Capítulo 4

Aprendizado de máquina e inferência em Grafos de Conhecimento

Daniel N. R. da Silva (LNCC), Artur Ziviani (LNCC) e Fabio Porto (LNCC)

Abstract

The increasing production and availability of massive and heterogeneous data bring forward challenging opportunities. Among them, the development of computing systems capable of learning, reasoning, and inferring facts based on prior knowledge. In this scenario, knowledge bases are valuable assets for the knowledge representation and automated reasoning of diverse application domains. Especially, inference tasks on knowledge graphs (knowledge bases' graphical representations) are increasingly important in academia and industry. In this short course, we introduce machine learning methods and techniques employed in knowledge graph inference tasks as well as discuss the technical and scientific challenges and opportunities associated with those tasks.

Resumo

A crescente produção e disponibilização de dados caracterizados por heterogeneidade e larga escala apresentam oportunidades desafiadoras à nossa sociedade. Dentre elas, como construir sistemas computacionais capazes de aprender, raciocinar e realizar inferências sobre fatos a partir de conhecimento prévio é uma tarefa relevante. Nesse cenário, bases de conhecimento são ativos importantes na representação e raciocínio automatizado do conhecimento de diversos domínios de aplicação. Em especial, a inferência de informação a partir de sua representação em rede — grafos de conhecimento — ganhou notoriedade na academia e indústria nos últimos anos. Em face ao exposto, neste curso, é apresentada uma introdução aos métodos e técnicas de aprendizado de máquina utilizadas em tarefas de inferência em grafos de conhecimento, discutindo-se os desafios e oportunidades tecnológicas e científicas desse tipo de tarefa.

4.1. Introdução

A representação computacional de conhecimento remonta ao nascimento da área de Inteligência Artificial. Ela é motivada pela necessidade de que a informação sobre o mundo

esteja descrita em uma forma processável e compreensível aos sistemas artificiais inteligentes [van Harmelen et al. 2008]. Nesse contexto, a representação de conhecimento na forma de uma rede tem atraído interesse da academia e indústria recentemente [Bonatti et al. 2019, Noy et al. 2019]. Esse tipo de representação, que remete ao surgimento de redes semânticas na década de 1960 [Lehmann 1992], ganhou novo fôlego no início dos anos 2010 na forma de grafos de conhecimento (*knowledge graphs*).[1]

Grafos de conhecimento têm se estabelecido como um arcabouço relevante para representação de conhecimento [Noy et al. 2019]. Eles fornecem uma estrutura semântica adequada para que sistemas computacionais sejam capazes de processar o conhecimento, assim como proveem uma representação próxima à linguagem natural. Esses grafos representam o conhecimento por meio da descrição de objetos (nós) e conexões (arestas) entre eles, sendo frequentemente imposto um esquema ou ontologia a esses objetos e conexões. Em geral, os nós desses grafos simbolizam entidades e classes do domínio de interesse, isto é, objetos do mundo real e categorias a que eles pertencem. Por sua vez, as arestas representam asserções sobre as entidades e classes de interesse. Em particular, uma aresta é usualmente disposta na forma de uma tripla (s, r, o), a qual indica que um tipo de relação r existe entre as entidades (e/ou classes) s e o.

Há várias maneiras de construir grafos de conhecimento. Eles podem ser produto de um processo de curadoria [Lenat 1995, Baker et al. 1998], de iniciativas de *crowdsourcing* [Vrandečić and Krötzsch 2014], extraídos a partir de bases contendo informação semiestruturada [Lehmann et al. 2015, Vrandečić and Krötzsch 2014], ou mesmo informação não estruturada [Dong et al. 2014]. Seja qual for a metodologia utilizada, o resultado da construção frequentemente está longe de ser perfeito [Paulheim 2017, Ratner et al. 2018]. Isso se deve a diversos fatores, incluindo a falta de informação digital sobre entidades de interesse e o processo, sujeito a falhas, empregado na construção desses grafos. A imperfeição inerente ao seu processo de construção implica diretamente na qualidade e utilidade de um grafo de conhecimento.

A qualidade e utilidade de grafos de conhecimento é atrelada a no mínimo três características: recentidade (*freshness*), exatidão (*correctness*) e completude (*coverage* ou *completeness*) [Paulheim 2017, Noy et al. 2019]. Recentidade diz respeito a se o conhecimento do grafo é atual, i.e., quão atualizada é a informação que ele contém. Já exatidão tange a se o grafo contém informação acurada, i.e., se essa informação retrata aquilo que é verdade. Por fim, completude tange a quanto do conhecimento de interesse está expresso no grafo. A dificuldade em se atender de forma abrangente a cada uma dessas três características promove a realização de tarefas de melhoria de grafos de conhecimento, isto é, de seu refino.

Tarefas de refino visam melhorar a qualidade de grafos de conhecimento ao inferir e adicionar conhecimento faltante ou ao identificar e remover erros. Nos últimos anos, essa tarefa tem sido abordada de forma desacoplada da construção de grafos de conhecimento. Por um lado, a construção é vista como um conjunto de operações (e.g., um *pipeline* analítico), realizadas sobre fontes de dados, que produzem um grafo de conhecimento. Por outro lado, o refino assume que os métodos de correção e/ou complementação serão aplicados em um grafo já existente. Perceba que o desacoplamento dessas duas ta-

[1] https://www.blog.google/products/search/introducing-knowledge-graph-things-not/

refas tem permitido o desenvolvimento de métodos de refino independentes de grafos de conhecimento [Paulheim 2017].

Técnicas de aprendizado de máquina são cada vez mais utilizadas no processo de refino de grafos de conhecimento, em particular na complementação desses grafos [Wang et al. 2017]. Em geral, essas abordagens mapeiam a complementação de um grafo de conhecimento em tarefas de aprendizado (supervisionado). Em outras palavras, um método de aprendizado é ajustado ao grafo de conhecimento a fim de realizar inferências de acordo com a tarefa mapeada. Por exemplo, se o objetivo da complementação é a adição de relacionamentos ao grafo, um classificador binário pode ser ajustado ao grafo de modo que na presença de uma tripla não observada, o classificador a atribua um escore de plausibilidade ou probabilidade relativo ao seu valor verdade. De acordo com esse escore, a tripla é ou não adicionada ao grafo de conhecimento.

Neste capítulo é apresentado sucintamente o emprego de técnicas de aprendizado de máquina em tarefas relacionadas a grafos de conhecimento; sobretudo na sua complementação. O capítulo está organizado da seguinte forma. Na Seção 4.2 é introduzido o emprego de grafos de conhecimento como forma de representação de conhecimento. Na Seção 4.3 são apresentados modelos de dados e sistemas para organização de grafos de conhecimento. Na Seção 4.4 são apresentadas algumas aplicações em que grafos de conhecimento têm sido empregados. Na Seção 4.5 são apresentadas tarefas em grafos de conhecimento, em particular, direcionadas a construção e complementação de grafos de conhecimento. Na Seção 4.6 discute-se como técnicas de aprendizado de máquina relacional são utilizadas em tarefas de complementação de grafos de conhecimento; em particular, técnicas baseadas no aprendizado de representações vetoriais. Por fim, na Seção 4.7 são feitas as considerações finais, ressaltando-se algumas oportunidades de pesquisa.

4.2. Contextualização

A popularidade recente de grafos de conhecimento tem trazido consigo ambiguidade ao termo. Tanto na indústria quanto na academia, o termo grafo de conhecimento é empregado, ora similarmente, ora distintamente de outros termos tais como base orientadas a grafos, base de conhecimento, ontologia e sistemas baseados em conhecimento [Ehrlinger and Wöß 2016]. Disto isto, a seguir são traçados alguns paralelos entre esses termos a fim de mostrar suas similaridades e diferenças.

Tradicionalmente, o termo base de conhecimento refere-se a estruturas informacionais destinadas à representação explícita de um domínio de conhecimento [Brodie and Mylopoulos 1986]. Além de conter asserções relativas às entidades desse domínio, essa classe de base é provida de uma semântica formal expressa em elementos como axiomas, definições e regras. Por exemplo, no domínio da biologia, uma base de conhecimento poderia conter um conjunto de fatos como *"O espécime Bo01 é uma borboleta Morpho Menelaus"* e axiomas como *"Toda borboleta é um inseto"*. Por causa da teoria que a base abarca (*"Toda borboleta é um inseto"*), mesmo que ela não contenha explicitamente o fato *"Bo01 é um inseto"*, ela o representa implicitamente.

A representação do conhecimento em uma base pode fazer uso de uma ontologia. Ontologias podem ser definidas como uma descrição formal dos conceitos (e.g., pessoa, localização e campo de estudo) — também chamados de classes — de um domínio de dis-

curso [Noy and Mcguinness 2001]. Em particular, conceitos contêm propriedades (e.g., a idade de uma pessoa) cujo domínio de valores é usualmente restrito. Conceitos também são arranjados em uma hierarquia taxonômica (e.g., insetos são animais) e participam de relações (e.g., pessoas podem trabalhar em empresas). Além dos conceitos, para que se permita processos de raciocínio e inferência, descreve-se regras e axiomas em ontologias. Esses processos são executados por um motor de raciocínio, acoplado à base, capaz de compreender a representação de conhecimento adotada.

Frequentemente, é possível dividir bases de conhecimento em dois componentes, um terminológico e um assertivo. Considerando que as entidades que instanciam os conceitos não são partes da ontologia, ontologias constituem o componente terminológico da base de conhecimento. Por sua vez, os fatos sobre as entidades do domínio compõem o componente assertivo da base.

Bases de conhecimento e banco de dados são estruturas distintas [Brodie and Mylopoulos 1986]. Como foi dito, bases de conhecimento devem estar associadas a uma teoria semântica que reflete o conhecimento do domínio de aplicação. Por sua vez, bancos de dados demandam uma teoria computacional concreta para o armazenamento e organização dos dados. A fim de exemplificar essa distinção, se remete ao exemplo biológico apresentado anteriormente. Nesse contexto, um banco de dados orientado a grafos conteria arestas relacionando espécimes a suas respectivas espécies (e.g., Bo01 é uma borboleta), assim como arestas relacionando taxonomicamente as espécies (e.g., Borboleta é um lepidóptero, que por sua vez, é um inseto). Perceba que o banco "desconhece" o fato *"Bo01 é um inseto"*, mesmo contendo informação suficiente para inferi-lo. Para realizar essa inferência seria necessário descrever, na forma de consulta, o caminho no grafo que explicite o encadeamento de raciocínio entre o espécime e conceito inseto.

