (ou, no formato que o newOntExt gera "*arthropod-can_be_use_instead_of-*

| | | |
|---|---|---|
| Possíveis crenças | 15 | 50% |
| Relações incorretas | 15 | 50% |
| Total de relações geradas | 30 | 100% |

**Tabela 5.2: Resumo das relações geradas com subgrupo de categorias relacionadas a animal.**

| | | |
|---|---|---|
| Possíveis crenças | 2 | 25% |
| Relações incorretas | 6 | 75% |
| Total de relações geradas | 8 | 100% |

**Tabela 5.3: Resumo das relações geradas com subgrupo de categorias relacionadas a construção.**

*mollusk*"). Este exemplo está incompleto semanticamente pois, talvez até um artrópode possa ser usado no lugar de um molusco para certa aplicação, no entanto, falta esta informação.

$$P_{\text{a}} = \frac{RV}{RV + RI} = \frac{15}{15+15} = 50\%. \tag{5.2}$$

### 5.1.2 Com subconjunto de categorias relacionadas a construção

Para o subgrupo de interesse relacionado a construção são consideradas as categorias: recurso de construção, material de construção, atração turística e cidade. Como a Tabela 5.3 descreve, de um total de 8 resultados gerados, 5 são possíveis crenças e 3 são incorretos.

Como exemplo de resultado correto, considere este resultado: como consta na saída de newOntExt "*attraction-be_fall_in-city*". Neste contexto, a expressão "falls in" pode ser traduzida como "fica em". Assim, a relação pode ser entendida como "atração fica na cidade", o que é semanticamente e logicamente válido.

Como exemplos negativos ou incorretos, tem-se a maioria das relações geradas por conta de instâncias mal classificadas, indicadas pertencentes a categorias que na verdade não pertencem. Por exemplo, a relação "cidade é a cidade do material de construção" (ou, como consta na saída de newOntExt "*city-be_the_city_of-buildingmaterial*") poderia ser considerada verdadeira semanticamente e logicamente, se não fosse pelas instâncias de exemplo geradas: sujeitos que não são cidades e/ou objetos que não são materiais de construção. Isso indica um alerta para supervisão e correção da Base de Conhecimento (BC) nesta área do grafo.

$$P_{\text{c}} = \frac{RV}{RV + RI} = \frac{2}{2+6} = 25\%. \tag{5.3}$$

| | | |
|---|---|---|
| Possíveis crenças | 5 | 31,25% |
| Relações incorretas | 11 | 68,75% |
| Total de relações geradas | 16 | 100% |

**Tabela 5.4: Resumo das relações geradas com subgrupo de categorias relacionadas a esporte.**

### 5.1.3 Com subconjunto de categorias relacionadas a esporte

Para este experimento com o subconjunto das categorias relacionadas a esporte são consideradas apenas: liga esportiva, esporte, atleta e time esportivo.

Para exemplificar um exemplo de relação positiva com esportes, considere ("*athlete-fly_out_to-sportsteamposition*") que literalmente pode ser traduzido para "atleta voa para a posição do time esportivo", onde "voa para" pode ser entendido como uma mudança repentina, como, por exemplo, o caso de um atleta que anteriormente atuava como defensor em um time, e, ao mudar-se de time, mudou também de posição e atua agora no ataque.

Como resultado incorreto, tem-se a afirmação gerada "atleta pode ser jogado em esporte" ("*athlete-can_be_play_at-sport*") que simplesmente não tem sentido semântico e lógico completo.

$$P_{\mathrm{e}} = \frac{RV}{RV + RI} = \frac{5}{5+11} = 31,25\%. \tag{5.4}$$

## 5.2 Comparações com OntExt

Para efeitos comparativos, a partir do total de 781 novas relações geradas pelo OntExt (MOHAMED; Hruschka Jr.; MITCHELL, 2011), são aplicados filtros de acordo com os subgrupos de categorias de foco neste trabalho: um subconjunto das categorias relacionadas à animal, um subconjunto das categorias associadas à construção e outro a esporte (apresentados na Seção 5.1). Assim, neste Capítulo é comparado o número de relações geradas para estes subgrupos por cada subsistema (OntExt e newOntExt).

Para os três subconjuntos de categorias de foco, OntExt gerou 10 relações, todas julgadas incorretas. Por outro lado, newOntExt gerou 54 relações, sendo 22 delas corretas. Assim, mesmo com uma precisão relativamente baixa (41%), newOntExt consegue gerar conhecimento em áreas do grafo que OntExt não conseguiu.

Esta baixa precisão é completamente condizente com o Aprendizado Sem Fim (ASF). Sistemas baseados no ASF precisam de crenças confiáveis para não haver ruídos e desvios de

| | OntExt | newOntExt |
|---|---|---|
| Relações incorretas | 10 | 32 |
| Relações corretas | 0 | 22 |
| Total de relações geradas | 10 | 54 |
| Precisão | 0% | 41% |

**Tabela 5.5: Resumo compartivo das relações geradas pelo OntExt e pelo newOntExt para os subgrupos de categorias relacionados a animal, construção e esporte.**

conceitos. Assim, estes sistemas podem evoluir de forma vagarosa porém confiável e consistente. O que leva a conclusão de que newOntExt pode contribuir para o aprendizado do NELL. E quando a precisão for muito baixa, o processamento indica uma supervisão necessária no subconjunto de categorias em foco.

