UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

APRENDIZADO DE SUBCATEGORIAS PARA *NEVER-ENDING LANGUAGE LEARNING*: UMA ABORDAGEM BASEADA EM PERGUNTAS E RESPOSTAS

WESLEY WILLY OLIVEIRA DE SOUZA

ORIENTADOR: PROF. DR. RICARDO AUGUSTO SOUZA FERNANDES

São Carlos – SP

Janeiro/2021

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

APRENDIZADO DE SUBCATEGORIAS PARA *NEVER-ENDING LANGUAGE LEARNING*: UMA ABORDAGEM BASEADA EM PERGUNTAS E RESPOSTAS

WESLEY WILLY OLIVEIRA DE SOUZA

Tese 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 Doutor em Ciência da Computação, área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Ricardo Augusto Souza Fernandes

São Carlos – SP

Janeiro/2021

UNIVERSIDADE FEDERAL DE SÃO CARLOS

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

Folha de Aprovação

Defesa de Tese de Doutorado do candidato Wesley Willy Oliveira de Souza, realizada em 08/01/2021.

Comissão Julgadora:

Prof. Dr. Ricardo Augusto Souza Fernandes (UFSCar)

Prof. Dr. Diego Furtado Silva (UFSCar)

Prof. Dr. Ricardo de Andrade Lira Rabelo (UFPI)

Prof. Dr. Fábbio Anderson Silva Borges (UESPI)

Prof. Dr. Vinicius Ponte Machado (UFPI)

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.
O Relatório de Defesa assinado pelos membros da Comissão Julgadora encontra-se arquivado junto ao Programa de Pós-Graduação em Ciência da Computação.

Dedico este trabalho ao meu tio Moacyr (*in memorian*),
ao meu sobrinho Leonardo e minha avó Nilpha.

AGRADECIMENTOS

Agradeço ao Professor Dr. Ricardo Augusto Souza Fernandes que apesar da intensa rotina de sua vida acadêmica aceitou me orientar. Obrigado pela confiança depositada, aprendizado, paciência e por estar à disposição em me orientar nesta pesquisa.

Agradeço ao Professor Dr. Estevam Rafael Hruschka Júnior, pelos ensinamentos, incentivo, por me fazer pensar e refletir a todo momento, essa troca de conhecimento fez grande diferença no resultado final desta pesquisa.

Agradeço a Maísa Cristina Duarte, minha grande amiga, chefe e minha madrinha de casamento, por ser essa pessoa incrível que fez e faz toda a diferença em minha vida.

Agradeço a Michelle, minha esposa, pelo apoio incondicional e por me manter motivado durante todo o processo.

Agradeço a Tia Teresa e Tio Dú por me darem oportunidades priorizando minha educação e pelo suporte e direcionamento para as minhas realizações e conquistas.

Agradeço ao meu irmão Hudson por ter estado presente em uma parte desta jornada, obrigado pela atenção, amizade e carinho.

Agradeço ao meu sogro Wilson e minha sogra Silvia, pelo apoio e incentivo que sempre me deram durante todos esses anos.

Agradeço à toda minha família por compreenderem os momentos em que estive ausente por causa do desenvolvimento desta pesquisa.

Agradeço aos meus amigos, Flávio Montoro, Pedro Migliatti, Élen Tomazela, Amandia, Antonio Petri e Diorge, que de alguma forma me apoiaram. Obrigado pela oportunidade do convívio e pela cooperação durante estes anos.

Agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) por ter apoiado o presente trabalho realizado através do financiamento 001.

"*We can only see a short distance ahead, but we can see plenty there that needs to be done*".

Alan Turing

RESUMO

Nos últimos anos, ontologias são utilizadas em sistemas de informação para indexar grandes *corpora* de documentos ou coleções de fatos, bem como apoiar diretamente na interação do usuário com o sistema através de funcionalidades como navegação e consultas. Assim, nota-se que tanto a estrutura quanto o conteúdo das ontologias devem acompanhar essas mudanças, temporalmente, sem perder coerência. A expansão de ontologias é a princípio um processo organizacional portanto, deve haver regras para a realização dos processos de atualizações, inserções e remoções da ontologia. Após aprender milhões de fatos extraídos da *web*, a NELL (*Never-ending Language Learning*), o primeiro sistema de aprendizado de máquina sem fim descrito na literatura que ininterruptamente extrai fatos (lendo a *web*) para aumentar sua base de conhecimento e aprender a ler melhor que o dia anterior, passou a adquirir além do conhecimento extraído e a inferir em novas crenças que ainda não havia lido anteriormente, tornando-se capaz de expandir sua ontologia inicial através de várias contribuições. Ainda assim, notou-se uma limitação no conhecimento da NELL, quanto a aprender novas subcategorias a partir das categorias já conhecidas em sua base de conhecimento. Neste sentido, a presente tese tem o objetivo de propor um componente modular sequencial que possibilite a expansão da ontologia da base de conhecimento da NELL, identificando e classificando subcategorias das categorias já conhecidas pela ontologia da NELL. O componente proposto recebe como entrada textos de perguntas em inglês do fórum de perguntas e respostas Yahoo Answers, um conjunto de artigos da Wikipédia em inglês, a base de conhecimento da NELL e um conjunto de exemplos sementes. Com isso, foram realizadas tarefas de pré-processamento dos dados com o intuito de extrair exemplos rotulados e não rotulados, os quais foram classificados por um algoritmo de aprendizado de máquina para definir exemplos candidatos a subcategorias. Um segundo módulo realiza um procedimento de validação baseado em probabilidade condicional. Os resultados mostraram que o componente, além de alcançar desempenhos adequados em termos do aprendizado de subcategorias, manteve uma taxa de falsos positivos relativamente baixa.

Palavras-chave: Inteligência Artificial, Aprendizado de Máquina, Aprendizado Sem Fim, Expansão de Ontologia.

ABSTRACT

In recent years, ontologies have been used in information systems to index large *corpora* of documents or collections of facts and directly support user interaction with the system through functionalities such as navigation and searches. Both structure and content of ontologies must come with these changes, over time, without losing coherence. Expansion of ontologies is primarily an organizational process and there must be rules for the processes of updating, inserting and exclusion from the ontology. After learning millions of facts extracted from the web, NELL (Never-ending Language Learning), the first never-ending machine learning system described in the literature that continuously extracts facts (reading the web) to increase its knowledge base and learn to read better than the previous day, began to learn beyond the knowledge extracted and to infer new beliefs that it had not yet read before, becoming able to expand its initial ontology through some contributions. In this way, the present thesis proposes a sequential modular computational model that allows the expansion of the ontology of the NELL knowledge base, identifying and classifying subcategories of the categories already known by the NELL ontology. The proposed component receives as inputs question texts in English from the Yahoo Answers forum, a set of English Wikipedia articles, the NELL knowledge base and a set of seed examples. From this, preprocessing tasks were done to extract labelled and unlabelled examples, which were classified by a machine learning algorithm that define the new candidates to subcategories. A second module performs a validation procedure based on conditional probability. The results showed that the component, in addition to achieve adequate performances in terms of subcategories learning, maintains a relatively low rate of false positives.

Keywords: Artificial Intelligence, Machine Learning, Never-ending Language Learning, Ontology Evolution.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

Capítulo 1

INTRODUÇÃO

A *Never-ending Language Learning* (NELL) é o primeiro sistema de aprendizado de máquina sem fim descrito na literatura (CARLSON et al., 2010a). Uma das características deste sistema é ser capaz de, autonomamente, interagir com usuários (seres humanos) de redes sociais para continuamente melhorar sua capacidade de aprendizado. Nesse sentido, a NELL foi também o primeiro sistema computacional a se utilizar do processo de *Conversing Learning* (tradução livre do inglês "aprender conversando") (PEDRO; HRUSCHKA, 2012; PEDRO; APPEL; HRUSCHKA, 2013).

Uma instância da NELL teve início em 2010 e está ativa até hoje com o objetivo de aprender fatos a partir de textos da *web* escritos na língua inglesa. A estrutura de sua base de conhecimento é uma ontologia, com categorias, instâncias, relações e fatos. Atualmente, sua base de conhecimento passou dos oitenta milhões de fatos. Entretanto, a base de conhecimento da NELL, mesmo sendo considerada grande, não possui especialização em nenhuma área do conhecimento.

Segundo Mitchell et al. (2018), durante os seis primeiros meses da NELL, suas únicas tarefas eram classificar os sintagmas nominais[1] em categorias e os pares de termos nominais em relações. A NELL, dado que alcançou algum nível de competência e cresceu consideravelmente sua base de conhecimento, tornou-se apta a enfrentar tarefas mais desafiadoras. Assim, algumas pesquisas foram iniciadas com o objetivo de expandir a base de conhecimento da NELL e a capacidade de melhorar sua aquisição de conhecimento, como em Pedro e Hruschka (2012), Dalvi et al. (2015) e Yang e Mitchell (2016).

[1] Os sintagmas são formados por constituintes, que são um conjunto de palavras de uma sentença que a divide, normalmente, em duas subpartes básicas: sintagma nominal e sintagma verbal. Quando o núcleo for um verbo, tem-se um sintagma verbal e quando for um substantivo têm-se um sintagma nominal (LOBATO, 1986 apud SANTOS, 2005)

Vale destacar a pesquisa de Souza e Hruschka (2016), em que foi proposta uma linguagem de conversação cognitiva[2](CCL) para agentes computacionais como *chatbots* e sistemas de perguntas e respostas[3]. O diferencial da CCL frente às outras é a possibilidade de receber regras que permitem que o agente de conversação possa acessar informações disponíveis na base de conhecimento da NELL. Portanto, o agente pode responder não apenas o que está nas regras, mas também pode elaborar respostas diferentes para várias categorias de perguntas com a mesma estrutura de pergunta e resposta. Outra vantagem da CCL é que a estrutura das regras pode ser criada, alterada ou removida não só manualmente, mas também de forma automática pelo próprio agente.

1.1 Justificativa e motivação

A construção, mineração e raciocínio automático de bases de conhecimento passaram a ser possíveis e consideradas ultimamente como pesquisa avançada em várias áreas, como: extração de informação, Processamento de Língua Natural (PLN), mineração de dados, *sites* de busca, aprendizado de máquina, banco de dados e integração de dados (REN et al., 2018).

A NELL (através de seus componentes) tem capacidade de aprender novos conceitos e novas relações, com o objetivo de popular cada vez mais e melhor a ontologia inicial.

Contudo, ainda existem desafios científicos e de engenharia a serem explorados no sentido de avanço e integração de metodologias, como no trabalho de Souza e Hruschka (2016), onde foi encontrada uma limitação na base de conhecimento da NELL. Notou-se que certas categorias da NELL não possuem subcategorias, as quais são essenciais para que um agente de conversação possa retornar uma informação válida. Um exemplo disso seria solicitar para um agente de conversação, que utiliza a CCL para acessar a base de conhecimento da NELL, uma lista de filmes de comédia. Nesse caso, a NELL possui uma categoria "filme"[4], mas não contém informações de quais instâncias da categoria "filme" são de comédia. Com isso, o agente retornaria uma lista de filmes, mas não necessariamente que sejam filmes de comédia. Identificou-se assim, que a NELL não possui uma técnica ou método de especialização do seu conhecimento sobre as categorias já aprendidas em sua base de conhecimento.

Esta limitação faz com que a base de conhecimento não seja tão adequada para ser usada por agentes de perguntas e respostas, pois o agente de conversação daria uma resposta incorreta mesmo que a regra da CCL esteja correta. Observa-se, portanto, que a proposição de uma

[2]Linguagem de Conversação Cognitiva: do inglês *Cognitive Conversation Language - CCL*.
[3]Sistemas de Perguntas e Respostas: tradução livre do inglês *Question Answering Systems - QAS*.
[4]O nome da categoria está originalmente em inglês (*movie*) na base de conhecimento.

abordagem de expansão autônoma de ontologia é necessária para ser possível responder perguntas com maior grau de acerto, principalmente ao se tratar de uma base de conhecimento como a NELL. Neste sentido, pode-se dizer que a principal contribuição desta tese está atrelada a garantir que a NELL possa ser utilizada em aplicações de perguntas e respostas.

1.2 Objetivos

O objetivo principal desta tese é propor um método que identifique novas subcategorias a partir das categorias já conhecidas pela base de conhecimento da NELL.

A partir do método proposto pretendeu-se desenvolver um componente para expandir a ontologia da base de conhecimento da NELL utilizando textos de fóruns de perguntas e respostas para classificar subcategorias das categorias já conhecidas pela NELL.

Com base no objetivo principal desta tese, ainda podem ser pautados objetivos específicos necessários. Assim, tais objetivos visam garantir que o componente:

- acesse a base de conhecimento da NELL e identifique categorias conhecidas pela mesma;
- receba como entrada os textos de perguntas feitas por humanos na língua inglesa;
- extraia atributos e gere exemplos candidatos a subcategorias;
- aprenda a classificar novos exemplos de candidatos a subcategoria a partir de um conjunto de exemplos rotulados;
- valide a classificação realizada por meio do uso de uma base de dados independente das utilizadas nos passos anteriores.

1.3 Estrutura do documento

Inicialmente, foi apresentado nesse documento a contextualização do trabalho seguida da justificativa, motivação e objetivos.

A fundamentação teórica, que servirá de base para entendimento do projeto, encontra-se dividida em dois capítulos. Assim, o Capítulo 2 apresenta trabalhos relacionados a ontologias com foco em abordagens de técnicas de expansão automática de bases de conhecimento. Na sequência, o Capítulo 3 é destinado à NELL, a qual representa uma nova abordagem para o aprendizado de máquina e que tem aumentado sua capacidade de adquirir conhecimento ao longo dos anos. Atualmente, a NELL está em sua fase de expansão da sua base de conhecimento, em que diferentes técnicas contribuem para o crescimento, atualização, correção e

expansão da mesma.

No Capítulo 4 é apresentado o componente de aprendizado automático de subcategorias a partir de textos de perguntas e respostas. Em seguida, o Capítulo 5 são detalhados os três experimentos realizados para a validação dos dois modelos computacionais propostos, assim como os resultados e análises.

Por fim, no Capítulo 6 estão as conclusões assim como os tópicos relevantes levantados durante o trabalho realizado, mas que estiveram fora do escopo do trabalho apresentado.

Capítulo 2

ONTOLOGIAS

Ontologias, na área da computação, são definidas na literatura de diferentes maneiras. De acordo com Zablith et al. (2015), ontologias são conceitos formais da engenharia de conhecimento, projetados para representar o conhecimento relacionado a um domínio específico (ontologia de domínio) ou genérico em conceitos relevantes (ontologia superior), relações entre esses conceitos e instâncias dos mesmos. Uschold (1996) define uma ontologia sendo um "*modelo do nível de conhecimento*"[1].

Encontram-se na literatura trabalhos de pesquisa em engenharia de conhecimento que abordam: (i) o desenvolvimento de ontologias; (ii) ferramentas e técnicas de manipulação, aquisição, elicitação e representação do conhecimento; e (iii) aplicações de ontologias em diferentes áreas (ZABLITH et al., 2015). Outras abordagens focam na expansão de ontologias[2] em diferentes aplicações, a partir de uma ontologia pré-existente. Cabe ainda mencionar que existem diferentes técnicas abordadas na literatura para expansão de ontologias (STOJANOVIC, 2004; NOY; KLEIN, 2004).

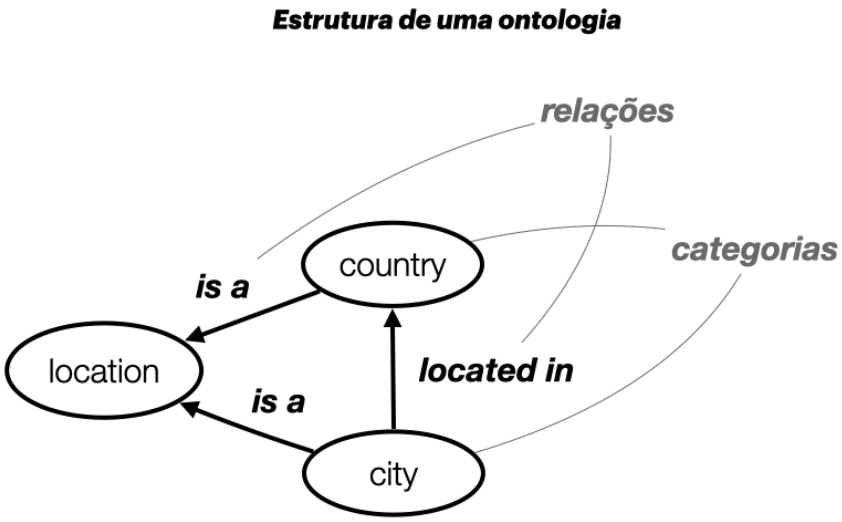
Destacam-se também outras abordagens que utilizam grandes bases de conhecimento utilizando estruturas baseadas em ontologias para representar o conhecimento humano a partir de dados não estruturados. Entre eles alguns se destacam por se basearem na *Wikipedia*, como DBPédia (LEHMANN et al., 2015), YAGO (SUCHANEK; KASNECI; WEIKUM, 2008; HOFFART et al., 2013) e Wikidata (VRANDEČIĆ; KRÖTZSCH, 2014).

De maneira geral, a base de conhecimento da NELL (CARLSON et al., 2010a) possui uma ontologia como estrutura para organizar e representar o conhecimento aprendido pelos seus componentes, os quais serão apresentados no capítulo 3. Nas Figuras 1 e 2 encontra-se ilustrada, de forma simplificada, como a NELL estrutura seu conhecimento adquirido em uma ontologia.

[1] Tradução livre do inglês Knowledge Level Model.

[2] Expansão de Ontologia: do inglês *Ontology Evolution* ou *Ontology Expansion*.

Figura 1: Exemplo de representação do conhecimento da NELL (CARLSON et al., 2010a) através da estrutura de uma ontologia.



Fonte: elaborada pelo autor

Na Figura 1 são ilustrados exemplos de categorias e relações em um grafo direcionado, onde os nós representam as categorias em uma ontologia como país, cidade e localização (*country*, *city* e *location* em inglês) e as arestas direcionais representam as relações entre as categorias, como "localizado em" e "é um" (*located in* e *is a* em inglês).

Na Figura 2 são listados exemplos das instâncias das categorias e relações apresentadas na Figura 2. As instâncias de categorias são listadas à esquerda da figura e as de relação à direita. Como exemplos da relação *locate_in(city, country)*, para expressar um fato de uma determinada cidade estar localizada em um determinado país pode-se armazenar a instância *located_in(amsterdam, netherland)*, ou seja, a cidade de *Amsterdam* está localizada na Holanda.

2.1 Expansão de ontologia

Uma ontologia pode ser considerada um modelo de representação de conhecimento de um determinado domínio. O conceito que uma ontologia representa sofre constantes alterações ou modificações durante o tempo, por isso a expansão da mesma é importante.

Nos últimos anos, ontologias são utilizadas em sistemas de informação para indexar grandes *corpora* de documentos ou coleções de fatos e apoiar diretamente na interação do usuário com o sistema através de funcionalidades como navegação e consulta. Tanto a estrutura quanto o conteúdo das ontologias devem acompanhar essas mudanças temporalmente sem perder coerência.

Figura 2: Exemplo de representação do conhecimento da NELL (CARLSON et al., 2010a) através das instâncias de uma ontologia.