Grafos de conhecimento representam o conhecimento de um domínio na forma de rede. Ainda não há uma definição formal para grafos de conhecimento [Ehrlinger and Wöß 2016]. Apesar disso, o termo grafo de conhecimento é utilizado frequentemente para se referir a uma estrutura (i) que representa o conhecimento de maneira similar à linguagem natural, isto é, em rede; (ii) que é útil na integração de dados de origens heterogêneas haja vista seu esquema flexível; (iii) que é frequentemente restrita por uma ontologia ou esquema de dados; e (iv) que está associada a aplicações e técnicas de inteligência artificial. Se o grafo contém uma formalização do conhecimento (e.g., uma ontologia), ele pode ser considerado um tipo de base de conhecimento no sentido tradicional do termo. Nessa perspectiva, grafos são bases de conhecimento estruturadas em grafo que armazenam informação factual na forma de relacionamentos. Para melhor compreensão daquilo a que grafos de conhecimento se referem, apresenta-se a seguir uma definição para eles. Ela se baseia nas definições apresentadas em trabalhos que propõem métodos de aprendizado para complementação de grafos de conhecimento.

É possível definir um grafo de conhecimento como o par $K = (\Delta, \Sigma)$ onde Σ e Δ denotam respectivamente os componentes terminológico (ontológico) e assertivo (de entidades) do grafo. O componente terminológico $\Sigma = (A, C, R_C, T_C, T_{C \to A}, V)$ é formado por (i) um conjunto C de conceitos; (ii) um conjunto R_C de (meta)-relações entre conceitos; (iii) um conjunto $A = \{A_j\}_{j=1}^{|A|}$ de atributos associados aos conceitos, onde cada atributo A_j toma valores em um conjunto de valores $V_j \in V$; (iv) um conjunto $T_C \subseteq C \times R_C \times C$

de relacionamentos entre conceitos; e (v) um conjunto $T_{C\to A} \subseteq C \times \{has\} \times A$ que associa atributos a conceitos, onde a constante *has* é utilizada para denotar que um conceito possuí determinado atributo (propriedade). Observe na Figura 4.1 que os nós *Pessoa* e *Organização* são conceitos, enquanto os nós *população* e *resumo* são atributos.

O componente assertivo $\Delta = (E, R_E, T_E, T_{E\to C}, T_{E\to V})$ é formado por (i) um conjunto E de entidades; (ii) um conjunto R_E de tipos de relações entre entidades; (iii) um conjunto de triplas $T_E \subseteq E \times R_E \times E$ contendo relacionamentos entre entidades; (iv) um conjunto $T_{E\to V} \subseteq E \times \cup_{j=1}^{|A|} \left(\{A_j\} \times V_j\right)$ contemplando relacionamentos atributivos, isto é, entre entidades e literais; e (v) um conjunto $T_{E\to C} \subseteq E \times \{isA\} \times C$ abarcando os relacionamentos de instanciação de entidades, onde a constante *isA* é utilizada para denotar que uma entidade instancia um conceito. Observe na Figura 4.1 que os nós *Elis* e *Maria* denotam entidades, a aresta (Elis, mãe-de, Maria) denota que Elis é mãe de Maria, enquanto a aresta (São Paulo, população, "11,967,825") denota que São Paulo possuí aproximadamente 12 milhões de habitantes.

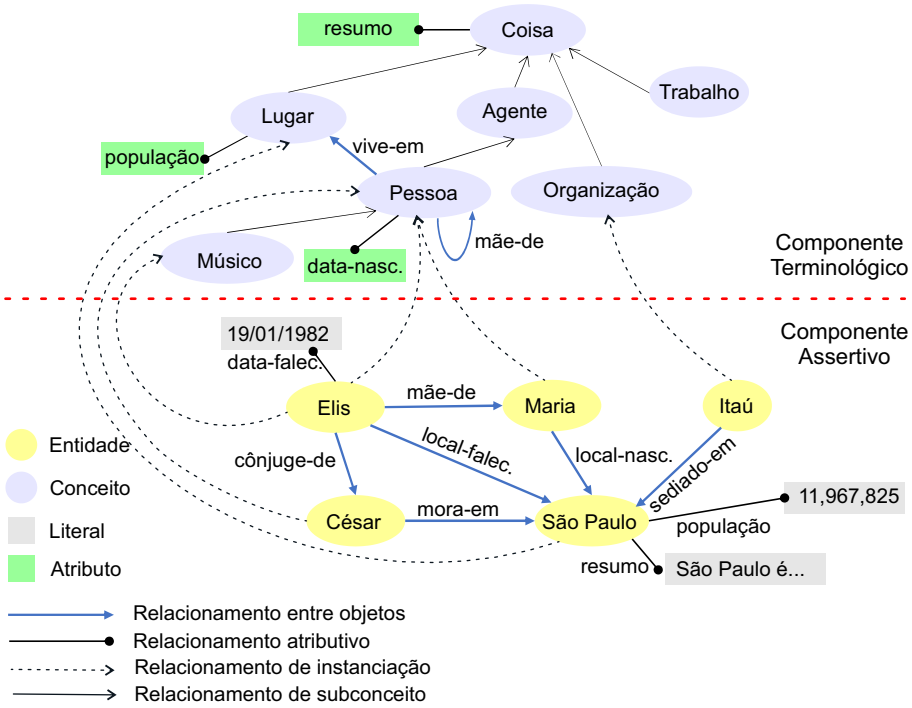

Figura 4.1. Exemplo de extrato de grafo de conhecimento.

Por fim, perceba que o conjunto de entidades e conceitos são disjuntos ($E \cap C = \emptyset$) assim como os conjuntos R_E, R_C e A também são disjuntos entre si ($(R_E \cup A) \cap R_C = \emptyset$, $R_E \cap A = \emptyset$). Além disso, o conjunto de triplas (arestas) T do grafo de conhecimento é tal que $T = T_E \cup T_{E\to C} \cup T_{E\to V} \cup T_C \cup T_{C\to A}$. Note que cada tripla $t \in T$ é da forma (s, r, o) onde s é a cabeça (sujeito), o a cauda (objeto) e r o tipo de relação.

4.3. Modelos de dados e sistemas

Para estruturar a informação referente a um grafo de conhecimento se recorre usualmente a um modelo de dados. Sobretudo, dois modelos (e suas ramificações) são usualmente

empregados na organização desse tipo de grafo: o modelo de dados do RDF (*Resource Description Framework*) e o grafo rotulado de propriedades (*Labeled Property Graph*).

O modelo do RDF[2] é o padrão W3C[3] de modelo de dados para a *Web* Semântica. Ele representa um grafo rotulado direcionado por meio de expressões da forma *sujeito-predicado-objeto*, conhecidas como triplas. Cada tripla (aresta) — identificada por um IRI (*Internationalized Resource Identifier*)[4] — representa uma asserção que relaciona dois nós do grafo. Cada nó é de um dos três tipos: recurso (*resource*), literal ou em branco (*blank*). Um nó recurso é identificado por um IRI e representa um elemento do domínio de interesse (entidades e conceitos). Por sua vez, um nó literal representa propriedades dos elementos do domínio. Para isso, ele possui um tipo de dados que define o intervalo de valores possíveis, e.g., cadeias de caracteres, números e datas. Por fim, um nó em branco representa um recurso para qual um IRI não foi dado. Note que em grafos RDF os nós e arestas não possuem uma estrutura interna, lhes distinguindo de grafos de propriedade.

Um grafo rotulado de propriedades é representado por nós, relações, propriedades e rótulos. Nesse grafo, cada nó possui um identificador único e um conjunto de propriedades (pares chave-valor) que os caracteriza. Além disso, um nó pode estar associado a zero ou mais rótulos, os quais podem representar classes, por exemplo. Por sua vez, os relacionamentos (arestas direcionadas) entre os nós devem estar associadas a um único tipo de relação. Mais ainda, de modo análogo aos nós do grafo, pode-se associar cada aresta a um conjunto de propriedades, o qual é definido pelo tipo de relação correspondente.

Nesse cenário, alguns tipos de sistemas são utilizados para o armazenamento e gerência de grafos. Em particular, de forma natural, grafos de conhecimento são usualmente armazenados e geridos em *triplestores* e sistemas de bancos de dados orientados a grafos (BDG). Dito isso, são citadas três ferramentas desenvolvidas com foco em grafos de conhecimento: Ontotext, Grakn e Amazon Neptune.

Ontotext GraphDB[5] é um triplestore com suporte a RDF e SPARQL[6]. A versão gratuita do sistema, implementada em Java, possui duas camadas: uma de inferência e outra de armazenamento, as quais empregam o *framework* RDF4J[7] de análise e consulta de dados RDF. O modelo de dados de Ontotext é baseado no RDFS (RDF Schema), o qual estende o vocabulário RDF ao permitir a descrição de taxonomias de classes e propriedades. Além disso, ele estende as definições de alguns dos elementos RDF, como o domínio e intervalo de propriedades.

Grakn[8] é um sistema de bancos de dados hiper-relacional dedutivo orientado ao armazenamento de grafos de conhecimento. O sistema lança mão de várias plataformas de computação distribuída e orientada a grafos, em especial, a JanusGraph[9], uma base de dados que implementa a API do Apache TinkerPop[10]. Além disso, Grakn provê um

[2]https://dvcs.w3.org/hg/rdf/raw-file/default/rdf-mt/index.html
[3]https://www.w3.org/.
[4]https://www.w3.org/International/iri-edit/draft-duerst-iri-05.txt
[5]https://www.ontotext.com/products/graphdb/
[6]https://www.w3.org/TR/rdf-sparql-query/
[7]https://rdf4j.eclipse.org/
[8]https://grakn.ai/
[9]https://janusgraph.org/
[10]https://tinkerpop.apache.org/

sistema de representação do conhecimento baseado em hipergrafos e no modelo de entidades e relacionamentos. Por fim, o sistema provê uma linguagem de consulta chamada Graql. Por meio dessa linguagem, o usuário define a ontologia do grafo na forma de um esquema e regras, realiza consultas declarativas (*Online Transaction Processing*) capazes de inferência e executa tarefas analíticas (*Online Analytical Processing)* como o cômputo de centralidade em grafos.