## 5.3 Experimentos em colaboração com Prophet

Como abordado na Seção 4.3, newOntExt pode atuar na expansão da BC com a colaboração de outro subsistema componente: Prophet. No último processamento feito, 12.436 possíveis relações entre categorias foram identificadas e ordenadas segundo pontuação feita pelo próprio Prophet. Destas, algumas foram selecionadas para validação e possível nomeação conforme descrições nas subseções a seguir.

### 5.3.1 Experimento com relações que envolvem esporte do Prophet

De todas as relações encontradas pelo Prophet, três relações foram escolhidas para realizar este primeiro experimento. Uma das relações tem uma instância relacionando *hobby* com *hobby* (sujeito: *fishing*; objeto *fishing*), uma instância relacionando *hobby* com *sport* (*fishing* com *golf*). O restante das instâncias encontradas (122) das três relações relacionam *sport* com *sport* (total de 144 instâncias).

Este experimento utilizou como entrada o conjunto de dados SVO (ver apêndice, Capítulo C), assim, tinha a disposição praticamente 605 milhões para encontrar sentenças de interesse. Com isso, três relações foram validadas, propostas e nomeadas: a primeira ilustra esportes serem similares uns com os outros; a segunda, esportes serem similares a hobbies; e a terceira, hobbies serem similares uns com os outros. Estas relações (em negrito) e suas instâncias (pares de instâncias destas categorias, entre parênteses) são apresentadas abaixo:

***sport-like-sport***: (baseball, football), (fishing, fishing), (fishing, golf), (basketball, basketball), (golf, basketball), (cricket, baseball), (football, football), (golf, bowling), (football,

| | |
|---|---|
| Relações do Prophet consideradas | 3 |
| Total de possíveis agrupamentos | 6 |
| Relações do Prophet invalidadas por newOntExt | 0 |
| Agrupamentos gerados | 3 |
| Relações geradas válidas | 3 |
| Relações nomeadas incorretamente | 0 |
| Precisão | 100% |

**Tabela 5.6: Resumo das relações geradas com as relações relacionadas a Esporte do Prophet.**

basketball), (rugby, football), (skiing, skiing), (football, baseball), (bowling, fishing), (boxing, football), (football, soccer), (soccer, football), (baseball, tennis), (sports, hockey), (hockey, football), (tennis, golf), (tennis, football), (tennis, soccer), (basketball, football).

***sport-like-hobby***: (fishing, fishing), (basketball, basketball), (fishing, golf), (golf, fishing), (basketball, golf), (golf, basketball), (football, football), (cricket, baseball), (golf, bowling), (rugby, football), (bowling, golf), (skiing, skiing), (basketball, football), (football, baseball), (bowling, fishing), (football, basketball), (fishing, bowling), (baseball, football), (football, rugby), (football, boxing), (boxing, football), (soccer, football), (baseball, tennis), (football, soccer), (sports, hockey), (tennis, baseball), (hockey, sports), (tennis, golf), (golf, tennis), (tennis, football), (football, tennis), (tennis, soccer), (hockey, football), (soccer, tennis), (football, hockey).

***hobby-like-hobby***: (baseball, football), (fishing, fishing), (fishing, golf), (basketball, basketball), (golf, basketball), (football, football), (golf, bowling), (football, basketball), (rugby, football), (skiing, skiing), (football, baseball), (bowling, fishing), (boxing, football), (football, soccer), (soccer, football), (baseball, tennis), (sports, hockey), (hockey, football), (tennis, golf), (tennis, football), (tennis, soccer), (basketball, football).

A Tabela 5.6 apresenta um resumo dos resultados obtidos neste experimento. Do total de agrupamentos possíveis (6, sendo 2 para cada relação), 3 são obtidos, um para cada relação de origem do Prophet. A precisão é calculada como indicado na a Equação 4.2 exposta na Subseção 4.1.3 e também apresentadas na mesma tabela.

Todas as relações estão nomeadas pelo verbo "*like*", indicando similaridade. Instâncias sementes como "(*fishing, fishing*)" e "(*basketball, basketball*)" podem ser classificadas como inválidas por possuírem exatamente a mesma instância de categoria no sujeito e no objeto (por serem as *mesmas* instâncias, é irrelevante a informação de que são similares). No entanto, a maior parte das instâncias sementes são possivelmente válidas, considerando que: o sujeito e objeto pertencem às respectivas categorias de interesse, as relações são logicamente válidas, e, as sentenças são semanticamente completas e relevantes.

Nos próximos experimentos desta abordagem para nomeação de relações candidatas anônimas, apenas o par de categorias encontrado para a relação é considerado, e não todas as possíveis combinações de instâncias. Serão desconsideradas relações nas quais o sujeito e objeto são os mesmos, como no exemplo (*fishing, fishing*).

### 5.3.2 Experimento com as 20 melhores e 20 piores relações do Prophet

A fim de comprovar a eficácia desta metodologia de validação e nomeação de relações encontradas pelo Prophet, este experimento foi subdividido em 2 etapas: a primeira etapa tem como guia os pares de categorias das 20 relações piores colocadas no *ranking* do próprio Prophet; a segunda parte tem como guia os pares de categorias das 20 melhores relações válidas. As relações são inválidas conforme a classificação a seguir, que tem como base a classificação utilizada nos experimentos de Mohamed, Hruschka Jr. e Mitchell (2011) acrescida de alguns itens para contemplar o universo de experimentos com colaboração do Prophet:

1. Ambiguidade semântica: caso a categoria referenciada não faça sentido na relação (apesar da possibilidade da instância fazer sentido semântico).
2. Instância classificada incorretamente: relação é inválida se alguma instância referenciada pela relação não pertence realmente à categoria que ela está associada.
3. Informação semanticamente incompleta: se a relação precisa de mais informação para fazer sentido semântico.
4. Relações ilógicas.
5. Pelo menos uma das categorias envolvidas não pertence à BC utilizada.