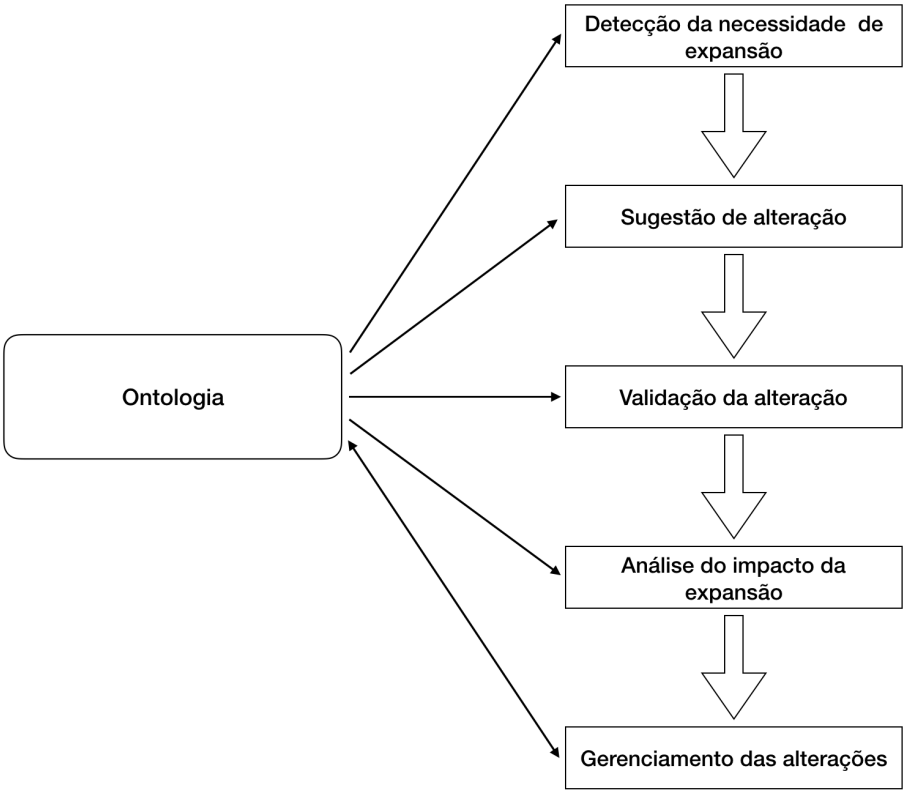
Fonte: elaborada pelo autor

Segundo Zablith et al. (2015), a expansão da ontologia pode ser definida como uma atualização baseada em mudanças necessárias, a qual pode ser gerenciada com um controle de versões. Neste sentido, os autores dividem todo o processo de expansão de ontologia em um fluxo de cinco fases, conforme mostrado na Figura 3. Nota-se ainda que esse processo pode ser executado de forma cíclica.

Cada um dos blocos que compõem a Figura 3 são definidos em maiores detalhes no que segue:

- *Detecção da necessidade de expansão* – essa é a fase que inicia o processo de expansão. A detecção pode ser iniciada a partir do uso ou análise dos dados fontes internos ou externos à ontologia;
- *Sugestão de alteração* – é o estado que representa ou sugere alterações que podem ser aplicadas à ontologia. Algumas abordagens utilizam técnicas que extraem padrões de dados textuais não estruturados que possuem representações do domínio da ontologia. Outras abordagens utilizam dados estruturados, como ontologias disponíveis na *web*, para fazer sugestões de alterações;
- *Validação da alteração* – é a etapa que avalia se a ontologia terá incoerências ou inconsistências após a atualização. As abordagens encontradas na literatura propõem basicamente validações de duas formas. A primeira é garantindo que a ontologia se mantenha logicamente consistente dentro de um limite especificado previamente. A outra valida o

Figura 3: Fluxo do processo de expansão da ontologia.



Fonte: elaborada pelo autor

domínio das alterações focando na relevância do mesmo após a alteração;

- *Análise do impacto da expansão* – esta etapa mede o impacto em componentes externos que são dependentes da ontologia, como outras ontologias e aplicações, por exemplo. Outra forma em que o impacto possa ser medido é pela determinação de critérios como cálculos de custos e benefícios das mudanças propostas;
- *Gerenciamento das alterações* – Nesta etapa são aplicadas as alterações requeridas, como adição, atualização ou remoção dos dados. Também é executado o controle de versões da ontologia, que é útil no caso que seja necessário realizar um *backup* ou retornar a ontologia para um estado anterior sem haver qualquer perda. Esta tarefa pode ser executada durante todo o processo de expansão da ontologia (ZABLITH et al., 2015).

2.2 Extração do conhecimento

Segundo Cowie e Lehnert (1996), a área que estuda a Extração da Informação[3] compreende qualquer processo que seleciona, combina ou estrutura dados em um conjunto de textos e os armazenam em uma base de dados. A Gestão do Conhecimento intersecciona-se com a área da Extração da Informação quando o objetivo da combinação, estruturação e armazenamento dos dados é, de certa forma, proporcionar qualquer tipo de conhecimento, no qual é possível realizar inferências além da informação explicita, encontrada nos dados. Pode-se assim afirmar que a manipulação automática de ontologias pode ser vista como uma subárea da Gestão do Conhecimento, podendo ser definida como uma área de pesquisa que integra a criação, captura, organização, acesso e uso de informações de uma determinada estrutura (GIRARD; GIRARD, 2015).

Dos trabalhos encontrados na literatura, alguns propõem e pesquisam linguagens de representação de ontologias, sendo a *OWL* (*Web Ontology Language*) uma das mais conhecidas (MCGUINNESS; HARMELEN et al., 2004).

Em Caraballo (1999) foi apresentado o modelo de desenvolvimento automático de hierarquias de hiperônimos de substantivos, o qual contribui em outras abordagens que dependem da identificação de relações semânticas entre as palavras em um determinado domínio.

O trabalho de Cimiano, Hotho e Staab (2005) apresenta uma abordagem para aprender automaticamente uma hierarquia de conceitos a partir de um *corpus* de texto, utilizando o método baseado em relações entre objetos e fundamentados nos atributos morfossintáticos extraídos dos mesmos, bem como os verbos, sujeito e as dependências entre eles. Os resultados obtidos na abordagem foram comparados com o modelo estatístico, em que o algoritmo de *k-means* é utilizado a partir da extração de *N* gramas. Assim, o modelo proposto foi capaz de obter um melhor resultado. Os autores concluíram que apesar de a abordagem ser totalmente automática, é importante a inclusão do usuário no processo, sendo que este deve ter uma interação mínima na avaliação dos resultados (ou *feedback*) para minimizar a propagação de erros que podem ocorrer durante o processo de expansão da ontologia.

Existem diferentes formas de extrair conhecimento e representá-los em uma ontologia, como a partir do uso de técnicas de PLN em textos de *corpora* ou de páginas da *web* (MAEDCHE; STAAB, 2001; CIMIANO; HOTHO; STAAB, 2005; STAAB; STUDER, 2010). Conforme os estudos feitos por Niklaus et al. (2018), existem cinco tipos de sistemas de extração do conhecimento, baseados em: (i) regras; (ii) cláusulas; (iii) relações entre proposições; (iv) aprendizagem de

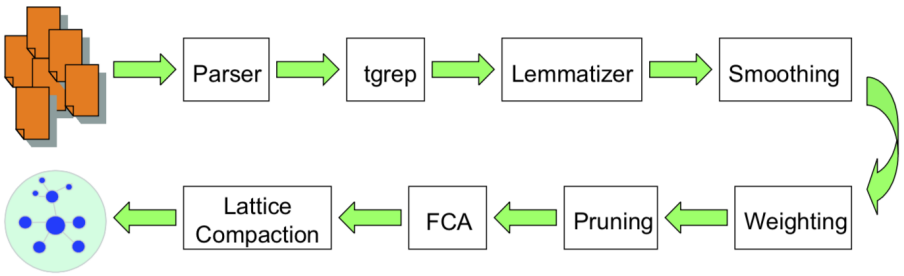
[3] Do inglês *Information Extraction* (IE).

máquina; e (v) leitura de máquina[4].

Vale destacar que o principal objetivo do *Machine Reading* é extrair automaticamente conhecimento de um *corpus* não estruturado (por exemplo, páginas da *web*) e representá-lo em um formato estruturado.

Um exemplo de aprendizado automático de ontologias é o trabalho de Cimiano, Hotho e Staab (2005) que se encontra ilustrado na Figura 4, onde foi proposto um modelo computacional que gera uma ontologia a partir de um *corpus*. Inicialmente, o *corpus* é processado por dois analisadores morfossintáticos responsáveis por gerar uma árvore sintática para cada sentença. Esta etapa é representada na Figura 4 pelo componente *Parser*. Em seguida, a partir da árvore gerada, são obtidos os pares de palavras de acordo com suas classes gramaticais. Os verbos e as primeiras palavras das sentenças são lematizados na sequência (componente *Lematizer*). Então, os pares extraídos e lematizados são contados em todo o *corpus* para se obter a quantidade de ocorrências para cada par. Com isso, é possível remover os dados que são considerados ruídos, ou seja, que ocorrem com menos frequência (essa etapa é representada pelo componente *Smoothing*). Medidas estatísticas são realizadas para calcular pesos (etapa representada pelo componente *Weighting*) dos pares. Desta forma, somente aqueles que obtiverem um peso acima de um valor crítico são transformados em um contexto formal. Na sequência é aplicada a Análise Formal de Conceitos (AFC ou FCA do inglês[5]), um método matemático utilizado para análise de dados. Então, o resultado do FCA passa por um algoritmo (componente *Lattice Compaction*) que gera os conceitos da ontologia.

Figura 4: Fluxo do processo de aprendizagem automática de ontologia.



Fonte: (CIMIANO; HOTHO; STAAB, 2005)

As abordagens que utilizam grafos de conhecimento o que inclui ortologias, têm sido exploradas em diferentes áreas para representar o conhecimento. Com isso, trabalhos recentes

[4]Leitura de Máquina: tradução livre do inglês *Machine Reading*.

[5]*Formal Concept Analysis*: nesse documento é usada a sigla FCA para corresponder à Figura 4 tirada do texto original.

abordam diferentes técnicas de extração automática do conhecimento populando, no que lhe concerne uma determinada ontologia.

Alguns trabalhos como os de Wang et al. (2017) e Yang et al. (2015) fazem a extração do conhecimento e populam bases de conhecimento gráficas realizam um processo chamado *embedding*, técnica que transforma os elementos da base de conhecimento (classes, relações e instâncias) em valores vetoriais contínuos, dispondo assim, os elementos da base de conhecimento em um espaço multidimensional. Esta técnica facilita algumas tarefas que podem ser realizadas com a base de dados por algoritmos que normalmente não utilizam valores nominais, sem deixar de preservar as relações semânticas entre os elementos.

A pesar das bases de dados com o processo de *embeddings*, ser um assunto relevante e atual, a técnica é utilizada para ser realizadas tarefas após a extração do conhecimento. Com uma ontologia (ou base de conhecimento gráfica) em expansão, o processo de *embedding* deve ser refeito para que as tarefas posteriores possam ser executadas em cima da base processada.

No trabalho de Dong et al. (2014), foi proposta uma base de conhecimento probabilística chamada *Knowledge Valt*, o trabalho utiliza diferentes fontes de dados e uma Base de dados pré-existente. Com os algoritmos que realizam tarefas como, tratar fontes correlatas, dados temporariamente verdadeiros, adição de novas informações e tratamento de erros, o resultado foi uma base de conhecimento 38 vezes maior do que a pré-existente utilizada.

2.3 Expansão de ontologias e bases de conhecimento

Segundo Stojanovic et al. (2002), a expansão de ontologias é um processo organizacional. Deve haver regras para a realização dos processos de atualizações, inserções e remoções da ontologia. É necessário verificar se a ontologia permite alterações, em caso positivo deve assegurar-se que essas manterão a coerência dos elementos que serão alterados, bem como de seus dependentes.

Outros trabalhos como Stojanovic (2004), Haase et al. (2005), Haase, Völker e Sure (2005) propõem abordagens com o objetivo de identificar e resolver inconsistências durante o processo de desenvolvimento ou expansão de ontologias de maneira automática, mesmo que a atualização da ontologia seja realizada de forma automática ou manual.

Em Stojanovic (2004) é proposto um método para sistemas de evolução de ontologias. No trabalho, o termo evolução é utilizado, pois, o domínio de aplicação se dá em cenário de mudanças eventuais no modelo de negócios. Assim, a ontologia receberá alterações relacio-

nadas à expansão do conhecimento, conforme a estrutura já existente e também deverá estar preparada para mudanças estruturais solicitadas por um usuário. Com o cenário proposto, o sistema deve garantir a consistência de todos os elementos da ontologia e todos os artefatos (que possuem alguma dependência), além de controlar as entradas de usuários com solicitações de mudanças sem perder históricos das mudanças anteriores. Segundo o autor, existem três principais desafios para a extensão eficiente da ontologia, a saber:

- Complexidade – ontologias são ricas de informações e possuem uma estrutura com muitas interdependências, portanto, cada mudança desencadeia uma sequência de tarefas a serem efetuadas;
- Dependência – ontologias podem estender ou reutilizar outras ontologias. Mudanças em uma ontologia podem afetar outras ontologias dependentes que, conforme a aplicação, podem requerer alterações nas mesmas;
- Distribuição Física – a construção da ontologia pode ser um processo descentralizado e colaborativo, no qual é preciso ter controle de todas as alterações aplicadas.

Em função desses desafios, o autor propõe uma abordagem multidimensional para evolução de uma ou mais ontologias, considerando o número de ontologias envolvidas e também a distribuição física das mesmas. Foi definida uma estrutura abrangente para a descoberta de mudanças baseadas em históricos e também no comportamento dos usuários que inserem as mudanças. A partir dos dados coletados foram propostos modelos heurísticos para identificação das mudanças. Dessa forma, a abordagem não representa um mero processo de gerenciamento de mudanças, mas um processo de melhoria contínua da ontologia.

Haase et al. (2005) propuseram uma abordagem com quatro diferentes versões para lidar com as inconsistências em ontologias. As quatro técnicas foram comparadas considerando aspectos sintáticos e semânticos, definições dependentes e não dependentes de linguagem, aspectos estruturais e lógicos. Por conta das quatro versões terem vieses diferentes, elas tornaram-se complementares na detecção de inconsistências. Por este motivo, os autores propuseram um *framework* multi-versão.

Na prática, alterar uma ontologia consiste potencialmente em introduzir inconsistências na mesma. Normalmente, essas inconsistências ocorrem quando as alterações são realizadas manualmente ou quando a ontologia é colaborativa e sofre alterações por diversos usuários. Assim, o tratamento de inconsistências é relevante não apenas durante o desenvolvimento da ontologia, mas também em tempo de execução de aplicações baseadas nessas ontologias.

Nos últimos anos, com o avanço computacional, tecnológico e das pesquisas na área, surgiram propostas de sistemas automatizados capazes de, a partir de uma base de conhecimento

inicial, extrair conhecimento de dados não estruturados, processá-los e popular a base de conhecimento inicial de modo a expandi-la, como *Snowball* (EUGENE; LUIS, 2000), *KnowItAll* (ETZIONI et al., 2004), *DeepDive* (SHIN et al., 2015) e *NELL* (CARLSON et al., 2010a).

Segundo Mitchell et al. (2018), um dos maiores desafios dos trabalhos que abordam sistemas automáticos de extração do conhecimento é o desenvolvimento de programas que, como os seres humanos, sejam capazes de:

- aprender diferentes tipos de conhecimento ou funções;
- auto-supervisionar a experiência adquirida;
- a partir do conhecimento aprendido previamente, aprender mais tipos de conhecimento;
- evitar que a auto-reflexão e a capacidade de formular novas representações e tarefas de aprendizagem permitam que o sistema aprendiz atinja platôs de estagnação e desempenho.

A NELL como um sistema que aprende extraindo conhecimento de textos da web para popular uma ontologia, depara-se com esses desafios. Ao se propor um novo componente para a NELL, diretamente ou indiretamente tem-se o objetivo de enfrentar um ou mais dos desafios listados.

Para garantir um sistema de expansão de ontologia eficiente, existem alguns requisitos que devem ser atendidos. Além disso, o processo de expansão de ontologia deverá ser capaz de lidar com mudanças (STOJANOVIC, 2004).

A partir dos trabalhos abordados, pode-se considerar que a expansão de ontologias é uma área de pesquisa promissora, dado que temporalmente as atualizações e expansões são essenciais para manter a ontologia coerente com o domínio no qual ela representa. Novos métodos e ferramentas para dar suporte a essa tarefa complexa pode ajudar na modificação fácil e consistente portanto, podem permitir o uso disseminado de ontologias em aplicações industriais e acadêmicas.

Capítulo 3

NEVER-ENDING LANGUAGE LEARNING - NELL

Proposta inicialmente por Carlson et al. (2010a), a NELL foi o primeiro estudo de caso de um agente que utiliza o aprendizado de máquina sem fim, o qual tem como tarefa aprender a ler a *web*, considerando as seguintes condições iniciais:

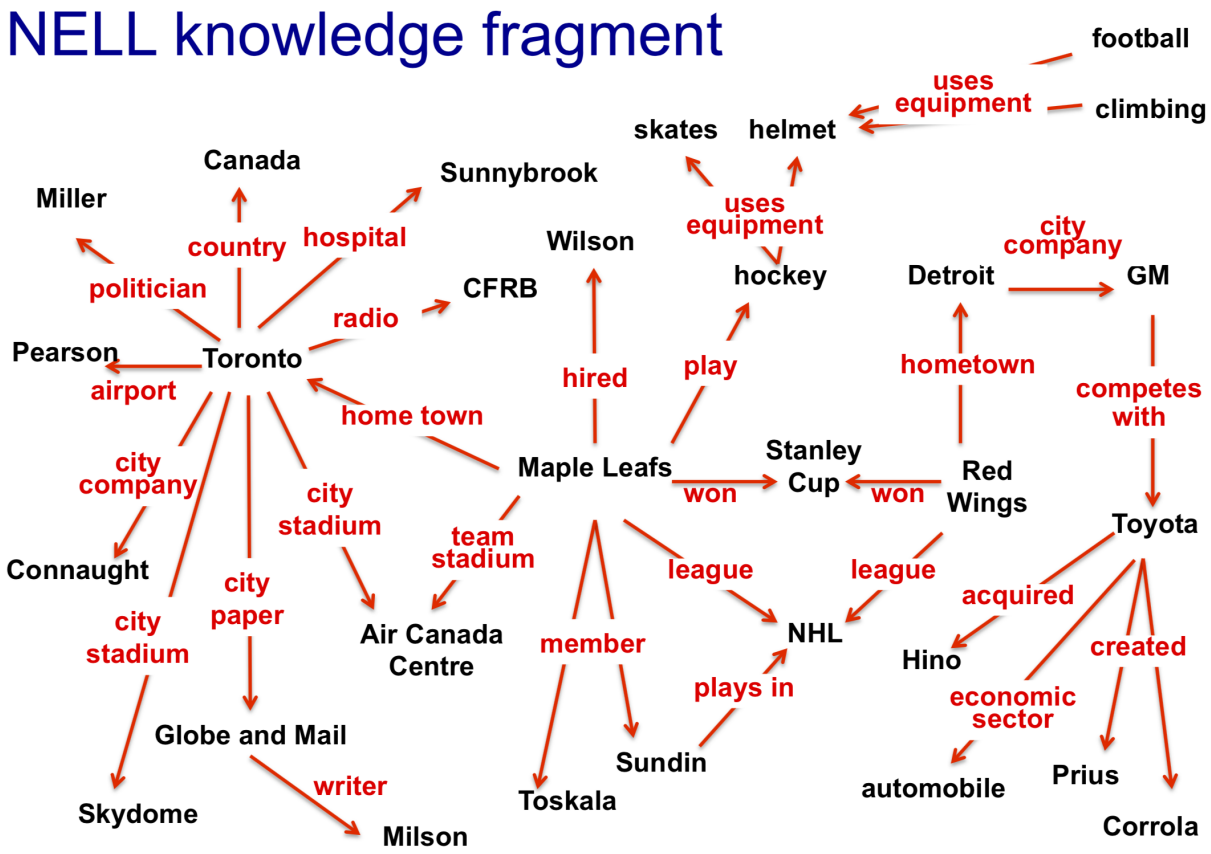
- Uma ontologia inicial com categorias definidas (exemplo: *Movie* e *Actor*) e relações binárias (exemplo: *ActorStarredInMovie(x,y)*);
- Cerca de uma dúzia de exemplos de treinamento rotulados para cada categoria e relação (para a categoria *Movie* pode-se incluir os sintagmas nominais "*Alien*" e "*Matrix*", por exemplo);
- A *web* (inicialmente uma coleção de quinhentas milhões de páginas *web* da ClueWeb (CALLAN et al., 2009), em 2009, e o acesso às páginas web através de uma API da Google com direito a cem mil pesquisas diárias);
- Interações ocasionais com seres humanos (por exemplo, através do *site* oficial da NELL: `http://rtw.ml.cmu.edu`).