Amazon Neptune[11] é um serviço e banco de dados orientado a grafos disponibilizado pela *Amazon Web Services*. O sistema suporta os modelos de dados grafo rotulado de propriedades e RDF, respectivamente, por meio da linguagem Gremlin do Apache TinkerPop e dos padrões W3C de Web Semântica RDF 1.1 e SPARQL 1.1. Especificamente, a unidade básica de dados é uma quádrupla (sujeito, objeto, predicado, grafo) — chamada de *quad* — baseada na tripla do RDF. Cada *quad* expressa a existência de um relacionamento entre dois recursos ou anexa um par chave-valor a um recurso. Além disso, segundo a representação adotada, RDF ou grafo de propriedades, o elemento *grafo* em cada quádrupla refere-se respectivamente a um *named graph identifier* ou identificador de aresta. Por fim, tanto os dados na forma de grafos de propriedades quanto RDF são armazenados no serviço Amazon S3; especificamente, em um volume virtual único que consiste de cópias dos dados ao longo de uma região AWS única.

4.4. Aplicações

Grafos de conhecimento têm se tornado uma tecnologia cada vez mais presente na indústria e academia, obtendo um papel de destaque em diversas aplicações. Na indústria, grafos de conhecimentos são adotados em aplicações como motores de busca, mecanismos de resposta a perguntas, sistemas de recomendação e agentes conversacionais. Na academia, espera-se que esses grafos promovam aplicações científicas — por exemplo, em biologia e medicina — por meio da integração de conhecimento acadêmico, assim como aplicações de grande impacto social como o combate à difusão de notícias falsas.

Atualmente os principais motores de busca — e.g., Baidu, Bing e Google – lançam mão de grafos de conhecimento na tarefa de resposta a consultas. Por exemplo, Freebase[12] foi utilizado na construção do grafo de conhecimento do Google. Esses motores de busca recorrem ao conhecimento enciclopédico e factual, expresso nesses grafos, sobre as entidades mais diversas, incluindo pessoas, localizações e instituições. Por exemplo, dada a consulta *Altura do Monte Everest*, além de apresentarem ao usuário documentos relacionados à consulta, esses motores também exibem painéis de informação, e.g., com a altura do monte e entidades relacionadas a ele.

Grafos de conhecimento também são empregados em sistemas computacionais como assistentes virtuais (e.g., Amazon Alexa, Google Assistant e Microsoft Cortanta), robôs conversacionais (e.g., ebay ShopBot e SalesForce Einstein Bot) e de respostas a perguntas (e.g., IBM Watson). Por exemplo, IBM Watson emprega grafos de conhecimento — e.g., DBpedia [Lehmann et al. 2015], Freebase e Yago [Rebele et al. 2016] — como fonte de informação estruturada [Nickel et al. 2016]. No contexto de comércio ele-

[11] https://aws.amazon.com/neptune/

[12] https://developers.google.com/freebase/data

trônico, ShopBot[13] foi um robô que recorria a grafos de conhecimento, contendo dados comportamentais e informação enciclopédica, a fim de compreender e refinar os pedidos de usuários em compras virtuais.

Cada vez mais sistemas de recomendação fazem parte da vida das pessoas, as sugerindo itens de interesse, por exemplo, filmes e músicas em serviços de *streaming*. Para realizar as recomendações é preciso que se modele nesses sistemas as interações entre os usuários e os itens de interesse. Isso é tradicionalmente feito ao recorrer-se a métodos de filtro colaborativo. Entretanto, o desempenho desses métodos sofre com a esparsidade dos relacionamentos entre os usuários e itens, além da falta de informação a respeito de usuários e itens recentes no sistema. Para enfrentar esses problemas, pode-se lançar mão da informação relativa aos usuários e itens. Em especial, grafos de conhecimento podem ser utilizados na estruturação dessa informação, promovendo recomendações mais adequadas [Wang et al. 2019a].

Grafos de conhecimento possuem grande potencial de aplicação nas áreas médicas e biológicas. Nessas áreas, grafos de conhecimento podem ser empregados na integração de conhecimento e informação biomédica. Por exemplo, a análise do grande e heterogêneo volume de literatura biomédica pode alavancar a descoberta de novos medicamentos. Nesse sentido, o método chamado GrEDel (*Graph Embedding based Deep Learning Method*) pode ser utilizado no processo de descoberta de fármacos [Sang et al. 2019]. Em resumo, esse método constrói um grafo de conhecimento a partir de resumos de artigos da literatura biomédica e aplica técnicas de aprendizado na descoberta de possíveis medicamentos.

Espera-se que grafos de conhecimento possam promover aplicações em medicina personalizada, a qual leva em conta a informação específica de cada paciente (e.g., variabilidade genética, ambiente e estilo de vida) na prevenção e tratamento de doenças. Essa abordagem médica depende da integração de um conjunto heterogêneo de informação sobre o paciente, o que pode incluir informação genética, além de dados sobre a administração de medicamentos e sobre o monitoramento das funções biológicas. Nesse contexto, a base (grafo) de conhecimento *Precision Medicine Knowledge Base* (PredMedKB) é uma iniciativa de integração. Em específico, esse grafo de conhecimento visa integrar informação e conhecimento sobre os quatro componentes fundamentais da medicina de precisão: doenças, genes, variantes genéticas e drogas [Yu et al. 2018].

Grafos de conhecimento também podem impulsionar o processo de checagem de fatos. Por causa dos danos sociais que a prática de propagação de notícias falsas incorre, iniciativas para checagem de fatos — e.g., o sítio *web* Snopes[14] — se fazem relevantes; em particular, iniciativas que realizam essa checagem de forma automatizada. Nesse contexto, grafos de conhecimento podem alavancar esse tipo de iniciativa ao serem empregados em métodos de detecção de notícias falsas baseada em conteúdo (*content based fake news detection*) [Pan et al. 2018].

[13]https://www.ebayinc.com/stories/news/say-hello-to-ebay-shopbot-beta/
[14]https://www.snopes.com/

4.5. Tarefas em grafos de conhecimento

Grafos de conhecimento estão associados a um conjunto de tarefas computacionais; desde a extração de informação até o uso do grafo na aplicação fim. Dentre essas, são ressaltadas, e apresentadas a seguir, a construção e o refino de grafos de conhecimento. Como o maior interesse deste capítulo é no uso de aprendizado de máquina no processo de refino do grafo, em particular, na inferência de elementos do grafo de conhecimento, o processo de construção é exposto de forma conceitual.

4.5.1. Construção automatizada de bases e grafos de conhecimento

O processo de construção, isto é, povoamento de bases e grafos de conhecimento com informação de interesse, tem se tornado uma tarefa cada vez mais automatizada. Em especial, a construção semiautomática desse tipo de base a partir da integração de dados estruturados, semiestruturados e não estruturados tem se tornado factível. Isso se deve em parte considerável a técnicas de extração de conhecimento baseadas em aprendizado de máquina. Em particular, aquelas que lançam mão de modelos de aprendizado profundo; estas têm obtido nos últimos anos desempenho estado da arte em subtarefas da construção de bases de conhecimento, operando diretamente nos dados de entrada como texto e imagem [Ratner et al. 2018]. Nesse contexto, descreve-se a seguir brevemente as tarefas de extração de entidades e relacionamentos, úteis no povoamento de bases de conhecimento; elas são descritas de forma conceitual e tomando em conta dados não estruturados (textuais). Em seguida, são apresentados os sistemas DeepDive e Fonduer, destinados à construção automatizada de bases e grafos de conhecimento.

4.5.1.1. Extração de entidades e relacionamentos

A tarefa de extração de entidades (*entity extraction*) visa obter entidades de interesse a partir de dados semiestruturados ou não estruturados [Yan et al. 2016]. Nesse contexto, o reconhecimento de entidades nomeadas e a ligação de entidades são duas subtarefas importantes desse tipo de extração. O objetivo do reconhecimento de entidades nomeadas (*Named Entity Recognition* - NER) é a identificação e classificação de entidades nomeadas (objetos do mundo real, e.g., pessoas e localizações) em documentos textuais. Por exemplo, ao ser considerado o fragmento textual *Einstein nasceu na Alemanha*, o resultado desejado do reconhecimento é a identificação e classificação dos termos *Einstein* e *Alemanha*, presentes no fragmento, como uma pessoa e país respectivamente. Em suma, dada uma sentença $x = (w_1, w_2, ..., w_n)$ o reconhecimento deve gerar como saída tuplas da forma (i_s, i_e, t) onde $i_s, i_e \in \{1, 2, ..., n\}$ são os índices inicial e final respectivamente de cada entidade nomeada, enquanto t é a classe/tipo a ela associada.

O objetivo da tarefa de ligação de entidades (*entity linking* ou *named entity desambiguation*) é vincular menções textuais as suas respectivas representações em um grafo de conhecimento de interesse [Yan et al. 2016]. Geralmente, essa tarefa está associada ao reconhecimento de entidades. Especificamente, ela realiza o processo de ligação a partir das menções (entidades nomeadas) produzidas durante o processo de NER. Por exemplo, o termo *Apple* nos trechos *Apple significa maçã em inglês* e *Apple é uma empresa de tecnologia* refere-se respectivamente a uma fruta e a uma empresa. Um método de ligação

de entidades deve associar o termo *Apple* no primeiro e segundo fragmento a entidades distintas no grafo de conhecimento. Em outras palavras, deve-se associar, se possível, cada entidade nomeada x a uma entidade $e \in E$ no grafo de conhecimento.

O objetivo da tarefa de extração de relacionamentos é obter fatos sobre as entidades de interesse a partir dos dados; por exemplo, o fato (Barack Obama, casado-com, Michelle Obama) a partir do fragmento textual *Barack Obama é casado com Michelle Obama* [Yan et al. 2016]. Usualmente, a tarefa de extração de relacionamentos é tomada como um problema de classificação binário. Por exemplo, dada uma sentença $x = (w_1, ..., e_1, ..., w_i, .., .e_2., .., w_n)$, onde e_1 e e_2 são entidades nomeadas e ϕ_x é o conjunto de características associado a x, deseja-se aprender um classificador f_r tal que $f_r(\phi_x) = 1$ se e_1 e e_2 são relacionadas pela relação r e $f_r(\phi_x) = 0$, caso contrário.

4.5.1.2. DeepDive

DeepDive [Zhang et al. 2016] é um sistema destinado à construção semiautomática de bases de conhecimento. A partir de uma coleção de dados estruturados, semiestruturados e não estruturados, o sistema extrai fatos, povoando uma base relacional. A principal motivação de DeepDive é aliviar o fardo de engenharia de características vinculado ao emprego de técnicas de aprendizado de máquina na construção desse tipo de base. Para isso, DeepDive implementa um conjunto de funcionalidades para extração de relacionamentos e emprega um modelo probabilístico na inferência do valor verdade dos elementos extraídos. Deve-se ressaltar que o sistema promoveu o desenvolvimento de aplicações em diversos domínios, incluindo no combate ao tráfico humano e em paleontologia.[15] Dito isso, o processo de construção adotado por DeepDive é descrito em termos gerais a seguir.