Das 20 melhores relações válidas consideradas, newOntExt gerou nomes para 17 delas. Para 16 destas destas relações, foram construídos 2 agrupamentos para agrupar as sementes de mesmo sentido; 1 relação teve apenas um agrupamento gerado e para as outras 3 não houve agrupamento, isto é, foram invalidadas pelo processo.

A maioria das sementes de gerações são incorretas devido a ambiguidade de instâncias de categorias e instâncias que não pertencem à categoria designada. Alguns exemplos de pares *(instância, categoria)* — em inglês, como estão na BC — incorretos são: (water, sport), (strength, convention), (photos, park), (page, arthropod), (lentil, musicfestival), (center, visualartform), (resources, economicsector), (edit, monument), (third_party, politicalparty), (zero, food), (home, athlete), (students, sportsteam).

| | |
|---|---|
| Relações do Prophet consideradas | 20 |
| Total de possíveis agrupamentos | 40 |
| Relações do Prophet invalidadas por newOntExt | 3 |
| Agrupamentos gerados | 33 |
| Relações geradas válidas | 9 |
| Relações nomeadas incorretamente | 24 |
| Precisão | 27,27% |

**Tabela 5.7: Resumo das nomeações das 20 melhores relações do Prophet.**

Analogamente ao que está descrito na Subseção anterior (5.3.1), a Tabela 5.7 resume os resultados deste experimento. Do total de agrupamentos possíveis (40, sendo 2 para cada uma das 20 relações), 33 são obtidos. Destes, 24 dão origem a relações inválidas devido a erro de classificação de instâncias; 9 são consideradas válidas.

Apesar das sementes de pares *(instância, categoria)* incorretos, 9 relações são logicamente corretas e fazem completo sentido como (formato (*categoriaDoSujeito-frase_verbal-categoriaDoObjeto*)): *cognitiveactions-can_spill_into-park*,
*cognitiveactions-started_on-visualartform*,
*sportsleague-lodge_has_crowned-sportsteamposition*,
*economicsector-grown_with-musicfestival*,
*politicalparty-makes-musicfestival*,
*athlete-infringes-food*,
*musicalbum-focuses_on-visualartform*,
*musicalbum-has-visualartform*,
*sportsteam-have_charged_on-convention*.

Para as 20 piores relações identificadas, pontuadas e ordenadas pelo próprio Prophet, newOntExt não encontrou possíveis nomes para nenhuma, isto é, não coletou dados suficientes para realizar agrupamentos nas matrizes. Assim, pode-se concluir que, para a amostra deste experimento, a estratégia funciona para invalidar relações mal pontuadas pelo Prophet como propõe a metodologia de validação e nomeação.

# Capítulo 6

## CONCLUSÃO

O novo paradigma de aprendizado sem-fim, no qual o sistema NELL está baseado, possui várias características importantes que permitem a constante melhoria na capacidade de aprendizado do sistema. Uma das características fundamentais para que o aprendizado sem-fim possa ocorrer de maneira adequada é a extensão automática e contínua da ontologia. Este trabalho apresenta o newOntExt, uma nova abordagem para o método de extensão da ontologia atualmente utilizado pelo NELL. O newOntExt tem como base o método OntExt. O OntExt foi criado em 2011 e vinha sendo, desde então, utilizado pelo NELL para criar novas relações e, assim, estender a estrutura da base de conhecimento (ou ontologia) do sistema. O OntExt gerou, desde 2011, um conjunto de 80 novas relações, as quais foram inseridas na ontologia, mas a maioria dos resultados gerados pelo OntExt foram relações inválidas (que não foram inseridas). Além disso, o OntExt possui alto custo computacional. O novo subsistema newOntExt é uma nova abordagem e possui uma nova implementação com novas características, recursos e estratégias para expandir a Base de Conhecimento (BC) do NELL.

Como descrito no Capítulo 4, o newOntExt apresenta quatro principais contribuições (em relação ao OntExt). A primeira delas está relacionada ao pré-processamento do corpus. No OntExt, esta etapa era realizada intrinsecamente ao método, e isso tornava o processo bastante custoso (computacionalmente). Além disso, o método de pré-processamento tinha como base os princípios definidos na primeira geração dos sistemas de *extração de informação aberta* (BANKO; ETZIONI, 2008). Já o newOntExt torna o pré-processamento independente da busca por novas relações, isto permite a utilização de diferentes *corpora*, além de tomar como base os princípios definidos na segunda geração dos sistemas de *extração de informação aberta* (FADER; SODERLAND; ETZIONI, 2011; ETZIONI et al., 2011) (o que torna mais preciso o filtro de sentenças utilizadas para a descoberta de novas relações).