Neste sentido, a cada iteração, a NELL executa ininterruptamente as seguintes tarefas: (1) Ler (extrair) mais crenças da *web* e remover as aprendidas incorretamente, para aumentar sua base de conhecimento; e (2) Aprender a ler melhor que o dia anterior.

Vale mencionar que a NELL está em execução desde janeiro de 2010, extraindo mais fatos da *web* e com auto-treinamento automático para melhorar suas competências. Até então, sua base de conhecimento possui mais de 80 milhões de crenças interconectadas. Um exemplo das crenças aprendidas pela NELL pode ser visto na Figura 5, onde cada aresta é uma tupla que representa uma crença da base de conhecimento da NELL (exemplo: *play(MapleLeafs, hockey)*) que está associada a um grau de confiança (não apresentado na ilustração).

Após aprender milhões de fatos extraídos da *web*, a NELL passou a aprender além do

Figura 5: Fragmento da base de conhecimento da NELL.



Fonte: (MITCHELL et al., 2018)

conhecimento extraído, inferindo em novas crenças que ainda não havia lido anteriormente. Portanto, agora, ela pode expandir sua ontologia inicial que fora concebida manualmente a partir do trabalho de Carlson et al. (2010a) e que possui diversas outras contribuições.

Segundo Mitchell et al. (2018), as tarefas que a NELL realiza para ler a *web* e aprender melhor a cada dia passam, atualmente, de 2500. Essas tarefas podem ser distribuídas nos grupos descritos a seguir:

- Classificação de categorias – funções que classificam sintagmas nominais através de categorias semânticas. A NELL, para cada categoria de sua base de conhecimento, aprende a partir de cinco diferentes funções que classificam os sintagmas nominais por visões distintas, são elas: recursos da cadeia de caracteres executado pelo sistema CMC (*Coupled Morphological Classifier*) (CARLSON et al., 2010a); distribuição de contextos textuais encontrados ao redor dos sintagmas nominais em 500 milhões de páginas *web* em inglês através do sistema CPL (*Coupled Pattern Learner*) (CALLAN et al., 2009; CARLSON et

al., 2010b); distribuição de contextos textuais encontrados ao redor dos sintagmas nominais utilizando páginas da *web* por *sites* de busca através do sistema OpenEval (SAMADI; VELOSO; BLUM, 2013); estrutura HTML de páginas *web* contendo sintagmas nominais através do sistema SEAL (WANG; COHEN, 2007); imagens associadas com os sintagmas nominais nos *sites* de busca de imagens através do sistema NEIL (CHEN; SHRIVASTAVA; GUPTA, 2013).

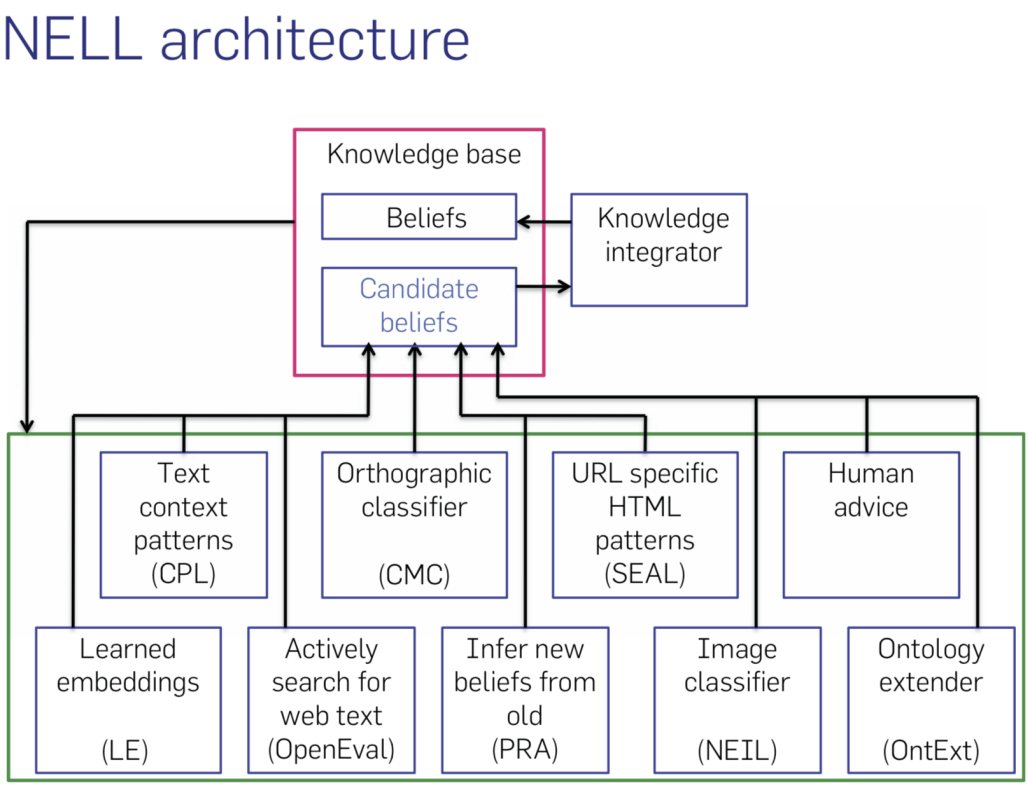
- Classificação de relações – funções que classificam pares de sintagmas nominais dependendo se esses satisfazem uma dada relação. A NELL utiliza três funções de classificação de relações baseadas em diferentes visões dos dados. Para classificar as relações, a NELL também utiliza os métodos do CPL e OpenEval baseando-se na distribuição dos contextos textuais encontrados entre dois sintagmas nominais. A NELL também utiliza o método de classificação SEAL para extrair conhecimento de páginas *web* através da leitura da estrutura da linguagem de marcação HTML;
- Definição de Entidade – funções que classificam pares de sintagmas nominais dependendo se os mesmos são sinônimos. Assim, esses pares podem ser considerados Entidades Nomeadas (ENs) (KRISHNAMURTHY; MITCHELL, 2011);
- Regras de inferência entre tuplas de crenças – funções que mapeiam a ontologia corrente da base de conhecimento da NELL para adicionar novas crenças. Para cada relação existente na ontologia da NELL, a função correspondente é representada por uma coleção de *Cláusulas de Horn*(DAVIS, 2007) aprendidas através do sistema PRA (LAO; MITCHELL; COHEN, 2011).

A arquitetura de *software* para a NELL é apresentada na Figura 6, onde a base de conhecimento (incluindo sua ontologia, entre outros elementos) está representada no retângulo vermelho. Também está representado com um retângulo verde, o componente com os módulos de inferência como CMC, CPL, SEAL, OpenEval e NEIL, com diferentes técnicas mencionadas anteriormente e outros como o LE (Learned vector embeddings) apresentado por Yang e Mitchell (2017) e o *OntExt* proposto por Mohamed, Hruschka e Mitchell (2011).

3.1 Expansão da base de conhecimento da NELL

A base de conhecimento da NELL, além de ser composta por atributos e características específicas para o uso do Aprendizado de Máquina sem fim, sua estrutura fundamental é uma ontologia, onde os conceitos e relações aprendidos são armazenados. Para expandir a ontologia da base de conhecimento da NELL, alguns trabalhos como Dalvi, Cohen e Callan (2013), Yang e Mitchell (2016), Mohamed, Hruschka e Mitchell (2011) e Settles (2011) propõem técnicas

Figura 6: Arquitetura da NELL.



Fonte: (MITCHELL et al., 2018)

para que essa expansão possa ser realizada de maneira automática.

Em Mohamed, Hruschka e Mitchell (2011), é proposta uma técnica de expansão da ontologia adicionando novas relações que conectam grupos de instâncias conhecidas entre pares de categorias. Assim, foi utilizada como entrada do componente um *corpus* externo para que a solução proposta retornasse resultados com baixo grau de incerteza.

No trabalho de Settles (2011), é proposta uma técnica para adicionar novas subcategorias, que utilizam as informações que já estão contidas na ontologia (*self-discovered*) através da análise de instâncias já conhecidas.

Outra técnica utilizada pela NELL para expandir sua ontologia se dá pelo componente *OntExt*, o qual busca novas relações que usa todos os pares de categorias na ontologia corrente da NELL. Dessa forma, busca-se evidências de uma nova relação que possa ocorrer frequentemente entre determinados pares.

A principal tarefa do *OntExt* é descobrir novas relações entre as principais categorias já conhecidas. O método proposto cria uma matriz de coocorrência dado um conjunto de contextos.

Assim, o modelo busca por categorias que possuem possivelmente alguma relação entre elas.

Para exemplificar seu funcionamento, pode-se considerar três categorias do domínio "futebol": "jogador", "clube" e "liga". Dadas as categorias, supõe-se que a NELL tem conhecimento prévio sobre duas relações entre essas categorias, são elas: "joga_em(jogador, clube)" (jogador joga em clube) e "participa_de(clube, liga)" (clube participa de liga), além do conhecimento de alguns fatos como "joga_em(Neymar, PSG)" e participa_de(PSG, Champions League).

Assim, com a matriz de coocorrência, outros dados e contextos disponíveis na Base de Conhecimento da NELL, o componente *OntExt* pode identificar e propor uma nova relação a partir de um novo fato como "Neymar jogou na Champions League" e, com isso, pode-se ter uma possível relação entre as categorias "jogador" e "liga"

Cabe mencionar que nenhum dos trabalhos relacionados à expansão da ontologia da NELL teve o objetivo de encontrar novas subcategorias das categorias já aprendidas. Outra observação a ser considerada é que o escopo desses trabalhos estão voltados somente à expansão do conhecimento e não à especialização do mesmo. Portanto, este pode ser destacado como um dos aspectos inovadores desta tese de doutorado.

Capítulo 4

APRENDIZADO AUTOMÁTICO DE SUBCATEGORIAS A PARTIR DE TEXTOS DE PERGUNTAS E RESPOSTAS

Conforme o que foi apresentado no Capítulo 1, o objetivo desta tese de doutorado é propor um componente para expandir a ontologia da base de conhecimento da NELL, utilizando textos de fóruns de perguntas e respostas para classificar sintagmas nominais como subcategorias das categorias já conhecidas pela NELL. Em outras palavras, o componente proposto será, através dos textos de entrada e de uma dada ontologia, um modificador de Entidades Nomeadas (EN), o qual especificará um conceito já existente. Portanto, nesse capítulo será apresentado o componente de expansão automática de ontologia, assim como seus submódulos de classificação e validação.

Conforme previamente apresentado, é possível extrair características de textos que são perguntas, as quais são úteis para contribuir com a expansão de uma ontologia. Portanto, um dos escopos deste projeto é explorar o ambiente de perguntas e respostas, focando em textos que são perguntas, não para propor melhorias nesse meio, mas sim contribuir para a expansão de determinadas ontologias.

Propõe-se um sistema modular que usa técnicas de aprendizado de máquina para extrair padrões dos textos com uso de ferramentas de PLN.

O componente foi desenvolvido na linguagem de programação Python, versão 3.6 (ROSSUM; DRAKE, 2009), através de frameworks de aprendizado de máquina como Scikit-Learn (BUITINCK et al., 2013) e Extreme Gradient Boosting (CHEN; GUESTRIN, 2016), além de Interfaces de Programação de Aplicativos (API) de Processamento de Língua Natural como NLTK (BIRD; KLEIN; LOPER, 2009) e Spacy (HONNIBAL; MONTANI, 2017). Para executar alguns ex-

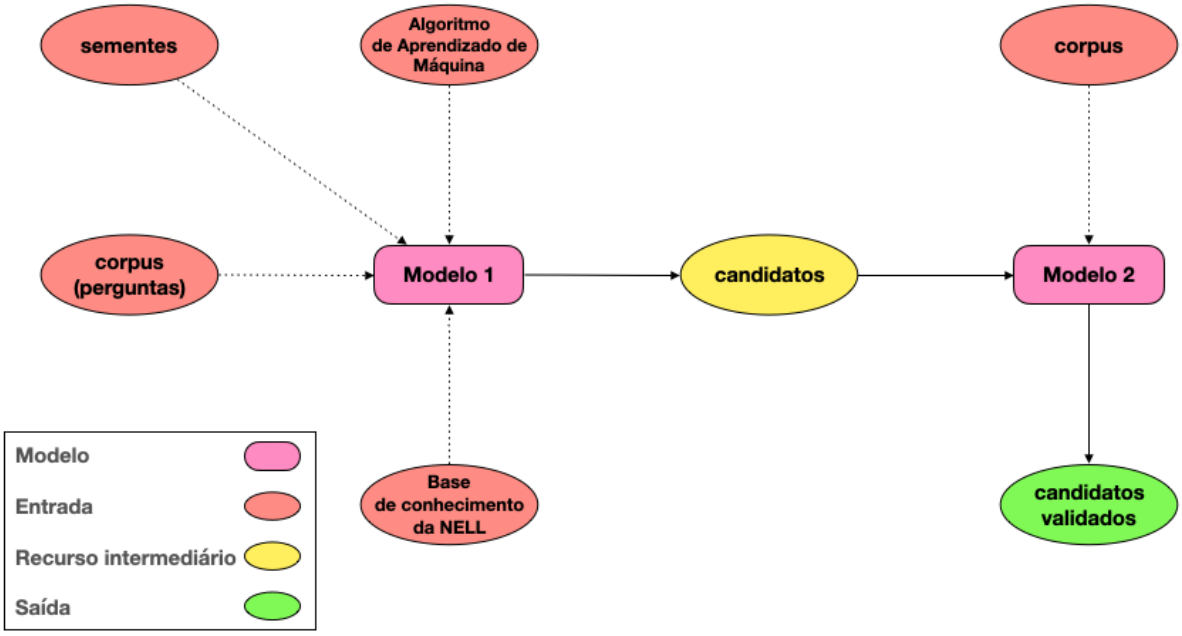
perimentos do componente em menor tempo, parte do código foi carregada em uma imagem de *container Docker* (MERKEL, 2014). Dessa forma, a mesma foi executado em uma máquina virtual no serviço de computação em Nuvem da AWS[1].

O sistema será dividido inicialmente em dois modelos computacionais (nomeados por Modelo 1 e Modelo 2), os quais serão descritos a seguir.

4.1 Visão geral do componente

Na Figura 7 é apresentada a visão geral do componente proposto, o qual é responsável por integrar os dois modelos computacionais. De maneira geral, o componente proposto irá identificar subcategorias de uma dada categoria, contida na base de conhecimento da NELL, a partir de textos extraídos de um *corpus* de perguntas e respostas.

Figura 7: Fluxo de execução geral do componente proposto.



Fonte: elaborada pelo autor

O Modelo 1 (primeiro modelo computacional que compõe o componente proposto) é responsável por extrair atributos de sentenças do *corpus* de perguntas a partir dos exemplos sementes. Com isso, o mesmo irá buscar por novos exemplos contidos no *corpus* e um classificador

[1] Amazon Web Services (AWS) - Cloud Computing Services, disponível em: `https://aws.amazon.com/` - acessado em: 07/12/2020.

baseado em aprendizado de máquina será treinado para identificar possíveis candidatos à subcategoria das categorias já conhecidas pela base de conhecimento da NELL. Esses candidatos selecionados servirão de entrada para o Modelo 2 (segundo modelo computacional que compõe o componente proposto) que, no que lhe concerne, fará uma validação probabilística utilizando um *corpus* separado em relação ao usado para treinamento do Modelo 1.

Como exemplificação, dada uma semente "*horror movie*", onde *horror* seria uma subcategoria para *movie*, o Modelo 1 irá buscar sentenças no *corpus* de perguntas, como essa: "*Does anyone know about any really good new horror movies that have come out lately?*[2]". Em seguida, são extraídos atributos da sentença formando um exemplo de um conjunto de dados como apresentado na Tabela 1.

Tabela 1: Exemplo de uma semente com atributos extraídos a partir de um *corpus* e da base de conhecimento da NELL.

Features	Values
ngrams	horror _
category	movie
occurrences	1,414347e-15
extraction pattern occurrences	2,649716e-17
min similarity	0,001154
max similarity	0,002719
signature	NOUN NOUN
label	1

Fonte: elaborada pelo autor

Na Tabela 1 é apresentado um exemplo de dado rotulado onde "*ngrams*" é o atributo de identificação do exemplo, "*category*" é a categoria conhecida na base de conhecimento a qual o exemplo está inserido, "*occurrences*" é o número de ocorrências em que o exemplo ocorre nos dados, "*extraction pattern occurrences*" é o número de ocorrências em padrões de extração aprendidos por um componente da NELL chamado CPL (CARLSON et al., 2010b), "*min similarity*" e "*max similarity*" são valores extraídos a partir de uma base de dados lexicais e conceituais chamada *WordNet* (MILLER, 1998) utilizando um cálculo de similaridade chamado *Wu-Palmer Similarity*, "*signature*" representa a sequência de classes morfossintáticas do exemplo e por fim, "*label*" representa o atributo classe do exemplo (nesse caso já rotulado) onde 1 e 0 são os possíveis valores que sinalizam ser um exemplo positivo ou negativo respectivamente. O processamento e pós-processamento executado para a obtenção dos dados serão apresentados no Item 4.2.3.

[2]Tradução: "Alguém sabe sobre algum filme de terror realmente bom que foi lançado recentemente?"

Na sequência, são identificados pares de possíveis subcategorias utilizando o *corpus* de perguntas com as sentenças que não foram utilizadas nas sementes. Assim, os mesmos atributos são extraídos e então é gerado um conjunto de exemplos que, dessa vez, não possui o atributo classe (*label*). Cabe mencionar que a validação realizada pelo Modelo 2 é independente dos atributos extraídos pelo Modelo 1, exceto pela redução do grau de incerteza existente devido aos exemplos já terem sido submetidos a um processo de classificação. Nas seções que seguem, serão apresentadas as propriedades de cada um dos modelos computacionais.

4.2 Modelo 1: Identificação de subcategorias a partir de perguntas e respostas

O Modelo 1 recebe como entrada a ontologia da base de conhecimento da NELL, um *corpus* de textos de perguntas e respostas e um conjunto de exemplos de subcategorias como sementes. A Figura 8 ilustra o fluxo de processos realizados pelo modelo computacional, os quais serão descritos em detalhes no decorrer desta seção. Além dos dados de entrada descritos, o modelo gera alguns recursos entre os processos como os dados selecionados, não selecionados, dados rotulados e não rotulados. O recurso de saída do modelo é um conjunto de dados classificados por um algoritmo de aprendizado de máquina.