Em primeiro lugar, a coleção de documentos provida pelo usuário é armazenada em um banco de dados relacional. Por padrão, cada documento dessa coleção é processado e armazenado no formato uma linha por sentença de texto. Nesse processo, são anexadas aos textos marcações produzidas por ferramentas de pré-processamento de linguagem natural disponibilizadas no sistema. Após a ingestão dos documentos, DeepDive executa dois tipos de consultas: mapeamentos de candidatos a relacionamento e associação de características. O primeiro tipo produz menções textuais, entidades e relacionamentos possíveis e o segundo associa características aos candidatos a relacionamentos.

Posteriormente, o usuário elabora, de forma assistida, o conjunto de treinamento empregado no ajuste do modelo probabilístico. Em particular, DeepDive associa a cada relação da base de conhecimento uma relação evidência de mesmo esquema, salvo um campo adicional que indica se uma tupla na relação é falsa ou verdadeira. O povoamento da relação evidência é feita por meio de rotulagem manual ou supervisão distante.

A fim de estimar a probabilidade de os candidatos serem verdade, o sistema adota um grafo de fatores (*factor graph*) como modelo probabilístico, similar a Redes Lógicas de Markov [Richardson and Domingos 2006], além de usar técnicas do sistema Tuffy [Niu et al. 2011]. Candidatos cujas estimativas são maiores do que um limiar estabelecido pelo usuário são promovidos a relacionamentos.

[15] http://deepdive.stanford.edu/showcase/apps

4.5.1.3. Fonduer

Fonduer [Wu et al. 2018] é um sistema para construção de bases de conhecimento a partir de documentos formatados de forma complexa. Em geral, os sistemas destinados a construção de bases de conhecimento realizam o processo de extração a partir de dados textuais semiestruturados e tabulares. De forma distinta, Fonduer visa efetuar a construção de bases de conhecimento levando em conta informação multimodal. Por exemplo, no contexto de relatório técnicos, o sistema pode extrair o fato *O lucro líquido no quarto bimestre foi de $100* a partir de uma tabela e suas evidências textuais.

O processo de extração em Fonduer é realizado de forma semisupervisionada com base em heurísticas do usuário, assim como métodos de supervisão fraca (*weak supervison*) e modelos de aprendizado profundo. Em primeiro lugar, o usuário estabelece um conjunto de documento de interesse (e.g., PDFs e páginas em HTML) e o esquema alvo (tipo de relação), por exemplo, triplas do tipo (Cônjuge A, casado-com, Cônjuge B). O sistema processa cada documento de entrada em um modelo de dados que associa características aos elementos de informação, e.g., a altura relativa à página de uma tabela em um documento PDF. Além disso, o usuário escreve um conjunto de funções arbitrárias para extrair menções a entidades, por exemplo, a partir de fragmentos textuais ou tabelas HTML. O produto cartesiano entre essas menções forma o conjunto de candidatas a relacionamentos.

Como o número de candidatos pode ser grande, um conjunto de funções heurísticas, escritas pelos usuários, é utilizado para eliminar parte dos candidatos a relacionamentos. O usuário ainda descreve um conjunto de funções rotuladoras que associam a cada candidato não eliminado um rótulo de crença: verdadeiro, falso ou abstenção. Note que para um mesmo candidato, uma rotuladora pode associar um rótulo verdadeiro, enquanto outra um rótulo falso. Com base nos rótulos produzidos para os candidatos, um modelo generativo, baseado em *Data Programming* [Ratner et al. 2016], é aprendido a fim de estimar o erro associado aos rótulos e produzir um rótulo único (estocástico) para cada candidato. Por fim, os candidatos e seus respectivos rótulos são passadas a uma rede neural BiLSTM (*Bidirectional Long Short Term Memory*) multimodal que lança mão das características associadas aos candidatos para os classificar como verdadeiros ou falsos. Os candidatos possuindo rótulos verdadeiros são adicionados à base.

4.5.2. Refino de Grafos de Conhecimento

Por causa da natureza de seu processo de construção, grafos de conhecimento frequentemente contêm informação faltante ou ruidosa. Por exemplo, relacionamentos entre entidades do grafo que existem na realidade e não estão expressos no grafo, ou que não existem e estão. Com isso, o refino (*refinement*) de grafos de conhecimento é um processo natural. Nesse sentido, tarefas de refino de grafos de conhecimento podem ser divididas em no mínimo três maneiras distintas: (i) *objetivo geral da tarefa*: complementação ou correção; (ii) *alvo do refino*: por exemplo, entidades, relacionamentos, atributos; e (iii) *uso de informação lateral*, por exemplo, emprego de fontes de informação externas ao grafo de conhecimento na execução da tarefa [Paulheim 2017]. Como o processo de complementação é aquele de maior interesse deste capítulo, não são abordadas tarefas de correção. Interessados na correção de grafos de conhecimento podem recorrer a [Paulheim 2017].

4.5.2.1. Complementação de Grafos de Conhecimento

O objetivo da complementação de grafos de conhecimento é a adição de informação faltante, isto é, nós ou arestas ao grafo. Dentre as tarefas de complementação, a inferência de fatos não observados a partir do grafo se destaca. Esse tipo de inferência se traduz na predição de arestas do grafo de conhecimento, portanto, na inferência de informação relativa ao seu conjunto de triplas. A seguir são apresentadas as principais tarefas relacionadas a inferências de fatos em grafos de conhecimento. Posteriormente, na Seção 4.6, são apresentadas de maneira explícita, algumas técnicas para resolver essas tarefas.

Há no mínimo cinco tarefas associadas à complementação de fatos: inferência do valor verdade de triplas, predição de ligações, predição de atributos, predição de relações e classificação de entidades (ver Tabela 4.1). No âmbito de aprendizado de máquina, cada uma dessas tarefas é resolvida por meio do ajuste de um modelo à informação expressa no grafo de conhecimento. Em particular, de acordo com a tarefa de interesse, esse modelo toma como entrada uma tripla do grafo e produz um escore de plausibilidade.

O objetivo da tarefa de classificação de triplas (*triple classification*) é inferir o valor verdade de triplas não observadas. Em outras palavras, deseja-se corretamente inferir se triplas de consulta $(s,r,o) \notin T$ pertencem ou não ao grafo de conhecimento; por exemplo, o valor verdade da tripla (Einstein, morreu-em, EUA). Essa tarefa pode ser tratada como um problema de classificação binário, onde uma classe indica a veracidade de uma tripla, enquanto outra sua falsidade.

Tabela 4.1. Exemplos ilustrativos de complementação de fatos.

Tarefa	Exemplo de tripla de consulta	Exemplo de resultado
Classificação de tripla	(Einstein, morreu-em, EUA)	(Sim, 90%)
Pred. de ligação (cauda)	(Elvis Presley, estrelou-em, ?)	(Feitiço Havaiano, ...)
Pred. de ligação (cabeça)	(?, estrelou-em, Casablanca)	(Humphrey Bogart, ...)
Predição de relação	(Einstein, ?, Alemanha)	(nasceu-em, ...)
Predição de atributo	(B. Obama, nacionalidade, ?)	(americano, queniano, ...)
Classificação de entidade	(Michael Jackson, isA,?)	(cantor, compositor, ...)

Tipicamente, o objetivo da tarefa de predição de ligações (*link prediction*) é predizer se uma entidade se relaciona com outra, ou se um conceito está associado a outro. Em particular, no caso das entidades, deseja-se saber quais entidades $e \in E$ satisfazem determinada tripla de consulta incompleta na forma $(?,r,o)$ (predição de sujeito/cabeça) ou $(s,r,?)$ (predição de objeto/cauda), onde o símbolo "?" denota o alvo de inferência. Por exemplo, as consultas podem ter como objetivo o conhecimento sobre o filme Casablanca — (?, estrelou-em, Casablanca) — e o cantor Elvis Presley — (Elvis Presley, estrelou-em, ?). Note que o resultado desta tarefa é uma lista ranqueada de entidades (e.g., começando com filmes na segunda consulta) de maneira decrescente pelo escore de plausibilidade. Note que quanto maior o escore de um item, mais o modelo acredita que ele é verdadeiro.

O objetivo da predição de relações é inferir os tipos de relação existentes entre entidades ou conceitos, isto é, inferir que elementos satisfazem triplas de consulta $(s,?,o)$. Essa tarefa pode ser abordada a partir da classificação de triplas ou de forma análoga à

predição de ligações. Na primeira abordagem, para uma consulta $(s, ?, o)$, avalia-se o rótulo de classificação de todas as triplas (s, r, o), isto é, para todos os tipos de relação ($r \in R_E$ ou $r \in R_C$). Por outro lado, de maneira análoga a predição de ligações, é possível avaliar a classificação dos tipos de relação para aquela consulta.

Por fim, as tarefas de classificação de entidades e predição de atributos podem ser abordadas como especializações da predição de ligações. O objetivo da tarefa de classificação de entidades é associar classes às entidades do grafo. Se as classes estiverem expressas no grafo (conceitos), essa tarefa pode ser simplesmente tratada como um problema de predição de ligações do tipo $(s, isA, ?)$, onde $s \in E$ e "?" $\in C$. Caso as classes não estejam expressas no grafo, o problema pode ser visto como uma tarefa de aprendizado multiclasse, caso apenas uma classe deva ser associada a cada entidade, ou multirrótulo, caso mais de uma classe possa ser associada a cada entidade. Por sua vez, a predição de atributos visa inferir relacionamentos atributivos, isto é, o valor de um atributo associado a determinada entidade. Por exemplo, se o domínio do atributo for finito (ou considerado como finito), esse tipo de predição pode ser traduzida na predição de ligações da forma $(s, a, ?)$ onde $s \in E$, $a \in A$ e "?" $\in V_a$.

4.6. Aprendizado de Máquina Relacional

Aprendizado de máquina relacional (AMR) destina-se à criação de modelos estatísticos para dados relacionais, isto é, dados cuja a informação relacional é tão ou mais importante que a informação individual de cada elemento. Essa classe de aprendizado tem sido utilizada em diversas aplicações, por exemplo, na extração de informação de dados não estruturados [Zhang et al. 2016] e na modelagem de linguagem natural [Vu et al. 2018]. Em particular, técnicas AMR têm sido amplamente empregadas em tarefas associadas a grafos de conhecimento, sobretudo na sua complementação [Nickel et al. 2016].