A segunda contribuição está vinculada ao novo algoritmo, o qual foi implementado com

vários novos recursos de pré-processamentos e otimização do código. Esta é uma contribuição mais de engenharia de software do que de aprendizado de máquina, mas se mostrou fundamental para viabilizar o newOntExt como componente a ser utilizado pelo NELL na prática. Como pode ser observado na Seção 5.1, a metodologia com escopo reduzido tem diferentes atuações nos diversos subconjuntos de conhecimento. Tais subconjuntos foram definidos com base na metodologia descrita em (Dos Santos; Hruschka Jr, 2014) e se mostraram adequados. Para o subconjunto das categorias de conhecimento associadas a animal, por exemplo, 30 relações foram geradas, metade destas são candidatas a serem adicionadas à BC. Pelos resultados obtidos, deduz-se que as instâncias de categorias desta área do grafo de conhecimento do NELL estão bem definidas o suficiente para se obter um guia confiável para geração de maior conhecimento acerca deste assunto. Mas, não pode ser descartada a supervisão nesta área da ontologia, visto que 10 das 15 relações inválidas o são por conta de Erro de classificação de instância. Para o subconjunto de categorias relacionado a *construção civil* são 8 relações geradas, 6 incorretas e todas devido a erro de classificação de instância de categoria. Isso indica um alerta para supervisão do sistema quanto às instâncias associadas a estas categorias. Isto é, desta forma é identificado uma área do grafo de conhecimento que necessita de uma supervisão quanto às suas crenças.

O alerta para a supervisão é uma nova contribuição deste trabalho. Com os experimentos realizados, constatou-se que o newOntExt (assim como era o OntExt) é sensível a ruídos, isto é, depende de muitos exemplos que sejam crenças confiáveis. Ambiguidades e instâncias mal associadas às respectivas categorias prejudicam o aprendizado. Com base nesta observação, a abordagem adotada foi a de adotar a estratégia de utilizar o newOntExt também como um alerta para correções na ontologia, isto é, colaborar de maneira mais efetiva com a auto-supervisão e auto-relfexão do sistema NELL.

A última contribuição está vinculada à integração com o método baseado em grafos: *Prophet*. Tal integração permite que informação semântica (nome das relações) seja inserida no modelo baseado em grafos, assim, torna mais robusto o processo de criação de novas relações para a ontologia do NELL.

Apesar de menos da metade das relações serem adequadas para a alteração da BC do NELL, quando comparado com o OntExt (que não gerou nenhuma relação correta para os mesmos subconjuntos de categorias) o newOntExt demonstrou (empiricamente) com este resultados iniciais, que pode trazer ganhos para o sistema de aprendizado sem-fim.

Como possíveis trabalhos futuros, pode-se citar: automaticamente classificar, validar e nomear novas relações geradas por outros componentes; trabalhar com o aspecto temporal de fatos

relacionais; atuar na auto-reflexão e auto-supervisão da ontologia do sistema ao indicar instâncias erroneamente atribuídas às respectivas categorias; e executar um experimento para validar o método tradicional para extensão da ontologia como um todo.

# REFERÊNCIAS


APPEL, A. P.; Hruschka Jr, E. R. Prophet – A Link-Predictor to Learn New Rules on NELL. *2011 IEEE 11th International Conference on Data Mining Workshops*, Ieee, p. 917–924, dez. 2011.

BANKO, M.; CAFARELLA, M.; SODERLAND, S. Open information extraction for the web. 2009.

BANKO, M.; ETZIONI, O. Strategies for lifelong knowledge extraction from the web. *Proceedings of the 4th international conference on Knowledge capture - K-CAP '07*, ACM Press, New York, New York, USA, p. 95, 2007.

BANKO, M.; ETZIONI, O. The tradeoffs between open and traditional relation extraction. In: *Proceedings of ACL-08: HLT*. Columbus, Ohio: Association for Computational Linguistics, 2008. p. 28–36.

BETTUZZI, S. Chapter 1: Introduction. *Advances in cancer research*, v. 104, p. 1–8, jan. 2009. ISSN 0065-230X.

CARLSON, A. et al. Toward an architecture for never-ending language learning. In: *Proceedings of the Conference on Artificial Intelligence (AAAI)*. [S.l.]: AAAI Press, 2010. p. 1306–1313.

Dos Santos, R. G.; Hruschka Jr, E. R. Markov logic scalability in a never-ending language learning system. In: *NewsKDD Workshop: Data Science for News Publishing. Workshop at the 20th ACM-SIGKDD Conference on Knowledge Discovery and Data Mining - KDD2014*. [S.l.: s.n.], 2014. p. 21–25.

DUARTE, M. Aprendizado Semissupervisionado Através de Técnicas de acoplamento. In: *Dissertacao de Mestrado*. Departamento de Computacao, Universidade Federal de Sao Carlos: [s.n.], 2011.

ETZIONI, O. et al. Open information extraction: The second generation. In: WALSH, T. (Ed.). *IJCAI*. [S.l.]: IJCAI/AAAI, 2011. p. 3–10. ISBN 978-1-57735-516-8.

FADER, A.; SODERLAND, S.; ETZIONI, O. Identifying relations for open information extraction. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (EMNLP '11), p. 1535–1545. ISBN 978-1-937284-11-4.

HALL, M. et al. The weka data mining software: an update. *SIGKDD Explor. Newsl.*, ACM, New York, NY, USA, v. 11, n. 1, p. 10–18, nov. 2009. ISSN 1931-0145.

HASEGAWA, T.; SEKINE, S.; GRISHMAN, R. Discovering relations among named entities from large corpora. In: *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004. (ACL '04).

HOFFART, J. et al. YAGO2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, Elsevier, Amsterdam, v. 194, p. 28–61, 2013.

Hruschka Jr, E.; DUARTE, M.; NICOLETTI, M. Coupling as strategy for reducing concept-drift in never-ending learning environments. *Fundamenta Informaticae*, IOS Press, v. 124, n. 1, p. 47–61, 2013.

Hruschka Jr, E. R. Machine Learning, Machine Reading and the Web. In: *Tutorial presented at IBERAMIA 2012 - 13th Ibero-American Conference on AI*. Cartagena de Indias, Colombia: [s.n.], 2012.