O *corpus* de entrada passa por um pré-processamento textual de limpeza antes de ser disponibilizado como entrada. Dentre os processos usados nesse pré-processamento estão: remoção de caracteres e pontuações duplicadas, transformação de todas as letras em minúsculas e remoção de alguns caracteres especiais.

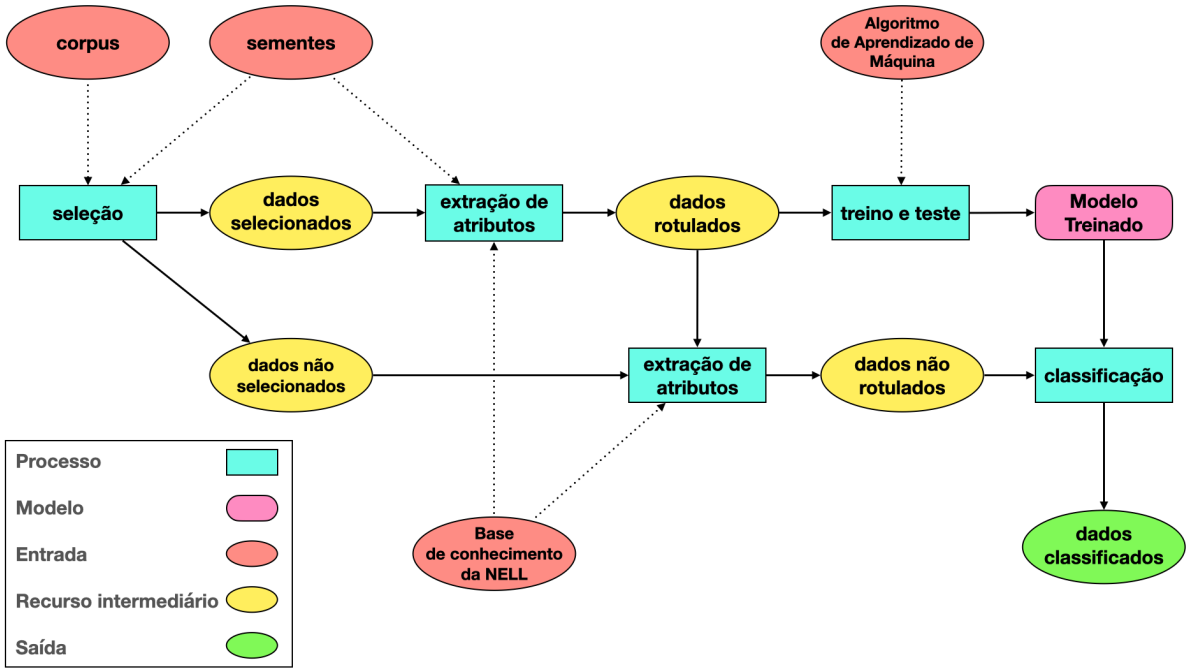
4.2.1 Seleção

Inicialmente é realizada a seleção dos dados a partir dos exemplos de subcategorias sementes. Nesse estágio, o modelo irá extrair dois conjuntos de dados. O primeiro é o conjunto de perguntas associadas aos exemplos sementes e o segundo conjunto de dados são as perguntas e respostas que não tiveram associações com os exemplos sementes.

Para exemplificar a seleção dos dados descrita, considere as seguintes as subcategorias da Tabela 2 como sementes dadas como entradas ao modelo.

Na Tabela 2, o exemplo semente representa que *horror* é subcategoria de *movie* (filme de horror). Com o exemplo de subcategoria dado, o algoritmo que faz a tarefa de seleção buscará no *corpus* de perguntas e respostas as sentenças onde ocorrem na sequência indicada pela sub-

Figura 8: Visão geral do fluxo de tarefas do Modelo 1



Fonte: elaborada pelo autor

Tabela 2: Exemplos de subcategorias para as categorias *movie* (linha 1) e *actor* (linha 2).

#	categoria	candidata	rótulo
1	movie	horror _	sim
2	actor	comic _	sim

Fonte: elaborada pelo autor

categoria. O modelo aceita exemplos negativos de modo a se obter um melhor desempenho na tarefa de aprendizado, reduzindo a taxa de erro na identificação das fronteiras de classificação pelo algoritmo de aprendizado utilizado.

Para que o processo de seleção não descarte exemplos relevantes, mas que não correspondem aos caracteres dos exemplos sementes, são utilizadas técnicas de pré-processamento textual, como: tokenização, *stemming* e uso de dicionário de sinônimos. As técnicas foram feitas com uso do pacote de recursos NLTK chamado *Wordnet* (BIRD; KLEIN; LOPER, 2009).

Para cada semente é gerada uma expressão regular considerando as posições da subcategoria e categoria na sentença. Para capturar mais exemplos e variações linguísticas, as palavras das sementes passam pelo processo de *stemming*. Por exemplo, na semente *"adventure movie"*,

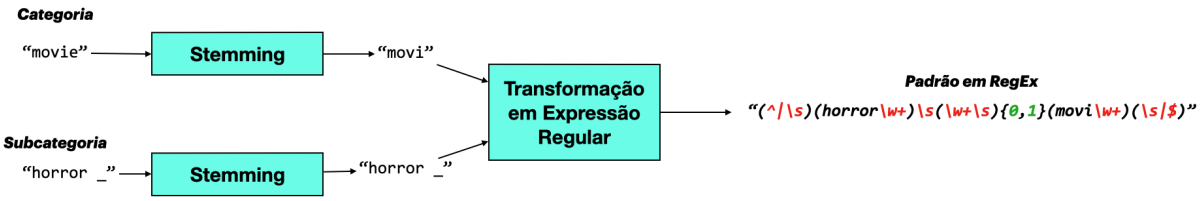
Tabela 3: Exemplos de sentenças associadas à subcategoria *horror movie* (filme de horror).

#	sentença
1	I need a list of horror movies
2	What is your favorite horror movie?

Fonte: elaborada pelo autor

a expressão regular gerada será "$(\wedge \mid \backslash s)(adventur\backslash w+)\backslash s(\backslash w+\backslash s)\{0,1\}(movi\backslash w+)(\backslash s \mid \$)$". Na Figura 9 é apresentado um exemplo do fluxo de transformação de determinada entrada (*horror movie*) em um padrão de expressão regular.

Figura 9: Fluxo de transformação de um exemplo em uma expressão regular.



Fonte: elaborada pelo autor

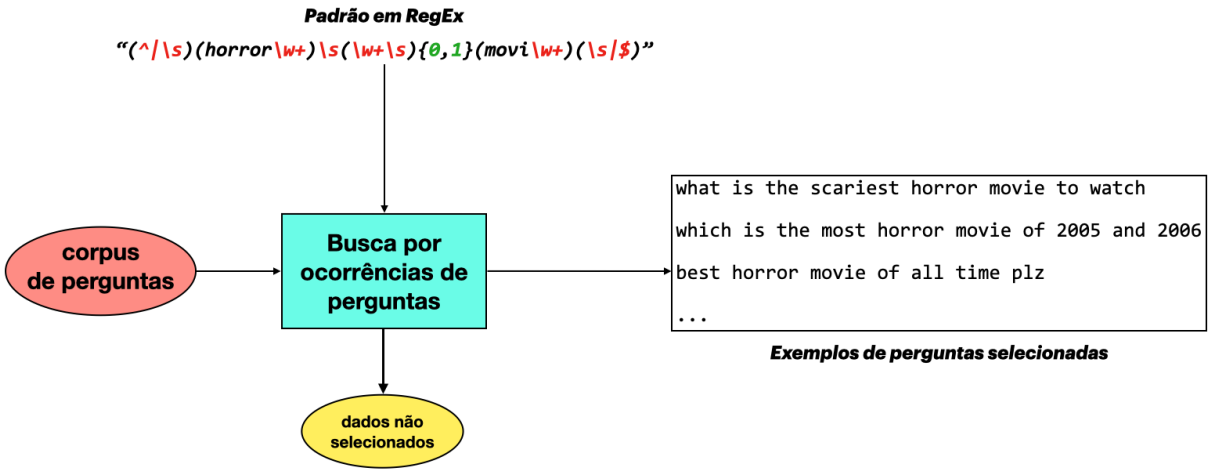
Com a expressão regular gerada, são selecionadas as sentenças do conjunto de perguntas que atendam os padrões da expressão regular, como as seguintes:

- what are good adventure movies;
- which are your favorite adventure movies;
- a good song does anyone kno a good song that will go alond with an adventure movie and is kind of sad cuz the main character dies.

Na Figura 10, é apresentado o fluxo de busca por perguntas, em que a tarefa principal é representada pelo retângulo com o texto "Busca por ocorrências de perguntas", que recebe como entrada o *corpus* de perguntas a expressão regular (padrão em *RegEx*).

Ao final do processo de seleção, cada exemplo semente terá um conjunto de sentenças associado. Esse conjunto de sentenças é então considerado o *conjunto selecionado* enquanto o conjunto de sentenças que não possui um exemplo semente associado é considerado o *conjunto não selecionado*. As perguntas do "conjunto não selecionado" serão separadas para a geração dos exemplos não rotulados descrito na subseção 4.2.4. Com o conjunto de perguntas selecionadas, é feita uma seleção aleatória de uma pergunta para cada exemplo.

Figura 10: Fluxo de execução da busca por perguntas a partir da expressão regular gerada.



Fonte: elaborada pelo autor

É importante considerar que, dependendo do *corpus* utilizado como entrada, o modelo pode não encontrar perguntas para determinados exemplos. Ou seja, eventualmente um exemplo semente pode gerar um padrão *RegEx* que não faça correspondência a pelo menos uma pergunta presente no *corpus*. Assim, aquele exemplo semente não terá perguntas que o represente, impossibilitando de ser utilizado nos próximos passos como o de extração de atributos linguísticos. Caso esse exemplo ocorra com com uma frequência notável, é recomendável investigar se a causa se dá pela representatividade do *corpus* em uso ou pela relevância da semente utilizada.

4.2.2 Extração de atributos linguísticos

Esse estágio é responsável pela execução completa de pré-processamentos textuais como tokenização e anotação morfossintática. Além disso, são identificadas as sentenças idênticas e trocadas por apenas um exemplo com um atributo numérico de ocorrência. Desta forma, caso uma sentença não possua outra igual, o valor de ocorrência será igual a 1.

Na Tabela 4 é ilustrado um exemplo para cada um dos pré-processamentos executados nesse estágio, com exceção da contagem de ocorrências. Na primeira linha, onde nenhum processo foi executado, a sentença está entre aspas representando uma sequência de caracteres. O processo de tokenização resulta em uma lista representada pelos colchetes e cada item da lista é separado por vírgula. No exemplo, cada palavra da sentença foi identificada como um *token*. Por fim, o processo de anotação morfossintática retorna uma lista de tuplas representadas pelos parênteses, em que cada tupla possui o *token* e sua classe gramatical.

Tabela 4: Exemplos de resultados dos pré-processamentos textuais utilizados na extração de atributos brutos em uma determinada sentença.

processo	resultado
nenhum	'i need a list of horror movies'
tokenização	['I', 'need', 'a', 'list', 'of', 'horror', 'movies']
anotação morfossintática	[('i', 'NOUN'), ('need', 'VERB'), ('a', 'DET'), ('list', 'NOUN'), ('of', 'ADP'), ('horror', 'NOUN'), ('movies', 'NOUN')]

Fonte: elaborada pelo autor

Os pré-processamentos textuais foram executados utilizando as ferramentas disponíveis no pacote NLTK (BIRD; KLEIN; LOPER, 2009). O conjunto de classes gramaticais utilizado foi o *Universal Part-of-Speech Tagset* (conjunto de anotações morfossintáticas universal) descrito na Tabela 5[3].

Tabela 5: *Universal Part-of-Speech Tagset* - conjunto de anotações morfossintáticas universal (BIRD; KLEIN; LOPER, 2009).

Anotação	*Significado aproximado*	*Exemplos em inglês*
ADJ	adjetivo	*new, good, high, special, big, local*
ADP	ad-posição	*on, of, at, with, by, into, under*
ADV	advérbio	*really, already, still, early, now*
CONJ	conjunção	*and, or, but, if, while, although*
DET	determinante	*the, a, some, most, every, no, which*
NOUN	substantivo	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	preposição	*at, on, out, over, per, that, up, with*
PRON	pronome	*he, their, her, its, my, I, us*
VERB	verbo	*is, say, told, given, playing, would*
.	pontuação	*. , ; !*
X	outros	*ersatz, esprit, dunno, gr8, univeristy*

Fonte: elaborada pelo autor

4.2.3 Extração de atributos

Após o pré-processamento linguístico das sentenças selecionadas, inicia-se o estágio de extração de atributos, de modo a gerar um conjunto de dados de entrada para o algoritmo de

[3]Os valores da coluna "Significado aproximado" é uma tradução baseada na gramática da língua portuguesa. Portanto, tanto a tradução quanto o processo de anotação utilizado não correspondem fielmente às classes gramaticais da língua portuguesa.

aprendizado de máquina. Esse estágio é o "núcleo duro" do processo de extração de atributos, pois tanto para os exemplos semente e os exemplos que serão extraídos e gerados a partir das perguntas que não foram selecionadas pelos exemplos sementes passarão por esse estágio. Na Tabela 6 são apresentados os atributos extraídos, os quais serão descritos na sequência.

Tabela 6: Atributos extraídos a partir dos exemplos sementes.

#	*Atributo*	*Tipo*
1	candidate	ID
2	category	ID
3	occurrences	Numérico
4	extraction_pattern_occurrences	Numérico
5	min_similarity	Numérico
6	max_similarity	Numérico
7	signature	Binário
8	label	Binário

Fonte: elaborada pelo autor

Na Tabela 6 os atributos do tipo "ID" fazem a identificação do exemplo em que foi realizada a extração dos atributos. O atributo *occurrences* representa o número de sentenças encontradas no conjunto de perguntas de entrada, onde há evidências da categoria e subcategoria do exemplo semente. O atributo *extraction_pattern_occurrences* representa o volume de exemplos de contextos de extração que contêm correspondências às expressões regulares do exemplo. Os atributos *min_similarity* e *max_similarity* representam respectivamente o maior valor dentre as menores medidas de similaridade e o menor valor dentre os maiores valores de similaridade, ambos encontrados entre o conjunto de palavras candidatas e a categoria. O atributo *signature* representa o conjunto de *tags* morfossintáticas da sentença relacionada ao exemplo. Por fim, o atributo *label* é do tipo binário e indica se determinado exemplo é ou não uma subcategoria.

Observando-se os resultados obtidos no trabalho de Souza et al. (2018), notou-se que ao treinar um classificador considerando a extração de atributos baseados em características linguísticas garantiam um desempenho adequado. Assim, foram definidos os atributos descritos no que segue.

Sequência de classes morfossintáticas (*Signature*) – a sentença representante da subcategoria, depois que passou pelo processo de extração de atributos linguísticos, como descrito na subseção 4.2.2, é utilizada para buscar a posição das palavras iguais aos exemplos. Então, as *Tags* ou classificações morfossintáticas são coletadas e armazenadas na mesma sequência em que as palavras estão dispostas, incluindo a categoria. Considerando o caso do candidato

"*horror*" para a categoria "*movie*" e a sentença "*i need a list of horror movies*", nota-se que na anotação morfossintática de tal sentença é possível identificar que a sequência de classes morfossintáticas seria "NOUN NOUN", conforme mostrado na Tabela 4.

Similaridade mínima e máxima – os atributos de similaridade máxima e mínima são extraídos com uso da base de dados lexicais e conceituais chamada WordNet (MILLER, 1998), a qual é interfaceada pela ferramenta NLTK (BIRD; KLEIN; LOPER, 2009). Além de fornecer informações semânticas de relações entre as palavras como sinonímia, antonímia, acarretamento e outras, a WordNet também fornece alguns algoritmos de similaridade entre duas palavras com base em relações ontológicas. Dentre as diferentes medidas disponíveis, o modelo utiliza a similaridade de Wu-Palmer (WU; PALMER, 1994), que retorna uma pontuação de similaridade baseada na profundidade dos dois sentidos da taxonomia e no nó antecessor mais específico. A comparação de similaridade é feita entre as palavras candidatas a subcategoria e a categoria, ou seja, caso um exemplo de subcategoria candidata possua duas ou mais palavras, então são feitas medidas de similaridades entre cada palavra e a categoria, gerando assim um conjunto de pontuação de similaridades. Além disso, também é considerado a polissemia e homonímia entre as palavras. Com isso, o modelo utiliza a classificação morfossintática das palavras para reduzir o fator de ambiguidade. Ainda assim, caso exista mais de um nó possível na taxonomia da WordNet para a uma determinada palavra, serão feitas medidas de similaridade de todas as opções restantes disponíveis e o irá modelo obter o maior e menor valores. Ao final do processo, cada palavra da subcategoria candidata terá um valor mínimo e máximo de similaridade, tornando possível selecionar o maior valor mínimo e o menor valor máximo de similaridade para compor os atributos *min_similarity* e *max_similarity*, respectivamente.

Ocorrências em padrões de extração da NELL – o componente da NELL chamado CPL proposto por Carlson et al. (2010b) utiliza coocorrência estatística entre sintagmas nominais e padrões de contexto para aprender novos padrões de extração que ajudam a aprender novas instâncias de categorias ao ler páginas da *web*. Atualmente, dependendo da categoria, a base de conhecimento possui centenas e em alguns casos milhares de padrões de extração em sua base de conhecimento. Por exemplo, na categoria "*movie*", pode-se encontrar o padrão "famous movies include _"[4]. Com esse padrão, a NELL busca na *web* por sintagmas nominais no lugar de "_" e então poderá encontrar possíveis exemplos de filmes. Notou-se que nesses conjuntos encontram-se padrões de extração como "*favorite horror movie is* _"[5], "*classic horror movies like* _"[6] ou "*horror movies such as* _"[7], entre outros. Dado que existe um processo

[4]Tradução:"filmes famosos incluindo _"
[5]Tradução: "favorito filme de terror é _"
[6]Tradução: "filmes clássicos de terror como _"
[7]Tradução: "filmes de terror como _"

de aprendizado para que os padrões de extração possam ser promovidos a bons exemplos de extração, a ocorrência das palavras candidatas nesses padrões de extração aumenta as probabilidades de que tais candidatas sejam exemplos positivos. Com isso, o modelo faz uma busca de ocorrências de padrões de extração para cada exemplo, relacionando as palavras candidatas e a categoria. Nas demonstrações dos dados que serão feitas, o nome do atributo representado aqui é apresentado como "*extraction_pattern_occurrences*'.'

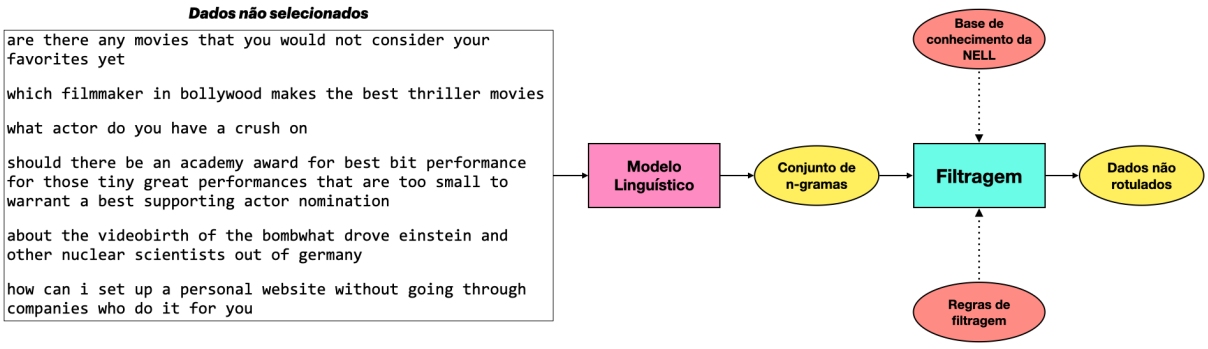
Occorrências – o número de ocorrências é dado pela quantidade de sentenças encontradas no *corpus* que possui a mesma sequência de palavras das subcategorias candidatas. Esse processo já foi descrito na fase de seleção na subseção 4.2.1. Portanto, nesse estágio é feita apenas a contagem de sentenças selecionadas por candidato à subcategoria. Na Tabela 6, o atributo é apresentado como "*occurrences*', sendo este o único atributo com um viés numérico.