A adoção de técnicas de aprendizado de máquina relacional em tarefas de complementação se baseia na ideia de existência de regularidades semânticas presentes no grafo de conhecimento. Essas regularidades, produto de padrões universais ou estatísticos, fazem com que o valor verdade de um relacionamento seja correlacionado com o valor verdade de outros relacionamentos. Por exemplo, em grafos de diversos domínios há uma tendência de entidades similares — i.e., que compartilham atributos comuns como faixa etária e crenças — se inter-relacionarem [Nickel et al. 2016]. Nesse caso, dadas duas entidades similares, se uma delas participa de um determinado tipo de relação, a chance da outra participar no mesmo tipo de relação aumenta.

Assumindo que os relacionamentos de interesse se deem apenas entre as entidades observadas no grafo de conhecimento, técnicas de AMR adotam três metodologias principais para abordar a existência e interdependência das triplas possíveis [Nickel et al. 2016]: (i) *modelos gráficos probabilísticos* assumem que a existência de cada tripla possível dependa da existência de um conjunto local de triplas; (ii) *modelos de características de grafo* assumem que a existência de cada tripla possível seja condicionalmente independente das demais, dadas as características **observadas** do grafo (e.g., caminhos) e parâmetros adicionais do modelo; e (iii) *modelos de características latentes* assumem que a existência de cada tripla possível seja condicionalmente independente das demais tripas dados os parâmetros do modelo e as características **não observadas** das entidades

s, o e relação r.

A seguir, essas metodologias de modelagem são apresentadas, sendo dada maior ênfase à apresentação de modelos de características latentes. A fim de simplificar a discussão, considere que, durante essa apresentação, apenas os relacionamentos entre entidades sejam de interesse, isto é, o domínio de triplas possíveis D seja tal que $D = E \times R_E \times E$. Ao discutir-se os aspectos de modelos de características latentes, são realizadas as devidas considerações sobre os demais tipos de relacionamento, e.g., ontológico e atributivo.

4.6.1. Modelos gráficos probabilísticos

No contexto de complementação, modelos gráficos probabilísticos assumem que a existência de uma tripla — isto é, ela representar uma proposição verdadeira — possa estar relacionada com as demais triplas [Raedt et al. 2016]. Em particular, para capturar a interdependência entre a existência de triplas, adota-se um grafo de dependências. Cada nó desse grafo representa uma variável estocástica $Y_{(s,r,o)} \in \{0,1\}$, a qual indica a existência de uma tripla possível $(s,r,o) \in D$. Por sua vez, cada aresta desse grafo de dependências modela a interdependência entre duas triplas. Uma vez que é impraticável considerar todas as $|D| \times (|D|-1)$ possíveis interdependências, é necessário que apenas aquelas mais relevantes sejam consideradas. Nesse contexto, usualmente emprega-se o modelo gráfico probabilístico não direcionado Campos Aleatórios de Markov como ferramenta de representação dessas interdependências. Particularmente no contexto de complementação são adotadas Redes Lógicas de Markov [Richardson and Domingos 2006], uma extensão desse modelo.

Redes Lógicas de Markov combinam Campos Aleatórios de Markov e lógica de primeira ordem. Nesse sentido, além do conjunto de triplas, emprega-se um conjunto de fórmulas lógicas que expressam regras e heurísticas do domínio do grafo de conhecimento, sendo cada fórmula associada a um peso real. Por exemplo, a fórmula (X, cônjuge_de, Y), (Y, mãe_de, Z) $\rightarrow$ (X,pai_de, Y) indica que usualmente o esposo da mãe de um indivíduo é seu pai. Essas fórmulas são utilizadas na definição de quais interdependências entre triplas devem ser consideradas. Em um processo chamado de instanciação, essas fórmulas são instanciadas (e.g., (João, cônjuge_de, Maria), (Maria, mãe_de, Lúcio) $\rightarrow$ (João, pai_de, Lúcio)) de forma coerente, isto é, obedecendo as restrições. Com base nesse processo, a probabilidade conjunta da existência de triplas é modelada por

$$P\left(\bigcap_{(s,r,o)\in D} Y_{(s,r,o)} \mid \theta\right) = \frac{1}{Z}\prod_i \exp\left(\theta_i \cdot x_i\right) \tag{1}$$

onde x_i e θ_i denotam respectivamente a quantidade de instanciações válidas e pesos associados à fórmula f_i. Além disso, Z é uma função de partição que assegura que P é uma distribuição de probabilidade.

Como o processo de inferência — estimativa da atribuição mais provável para os $Y_{(s,r,o)}$ — é um problema computacionalmente intratável, emprega-se abordagens heurísticas, por exemplo, amostragem de Gibbs e MC-SAT. Além disso, como o aprendizado de parâmetros θ por maximização de verossimilhança ou probabilidade a *posteriori* recorre a etapa de inferência, são empregadas aproximações como pseudo-verossimilhança.

4.6.2. Modelos de características de grafo

Modelos de características de grafo lançam mão de representações baseadas em elementos observáveis na estrutura do grafo, por exemplo, caminhos e vizinhanças. Esse tipo de método parte da premissa de que existem padrões expressos no grafo que possuem poder preditivo. Por exemplo, a quantidade de caminhos entre duas entidades pode ser um indicador da existência de determinado relacionamento entre elas. Nesse contexto, algumas abordagens para inferência de triplas incluem o uso de índices de similaridade, mineração de regras e programação lógica indutiva [Nickel et al. 2016]. Dentre essas, destaca-se o método *Path Ranking Algorithm*.

Path Ranking Algorithm [Lao et al. 2011] é um algoritmo para produção de modelos de características de grafo. Ele emprega a exploração aleatória de caminhos de comprimento limitado no grafo de conhecimento a fim de construir representações vetoriais (vetores de características) para suas triplas. A construção dessas representações é dividida em duas etapas, extração de características e treinamento. Na etapa de extração de características, um conjunto de caminhos de relação é selecionado; por exemplo, um conjunto $P = \{p_i\}_{i=1}^{|P|}$ de caminhos de comprimento n. Cada um desses caminhos segue a forma $p = (r_1, r_2, ..., r_n)$ onde cada r_i é um tipo de relação. Por exemplo, p = (cônjuge_de, mãe_de) é um caminho de relação de comprimento dois.

Após serem extraídos os caminhos de relação, um conjunto de treinamento é selecionado a partir do conjunto de triplas. Para cada tripla (s, r^*, o) no conjunto de treinamento e cada caminho de relação $p \in P$ computa-se a probabilidade que ao se iniciar o caminho p em s se chegue a o de forma consistente, isto é, seguindo os tipos de relação expressos em p. Note que o cômputo dessa probabilidade é feito de forma uniforme, isto é, a probabilidade de navegar-se "para fora" de um nó s através de um determinado tipo de relação r' é proporcional a quantidade de vizinhos associados a s por r'.

Após computado, o conjunto de probabilidades é empilhado em um vetor de características $f^{\text{PRA}}_{s,r^*,o} \in \mathbb{R}^{|P|}$ e associado à tripla (s, r^*, o). Computadas as representações vetoriais para as triplas de um conjunto de treinamento, um modelo de aprendizado "de prateleira" é ajustado. Por exemplo, ao ser empregado um modelo de regressão logística, define-se o escore dado a uma tripla (s, r, o) como

$$\phi^{\text{PRA}}_{(s,r,o)} := \sigma\left(\mathbf{v}_r^\top f^{\text{PRA}}_{s,r,o}\right) \tag{2}$$

onde $\mathbf{v}_r \in \mathbb{R}^{|P|}$ denota o vetor de pesos (a ser aprendido) associado ao tipo de relação r e $\sigma(x) = 1/(1+\exp^{-x})$ é a função sigmoide. Note que a cada nova consulta sobre a existência de uma tripla é necessário computar o vetor de características a ela associado.

4.6.3. Modelos de características latentes

Nos últimos anos, o desenvolvimento de modelos de características latentes tem se tornado a linha de pesquisa dominante na tarefa de complementação [Kejriwal 2019], sendo possível enunciar alguns fatores para isso. Em primeiro lugar, há o sucesso recente da área de pesquisa de aprendizado de representações (*embeddings*) [Bengio et al. 2013, Hamilton et al. 2017]. No caso particular de grafos de conhecimento, a ideia é que as representações das entidades e relacionamentos, necessárias para o melhor desempenho

de um modelo, precisam ser aprendidas. Em outras palavras, elas devem ser produzidas durante o processo de aprendizado de um modelo e não engendradas minuciosamente a priori [Hamilton et al. 2017]. Isso se contrapõe às abordagens de características de grafo, as quais definem a priori vetores de características com base em propriedades do grafo (e.g., estatísticas sumarizantes). Em segundo lugar, modelos de características latentes não pressupõem uma representação simbólica mais formal do conhecimento (e.g., definição de regras) e têm demonstrado serem escaláveis a grafos com milhões de entidades [Nickel et al. 2016]. Isso vai de encontro a grande parte dos modelos gráficos probabilísticos usados em complementação, como as Redes Lógicas de Markov, apresentadas anteriormente. Essa última classe de modelos, apesar de ter sido dominante na tarefa de complementação no passado, perdeu popularidade por causa de dificuldades tangentes à escalabilidade dos processos de inferência [Kejriwal 2019].

De forma geral, modelos de características latentes, também chamados de modelos de *embedding*, embutem entidades e relações em espaços vetoriais reais e complexos [Wang et al. 2017]. O modelo é ajustado para que a estrutura do espaço de *embedding* reflita a estrutura do grafo de conhecimento; por exemplo, mantendo uma certa similaridade entre os relacionamentos geométricos das representações vetoriais e seus correspondentes expressos simbolicamente no grafo de conhecimento. Além disso, a dimensão desse espaço escolhido precisa ser bem menor do que a quantidade de entidades presentes no grafo. Desse modo, uma maior quantidade de regularidades presentes no grafo pode ser capturada. Entretanto, esse número não deve ser muito baixo a ponto de as representações vetoriais não serem capazes de modelar a semântica do grafo.