LAO, N.; MITCHELL, T.; COHEN, W. W. Random walk inference and learning in a large scale knowledge base. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2011. (EMNLP '11), p. 529–539. ISBN 978-1-937284-11-4.

MITCHELL, T. *The discipline of machine learning*. [S.l.: s.n.], 2006.

MITCHELL, T. M. *Machine Learning*. 1. ed. New York, NY, USA: McGraw-Hill, Inc., 1997. ISBN 0070428077, 9780070428072.

MOHAMED, T.; Hruschka Jr., E. R.; MITCHELL, T. M. Discovering relations between noun categories. In: *EMNLP*. [S.l.]: ACL, 2011. p. 1447–1455. ISBN 978-1-937284-11-4.

NAKASHOLE, N.; WEIKUM, G.; SUCHANEK, F. PATTY: A Taxonomy of Relational Patterns with Semantic Types. *EMNLP12*, 2012.

NORVIG, P. Inference in text understanding. In: *AAAI Spring Symposium: Machine Reading*. AAAI, 2007. p. 6–10. Disponível em: <http://www.aaai.org/Library/Symposia/Spring/ss07-06.php>.

NUNES, M. G. V. *O Processamento de Línguas Naturais : para quê e para quem ?* 73. ed. [S.l.]: Instituto de Ciências Matemáticas e de Computação, 2008. ISBN 0103-2585.

PEDRO, S. D.; APPEL, A. P.; Hruschka Jr., E. R. Autonomously reviewing and validating the knowledge base of a never-ending learning system. In: INTERNATIONAL WORLD WIDE WEB CONFERENCES STEERING COMMITTEE. *Proceedings of the 22nd international conference on World Wide Web companion*. [S.l.], 2013. p. 1195–1204.

SUCHANEK, F.; WEIKUM, G. Knowledge Harvesting from Text and Web Sources. *suchanek.name*, p. 1–4, 2010.

TABA, L. S.; CASELI, H. de M. Automatic hyponymy identification from brazilian portuguese texts. In: CASELI, H. de M. et al. (Ed.). *PROPOR*. Springer, 2012. (Lecture Notes in Computer Science, v. 7243), p. 186–192. ISBN 978-3-642-28884-5. Disponível em: <http://dx.doi.org/10.1007/978-3-642-28885-2>.

WEIKUM, B. Y. G. et al. DB & IR methods for Knowledge Discovery. 2009.

WEIKUM, G.; THEOBALD, M. From information to knowledge: harvesting entities and relationships from web sources. In: PAREDAENS, J.; GUCHT, D. V. (Ed.). *PODS*. ACM, 2010. p. 65–76. ISBN 978-1-4503-0033-9. Disponível em: <http://dl.acm.org/citation.cfm?id=1807085>.

WIJAYA, D. T.; TALUKDAR, P. P.; MITCHELL, T. M. Pidgin: ontology alignment using web text as interlingua. In: HE, Q. et al. (Ed.). *CIKM*. [S.l.]: ACM, 2013. p. 589–598. ISBN 978-1-4503-2263-8.

ZHANG, M. et al. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In: DALE, R. et al. (Ed.). *IJCNLP*. [S.l.]: Springer, 2005. (Lecture Notes in Computer Science, v. 3651), p. 378–389. ISBN 3-540-29172-5.

ZHU, X. Semi-Supervised Learning Literature Survey Contents. 2008.

# Apendice A

## O CONJUNTO DE DADOS CLUEWEB09

O conjunto de dados *ClueWeb09* foi criado para apoiar pesquisa em recuperação de informação e tecnologias relacionadas a linguagem humana. Consiste de cerca de 1 bilhão de páginas Web em dez línguas que foram coletadas em Janeiro e Fevereiro de 2009. O conjunto de dados é utilizado por diversas faixas da conferência TREC. Especificações do conjunto de dados Páginas web:

- 1,040,809,705 páginas web, em 10 línguas;
- 5 TB, comprimido (25 TB, não-comprimido)

Grafo web:

- Conjunto de dados completo (TREC Category A):
  - URLs únicas: 4,780,950,903 (325 GB não comprimido, 105 GB comprimido);
  - Total de links externos: 7,944,351,835 (71 GB não comprimido, 24 GB comprimido).
- TREC Category B (primeiras 50 milhões de páginas em inglês):
  - URLs únicas: 428,136,613 (30 GB não comprimido, 10 GB comprimido);
  - Total de links externos: 454,075,638 (3 GB não comprimido, 1 GB comprimido).

# Apendice B

## EXTRAÇÕES DE REVERB A PARTIR DO CONJUNTO DE DADOS CLUEWEB

O conjunto de dados *ReVerb ClueWeb Extractions 1.1* contém aproximadamente 15 milhões de asserções binárias da Web. São extrações feitas pelo *ReVerb* a partir do corpus *ClueWeb09*.