Após completar a extração de atributos guiada pelos exemplos sementes, o modelo terá um conjunto de dados rotulados que será utilizado como conjunto de treinamento e validação ao algoritmo aprendizado de máquina descrito na subseção 4.2.6, além de um conjunto de dados para limitar a busca por exemplos no estágio de geração de exemplos não rotulados.

4.2.4 Geração de exemplos não rotulados

Nesse estágio, novos exemplos candidatos a subcategorias são extraídos a partir do *corpus* chamado *conjunto não selecionado*, resultante do processo de seleção descrito na subseção 4.2.1. O particionamento do *corpus* diminui a incidência de sobreposições[8], salvos somente por outras características da língua como, por exemplo, sinonímia.

Figura 11: Fluxo de geração de exemplos não rotulados.



Fonte: elaborada pelo autor

[8]Também chamado de *overlap* na literatura, inclusive em português.

Inicialmente, é realizada uma pré-seleção a partir de um modelo de linguagem estatístico com algoritmo de *n*-gramas, por meio da ferramenta de aprendizado de máquina chamada *Scikit-learn* (PEDREGOSA et al., 2011). Cada *n*-grama gerado é uma sequência de *n* elementos (palavras no escopo desse trabalho). O modelo de linguagem retorna uma lista de *n*-gramas ao Modelo 1, o qual faz uma pré-seleção antes de iniciar a extração dos atributos.

A lista retornada pelo modelo de linguagem contém os *n*-gramas mais relevantes do *corpus*, porém, nem todos são candidatos a subcategoria e passam por uma seleção realizada através de uma regra de filtragem estática e um conjunto de regras que podem ser definidas manualmente.

A regra de filtragem de *n*-gramas estática é baseada nas categorias. Uma das palavras do conjunto de palavras deverá ser a categoria dada a partir da comparação com os radicais das palavras. Por exemplo, para encontrar *n*-gramas para a categoria "movie", são buscadas palavras com radicais "*movi*". Caso nenhuma palavra seja retornada pela busca, então o *n*-grama é descartado. Caso contrário, o *n*-grama passará para os próximos filtros definidos manualmente.

Ao final do processo, os *n*-gramas são formatados como um conjunto de candidatos a subcategorias (conforme mostrado na Tabela 2), porém sem o atributo *rótulo*. Esses exemplos passarão pelo mesmo processo de extração de atributos descritos na subseção 4.2.3 e terão os mesmos atributos apresentados na Tabela 6 sem o atributo *label*. A partir de então considera-se o conjunto de exemplos não rotulados.

4.2.5 Transformação

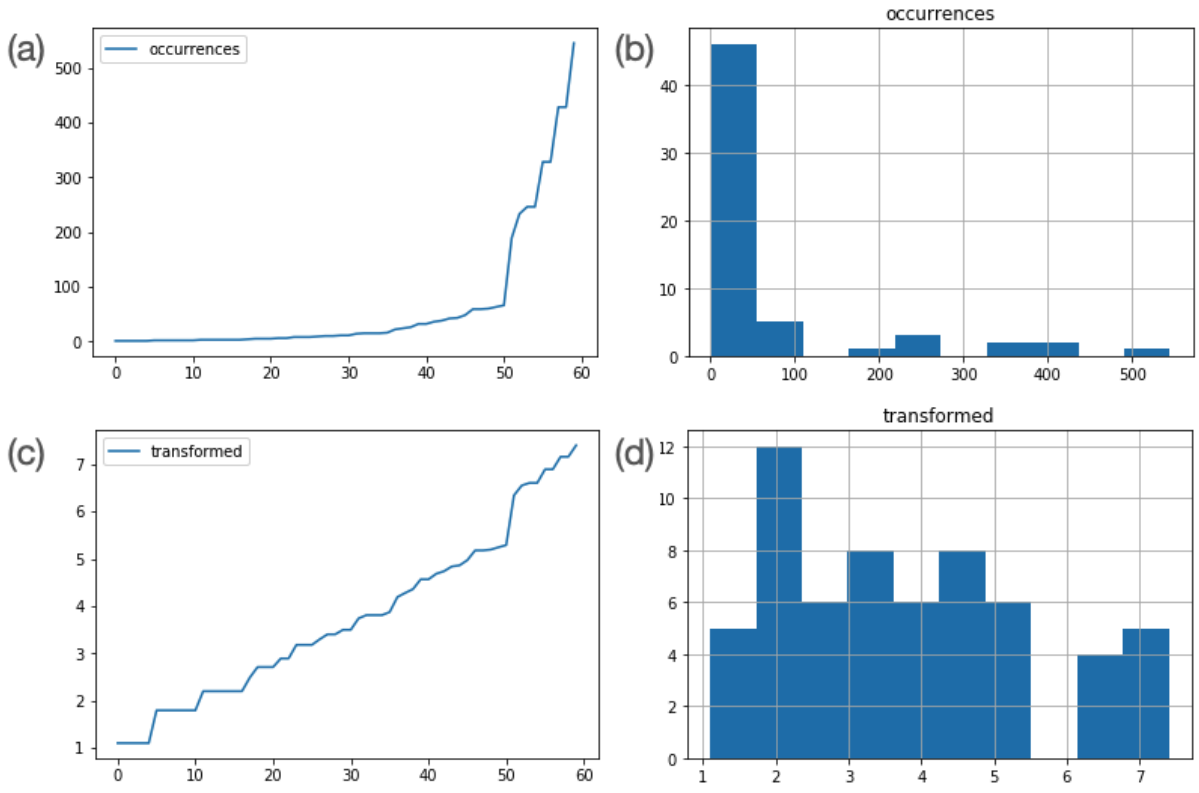
Antes das fases de treinamento e validação do algoritmo de aprendizado de máquina, ocorrem dois tipos de transformação nos dados rotulados e não rotulados, a primeira é a exponenciação dos valores numéricos e a segunda é a transformação de valores nominais em binários.

Segundo Hazarika (2013), transformar um conjunto de dados de modo a aumentar sua linearidade, de maneira geral, pode trazer benefícios no ajuste entre variáveis e, no caso em questão, auxiliar na tarefa de classificação dos dados.

Para exemplificar as transformações dos dados numéricos realizadas nesse trabalho, será apresentada uma das que foi realizada no atributo "*occurrences*". Após a extração dos valores em 60 exemplos, foi realizada uma ordenação crescente dos dados. Ao plotar uma linha onde o eixo *x* representa os exemplos ordenados e o eixo *y* o número de ocorrências, obtém-se o gráfico da Figura 12(a).

Ainda com os dados em seus valores originais, pode-se plotar um histograma, com o obje-

Figura 12: Exemplo de uma transformação exponencial no atributo "*occurrences*" em um conjunto de dados com 60 exemplos: (a) disposição dos valores ordenados: (b) histograma dos exemplos (valor por quantidade); (c) disposição dos valores transformados ordenados; (d) histograma dos exemplos com valores transformados.



Fonte: elaborada pelo autor

tivo de se obter uma visão da distribuição dos exemplos. Pode se observar no histograma gerado na Figura 12(b) que existe uma concentração maior de exemplos com número de ocorrências entre 0 e 50. Após a transformação dos dados a partir da função apresentada na Equação 4.1, pode-se gerar os mesmos gráficos e observar as seguintes mudanças.

$$f(x) = \log(3x). \tag{4.1}$$

Observando o gráfico da Figura 12(c), nota-se um aumento nos valores, respeitando um crescimento próximo ao linear. No gráfico da Figura 12(d), é possível observar os exemplos mais distribuídos entre os possíveis valores do atributo.

A transformação de valores nominais em binários se faz necessária pelo uso de algoritmos que aceitam apenas atributos numéricos e binários. Nesse trabalho, o atributo "*signature*" foi submetido a essa transformação. Na Tabela 7 é apresentado um conjunto de exemplos com seus

respectivos valores para tal atributo.

Tabela 7: Exemplo de um conjunto de dados com o atributo "*signature*".

candidate	category	signature
adventure _	movie	ADJ NOUN
considered _	scientist	VERB NOUN
music _	website	NOUN NOUN

Fonte: elaborada pelo autor

A partir dos valores do atributo em questão, são identificados os valores únicos. Então, cada valor único transforma-se em uma coluna e cada exemplo receberá o valor binário "*VERDADEIRO*" (ou 1) na coluna em que representava o valor do exemplo na coluna "*signature*" e "*FALSO*" (ou 0) nas demais colunas. Ao final, a coluna "*signature*" é descartada. Continuando o exemplo apresentado na Tabela 7, o conjunto de dados resultantes após esse processo de transformação é apresentado na Tabela 8.

Tabela 8: Exemplo de um conjunto de dados após a transformação do atributo"*signature*"

candidate	category	ADJ NOUN	VERB NOUN	NOUN NOUN
adventure _	movie	1	0	0
considered _	scientist	0	1	0
music _	website	0	0	1

Fonte: elaborada pelo autor

4.2.6 Treinamento e Validação do Classificador

Nesse último estágio do Modelo 1, é recebido como entrada o conjunto de dados rotulados e não rotulados gerados nos estágios supramencionados. Assim, a tarefa nesse estágio é relacionada à indução modelo de classificação, buscando torná-lo hábil a classificar os exemplos candidatos a subcategoria de suas respectivas categorias. Na Figura 8, esta etapa é representada pelos retângulos "treino e teste" e "classificação"

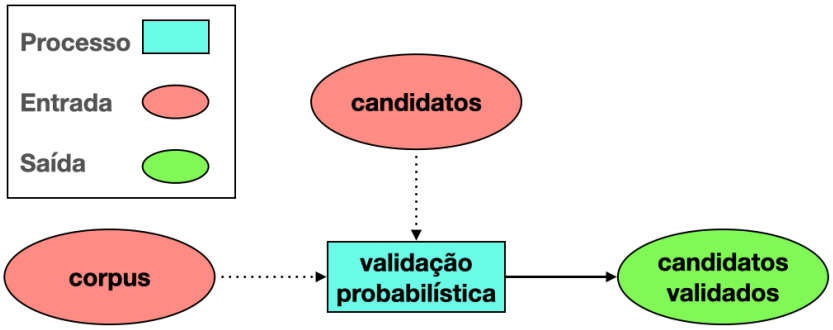
O Modelo 1 permite o uso de diferentes algoritmos de aprendizado de máquina supervisionado ou semi-supervisionado de classificação binária. Nesse estágio, é possível realizar validações do modelo gerado a partir de um algoritmo de aprendizado de máquina predeterminado, além de realizar ajustes de parâmetros dos algoritmos. O algoritmo utilizado neste estágio foi o *XGBOOST* que será detalhado no Capítulo 5.

Por fim, tem-se o conjunto de exemplos que servirá como entrada ao segundo modelo computacional (Modelo 2) que compõe o componente proposto, o qual é responsável por um mecanismo de validação automática. Vale comentar que somente os exemplos classificados como positivos são enviados como entrada ao Modelo 2 descrito na próxima seção.

4.3 Modelo 2: Validação automática

O Modelo 2 tem seu fluxo de execução apresentado na Figura 13, onde recebe como entrada um *corpus* não anotado de artigos distintos da fonte de entrada do Modelo 1 e o par $p(C,W)$, onde C representa a categoria já conhecida pela Base de Conhecimento da NELL e W representa o conjunto de palavras candidato a subcategoria de C.

Figura 13: Fluxo de execução do Modelo 2.



Fonte: elaborada pelo autor

O modelo computacional de validação proposto baseia-se no teorema de Bayes (BAYES, 1763). A tarefa do algoritmo é calcular uma pontuação de confiança de um par formado e classificado pelo Modelo 1. Então, dadas as probabilidades desse par, $p(C,W)$, ocorrer em um determinado *corpus*, o algoritmo determina se o par é um exemplo positivo.

Considerando que o Modelo 2 recebeu como entrada um determinado par, $p(C,W)$, e um determinado *corpus* com N artigos, a princípio, calcula-se a quantidade de artigos em que ocorrem C e W. Sabe-se que C e W representam conjunto de palavras de tamanho maior ou igual a 1. Portanto, para se calcular a ocorrência desses conjuntos de palavras, o modelo considera uma ocorrência em um artigo apenas quando uma das possíveis combinações de mesmo tamanho do conjunto de palavras ocorre no artigo. Assim, para melhor entendimento dos próximos passos do processo, c representa a quantidade de artigos que contém a categoria C no par $p(C,W)$ e w representa a quantidade de artigos que contém o conjunto de palavras candidato a subcategoria

de C.

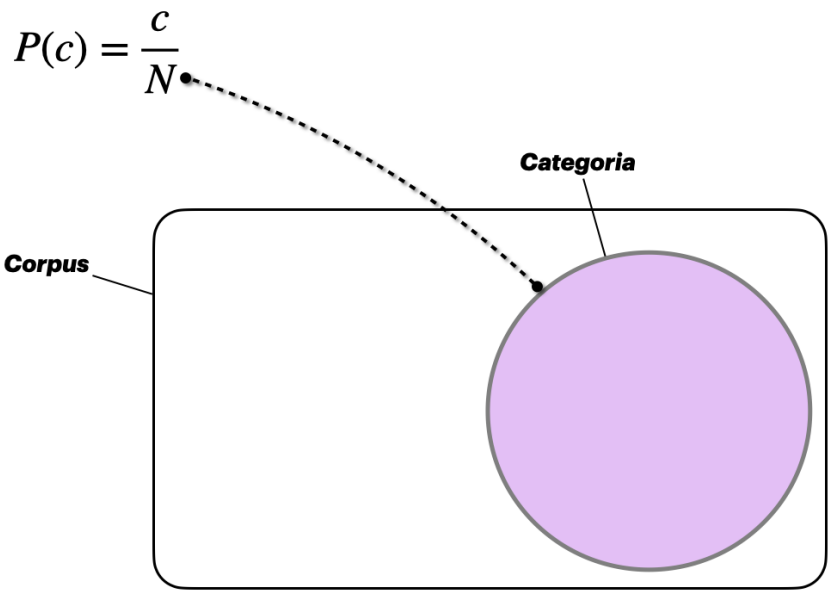
A partir da obtenção da frequência, são realizados os cálculos com as seguintes equações:

$$P(c) = \frac{c}{N}, \tag{4.2}$$

$$P(w) = \frac{w}{N}, \tag{4.3}$$

onde $P(c)$ representa a probabilidade da categoria (c) ocorrer dado um *corpus* de tamanho N e $P(w)$ é a probabilidade do conjunto candidato (w) a subcategoria ocorrer dado um *corpus* de tamanho N. As equações 4.2 e 4.3 são ilustradas em diagramas de Venn nas figuras 14 e 15 respectivamente.

Figura 14: Representação da equação 4.2 em um diagrama de Venn.



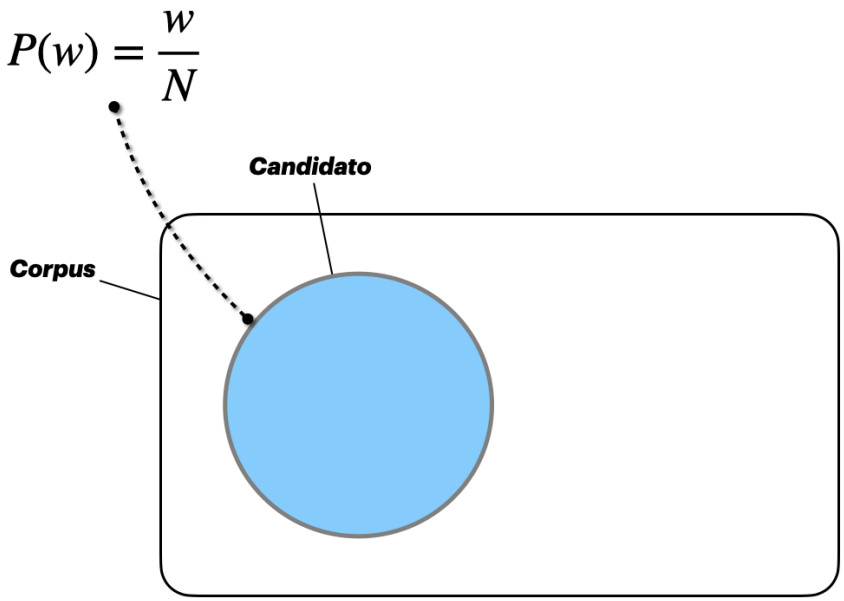
Fonte: elaborada pelo autor

Em seguida, calcula-se quanto o conjunto candidato a subcategoria intersecciona-se com a categoria. Esse valor é representado por $P(A)$, conforme a Equação 4.4 e ilustração na Figura 16:

$$P(A) = \frac{P(c).P(w)}{P(w)}, \tag{4.4}$$

sendo A um conjunto que representa os possíveis valores de $P(A)$, então $A = \{x \in P(A) :$

Figura 15: Representação da equação 4.3 em um diagrama de Venn.



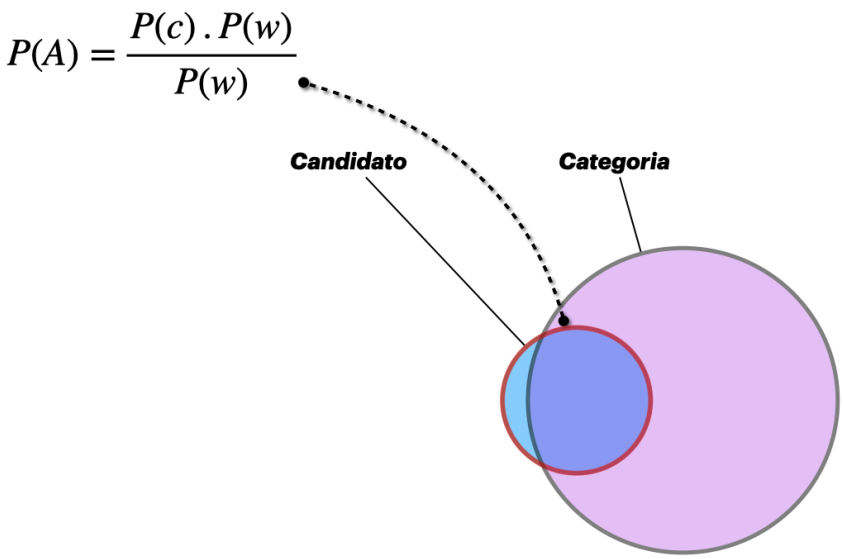
Fonte: elaborada pelo autor

$0 \leq x \leq 1\}$. Caso o valor resultante de $P(A)$ seja próximo a 0, então pode-se assumir que a relação entre a categoria e o conjunto candidato a subcategoria é fraca por não ter muitos exemplos em que o par ocorre concomitantemente em relação à quantidade de vezes que o conjunto de palavras que representa a subcategoria ocorre sem a presença da categoria. Por outro lado, nos casos em que $P(A)$ aproxima-se de 1, pode-se assumir que a relação do par é forte, ou seja, o conjunto de palavras candidato a subcategoria ocorre em conjunto com a categoria na maioria das vezes, independente da quantidade de vezes em que ocorre a categoria. Destaca-se que, nos casos em que o valor de $P(A)$ seja igual a zero, não é possível assumir que a relação do par seja nula, pois o *corpus* utilizado, mesmo que por inteiro, não representa todo o domínio de conhecimento da língua. Portanto, trata-se de um ambiente de incerteza no qual não é recomendado assumir que o par não tenha qualquer relação entre si, mesmo que não existam ocorrências no *corpus* analisado. Nesses casos, é recomendado transformar os valores de $P(A)$ para um valor muito baixo. Nos experimentos realizados e que serão apresentados nesta tese, os valores de $P(A)$ iguais a zero foram transformados para $1e^{-6}$.