As técnicas baseadas em *embeddings* podem ser categorizadas em dois grupos: modelos de distância translacional (*translational distance models*) e modelos de correspondência semântica (*semantic matching models*) [Wang et al. 2017]. Modelos de distância translacional exploram funções de escore baseadas em distância. Isto é, eles medem a plausibilidade de um fato como algum tipo de distância entre as representações vetoriais das entidades envolvidas nesse fato, usualmente após a translação pelo tipo de relação correspondente. Por sua vez, modelos de combinação exploram funções de escore baseadas em similaridade. Eles medem a plausibilidade de um fato ao combinar a semântica latente de entidades e relacionamentos. A seguir são apresentados alguns desses modelos. Posteriormente, o processo de treinamento utilizado no aprendizado desses modelos é discutido.

4.6.3.1. Modelos de distância translacional

TransE [Bordes et al. 2013] foi um dos primeiros modelos de *embedding* propostos para grafos de conhecimento; sendo ele de certo modo o "pai" dos modelos translacionais. Entretanto, apesar de sua idade, ele continua sendo relevante, tanto como medida de comparação quanto base para novos modelos. Por exemplo, TransH [Wang et al. 2014], TransR [Lin et al. 2015] e TransA [Jia et al. 2016] estendem as ideias de TransE como é disposto na Figura 4.2 e Tabela 4.2. Em particular, em TransE os relacionamentos são representados como translações em um espaço de *embedding*. Uma das motivações para esse tipo de abordagem vem do uso de aprendizado de representações no processamento

de linguagem natural. Nesse contexto, observou-se que alguns modelos de *embedding* representavam as palavras referentes a relacionamentos (e.g., *capital-de*) como translações [Bouraoui et al. 2018].

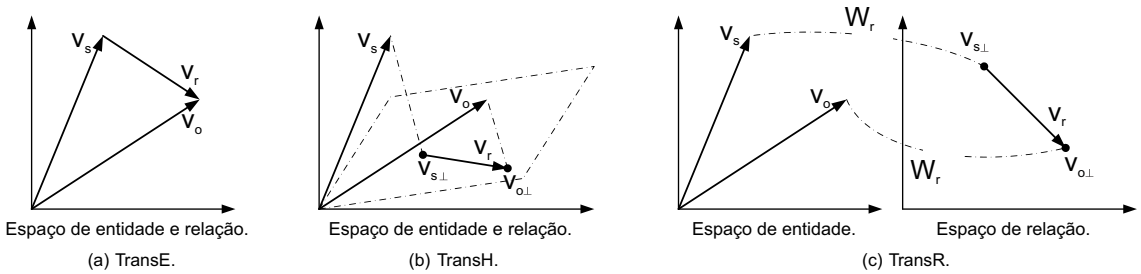

Figura 4.2. Ilustrações de modelos translacionais. Adaptado de [Wang et al. 2017, p.3]. Note que enquanto TransE aplica a ideia de translação de forma direta, TransH [Wang et al. 2014] e TransR [Lin et al. 2015] primeiro projetam as entidades em um hiperplano e espaço específico de relação respectivamente.

TransE gera *embeddings* para as entidades e relações de tal forma que a representação vetorial do objeto de uma tripla seja aproximadamente igual à translação da representação do sujeito. Em outras palavras, para cada tripla $(s,r,o) \in T_E$, $\mathbf{v}_s + \mathbf{v}_r \approx \mathbf{v}_o$ onde $\mathbf{v}_s, \mathbf{v}_r, \mathbf{v}_o \in \mathbb{R}^d$ ($d \in \mathbb{N}$) são as representações vetoriais, respectivamente, de s, r e o. Em particular, o modelo é definido pela função de escore:

$$\phi^{\text{TransE}}_{(s,r,o)} := -\|\mathbf{v}_s + \mathbf{v}_r - \mathbf{v}_o\|_{1\vee 2} \tag{3}$$

onde $\|\cdot\|_{1\vee 2}$ é a norma L_1 ou L_2. Perceba que apesar de TransE ser capaz de abarcar relações 1:1, ele apresenta dificuldades ao lidar com relações do tipo 1:N, N:1 ou M:N. Tome como exemplo o tipo de relação M:N *atua-em*; ela indica que um ator atua em um filme. Se houver duas triplas no grafo (a, atua-em, f1) e (a, atua-em, f2), o modelo poderá aprender representações similares para f1 e f2 ($\mathbf{v}_{f1} \approx \mathbf{v}_{f2}$), mesmo se f1 e f2 forem elementos muito distintos.

Por causa das dificuldades apresentadas, alguns modelos têm sido propostos, dentre os quais, apresenta-se o TransH. Ao invés de utilizar apenas um vetor para cada tipo de relação, TransH [Wang et al. 2014] emprega dois vetores. Em particular, um vetor $\mathbf{v}_r \in \mathbb{R}^d$ de norma de um hiperplano e um vetor $\mathbf{w}_r \in \mathbb{R}^d$ de projeção. A ideia é que para triplas verdadeiras (s,r,o) a projeção de $\mathbf{v}_s$ e $\mathbf{v}_o$ estejam aproximadamente conectadas por $\mathbf{v}_r$. Com essa mudança o método é capaz de modelar de maneira mais adequada tipos de relação que não são funcionais e nem injetivos. Dito isso, a função de escore de TransH é definida como:

$$\phi^{\text{TransH}}_{(s,r,o)} := -\|(\mathbf{v}_s - \mathbf{w}_r^\top \mathbf{v}_s \mathbf{w}_r) + \mathbf{v}_r - (\mathbf{v}_o - \mathbf{w}_r^\top \mathbf{v}_o \mathbf{w}_r)\|_2^2 \tag{4}$$

4.6.3.2. Modelos de correspondência semântica

Diversos modelos de correspondência semântica têm sido propostos nos últimos anos; por exemplo, RESCAL, ANALOGY, SimplE, ConvE e R-GCN.

Tabela 4.2. Parâmetros de modelos translacionais.

Método	Embedding de entidade	Embedding de relação	Função de Escore
TransE	$\mathbf{v}_s, \mathbf{v}_o \in \mathbb{R}^d$	$\mathbf{v}_r \in \mathbb{R}^d$	$-\|\mathbf{v}_s + \mathbf{v}_r - \mathbf{v}_o\|_{1\vee 2}$
TransH	$\mathbf{v}_s, \mathbf{v}_o \in \mathbb{R}^d$	$\mathbf{v}_r, \mathbf{w}_r \in \mathbb{R}^d$	$-\|(\mathbf{v}_s - \mathbf{w}_r^\top \mathbf{v}_s \mathbf{w}_r) + \mathbf{v}_r - (\mathbf{v}_o - \mathbf{w}_r^\top \mathbf{v}_o \mathbf{w}_r)\|_2^2$
TransR	$\mathbf{v}_s, \mathbf{v}_o \in \mathbb{R}^d$	$\mathbf{v}_r \in \mathbb{R}^k, \mathbf{W}_r \in \mathbb{R}^{k\times d}$	$-\|\mathbf{W}_r\mathbf{v}_s + \mathbf{v}_r - \mathbf{W}_r\mathbf{v}_o\|_2^2$
TransA	$\mathbf{v}_s, \mathbf{v}_o \in \mathbb{R}^d$	$\mathbf{v}_r \in \mathbb{R}^d, \mathbf{W}_r \in \mathbb{R}^{d\times d}$	$-\lvert\mathbf{v}_s + \mathbf{v}_r - \mathbf{v}_o\rvert^\top \mathbf{W}_r \lvert\mathbf{v}_s + \mathbf{v}_r - \mathbf{v}_o\rvert$

RESCAL [Nickel et al. 2011] modela a plausibilidade de uma tripla por meio das interações par a par entre as características latentes das entidades nela retratadas. Especificamente, ele modela o escore de uma tripla (s,r,o), isto é, sua plausibilidade de ser verdadeira, como:

$$\phi^{\text{RESCAL}}_{(s,r,o)} := \mathbf{v}_s^\mathsf{T} \mathbf{W}_r \mathbf{v}_o = \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbf{W}_{k_{ij}} \mathbf{v}_{s_i} \mathbf{v}_{o_j} \tag{5}$$

onde $d \in \mathbb{N}$ é a dimensão do espaço de *embedding* de entidades e $\mathbf{v}_s \in \mathbb{R}^d$, $\mathbf{v}_o \in \mathbb{R}^d$ e $\mathbf{W}_r \in \mathbb{R}^{d\times d}$ são respectivamente as representações vetoriais para s, o e r. Perceba que cada escalar $\mathbf{W}_{r_{ij}}$ especifica o quanto as características não observadas i e j, respectivas às representações de s e o, interagem na relação r.

ANALOGY [Liu et al. 2017] lança mão da ideia de que propriedades analógicas entre entidades e relações ajudam na predição de fatos. Por exemplo, suponha a analogia "*homem é para rei aquilo que mulher é para rainha*". A ideia é que o conjunto $\{(homem, r_1, rei), (mulher, r_1, rainha), (homem, r_2, mulher), (rei, r_2, rainha)\}$ forme uma estrutura analógica, onde r_1 e r_2 denotam tipos de relação. Essa estrutura indica que a relação entre *homem* e *rei* ajuda a predizer os relacionamentos não observados entre *mulher* e *rainha*. Nesse sentido, ANALOGY emprega a mesma função de escore de RESCAL:

$$\phi^{\text{ANALOGY}}_{(s,r,o)} := \mathbf{v}_s^\mathsf{T} \mathbf{W}_r \mathbf{v}_o \tag{6}$$

entretanto, para capturar estruturas analógicas, o modelo impõe que as matrizes de relação sejam normais ($\mathbf{W}_r \mathbf{W}_r^\mathsf{T} = \mathbf{W}_r^\mathsf{T} \mathbf{W}_r$) e comutem entre si ($\mathbf{W}_{r_1} \mathbf{W}_{r_2} = \mathbf{W}_{r_2} \mathbf{W}_{r_1}$).