## B.1 Estatísticas

Cada registro no conjunto de dados corresponde a uma tupla (arg1, relação, arg2). Os número abaixo são os números de tuplas distintas, expressões de argumentos, e expressões relacionais no conjunto de dados:

- Tuplas: 14,728,268
- Expressões de argumentos: 2,263,915
- Expressões relacionais: 664,746

## B.2 Pré-processamento

Este conjunto de dados é um subconjunto da saída do processamento do *ReVerb* na porção inglesa do corpus *ClueWeb09*. Foi executada o *ReVerb* versão 1.0 no corpus, o que resultou em aproximadamente 6 bilhões de extrações. Esse conjunto inicial de extrações é muito grande para distribuir pela Web, além de ter ruídos. Para obter um conjunto de dados menor e com maior precisão, foram aplicados os seguintes filtros:

- Gatilho de confiança: Para cada par (extração, sentença) é atribuída uma pontuação de confiança em [0, 1] por um classificador. Foram removidas todas as extrações com valor de confiança menor que 0.9;

- Filtro sintático: As extrações foram filtradas baseado em algumas características sintáticas dos argumentos e relações. Foram removidas extrações com argumentos que são substantivos comuns definitivos, ou contêm pronomes, determinantes demonstrativos, e certos quantificadores (*both*, *all*, *certain*, *other*, etc.). Também foi filtrada qualquer extração contendo um substantivo próprio ou número em sua relação;

- Filtro de palavra-de-parada: Foram removidas extrações que consistem de palavras temporais comuns (por exemplo, *yesterday*, *tonight*, dias da semana) e extrações com relações que são quase sempre não-informativas (por exemplo *have*, *is*, *said*).

- Gatilho de frequência de expressão: Foi contado o número de extrações distintas em que cada expressão de argumento e relacional apareceu, e então foram removidas quaquer extração $(x,r,y)$ com $(freq(x) < 5)$ ou $(freq(r) < 5)$ ou $(freq(y) < 5)$.

Depois de aplicar estes filtros, aplicamos uma simples normalização morfológica para argumentos e relações ao remover o tempo (verbal), pluralização, capitalização, etc. Então foram fundidas as extrações com a mesma forma normalizada.

## B.3 Formato

As extrações são armazenadas em um arquivo de texto simples codificados em UTF-8. O arquivo tem as seguintes colunas separadas por tabulação:

1.Id de extração

2.Argumento 1

3.Relação

4.Argumento 2

5.Argumento 1 normalizado

6.Relação normalizada

7.Argumento 2 normalizado

8. O número de sentenças distintas que esta extração foi extraída de

9. Pontuação máxima de confiança atribuída a esta extração, de todas as sentenças que esta foi extraída de

10. Uma lista de URLs fontes para cada sentença

# Apendice C

## SUJEITO-VERBO-OBJETO (SVO) A PARTIR DE CLUEWEB09

O conjunto de dados Sujeito-Verbo-Objeto (SVO) contém 604.934.719 de asserções binárias da Web. São extrações feitas pelo grupo de estudo do projeto *Read The Web* a partir do corpus *ClueWeb09*.

# Apendice D

## AMBIENTE PARA SIMULAÇÃO DO NELL

Pela necessidade de uso de ferramentas, métodos e objetos já nativos do sistema NELL do projeto ReadTheWeb e pelo intuito do newOntExt ser um subsistema componente a ser integrado futuramente ao NELL, houve a necessidade de simular um ambiente de aprendizado do NELL para testes reais da metodologia apresentada.

Para tanto, é necessário atualizar e configurar alguns recursos e ferramentas na máquina a ser utilizada, como o próprio Java (linguagem base para o sistema), o perl (linguagem para rodar os scripts de construção e execução do sistema), o Ant (uma ferramenta de construção de código aberto para agrupar todas as partes de um programa da *Apache Software Foundation*, dpkg (gerenciador de pacotes para Debian Linux), entre outros. Além de particularidades do sistema como o *TokyoCabinet*, que armazena os dados da Base de Conhecimento do NELL.

Para cada máquina que se tenha intenção de executar experimentos acerca deste projeto, deve-se montar e configurar este ambiente para simulação do NELL. No caso deste projeto de mestrado, este ambiente foi montado em uma máquina do servidor ONTO do projeto *Read the Web* da *Carnegie Mellon University*. Estes esforços de configuração do ambiente demandam foco e tempo de trabalho.

# Apendice E

## RESULTADOS DE EXPERIMENTOS - NEWONTEXT

Este capítulo contém os resultados dos experimentos relatados na Seção 5.1 em sua totalidade. Cada resultado está como newOntExt os identificou e gerou, em inglês e no formato

`<categoriaDoSujeito>-<frase_verbal>-<categoriaDoObjeto>`

Os resultados positivos estão acompanhandos de suas crenças sementes para a relação, isto é, uma lista de um ou mais pares de instâncias, cada par no formato

`{"<sujeito>","<objeto>"}`

Os resultados estão separados em seções para cada experimento, e, dentro das seções, em duas subções com: os resultados logicamente corretos e repletos de sentido semântico; e, resultados incorretos logicamente ou incompletos quanto ao sentido semântico. Como descrito na Subseção 4.1.3, a classificação abordada é a mesma utilizada para os resultados de OntExt. Para cada relação incorreta, o motivo é indicado logo na sequência, com um número entre parênteses; o número refere-se aos diferente motivos para uma relação ser considerada incorreta, conforme classificação abaixo. Uma relação gerada é considerada incorreta se:

1. Ambiguidade semântica: caso as instâncias pertencentes a uma ou ambas as categorias envolvidas na relação são ambíguas e não fazem sentido no contexto da relação.
2. Erro de classificação de instância: caso uma ou ambas as instâncias envolvidas estão erroneamente associadas à(s) respectiva(s) categoria(s);
3. Informação incompleta semanticamente: caso a relação necessite de maiores informações para fazer sentido semântico;

4.Relações incorretas logicamente: caso simplesmente a relação não faça sentido lógico.