Observado o valor de $P(A)$, o próximo valor a ser calculado é $P(B|A)$, conforme apresentado na Equação 4.5:

$$P(B|A) = \frac{P(c).P(w)}{N} \tag{4.5}$$

Figura 16: Representação da equação 4.4 em um diagrama de Venn.



Fonte: elaborada pelo autor

onde $P(B|A)$ representa a probabilidade do par $p(C,W)$ ocorrer em um mesmo artigo no *corpus*. Ou seja, $P(B|A)$ pode ser visto como a intersecção de $P(c)$ e $P(w)$, conforme ilustrado em um gráfico de Venn na Figura 17.

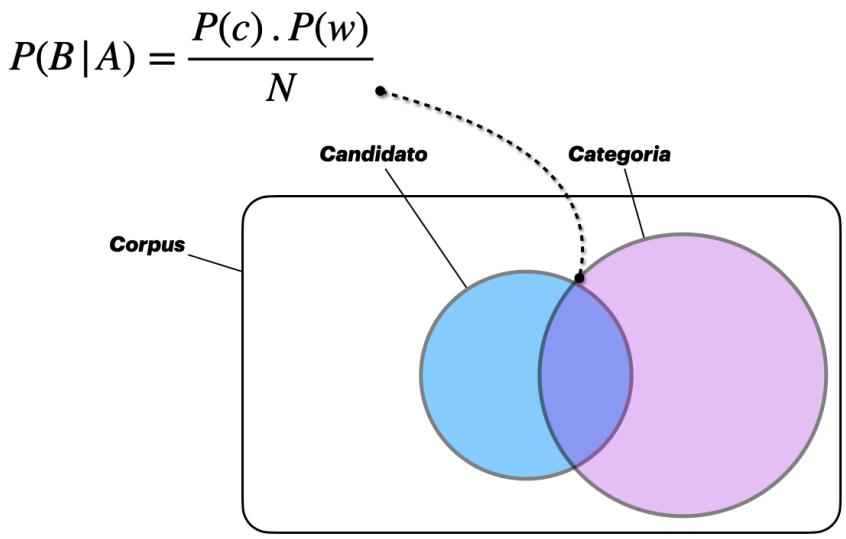
Dado o tamanho do *corpus*, essa intersecção geralmente irá trazer um valor muito baixo (próximo a zero), pois normalmente o *corpus* utilizado aborda uma grande variedade de assuntos.

Outro fator que também pode ocorrer com valores muito baixos é que a diferença entre as probabilidades de dois pares pode ser muito baixa para serem comparadas, dificultando a tomada de decisão do Modelo 2, como exemplificado a seguir. Dado um *corpus* de tamanho $N = 20.000$, um par $p_1(C_1,W_1)$ onde a intersecção entre C_1 e W_1 ocorrem 10 vezes no *corpus* e um par $p_2(C_2,W_2)$ onde a intersecção entre C_2 e W_2 ocorrem 100 vezes no *corpus*, a diferença entre $P(B|A)_1$ e $P(B|A)_2$ será de $4,5e^{-3}$ mesmo sabendo que a intersecção de $p_2(C_2,W_2)$ ocorre 10 vezes mais. Para minimizar os fatores que poderão dificultar a tomada de decisão do Modelo 2, é realizada uma transformação de $P(B|A)_t$ como descrito na Equação 4.6:

$$P(B|A)_t = \frac{\log(P(B|A) \times e^3 + 1)}{3}. \tag{4.6}$$

As constantes escolhidas na Equação 4.6 baseiam-se nos tamanhos de *corpus* utilizados nos experimentos realizados nesta tese, portanto, caso seja necessária uma replicação dos experi-

Figura 17: Representação da equação 4.5 em um diagrama de Venn.



Fonte: elaborada pelo autor

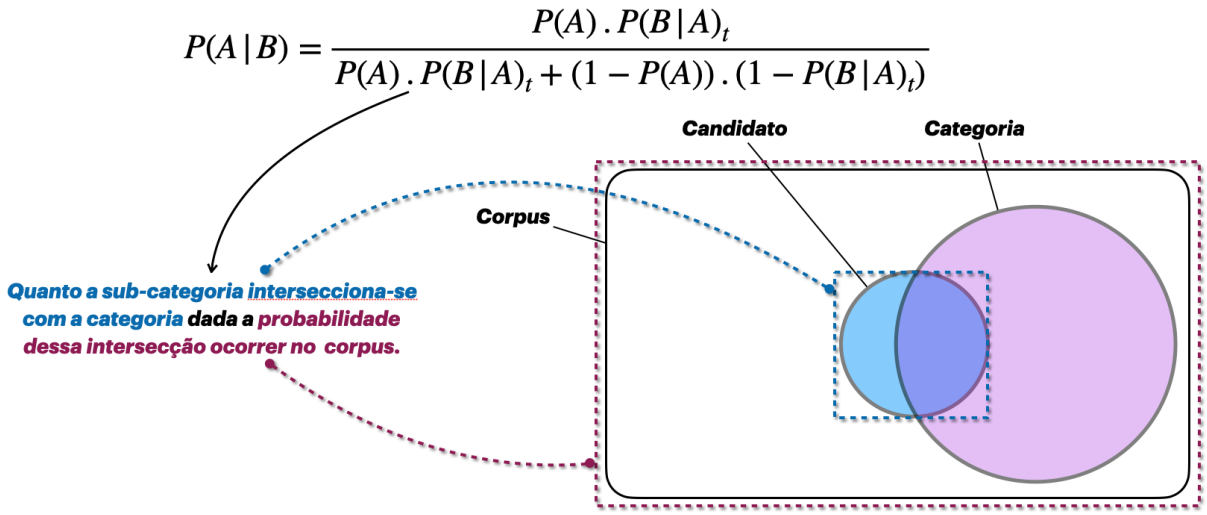
mentos com *corpora* de tamanhos diferentes dos utilizados, torna-se necessário alterar/rever os valores das constantes dessa equação.

Com $P(A)$ e $P(B|A)_t$ calculados, o validador calcula $P(A|B)$ como descrito na Equação 4.7:

$$P(A|B) = \frac{P(A).P(B|A)_t}{P(A).P(B|A)_t + (1-P(A)).(1-P(B|A)_t)}, \tag{4.7}$$

onde $P(A|B)$ representa a probabilidade condicional de quanto a subcategoria intersecciona-se com a categoria dada a probabilidade dessa intersecção ocorrer no *corpus* como pode-se notar no diagrama de Venn apresentado na Figura 18. O passo final do validador é classificar se W pertencente ao par $p(C,W)$ é uma possível subcategoria de C ou não. Esse passo é realizado a partir da definição de um limiar de corte l do resultado de $P(A|B)$, onde valores resultantes no intervalo entre 0 e l indicam que o par não possui exemplos suficientes para sustentar uma validação estatística de que o conjunto de palavras em questão é subcategoria de uma determinada categoria. Recomenda-se um valor de l próximo à $0,2$, porém é importante enfatizar que este valor pode ser modificado conforme a parametrização feita no Modelo 2. O tamanho e as características do *corpus* utilizado pode influenciar no valor de l a ser determinado, assim como o ajuste das constantes da Equação 4.6.

Figura 18: Representação da equação 4.7 em um diagrama de Venn.



Fonte: elaborada pelo autor

Capítulo 5

EXPERIMENTOS, RESULTADOS E ANÁLISES

Nesse capítulo serão apresentados os três principais experimentos realizados para validação da abordagem proposta nesta tese e apresentada no Capítulo 4. O primeiro experimento realizado, descrito na seção 5.1 teve como objetivo fazer a validação do Modelo 1 (seção 4.2), ou seja, o experimento investigou a viabilidade da capacidade de um ou mais algoritmos de aprendizado de máquina serem capazes de aprender a tarefa de classificar subcategorias. Nesse experimento, os atributos foram gerados manualmente por especialista. O segundo experimento descrito na seção 5.2 foi realizado com o objetivo de validar o segundo modelo computacional, descrito na seção 4.3, utilizando um conjunto de dados rotulados manualmente. Por fim, o terceiro experimento apresentado na seção 5.3 validou o componente completo com os dois modelos computacionais propostos, onde o componente recebeu as sementes de entrada e executou todo o processo automaticamente apenas utilizando os outros recursos de entrada conforme proposto.

5.1 Experimento preliminar visando a identificação de subcategorias executada pelo Modelo 1

Para validar a possibilidade de desenvolver um agente computacional que aprende novas categorias para a base de conhecimento da NELL a partir da leitura de textos em um ambiente de perguntas e respostas, foram realizados experimentos que também descritos em Souza et al. (2018). O trabalho aborda uma comparação de algoritmos cuja tarefa é a classificação de pares de palavras candidatas a subcategoria na base de conhecimento da NELL. Foram extraídos textos de fóruns de perguntas e respostas (`www.answers.yahoo.com` e `www.answers.com`) que, consequentemente, são transformados em um conjunto de sentenças por meio do processo de tokenização. As sentenças, no que lhe concerne, são processadas de forma que sejam extraídos

sintagmas nominais. As palavras dos sintagmas nominais extraídas são comparadas com a ontologia da NELL. Caso uma das palavras seja identificada como uma categoria já conhecida, são formados pares de palavras entre a palavra conhecida como categoria e as demais palavras do sintagma nominal.

Um exemplo desse processo, supondo que o agente computacional extraiu dos textos dos fóruns a sentença "*I need a list of good horror movies*"[1], ao passar a frase pelo algoritmo extrator de sintagmas nominais, se tem as listas contendo as palavras: *[list]* e *[good, horror, movies]*. Como a primeira lista tem apenas uma palavra, a mesma será descartada, pois, não é possível formar pares de palavras. Entretanto, a segunda lista será utilizada para comparar cada palavra com as categorias conhecidas da ontologia da NELL, onde se emprega um lematizador em todas as palavras para reduzi-las apenas ao lema (por exemplo, o lema de *movies* é *movie*). Nesse exemplo supõe-se que a NELL conheça apenas a palavra *movie* como categoria. Portanto, serão formados dois pares de palavras: *[good, movie]* e *[horror, movie]*. Neste sentido, cada par de palavra indica que a primeira palavra possui uma chance de ser uma subcategoria da segunda palavra que já é conhecida como categoria da ontologia em questão.

Com os pares criados, as sentenças nas quais os pares pertencem são submetidas a outros processamentos textuais para que seja possível gerar atributos relevantes aos algoritmos de classificação. As cinco primeiras linhas do conjunto de dados criado são apresentadas na Tabela 9, onde a coluna *word* representa a palavra que será categorizada como sendo ou não uma candidata a subcategoria de uma determinada categoria (coluna *category*).

Utilizando ferramentas de processamento textual disponíveis no *NLTK*, foram definidos os atributos mais relevantes e que carregam informações que poderão auxiliar os algoritmos de aprendizado de máquina a se tornarem mais eficientes na tarefa de classificação. Os atributos *word_tag* e *category_tag* representam quais classes gramaticais as palavras que formam o par *(word, category)* pertence. O conjunto de classes gramaticais utilizado foi o *Penn Treebank* (padrão do *NLTK*). Na coluna *is_category* é apresentado se a palavra do par também é uma categoria existente na base de conhecimento da NELL. O atributo *occurrences* representa a quantidade de vezes que o par ocorre na base de dados. O atributo *is_category* é o rótulo do par. Na Tabela 9 esses pares ainda não foram rotulados.

Para esse experimento preliminar, o processo de rotulação foi realizado manualmente. Para uma melhor concordância ao realizar o processo de rotulação, foram criadas algumas regras para considerar quais palavras podem ou não ser candidatas a subcategorias da base de conhecimento da NELL.

[1] Do inglês "Eu preciso de uma lista de bons filmes de horror"

Tabela 9: Cinco primeiras linhas da base de dados sem rótulo.

sentence	word	category	word tag	category tag	is_category	occurrences
bad rap actors good rap actors list?	bad	actor	JJ	NNS	no	3
bad rap actors good rap actors list?	rap	actor	NN	NNS	no	2
bad rap actors good rap actors list?	good	actor	NN	NNS	no	11
bad rap actors good rap actors list?	rap	actor	NN	NNS	no	2
bad rap actors good rap actors list?	list	actor	NN	NNS	no	7

Fonte: elaborada pelo autor

Definiu-se um critério de rotulação onde se restringiu alguns grupos de palavras de acordo com suas características, como mostrado a seguir:

- Conjunto de caracteres que não formam palavras na língua inglesa;
- Pares que não fazem sentido, exemplo: "*(year, actor)*" (ator, ano);
- Características subjetivas, exemplos: "*good, fast, bad*" (bom, rápido, mau);
- Características que mudam com o tempo, exemplo: "*young*" (novo);
- Categorias baseadas em gênero na ontologia, exemplo: *(male, actor)* (ator, homem).

Com os dados prontos, o próximo passo foi realizar o experimento a partir da seleção de alguns algoritmos de classificação.

Ao final da rotulação, o atributo classe (*is_category_candidate*) da base de dados, com 370 instâncias, assume dois valores possíveis: *no* e *yes* [2], sendo 269 instâncias rotuladas com o valor *no* e 101 instâncias rotuladas com o valor *yes*.

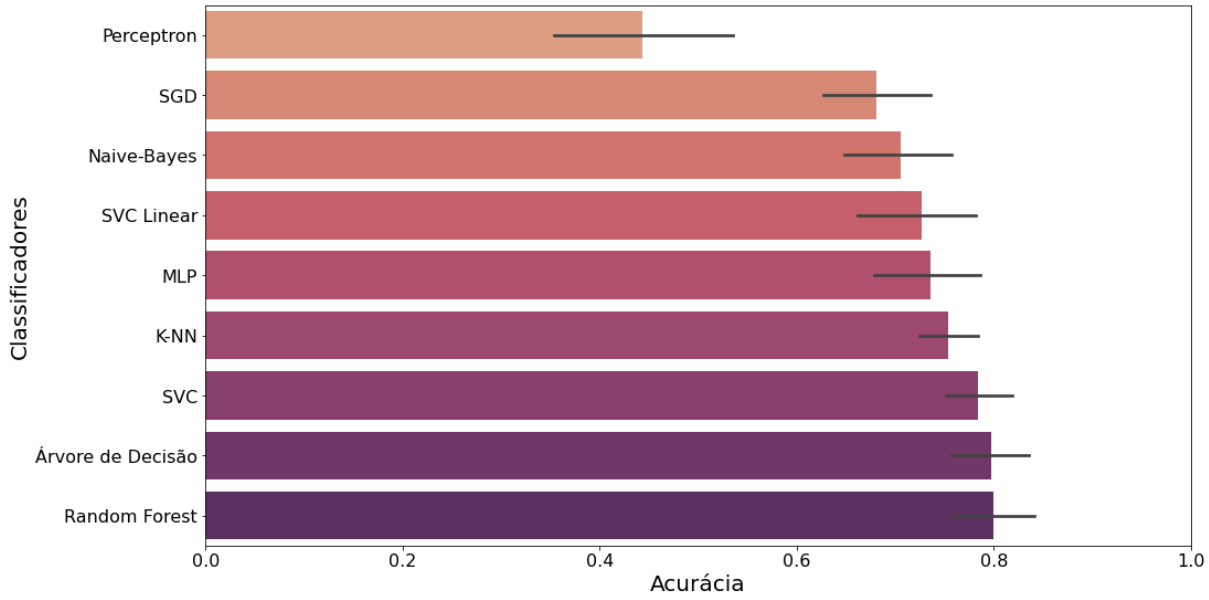
Em seguida, foi realizada a seleção dos atributos que farão a composição do conjunto de dados utilizado na avaliação dos algoritmos, que são: *word_tag*, *nell_category_tag*, *word_is_category*, *number_of_occurrences* e o atributo classe *is_category_candidate*.

Para a execução do experimento, foram escolhidos três algoritmos lineares e cinco não lineares. Os algoritmos lineares selecionados foram o *Perceptron*, *Linear SVC* e SGD (*Stochastic Gradient Descent*). O objetivo de utilizar esses classificadores é identificar se os dados são ou não linearmente separáveis, indicado pelos seus valores de pontuação. Para os algoritmos não lineares, foram escolhidos o SVM (*Support Vector Machine*) chamado de SVC, um kNN (*K-Nearest Neighbors*) com $k = 7$, um Naïve Bayes e dois algoritmos baseados em árvore de decisão (sendo *CART* e o *ensemble* denominado *Random Forest*. A base de dados utilizada e o experimento estão disponíveis para acesso público em `https://github.com/MaLL-UFSCar/NELL-subcategories-QnA`.

[2] Do inglês "não" e "sim".

Os algoritmos de aprendizado escolhidos foram avaliados utilizando uma validação cruzada com 10 *folds*, com a mesma separação de dados. Em cada *fold*, foi considerada a acurácia do classificador no conjunto de validação. A Figura 19 mostra as médias de acurácia de cada método em suas dez execuções, com uma demonstração de intervalo de confiança usando o primeiro decil e o nono decil (ou 10^{o} e 90^{o} percentis).

Figura 19: Média da acurácia por atributo com intervalo mínimo e máximo



Fonte: (SOUZA et al., 2018)

Aplicando o método *ANOVA* aos resultados obtidos, tem-se que a *Random Forest*, candidato a melhor classificador, é superior aos algoritmos Perceptron, SGD e Naïve-Bayes com significância estatística ($p \leq 0,05$). Os resultados são inconclusivos para os demais algoritmos de classificação.

Com os resultados obtidos, foi possível validar a possibilidade de desenvolver um agente computacional que aprende novas categorias para a base de conhecimento da NELL a partir da leitura de textos em um ambiente de perguntas e respostas.

5.2 Experimento preliminar visando a validação automática executada pelo Modelo 2

Validada a hipótese descrita na seção 5.1, torna-se possível realizar o experimento para a validação do Modelo 2 que executa a tarefa de validação automática dada a saída do Mo-

delo 1. Antes de avançar na demonstração do experimento, inicialmente, serão abordadas algumas considerações para que as demonstrações sejam compreendidas com maior facilidade.

Não foi computacionalmente possível utilizar um *corpus* completo durante a execução do experimento. Por isso, para ser possível identificar se o tamanho do *corpus* impacta nas medidas de desempenho do Modelo 2, foram executados cinco experimentos similares com alterações no número de artigos contidos no *corpus*. Com essa execução, é possível verificar se existe uma tendência de convergência que indique a necessidade apenas de um número suficiente de exemplos de textos em língua inglesa que possam validar uma quantidade significativa de exemplos de subcategorias dadas as categorias já conhecidas.

Conforme o que foi apresentado no Capítulo 4, o Modelo 2 recebe como entrada um par $p(C,W)$ que foi classificado pelo Modelo 1 e um *corpus* não anotado.

Foi gerado manualmente um conjunto de entrada com exemplos positivos e negativos de candidatos a subcategorias das categorias *movie* e *actor* no mesmo formato de saída do Modelo 1.