SimplE [Kazemi and Poole 2018] é um modelo baseado na decomposição de posto tensorial. Nele são considerados dois vetores $\mathbf{v}_e^{(+)} \in \mathbb{R}^d$ e $\mathbf{v}_e^{(-)} \in \mathbb{R}^d$ para representar cada entidade $e \in E$. Os vetores $\mathbf{v}_e^{(+)}$ e $\mathbf{v}_e^{(-)}$ são respectivamente as representações de e como sujeito e objeto das relações. De mesmo modo, dois vetores são considerados para cada relação $r \in R_E$, $\mathbf{v}_r^{(+)}$ e $\mathbf{v}_r^{(-)}$, onde $\mathbf{v}_r^{(-)}$ visa representar a relação inversa de r. Dito isto, o escore dado por SimplE é definido como:

$$\phi^{\text{SimplE}}_{(s,r,o)} := \frac{1}{2}\sum_{i=1}^{d} \left(\mathbf{v}_{s_i}^{(+)} \mathbf{v}_{r_i}^{(+)} \mathbf{v}_{o_i}^{(-)} + \mathbf{v}_{o_i}^{(+)} \mathbf{v}_{r_i}^{(-)} \mathbf{v}_{s_i}^{(-)}\right) \tag{7}$$

Além disso, para capturar conhecimento ontológico existente — em especial, relações simétricas, antissimétricas e inversas — são feitas restrições aos vetores de *embedding* das relações; por exemplo, no caso de r ser simétrica, impõe-se que $\mathbf{v}_r^{(+)} \approx \mathbf{v}_r^{(-)}$ e que os vetores de *embedding* sejam não negativos.

Os modelos de *embedding* apresentados acima empregam diretamente as representações vetoriais no cômputo do escore de predição. Uma das desvantagens desse tipo de abordagem é que a única maneira de aumentar a expressividade de uma representação — i.e., a quantidade de características latentes — é adotar um espaço de *embeddings* com maior dimensão. Todavia, isso não escala para grafos de larga escala, uma vez que o número de parâmetros do *embedding* é da ordem do grafo. O aumento da quantidade de características de forma independente do espaço de *embedding* requer o uso de múltiplas camadas de características. Entretanto, esse tipo de abordagem exige cuidados adicionais para que o modelo gerado não superajuste (*overfitting*) aos dados de treinamento e consequentemente não generalize [Nickel et al. 2016].

ConvE [Dettmers et al. 2018] é um modelo convolucional que ataca os desafios apresentados. Em particular, ele emprega camadas de convolução bidimensional e totalmente conectadas na modelagem de interações entre as representações vetoriais de relação e entidades. O modelo utiliza uma camada de convolução para capturar a interação entre as representações da entidade s e relação r e camadas não lineares para aumentar a expressividade das interações entre s, r e o. Em suma, ConvE possui uma arquitetura definida por três camadas: convolução, projeção e produto interno, sendo sua função de escore definida como:

$$\phi^{\text{ConvE}}_{(s,r,o)} := \underbrace{f_2\left(\underbrace{\text{vec}\left(f_1\left(\underbrace{\text{conv}_{\omega}\left(\text{concat}(\bar{\mathbf{v}}_s, \bar{\mathbf{v}}_r)\right)}_{\text{Convolução}}\right)\right)\mathbf{W}}_{\text{Projeção}}\right)\mathbf{v}_o}_{\text{Produto Interno}} \tag{8}$$

onde (i) f_1, f_2 são funções de ativação não linear (e.g., $f_1 = ReLU$[16] e $f_2 =$ sigmoide); (ii) concat concatena duas matrizes uma embaixo da outra; (iii) conv_{ω} é a camada de convolução parametrizada pelos filtros ω; (iv) $vec(\cdot)$ é uma operação de achatamento, a qual ordena um tensor ou matriz na forma de um vetor; (v) $\mathbf{W}$ é uma matriz de parâmetros utilizada na projeção ao espaço de *embedding*; e finalmente (vi) $\bar{\mathbf{v}} \in \mathbb{R}^{m \times n}$ é a representação matricial adotada para $\mathbf{v} \in \mathbb{R}^d$ $(m \times n = d)$[17]. Na Figura 4.3 é mostrada graficamente a arquitetura do modelo ConvE.

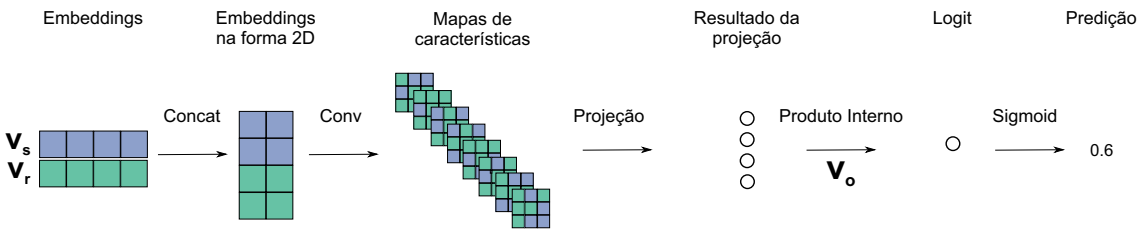

Figura 4.3. Ilustração do modelo ConvE. Fonte: Adaptado de [Dettmers et al. 2018, p.1814].

[16] $ReLU(x) = max(0, x)$

[17] Convoluções bidimensionais esperam que o dado de entrada seja bidimensional.

R-GCNs (*Relational Graph Convolution Neural Network*) [Schlichtkrull et al. 2018] são modelos de aprendizado que estendem redes de convolução de grafos para o cenário multirrelacional. Na complementação de grafos de conhecimento, esse tipo de modelo é utilizado como codificador em um modelo auto-codificador $\phi^{\text{auto-encoder}}_{(o,r,s)} = h_r(g)$. Especificamente, o codificador $g : E \to \mathbb{R}^d$ embute as entidades do grafo no espaço de *embedding* e o decodificador $h_r : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ (parametrizado pelo tipo de relação $r \in R_E$) dá um escore de plausibilidade para uma tripla. Note que o decodificador pode ser qualquer um dos modelos apresentados anteriormente. Além disso, o processo de aprendizado é feito *end-to-end*, isto é, o ajuste do modelo auto-codificador é feito de forma conjunta (codificador mais decodificador).

De forma concreta, R-GCNs são redes neurais multicamada, que quando utilizadas como codificadores, visam aprender representações vetoriais para entidades. Em particular, R-GCNs implementam duas ideias básicas (i) as representações vetoriais de uma entidade devem ser o produto de múltiplas camadas; e (ii) a representação vetorial de uma entidade deve estar relacionada com as representações vetoriais das suas entidades vizinhas. Cada camada oculta $l \in 1, 2, .., L$ da rede é da forma:

$$\mathbf{v}_e^{(l)} = f\left(\underbrace{\mathbf{W}_0^{(l-1)}\mathbf{v}_e^{(l-1)}}_{\text{Própria entidade.}} + \underbrace{\sum_{r\in R_E}\sum_{e'\in N_e^r}\frac{1}{c_{e,r}}\mathbf{W}_r^{(l-1)}\mathbf{v}_{e'}^{(l-1)}}_{\text{Entidades vizinhas.}}\right) \tag{9}$$

onde (i) $\mathbf{v}_e^{(l)} \in \mathbb{R}^{d_l}$ é a representação de $e \in E$ na camada l; (ii) $\mathbf{W}_r^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ é uma matriz de parâmetros para a relação r; (iii) $\mathbf{W}_0^l \in \mathbb{R}^{d_l \times d_{l-1}}$ transforma a representação de e da camada anterior para o espaço da camada atual; (iv) $N_e^r = \{o \mid (e,r,o) \in T_E\}$ é a vizinhança de e[18]; (v) $c_{e,r}$ é uma constante de normalização; (vi) f é uma função de ativação não linear; e (vii) $d^l \in \mathbb{N}$ é o tamanho da dimensão das representações ocultas das entidades na camada l. Note que as representações $\{\mathbf{v}_e^{(L)} \in \mathbb{R}^d \mid e \in E\}$ são o resultado da codificação. Na Figura 4.4 o cômputo das representações é exemplificado.

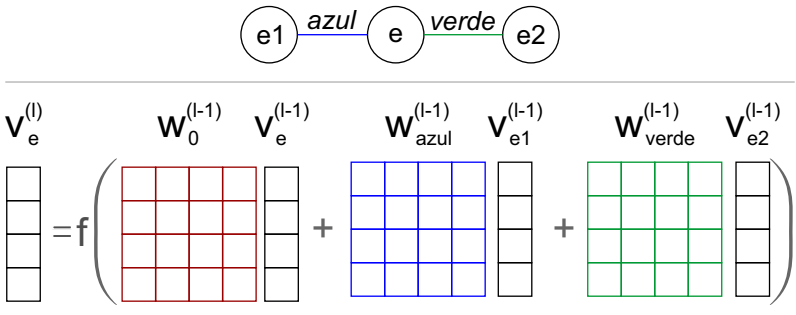

Figura 4.4. Representação de uma camada do modelo R-GCN. A ilustração mostra o cômputo da representação da entidade e na camada l. A entidade e relaciona-se com as entidades $e1$ e $e2$ por meio da relação "azul" e "verde", respectivamente.

[18]Para o cômputo da representação de uma entidade e, além das arestas saintes, também são consideradas as entrantes já que no modelo, R_E contém as inversas de relações (e.g., *pai-de* e sua inversa *pai-de*$^{-1}$).

4.6.3.3. Literais e relacionamentos atributivos

Na Seção 4.2 mostrou-se que grafos de conhecimento usualmente contêm um conjunto de triplas $T_{E \to V}$ atributivas, que ligam entidades a valores literais (de atributo). Esses valores podem ser de diferentes tipos, incluindo dados textuais (e.g., nomes e comentários), numéricos (e.g., altura e ano) ou mesmo imagens. Note que os considerar na modelagem de grafos de conhecimento pode ajudar a produzir melhores representações vetoriais para entidades e, portanto, modelos de *embedding* mais adequados [Gesese and Russa Biswas 2019]. Em particular, eles podem ser úteis no aprendizado de representações para entidades que possuem poucos ou nenhum relacionamento observado no grafo.

A maior parte dos modelos de *embedding* (incluindo os discutidos anteriormente) não leva em conta de forma explícita esse tipo de informação. Isso dificulta, por exemplo, a realização da tarefa de predição de atributos. Nesse contexto, uma abordagem imediata seria modelar literais da mesma forma que entidades. Entretanto, apesar de sua simplicidade, essa abordagem sofre como alguns problemas, em especial, ela pode aumentar drasticamente a quantidade de parâmetros a serem aprendidos, assim como apenas consegue lidar com atributos categóricos [Wang et al. 2017]. Em face ao exposto, modelos de *embedding* capazes de lidar com valores literais têm sido propostos [Gesese and Russa Biswas 2019]; dentre eles, destaca-se MKBE.

Multimodal Knowledge Base Embeddings (MKBE) [Pezeshkpour et al. 2018] é um modelo de complementação de grafos de conhecimento que emprega diferentes codificadores neurais no aprendizado de *embedding* para tipos diversos de dados (textual, numérico e imagens). Assim como, o modelo R-GCN, apresentado anteriormente, MKBE adota uma abordagem de auto-codificador que é descrita a seguir.