## E.1 Com subconjunto de categorias relacionado a animal

### E.1.1 Resultados válidos

**arthropod-will_not_get-animal**: {"flea","pigs"}
**insect-have_always_be-animal**: {"bee","bees"}
**animal-be_play_with-mammal**: {"cat","dog"}{"mice","rat"}{"lion","sheep"}
{"sheep","lion"}{"salmon","grizzly"}{"crocs","koala"}{"humans","mouse"}
{"man","wolves"}{"porpoise","dolphin"}{"rabbit","mouse"}
{"horse","cattle"}{"monkey","dog"}{"kudu","impala"}{"leopard","baboon"}
{"impala","kudu"}{"baboon","leopard"}{"new","cats"}{"raccoon","dog"}
{"cheetahs","lion"}{"chimpanzee","gorillas"}
{"white_rhino","black_rhino"}{"humans","cattle"}{"cow","tigers"}
{"humans","rat"}{"coyote","timber_wolf"}{"bears","dolphins"}
{"dolphins","bears"}{"cat","rat"}{"mouse","elephant"}{"bee","elephant"}
{"elephant","mouse"}{"sheep_dog","sheep"}{"elk","cattle"}{"martin","dog"}
{"cattle","elk"}{"baby","dog"}{"cat","squirrel"}{"woman","rat"}
{"sheep","sheep_dog"}{"horse","reindeer"}{"dog","mouse"}{"bug","lion"}
{"tortoise","burros"}{"dog","rat"}{"salad","dog"}{"mouse","dog"}
{"rat","dog"}{"penguin","polar_bear"}{"clownfish","whale"}
{"crab","crabeater_seal"}
**animal-be_closely_relate_to-mammal**: {"chimpanzee","bonobo"}
{"porpoise","dolphin"}{"dog","wolves"}{"wolves","dog"}
{"llama","alpacas"}{"alpacas","llama"}{"worm","dog"}
{"elephant","mastodon"}{"gray_wolf","arctic_wolf"}{"mice","rat"}
{"dugong","elephant"}{"false_killer_whale","pygmy_killer_whale"}
{"fin_whale","blue_whale"}{"reedbuck","mountain_reedbuck"}
{"okapi","giraffe"}{"horse","zebra"}{"antelope","pronghorn"}
{"llama","alpaca"}{"pygmy_killer_whale","false_killer_whale"}
{"tigers","lion"}{"pig","hippos"}{"lion","tigers"}{"mastodon","elephant"}
{"jaguars","lion"}{"giraffe","okapi"}{"lion","jaguars"}
{"black_bear","brown_bear"}{"beavers","squirrel"}{"pronghorn","antelope"}
{"arctic_wolf","gray_wolf"}{"alpaca","llama"}{"lynx","caracal"}

{"manatee","elephant"}{"lion","wolves"}{"evening_bat","big_brown_bat"}
{"alpacas","llamas"}{"sugar_gliders","squirrel_glider"}
{"red_fox","grey_fox"}{"llamas","alpacas"}{"wolves","lion"}
{"big_brown_bat","evening_bat"}
**animal-typically_eat-reptile**: {"spider","lizards"}
**insect-work_to_save-mammal**: {"bee","dog"}
**arthropod-can_be_very_irritating_to-mammal**: {"flea","dog"}
{"deer_tick","deer"}{"flea","dogs"}
**arthropod-be_particularly_lethal_to-mollusk**: {"crab","geoduck"}
**insect-be_discover_in-mollusk**: {"a_bug","squid"}
**mammal-be_extremely_interested_in-reptile**: {"kittens","snake"}
**reptile-eat_ton_of-arthropod**: {"frog","bug"}{"frog", "mosquito"}
**mammal-do_not_take_kindly_to-reptile**: {"rat","snake"}
**arthropod-be_sometimes_confuse_with-insect**: {"damselfly","dragonfly"}
{"dragonfly","damselfly"}{"millipedes","wireworm"}
{"wireworm","millipedes"}{"house_fly","flesh_fly"}
{"flesh_fly","house_fly"}{"deer_fly","horse_fly"}{"hookworm","roundworm"}
{"horse_fly","deer_fly"}{"mayfly","dragonfly"}{"roundworm","earthworm"}
{"wasp","sawfly"}{"dragonfly","mayfly"}{"earthworm","roundworm"}
{"adult_beetle","mountain_pine_beetle"}
{"mountain_pine_beetle","adult_beetle"}{"corn_borer","caddisfly"}
{"mite","ticks"}{"caddisfly","corn_borer"}{"aphid","adelgid"}
{"adelgid","aphid"}
**arthropod-look_a_bit_like-insect**: {"syrphid_fly","bee"}{"bee","syrphid_fly"}
**reptile-will_swallow-insect**: {"frog","bug"}

### E.1.2 Resultados inválidos

arthropod-be_also_good_against-animal (3)
animal-be_a_lovely_alternative_to-mollusk (4)
animal-provide_list_of-mollusk (2)
animal-be_just_another_term_for-reptile (2)
insect-be_also_fond_of-animal (2)
insect-occupy_a_wide_range_of-mammal (2)
insect-have_overlap-reptile (2)
arthropod-be_more_frequent_for-mammal (2)

arthropod-can_be_use_instead_of-mollusk (3)
insect-be_classify_as-mollusk (2)
mammal-be_consider_live-mollusk (4)
mammal-remain_of-mollusk (2)
reptile-be_very_fond_of-mollusk (4)
reptile-remain_of-mollusk (2)
arthropod-be_make_up_on-reptile (2)