O *corpus* não anotado utilizado foi gerado a partir da Wikipédia[3] em inglês. Na data em que foram realizados os experimentos, o *corpus* completo continha cerca de 5 milhões de artigos. Após o *corpus* completo ser gerado, gerou-se, a partir do mesmo, cinco versões de tamanhos menores, contendo 1.000, 5.000, 10.000, 15.000 e 20.000 artigos.

Na Figura 20 são apresentadas as distribuições obtidas a partir dos experimentos realizados com os diferentes tamanhos de *corpus* em gráficos de caixa[4]. após a aplicação da função de validação automática, foi aplicado um limiar de corte em $0,2$, considerando-se "falso" os resultados que estiverem abaixo do limiar e "verdadeiro" (ou possível candidato) caso contrário.

Na Tabela 10, tem-se na primeira coluna o tamanho do *corpus* de cada experimento. Nas colunas seguintes são apresentados os resultados das métricas aplicadas, a saber: acurácia, precisão, cobertura e *f1-score*.

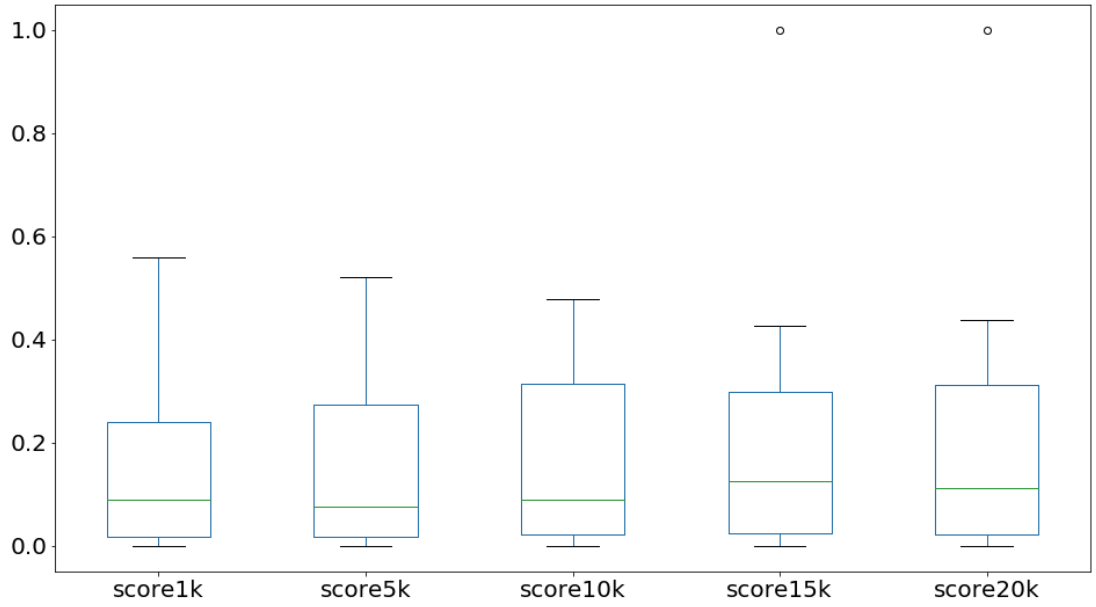
Por meio da Tabela 10 é possível identificar bons resultados a partir do experimento com *corpus* de tamanho 1k, que equivale a 0,025% do tamanho total do *corpus* original. Conforme o tamanho do *corpus* aumenta, observa-se um aumento nas medidas de acurácia, cobertura e *f1-score*. De maneira oposta, a medida de precisão tem uma leve tendência de queda.

Ainda, conforme os resultados apresentados na Figura 20, a mediana dos valores obtidos

[3]Foi utilizado um arquivo de despejo da Wikipédia em inglês chamado *enwiki-latest-pages-articles.xml.bz2* disponível em `https://dumps.wikimedia.org/enwiki/latest/`, último acesso em 3 de agosto de 2020. Após o download, o *corpus* foi construído utilizando a ferramenta *gensim* seguindo os passos descritos por Mayo (2017).

[4]Gráfico de caixa, mais conhecido como *box plot* do inglês.

Figura 20: Distribuição dos valores obtidos pelo Modelo 2 para diferentes tamanhos de *corpus*.



Fonte: elaborada pelo autor

nos cinco testes realizados durante o experimento aproximou-se de $0,1$, mesmo com o uso da Equação 4.6. Ainda assim, foi possível alcançar bons resultados nas métricas após a utilização do limiar de corte. Portanto, pode-se considerar que a Equação 4.6 distribuiu os valores de forma que foi possível distinguir exemplos que possuem uma relevância para serem considerados como exemplos positivos dos negativos.

A utilização do limiar de corte para separar os exemplos que obtiveram uma pontuação baixa contribuiu na eliminação dos falsos positivos. O que pode ser notado nos valores de precisão obtidos.

Portanto, pode-se dizer que o Modelo 2 provou-se válido para o escopo em estudo, ou seja, para a base de conhecimento da NELL. Assim, considera-se o Modelo 2 suficiente para compor o processo de validação do Modelo 1 apresentado na seção 5.3.

5.3 Testes do componente proposto para identificação de subcategorias a partir de perguntas e respostas

Com os modelos computacionais 1 e 2 devidamente validados nos experimentos preliminares, houve a necessidade de realizada uma etapa de testes do componente proposto nesta tese para a identificação de subcategorias oriundas de categorias pertencentes à base de conheci-

Tabela 10: Métricas obtidas do validador automático a partir da adição do limiar de corte em 0,2.

Tamanho	Acurácia	Precisão	Cobertura	F1-score
1k	0,775	1,000	0,55	0,710
5k	0,825	1,000	0,65	0,788
10k	0,850	1,000	0,70	0,824
15k	0,825	0,933	0,70	0,800
20k	0,850	0,937	0,75	0,833

Fonte: elaborada pelo autor

mento da NELL.

5.3.1 Parametrização

Regras de filtragem de *n*-gramas – notou-se empiricamente que alguns *n*-gramas selecionados, mesmo que filtrados pela presença da categoria, poderiam ser identificados como exemplos negativos a partir de regras simples utilizando expressões regulares. Os seguintes filtros foram empregados para todos os experimentos:

- **Menor** *n*' – o menor valor de *n* definido foi 2, assumindo-se que ao menos uma das palavras seja a categoria e a outra a subcategoria;
- **Maior** *n* – o maior valor de *n* definido foi 5;
- **Caracteres não alfabéticos** – *n*-gramas com *tokens* que possuem caracteres não alfabéticos foram removidos no experimento;
- **Categoria no final** – Foram pré-selecionados apenas os *n*-gramas que continham as categorias ao final da sequência de palavras.

Conjunto de perguntas e respostas de entrada – foi utilizada uma base de dados em formato *csv*[5] disponibilizado por Rakshit (2007) com 4.483.032 perguntas do Yahoo Answers e suas respectivas respostas, com temas que variam entre sociedade e cultura, ciência e matemática, saúde, educação, informática, negócios e finanças, entretenimento e música, família e relacionamento e política e governo. Foi realizada uma limpeza, normalização e *tokenização* de sentenças nos textos antes de serem dados como entrada para o componente.

Algoritmo de classificação – foi utilizado o *Extreme Gradient Boosting* (*XGBoost*)(CHEN; GUESTRIN, 2016), uma técnica que gera um modelo de predição a partir da junção de mode-

[5]CSV: é uma extensão de arquivo de computador onde os dados são armazenados em um documento de texto e os valores podem ser divididos por colunas normalmente com uso de vírgulas.

los de predição mais simples, que de maneira geral são otimizados a partir de uma função de perda. Nos testes, o algoritmo foi ajustado para ser utilizada a função de perda para tarefa de classificação binária.

Sementes – foram geradas manualmente um conjunto com 87 exemplos sementes de 4 diferentes categorias distribuídas como apresentado na Tabela 11:

Tabela 11: Exemplos de sementes de entrada apresentados aos testes do Modelo 1.

Categoria	Exemplos Positivos	Exemplos Negativos	Total
actor	10	10	20
movie	11	14	25
scientist	10	10	20
website	10	12	22
total	41	46	87

Fonte: elaborada pelo autor

Base de conhecimento da NELL – atualizada em sua iteração de número 1115, foram utilizadas as categorias em que a NELL possui grau de confiança acima de 90% e exemplos de padrões de extração (*extractionPatterns*).

Corpus – não anotado foi gerado a partir da Wikipédia, ou seja, o mesmo utilizado no experimento descrito na seção 5.2. Entretanto, a partir desse *corpus*, foram geradas versões menores contendo 1.000, 5.000, 10.000, 20.000, 40.000, 60.000, 80.000 e 100.000 artigos.

Após os processos de seleção e extração de atributos, foram gerados dois conjuntos de dados (rotulados e não rotulados). Das 87 sementes dadas como entrada, 75 geraram exemplos rotulados distribuídos conforme a Tabela 12.

Tabela 12: Distribuição do conjunto de dados rotulados.

Categoria	Exemplos Positivos	Exemplos Negativos	Total
actor	7	10	17
movie	11	14	25
scientist	6	7	13
website	8	12	20
total	32	43	75

Fonte: elaborada pelo autor

O conjunto de dados não rotulados possui 182 exemplos (valor amostral). Para a análise

dos resultados obtidos pelo componente, os 182 exemplos foram rotulados manualmente. Assim, os exemplos positivos e negativos dos dados, por categoria, ficaram distribuídos conforme apresentado na Tabela 13.

Tabela 13: Distribuição dos candidatos a subcategoria com rotulagem manual.

Categoria	Exemplos Positivos	Exemplos Negativos	Total
actor	8	28	36
movie	13	45	58
scientist	3	40	43
website	2	43	45
Total	26	156	182

Fonte: elaborada pelo autor

Para a validação, foram escolhidas 5 medidas de desempenho, são elas: acurácia, precisão, cobertura, *f1-score* e *brier Score Loss*(*bsl*)[6]. Nos gráficos e tabelas dos resultados, essas medidas serão apresentadas em inglês: *accuracy*, *precision*, *recall*, *f1-score* e *bsl*, respectivamente.

5.3.2 Resultados do Modelo 1

Após a execução do teste no Modelo 1, obteve-se os resultados apresentados na Tabela 14. Os valores das medidas de desempenho são divididos pelas colunas. Com os resultados obtidos, foi possível analisar seu desempenho com todo o conjunto de dados e também foram agrupados por categoria. O valor *all* na coluna *category* representa os resultados do teste do conjunto de dados completo com todas as categorias.

Tabela 14: Desempenho do Modelo 1 com XGBOOST.

Category	Accuracy	Precision	Recall	F1-score	Bsl
all	0,807692	0,400000	0,692308	0,507042	0,192308
actor	0,722222	0,437500	0,875000	0,583333	0,277778
movie	0,793103	0,529412	0,692308	0,600000	0,206897
scientist	0,860465	0,200000	0,333333	0,250000	0,139535
website	0,844444	0,142857	0,500000	0,222222	0,155556

Fonte: elaborada pelo autor

Nota-se uma acurácia acima de 0,8 no conjunto de dados completo, porém, o mesmo não

[6] *Brier Score Loss*: Uma medida de desempenho por erro proposta por Glenn W. Brier (1950), normalmente utilizada para em abordagens probabilísticas.

se repete por categoria. Um padrão observado é que nas categorias em que o Modelo 1 obteve uma acurácia mais baixa, como *actor* e *movie*, há um aumento inversamente proporcional na acurácia, cobertura, *f1-score* e *bsl*.

Dos 182 exemplos de entrada, apenas 45 considerados pelo classificador, sendo 27 exemplos negativos e 18 exemplos positivos. Analisando os exemplos de entrada do Modelo 1, apenas 17% dos exemplos negativos foram classificados como positivos enquanto 81% dos exemplos positivos foram classificados como positivos, evidenciando assim, taxas baixas de falsos negativos e, mesmo que maior em quantidade no experimento realizado, falsos positivos.

O conjunto de exemplos de entrada para o Modelo 2, contendo os candidatos a subcategoria está distribuído da forma como descrito na Tabela 15.

Tabela 15: Candidato a subcategorias classificados como exemplos positivos pelo Modelo 1

Categoria	Exemplos Positivos	Exemplos Negativos	Total
actor	7	9	16
movie	9	8	17
scientist	1	4	5
website	1	6	7
Total	18	27	45

Fonte: elaborada pelo autor

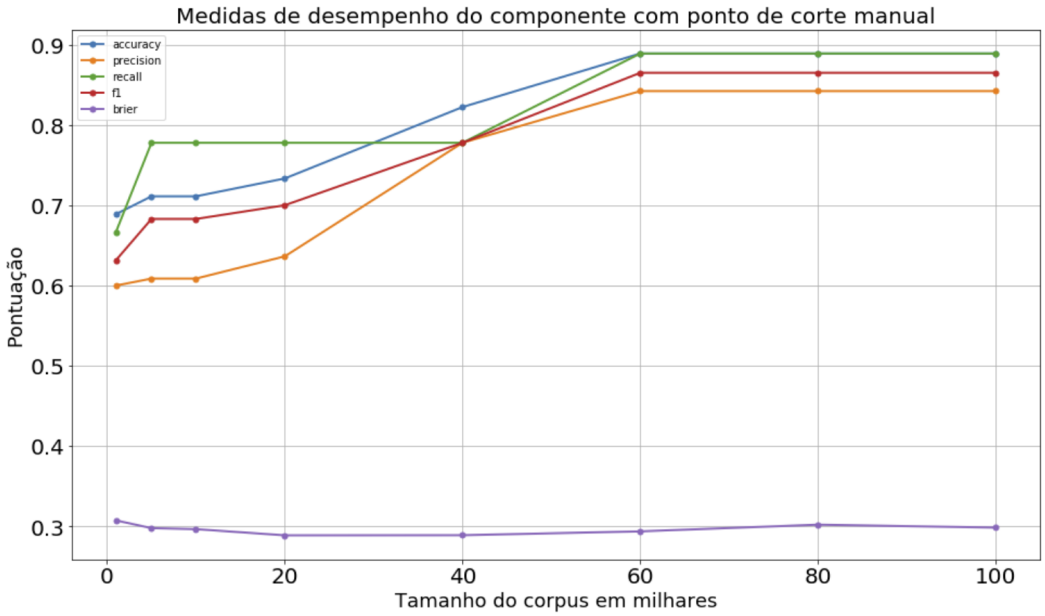
5.3.3 Resultados do Modelo 2

Como realizado na validação do Modelo 2, seção 4.2, após a execução foi definido manualmente um ponto de corte para separar exemplos negativos dos positivos. Assim, os valores retornados puderam ser submetidos às métricas de desempenho como demonstrado na Figura 21, bem como detalhados na Tabela 16 do Anexo A.

No gráfico da Figura 21, nota-se uma convergência nas métricas a partir do experimento com *corpus* de tamanho 60k. o Modelo 2 atingiu um platô em todas as métricas com exceção da *bsl*. Devido à *bsl* analisar o erro pela probabilidade de predição, a mesma apresentou pequenas variações.

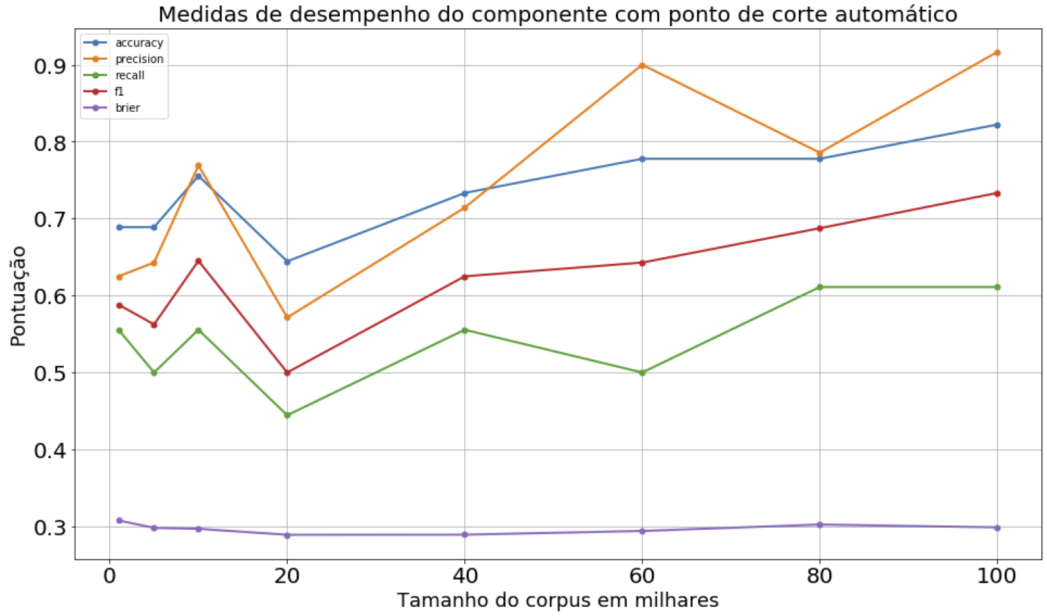
Além dos resultados com corte manual, foi realizado um experimento com um algoritmo de regressão com árvore de decisão baseado na proposta de Breiman et al. (1984), o qual busca definir um ponto de corte automaticamente. O resultado obtido é apresentado na Figura 22 e, maiores detalhes numéricos podem ser observados na Tabela 17 do Anexo A.

Figura 21: Gráfico de desempenho do Modelo 2 com corte manual em função do tamanho do *corpus*.



Fonte: elaborada pelo autor

Figura 22: Gráfico de medidas de desempenho do Modelo 2 com corte automático em função do tamanho do *corpus*



Fonte: elaborada pelo autor

Nota-se que nos dois resultados, quanto maior o número da amostra do *corpus* do Modelo 2, melhores são os resultados. Porém, não se pode concluir que a validação com corte manual se manterá com bons resultados em corpora maiores de 100 mil artigos. Além disso, foi possível notar que a simplificação do problema em uma tarefa de classificação binária facilitou a tarefa de aprendizado, mesmo que existam pontos de otimização e *spin-offs* que podem surgir a partir dos resultados obtidos.

É importante mencionar que o classificador, ainda que tenha usado um conjunto de dados balanceados, recebeu como entrada poucos exemplos sementes. Por este motivo, notou-se, geralmente, uma quantidade maior de falsos positivos do que falsos negativos. Ainda foi possível observar que o validador probabilístico do Modelo 2 atuou de forma efetiva ao validar exemplos negativos.

Ao utilizar um algoritmo de árvore de decisão para definir um ponto de corte automático na saída do validador probabilístico, notou-se uma melhora nos resultados ao utilizar *corpora* maiores. Diferente do comportamento estável obtido para o corte manual com *corpus* maior que 60k, no corte automático há uma tendência de incremento de grande parte das métricas de desempenho, principalmente após os 60k.

Dessa forma, ainda que o desempenho com um ponto de corte manual tenha se mostrado melhor nos testes em comparação com o uso de um classificador, não é possível concluir que a validação manual se manterá com bons desempenhos em *corpora* maiores.

Capítulo 6

Conclusões e Lacunas de Pesquisa

Notou-se na literatura um crescimento no uso de ontologias para representação do conhecimento humano. O uso de ontologias na maioria dos domínios de conhecimento requer tarefas constantes de atualização e expansão das mesmas.

Conforme previamente mencionado, a NELL foi o primeiro estudo de caso de um agente que utiliza o aprendizado de máquina sem fim, o qual tem como tarefa aprender a ler a *web*. Segundo Mitchell et al. (2018), a NELL alcançou um nível de competência, cresceu consideravelmente sua base de conhecimento e tornou-se apta a enfrentar tarefas mais desafiadoras do que apenas classificar sintagmas nominais em categorias e os pares de termos nominais em relações. Com base nisso, é possível notar estudos com o objetivo de expandir a base de conhecimento da NELL (SETTLES, 2011; MOHAMED; HRUSCHKA; MITCHELL, 2011; SAMADI; VELOSO; BLUM, 2013; SAMADI et al., 2015). No entanto, notou-se que os trabalhos que abordam a expansão de ontologias e bases de conhecimento de forma automática não as executam em busca de novos conceitos. Entre os trabalhos apresentados para a expansão da base de conhecimento da NELL, nenhum deles teve o objetivo de encontrar novas subcategorias das categorias já aprendidas, sendo este um dos aspectos inovadores desta tese de doutorado. Além disso, o aprendizado de subcategorias contribui para a melhoria de agentes conversacionais na busca por informações corretas que são dependentes de perguntas.

Neste sentido, foi proposto um componente modular sequencial, subdividido em dois modelos computacionais, onde o Modelo 1 recebe como entrada a ontologia da base de conhecimento da NELL, um *corpus* de textos de perguntas e respostas e um conjunto de exemplos de subcategorias como sementes, enquanto Modelo 2 recebe do Modelo 1 um conjunto de candidatos que, no que lhe concerne, faz uma validação probabilística e retorna um conjunto de novas subcategorias das categorias já conhecidas pela base de conhecimento da NELL.

Nas fases de validação probabilística do Modelo 2 descritas nas seções 5.2 e 5.3, aumenta-se o esforço computacional de maneira exponencial conforme aumenta o número de exemplos a serem validados e o número de documentos a serem comparados. Para viabilizar a execução dos testes foi necessária a utilização de uma infraestrutura computacional não convencional. Assim, os testes foram realizados com uso de máquinas virtuais por meio dos serviços de computação em nuvem da AWS, contendo de 16 a 96 núcleos de processamento. Os testes realizados na seção 5.2 com 20 mil artigos e 40 exemplos, dispenderam entre 48 e 56 horas de execução[1]. No intervalo entre os experimentos

Durante a execução do componente, notou-se um efeito funil, onde o volume de dados a serem calculados e processados diminui a cada etapa de execução. Esse fenômeno limitou, de certa forma, a seleção das categorias durante o experimento. A quantidade de exemplos com cientistas acabou sendo bem menor do que de atores, pois a frequência da palavra ator é maior que a palavra cientista no conjunto de dados utilizado. Outro fator importante desse efeito funil, se dá pelo desbalanceamento notável durante os passos que geram os exemplos rotulados e não rotulados, pois a tendência durante os testes é sempre existir mais exemplos negativos do que positivos. Esse comportamento mais restritivo do que permissivo do componente dá-lhe a vantagem de diminuir a incidência de falsos positivos.

Com base nos resultados, pode-se comprovar a validade do componente proposto ao identificar e propor candidatos a subcategorias da base de conhecimento da NELL dadas as categorias já conhecidas.

No processo de seleção do Modelo 1 descrito no Capítulo 5.3, após a identificação de sentenças do *corpus* de entrada é selecionada uma das sentenças para ser realizada a extração de atributos linguísticos. Na versão final, essa seleção é feita de maneira pseudoaleatória através de uma funcionalidade oferecida pelo pacote de recursos chamado *Pandas*. Em versões anteriores, todas as sentenças passavam pela extração de atributos linguísticos e na fase seguinte os valores resultantes dos atributos numéricos eram médias do conjunto que representava o exemplo e moda nos casos de booleanos ou categóricos. A princípio, essa técnica deveria representar melhor aquele conjunto, mas, em alguns casos, exemplos discrepantes poderiam afastar as médias dos exemplos mais comuns, tornando os valores menos representativos. Outro fator agravante é que o conjunto de sentenças de entrada não é grande o suficiente para que os valores dos atributos sejam menos influenciados pelas discrepâncias. A utilização de uma seleção única e aleatória, mesmo que existam chances de que uma discrepância seja selecionada, resulta em maiores chances de serem representativos que uma média enviesada. O aprofundamento nesse

[1] Os tempos exatos de execução não foram coletados por não fazerem parte do escopo deste trabalho.

tópico, com o objetivo de otimizar os resultados obtidos através da investigação e propostas de novas soluções, pode trazer boas contribuições em trabalhos futuros.

Conforme descrito na seção 5.2, o Modelo 2 utilizou um *corpus* não anotado a partir da Wikipédia em inglês. Tal *corpus*, por passar por um processamento, perde alguns dados que poderiam ser úteis para otimizar o processo de validação automática. Os artigos utilizados não possuem pontuação que dividem as sentenças. Dividir os artigos em sentenças pode trazer vantagens para o processo de validação. Por exemplo, o ponto final pode indicar uma quebra de sentença, separando palavras que representam uma categoria de outras que não estão no mesmo escopo, mas que poderiam ser classificadas como uma possível subcategoria apenas por estarem próximas. Mesmo que a proximidade entre as palavras não tenha sido utilizada como medida para o Modelo 2, o uso dos sinais de pontuação, assim como outras informações, poderão ser implementadas em novas pesquisas.

Uma opção de otimização que pode ser aplicada é a utilização da biblioteca de sinônimos ou possíveis variações de entidades nomeadas e sintagmas nominais, tanto na categoria quanto nas candidatas a subcategorias. Essa técnica de otimização pode ser aplicado no Modelo 1 para identificar possíveis conjuntos de palavras candidatas a subcategoria, reduzindo assim possíveis duplicidades. Além disso, é possível aplicar a mesma técnica no Modelo 2 para identificar variações dos conjuntos de palavras a serem validados, aumentando as chances de acerto dos pares que possuem algumas variações, mas que possuem o mesmo significado.

As regras para extração de atributos apresentadas na subseção 4.2.4 foram configuradas de maneira manual nesse trabalho. Nota-se, portanto uma oportunidade de pesquisa em se buscar métodos e técnicas para otimizar e automatizar essa etapa de extração de atributos, quando ainda não se tem exemplos rotulados suficientes.

Também pode ser realizada a implementação do componente proposto em um ambiente de aprendizado sem fim como o da NELL. É possível integrar o componente à base de conhecimento da NELL de forma que o sistema seja retroalimentado. Com isso, espera-se uma redução de falsos positivos e negativos a partir do processo iterativo. Essa redução na taxa de erros pode ser realizada com velocidade ainda maior com uma supervisão humana nas primeiras iterações do componente. Com a implementação de um integrador, a ontologia poderá ser expandida de maneira recursiva, de modo que o componente proposto possa encontrar subcategorias das subcategorias, deixando sua ontologia com conhecimentos mais específicos sobre determinados conceitos.

Além da integração com o ambiente de aprendizado sem fim, é possível investigar a viabilidade da abordagem proposta ser implementada em sistemas de perguntas e respostas que

utilizem ontologias, como, por exemplo o Freebase (BOLLACKER et al., 2008).

REFERÊNCIAS

BAYES, T. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, The Royal Society London, n. 53, p. 370–418, 1763.

BIRD, S.; KLEIN, E.; LOPER, E. *Natural language processing with Python: analyzing text with the natural language toolkit*. [S.l.]: "O'Reilly Media, Inc.", 2009.

BOLLACKER, K. et al. Freebase: a collaboratively created graph database for structuring human knowledge. In: ACM. *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. [S.l.], 2008. p. 1247–1250.

BREIMAN, L. et al. *Classification and regression trees*. [S.l.]: CRC press, 1984.

BRIER, G. W. Verification of forecasts expressed in terms of probability. *Monthly weather review*, v. 78, n. 1, p. 1–3, 1950.

BUITINCK, L. et al. API design for machine learning software: experiences from the scikit-learn project. In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*. [S.l.: s.n.], 2013. p. 108–122.

CALLAN, J. et al. *Clueweb09 data set*. 2009. Disponível em: <https://lemurproject.org/clueweb09.php/>.

CARABALLO, S. A. Automatic construction of a hypernym-labeled noun hierarchy from text. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. [S.l.], 1999. p. 120–126.

CARLSON, A. et al. Toward an architecture for never-ending language learning. In: *AAAI*. [S.l.: s.n.], 2010. v. 5, p. 3.

CARLSON, A. et al. Coupled semi-supervised learning for information extraction. In: ACM. *Proceedings of the third ACM international conference on Web search and data mining*. [S.l.], 2010. p. 101–110.

CHEN, T.; GUESTRIN, C. XGBoost: A scalable tree boosting system. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: ACM, 2016. (KDD '16), p. 785–794. ISBN 978-1-4503-4232-2. Disponível em: <http://doi.acm.org/10.1145/2939672.2939785>.

CHEN, X.; SHRIVASTAVA, A.; GUPTA, A. Neil: Extracting visual knowledge from web data. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2013. p. 1409–1416.

CIMIANO, P.; HOTHO, A.; STAAB, S. Learning concept hierarchies from text corpora using formal concept analysis. *Journal of artificial intelligence research*, v. 24, p. 305–339, 2005.

COWIE, J.; LEHNERT, W. Information extraction. *Communications of the ACM*, ACM New York, NY, USA, v. 39, n. 1, p. 80–91, 1996.

DALVI, B.; COHEN, W. W.; CALLAN, J. Classifying entities into an incomplete ontology. In: ACM. *Proceedings of the 2013 workshop on Automated knowledge base construction*. [S.l.], 2013. p. 31–36.

DALVI, B. et al. Automatic gloss finding for a knowledge base using ontological constraints. In: *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. [S.l.: s.n.], 2015.

DAVIS, E. *Horn clause logic*. 2007. Disponível em: <https://cs.nyu.edu/courses/spring03/G22.2560-001/horn.html>.

DONG, X. et al. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In: *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA: Association for Computing Machinery, 2014. (KDD '14), p. 601–610. ISBN 9781450329569. Disponível em: <https://doi.org/10.1145/2623330.2623623>.

ETZIONI, O. et al. Web-scale information extraction in knowitall:(preliminary results). In: ACM. *Proceedings of the 13th international conference on World Wide Web*. [S.l.], 2004. p. 100–110.

EUGENE, A.; LUIS, G. Extracting relations from large plain-text collections. *Proc. ACM*, v. 2000, 2000.

GIRARD, J.; GIRARD, J. Defining knowledge management: Toward an applied compendium. *Online Journal of Applied Knowledge Management*, v. 3, n. 1, p. 1–20, 2015.

HAASE, P. et al. A framework for handling inconsistency in changing ontologies. In: SPRINGER. *International semantic web conference*. [S.l.], 2005. p. 353–367.

HAASE, P.; VÖLKER, J.; SURE, Y. Management of dynamic knowledge. *Journal of Knowledge Management*, Emerald Group Publishing Limited, v. 9, n. 5, p. 97–107, 2005.

HAZARIKA, N. *Correlation and Data Transformations*. setembro 2013. Disponível em: <https://blog.majestic.com/case-studies/correlation-data-transformations/>.

HOFFART, J. et al. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, Elsevier, v. 194, p. 28–61, 2013.

HONNIBAL, M.; MONTANI, I. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear. 2017.

KRISHNAMURTHY, J.; MITCHELL, T. M. Which noun phrases denote which concepts? In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. [S.l.], 2011. p. 570–580.

LAO, N.; MITCHELL, T.; COHEN, W. W. Random walk inference and learning in a large scale knowledge base. In: ASSOCIATION FOR COMPUTATIONAL LINGUISTICS. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. [S.l.], 2011. p. 529–539.

LEHMANN, J. et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, IOS Press, v. 6, n. 2, p. 167–195, 2015.

LOBATO, L. M. P. *Sintaxe gerativa do Português: da teoria padrão à teoria da regência e ligação*. [S.l.]: Vigília, 1986.

MAEDCHE, A.; STAAB, S. Learning ontologies for the semantic web. In: CEUR-WS. ORG. *Proceedings of the Second International Conference on Semantic Web-Volume 40*. [S.l.], 2001. p. 51–60.

MAYO, M. *Building a Wikipedia Text Corpus for Natural Language Processing*. 2017. Disponível em: <https://www.kdnuggets.com/2017/11/building-wikipedia-text-corpus-nlp.html>.

MCGUINNESS, D. L.; HARMELEN, F. V. et al. Owl web ontology language overview. *W3C recommendation*, v. 10, n. 10, p. 2004, 2004.

MERKEL, D. Docker: lightweight linux containers for consistent development and deployment. *Linux journal*, v. 2014, n. 239, p. 2, 2014.

MILLER, G. A. *WordNet: An electronic lexical database*. [S.l.]: MIT press, 1998.

MITCHELL, T. et al. Never-ending learning. *Communications of the ACM*, ACM, v. 61, n. 5, p. 103–115, 2018.

MOHAMED, T.; HRUSCHKA, E.; MITCHELL, T. Discovering relations between noun categories. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, 2011. p. 1447–1455. Disponível em: <http://www.aclweb.org/anthology/D11-1134>.

NIKLAUS, C. et al. A survey on open information extraction. *arXiv preprint arXiv:1806.05599*, 2018.

NOY, N. F.; KLEIN, M. Ontology evolution: Not the same as schema evolution. *Knowledge and information systems*, Springer, v. 6, n. 4, p. 428–440, 2004.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

PEDRO, S. D.; APPEL, A. P.; HRUSCHKA, E. R. Autonomously reviewing and validating the knowledge base of a never-ending learning system. In: ACM. *Proceedings of the 22nd International Conference on World Wide Web*. [S.l.], 2013. p. 1195–1204.

PEDRO, S. D. S.; HRUSCHKA, E. R. Conversing learning: Active learning and active social interaction for human supervision in never-ending learning systems. In: SPRINGER. *Ibero-American Conference on Artificial Intelligence*. [S.l.], 2012. p. 231–240.

RAKSHIT, S. *Yahoo Answers Dataset — Kaggle*. 2007. `https://www.kaggle.com/soumikrakshit/yahoo-answers-dataset`. (Accessed on 12/09/2020).

REN, X. et al. First workshop on knowledge base construction, mining and reasoning. In: ACM. *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. [S.l.], 2018. p. 793–794.

ROSSUM, G. V.; DRAKE, F. L. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009. ISBN 1441412697.

SAMADI, M. et al. Askworld: Budget-sensitive query evaluation for knowledge-on-demand. In: *International Joint Conference on Artificial Intelligence (IJCAI)*. [S.l.: s.n.], 2015.

SAMADI, M.; VELOSO, M. M.; BLUM, M. Openeval: Web information query evaluation. In: *AAAI*. [S.l.: s.n.], 2013.

SANTOS, C. N. dos. *Aprendizado de máquina na identificação de sintagmas nominais: o caso do português brasileiro*. Tese (Doutorado) — Instituto Militar de Engenharia, 2005.

SETTLES, B. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In: *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK.: Association for Computational Linguistics, 2011. p. 1467–1478. Disponível em: <http://www.aclweb.org/anthology/D11-1136>.

SHIN, J. et al. Incremental knowledge base construction using deepdive. *Proceedings of the VLDB Endowment*, VLDB Endowment, v. 8, n. 11, p. 1310–1321, 2015.

SOUZA, W. W. O. et al. Nell's subcategories from a question answering environment. In: *Encontro Nacional de Inteligencia Artificial e Computacional*. [S.l.: s.n.], 2018.

SOUZA, W. W. O.; HRUSCHKA, E. R. Cognitive conversation language-ccl. In: SPRINGER. *International Conference on Intelligent Systems Design and Applications*. [S.l.], 2016. p. 309–318.

STAAB, S.; STUDER, R. *Handbook on ontologies*. [S.l.]: Springer Science & Business Media, 2010.

STOJANOVIC, L. Methods and tools for ontology evolution. Karlsruhe Institute of Technology, Germany, 2004.

STOJANOVIC, L. et al. User-driven ontology evolution management. In: SPRINGER. *International Conference on Knowledge Engineering and Knowledge Management*. [S.l.], 2002. p. 285–300.

SUCHANEK, F. M.; KASNECI, G.; WEIKUM, G. Yago: A large ontology from wikipedia and wordnet. *Web Semantics: Science, Services and Agents on the World Wide Web*, Elsevier, v. 6, n. 3, p. 203–217, 2008.

USCHOLD, M. Euroknowledge glossary. 1996.

VRANDEČIĆ, D.; KRÖTZSCH, M. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, ACM, v. 57, n. 10, p. 78–85, 2014.

WANG, Q. et al. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, v. 29, n. 12, p. 2724–2743, 2017.

WANG, R. C.; COHEN, W. W. Language-independent set expansion of named entities using the web. In: IEEE. *icdm*. [S.l.], 2007. p. 342–350.

WU, Z.; PALMER, M. Verb semantics and lexical selection. *arXiv preprint cmp-lg/9406033*, 1994.

YANG, B.; MITCHELL, T. Joint extraction of events and entities within a document context. In: *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL)*. [S.l.: s.n.], 2016.

YANG, B.; MITCHELL, T. Leveraging knowledge bases in lstms for improving machine reading. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. [S.l.: s.n.], 2017. v. 1, p. 1436–1446.

YANG, B. et al. *Embedding Entities and Relations for Learning and Inference in Knowledge Bases*. 2015.

ZABLITH, F. et al. Ontology evolution: a process-centric survey. *The knowledge engineering review*, Cambridge University Press, v. 30, n. 1, p. 45–75, 2015.

GLOSSÁRIO

AFC – *Análise Formal de Conceitos*

API – *Application Programming Interface*

CCL – *Cognitive Conversation Language*

CMC – *Coupled Morphological Classifier*

CPL – *Coupled Pattern Learner*

CSV – *Comma Separated Values*

EN – *Entidade Nomeada*

FCA – *Formal Concept Analysis*

IE – *Information Extraction*

KB – *Knowledge Base*

LE – *Learned vector embeddings*

NEIL – *Never Ending Image Learner*

NELL – *Never-ending Language Learning*

NLTK – *Natural Language Toolkit*

OWL – *Web Ontology Language*

PLN – *Processamento de Língua Natural*

PRA – *Path Ranking Algorithm*

QAS – *Question Answering Systems*

SEAL – *Set Expander for Any Language*

XGBOOST – *Extreme Gradient Boosting*

Apendice A

RESULTADOS DETALHADOS DOS TESTES REALIZADOS COM O COMPONENTE

A.1 Resultados com corte definido manualmente

Tabela 16: Medidas de desempenho do Modelo 2 com corte manual por tamanho do corpus.

size	accuracy	precision	recall	f1-score	bsl
1k	0,688889	0,600000	0,666667	0,631579	0,307690
5k	0,711111	0,608696	0,777778	0,682927	0,298065
10k	0,711111	0,608696	0,777778	0,682927	0,296798
20k	0,733333	0,636364	0,777778	0,700000	0,289099
40k	0,822222	0,777778	0,777778	0,777778	0,289230
60k	0,888889	0,842105	0,888889	0,864865	0,294033
80k	0,888889	0,842105	0,888889	0,864865	0,302344
100k	0,888889	0,842105	0,888889	0,864865	0,298726

Fonte: elaborada pelo autor

A.2 Resultados com corte definido automaticamente

Tabela 17: Medidas de desempenho do Modelo 2 com corte feito por um classificador por tamanho do corpus.

size	accuracy	precision	recall	f1-score	bsl
1k	0,688889	0,625000	0,555556	0,588235	0,307690
5k	0,688889	0,642857	0,500000	0,562500	0,298065
10k	0,755556	0,769231	0,555556	0,645161	0,296798
20k	0,644444	0,571429	0,444444	0,500000	0,289099
40k	0,733333	0,714286	0,555556	0,625000	0,289230
60k	0,777778	0,900000	0,500000	0,642857	0,294033
80k	0,777778	0,785714	0,611111	0,687500	0,302344
100k	0,822222	0,916667	0,611111	0,733333	0,298726

Fonte: elaborada pelo autor