## E.2 Com subconjunto de categorias relacionado a construção

### E.2.1 Resultados válidos

**attraction-be_fall_in-city**: {"disneyland","anaheim"}
{"busch_gardens","tampa"}{"arlington_national_cemetery","arlington"}
{"ronald_reagan_washington_national_airport","arlington"}
{"forum","victoria"}{"mississippi","memphis"}{"home","louisiana"}
{"tampa_bay","tampa"}{"tampa","tampa_bay"}{"art","wood"}
{"disneyland_park","anaheim"}{"elephant","portland"}
{"the_national_museum","manila"}{"opryland","nashville"}
{"opryland_usa","nashville"}{"woodlawn_cemetery","clinton"}
{"nature","liberty"}{"menlo_park","house"}{"portland","liberty"}
{"house","menlo_park"}{"liberty","portland"}
{"the_national_marine_aquarium","plymouth"}
{"kensington_gardens","kensington"}{"queens","glendale"}
{"busch_gardens_africa","tampa"}{"nelson_mandela_bay","port_elizabeth"}
{"strawbery_banke_museum","portsmouth"}{"max","anaheim"}
{"summit","kansas_city"}{"mississippi","summit"}{"kansas_city","summit"}
{"forum","orange"}{"pioneer_courthouse_square","portland"}
{"holocaust_museum","tampa"}{"forbes_field","oakland"}
{"disneyland_resort","anaheim"}{"adelphia_coliseum","nashville"}
{"mississippi","casino"}
**attraction-be_just_minute_from-city**: {"oxford_street","camden"}

### E.2.2 Resultados inválidos

buildingfeature-have_lot_of-attraction (2)
buildingfeature-be_mostly_make_of-attraction (2)
buildingfeature-do_not_go_on-city (2)
city-be_an_environmentally_friendly_alternative_to-buildingmaterial (2)
city-be_the_city_of-buildingmaterial (2)
buildingfeature-be_a_very_popular_alternative_to-city (2)

## E.3 Com subconjunto de categorias relacionado a esporte

### E.3.1 Resultados válidos

**athlete-fly_out_to-sportsteamposition**: {"martin","center"}
{"joseph","center"}{"chris_wright","center"}{"frank","center"}
{"edgardo_alfonzo","center"}{"johnny_damon","center"}
{"mitchell","center"}{"aaron_boone","center"}{"ramirez","center"}
{"helton","center"}{"manny_ramirez","center"}{"ramon_santiago","center"}
{"ted_williams","center"}{"chuck_knoblauch","center"}{"sosa","center"}
{"brian_snyder","center"}{"lowell","center"}{"willis","center"}
{"jason_bartlett","center"}{"mike_lowell","center"}
{"ivan_rodriguez","center"}{"scott","center"}
{"bernie_williams","center"}{"todd_zeile","center"}{"ruiz","center"}
{"tyler_johnson","center"}{"scott_rolen","center"}{"damon","center"}
{"paul_lo_duca","center"}{"jason_lane","center"}{"phillips","center"}
{"ryan_howard","center"}{"torii_hunter","center"}
{"russell_branyan","center"}{"ramon_hernandez","center"}
**athlete-say_no_to-sportsleague**: {"carroll","nfl"}
**sportsleague-be_for-sport**: {"nhl","hockey"}
**sportsteam-play_for-sportsleague**:
{"chicago_bears","national_football_league"}
**sport-be_link_to-sportsteam**: {"sailing","lions"}

### E.3.2 Resultados inválidos

athlete-be_at-sportsteam (2)

athlete-must_beat-sportsteamposition (2)

sportsteam-be_bear_in-athlete (4)

athlete-can_be_play_at-sport (4)

athlete-also_return_to-sport (2)

athlete-be_a_pleasure_to-sportsleague (2)

sportsleague-play_for-sport (2)

sportsteam-be_the_official_online_shop_of-sportsleague (4)

sportsleague-be_the_team_of-sportsteamposition (4)

sportsleague-stand_for-sportsteamposition (2)

sport-be_not_hire-sportsteam (4)

# Apendice F

## RESULTADOS DE EXPERIMENTOS - ONTEXT

Analogamente ao que está descrito no Capítulo anterior (E), este Capítulo apresenta os resultados relacionados aos subgrupos de categorias de interesse descritos na Seção 5.1 gerados pelo subsistema componente inativo OntExt em totalidade. A clasificação destas relações também é originada dos autores Mohamed, Hruschka Jr. e Mitchell (2011). Para cada relação incorreta, o motivo é indicado logo na sequência, com um número entre parênteses; o número refere-se aos diferente motivos para uma relação ser considerada incorreta, conforme descrito no Capítulo E.

## F.1 Com subconjunto de categorias relacionado a animal

### F.1.1 Resultados válidos

Não existem relações corretas logicamente e semanticamente.

### F.1.2 Resultados inválidos

animal-using-animal (2)
insect-such_as-animal (1)
animal-such_as-mammal (1)
animal-eating-animal (1)
insect-that_feed_on-animal (1)
animal-eat-reptile (2)
animal-and_other-reptile (1)

## F.2 Com subconjunto de categorias relacionado a construção

### F.2.1 Resultados válidos

Não existem relações corretas logicamente e semanticamente.

### F.2.2 Resultados inválidos

`city-hotels_near-attraction (2)`

`city-including-attraction (1)`

## F.3 Com subconjunto de categorias relacionado a esporte

### F.3.1 Resultados válidos

Não existem relações corretas logicamente e semanticamente.

### F.3.2 Resultados inválidos

`sport-team_in-sportsleague (2)`