O processo de codificação emprega para cada tipo de elemento — imagem, número, texto, entidade, relação — uma rede neural distinta. A representação vetorial de uma imagem é obtida por meio de uma rede neural de convolução; especificamente, uma VGGNet pré-treinada na base de dados ImageNet[19]. No que lhe diz respeito, o vetor de *embedding* de um literal numérico é obtido por meio de uma rede neural *feed-forward*. Por sua vez, emprega-se dois tipos de redes neurais profundas na codificação de literais textuais. Em particular, a representação de textos de menor comprimento (e.g., nomes) é obtida por uma arquitetura recorrente bidirecional GRU (*Gated Recurrent Unit*) a nível de caractere, enquanto *embeddings* de textos de maior comprimento (e.g., descrições textuais), por meio de uma rede de convolução. Por fim, emprega-se duas camadas densas de rede neural no aprendizado de *embeddings* de entidades e tipos de relação.

O decodificador de MKBE realiza o processo de inferência. Notadamente, ele lança mão dos *embeddings* produzidos pelo codificador para produzir o escore referente a uma tripla. O decodificador pode ser, por exemplo, o modelo ConvE, apresentado anteriormente. Perceba que o processo de aprendizado é realizado *end-to-end*, isto é, os parâmetros do codificador e decodificador são ajustados de forma conjunta.

Na Figura 4.5 é apresentada a arquitetura de MKBE. Os vetores $\mathbf{s} \in \{0,1\}^{|E|}$ e

[19] http://www.image-net.org/

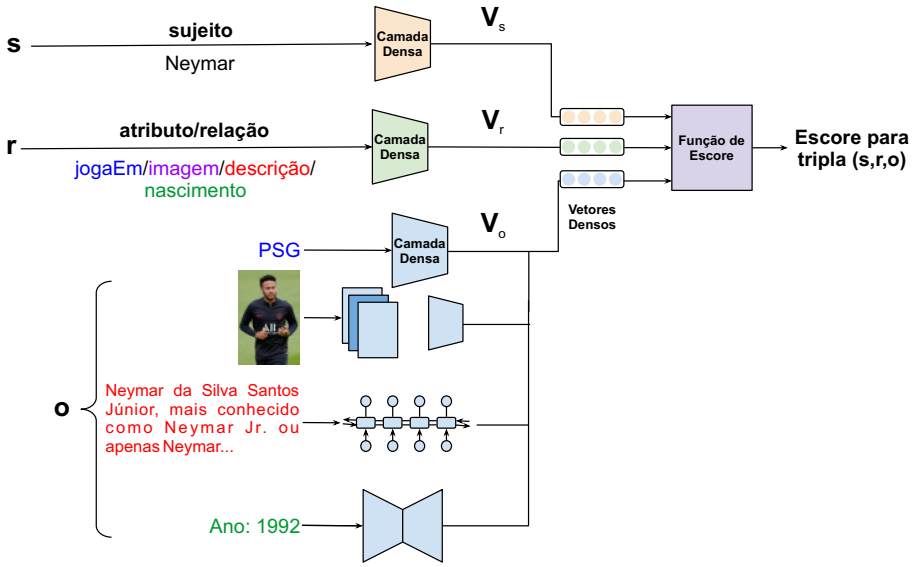

Figura 4.5. Ilustração da arquitetura do modelo MKBE. Informação referente ao jogador de futebol Neymar é utilizada como exemplo. Fonte: Adaptado de [Pezeshkpour et al. 2018, p.3211].

$\mathbf{r} \in \{0,1\}^{|R_E \cup A|}$ são a codificação *one-hot* da entidade s e da relação r respectivamente.[20] Por sua vez, o significado do objeto $\mathbf{o}$ depende daquilo a que ele se refere: entidade (*one-hot encoding*), imagem (tensor real tridimensional), texto (*embeddings* textuais) e literal numérico (números reais padronizados). Perceba ainda os *embeddings* de entidades e atributos $\mathbf{v}_s, \mathbf{v}_o \in \mathbb{R}^d$. Os *embedding* de relação $\mathbf{v}_r$ depende da função de escore empregada; por exemplo, se for adotado ConvE, então $\mathbf{v}_r \in \mathbb{R}^d$.

4.6.3.4. Emprego de ontologias

Conforme exposto na Seção 4.2, grafos de conhecimento geralmente incluem um componente terminológico; em particular, na forma de informação ontológica. Entretanto, ainda é incipiente o uso dessa classe de informação no desenvolvimento de modelos de complementação baseados em *embeddings*. Nesse sentido, a fim de que se modele de forma mais adequada grafos de conhecimento, recentemente alguns trabalhos recorrem a ontologias. Dentre esses, destaca-se aqueles que as empregam no desenvolvimento de novos modelos e na restrição do intervalo de valores das representações vetoriais a serem aprendidas.

JOIE [Hao et al. 2019] é um modelo de complementação que codifica de forma conjunta tanto o componente assertivo quanto terminológico do grafo de conhecimento. Em particular, JOIE é formado por dois componentes. O primeiro deles, chamado de *cross-view association model*, tem como intuito associar o *embedding* de uma entidade ao seu respectivo *embedding* de conceito; por exemplo, a representação vetorial da entidade Albert Einstein à representação da classe pessoa. Para isso, duas técnicas são empregadas, *cross-view grouping* e *cross-view transformation*. Em resumo, a primeira delas visa agrupar entidades referentes ao mesmo conceito, enquanto a segunda, mapear o es-

[20] O vetor *one-hot* $\mathbf{v}$ associado à n-ésima entidade (relação/atributo) é tal que $\mathbf{v}_n = 1$ e $\mathbf{v}_{j \neq n} = 0$.

paço representacional das entidades ao de conceitos. O segundo componente, chamado de *intra-view embedding model*, visa caracterizar as triplas pertencentes aos componentes terminológico e assertivo em dois espaços de *embeddings* distintos. Para isso, lança-se mão de uma função de escore para as triplas T_E e outra para T_C. Essas funções de escore podem ser os modelos apresentados anteriormente, por exemplo, TransE.

A fim de melhor modelar informação ontológica, e consequentemente melhorar o desempenho de modelos, alguns trabalhos propõem abordagens que limitam o espaço de *embedding* associado ao grafo de conhecimento. Nesse contexto, [Ding et al. 2018] propõem alterações no modelo Complex [Trouillon et al. 2017]. Em particular, eles impõem que os *embeddings* das entidades sejam não negativos e seus valores contidos em $[0,1]^d$. Além disso, eles restringem os valores dos *embeddings* das relações a fim de melhor capturar subsunções aproximadas (e.g., a relação nascido-em usualmente implica na relação nacionalidade). De modo similar, [Fatemi et al. 2019b] adotam uma estratégia para garantir que o modelo SimplE seja capaz de capturar subsunções (e.g., $(X, r_1, Y) \rightarrow (X, r_2, Y)$) entre tipos de relação. Em particular, eles impõem que os *embeddings* de entidades sejam não negativos e que o *embedding* de um tipo de relação seja sempre menor ou igual aos *embeddings* das relações que ele subsume.

4.6.3.5. Avaliação e treinamento de modelos

Em sua maioria, o desempenho preditivo de modelos de *embedding* é avaliado por meio dos protocolos de classificação de triplas (*triple classfication*) e ranqueamento de entidades (*entity ranking*), sendo o último mais frequentemente utilizado [Wang et al. 2019c]. Em ambos os protocolos, segundo a prática usual em aprendizado de máquina, a coleção de triplas T do grafo de conhecimento é dividida em três conjuntos disjuntos, nomeadamente, treinamento $T^{(+)}_{train} \subset T$, validação $T^{(+)}_{val} \subset T$ e teste $T^{(+)}_{test} \subset T$. Além disso, como em tarefas de complementação, assume-se que o grafo de conhecimento não abarque proposições falsas, os conjuntos acima contêm apenas triplas tidas como verdadeiras.

O objetivo do protocolo de *classificação de triplas* é testar a habilidade de um modelo ϕ em discriminar triplas verdadeiras das falsas [Wang et al. 2019c]. Esse protocolo está associado, por exemplo, com a tarefa de inferência de triplas. Nesse cenário, a fim de avaliar o modelo, triplas pseudonegativas são geradas. Essa geração pode ser realizada ao substituir de maneira aleatória o sujeito ou objeto de cada tripla de teste por outro elemento do grafo que aparece como sujeito ou objeto respectivamente. Além disso, a tripla (s,r,o) é classificada como verdadeira se o escore $\phi_{(s,r,o)}$ exceder um limiar λ_r dependente do tipo de relação r, o qual é ajustado durante o processo de treinamento e validação do modelo. O desempenho do modelo é medido a partir dos rótulos das triplas de teste por meio de métricas de classificação, incluindo acurácia, precisão e revocação.

No que lhe diz respeito, o objetivo do protocolo de *ranqueamento de entidades* é avaliar o desempenho de um modelo ϕ na inferência de determinadas consultas [Wang et al. 2019c]. Esse protocolo está associado, por exemplo, com a tarefa de predição de ligações. Em particular, para cada tripla de teste $t = (s,r,o)$ duas consultas são produzidas $q_s = (?,r,o)$ e $q_o = (s,r,?)$. Substitui-se "?"em q_s e q_o por cada elemento de interesse x (e.g, entidade) do grafo e ranqueia-se em ordem decrescente, com base nos escores $\phi_{(s,r,x)}$

e $\phi_{(x,r,o)}$, as triplas (s,r,x) e (x,r,o), respectivamente. Com base na posição de cada tripla de teste (s,r,o) nesse *ranking*, o desempenho do modelo é avaliado. Essa avaliação usualmente emprega métricas de recuperação de informação (*information retrieval*), por exemplo, *hits@k* e *mean recipocral ranking*. Além disso, para evitar resultados enganosos, na avaliação usualmente são desconsideradas as triplas (s,r,x) e (x,r,o) presentes no conjunto de treinamento e validação.

Tabela 4.3. Funções de custo utilizadas no treinamento de modelos de *embedding*.

	Erro quadrático	*Hinge*	*Logística*
Pontuais	$\frac{1}{2}\sum_{t\in T_{train}}(\phi_t - y_t)^2$	$\sum_{t\in T_{train}}[\lambda + (-1)^{y_t}\phi_t]_+$	$\sum_{t\in T_{train}}\log(1+\exp((-1)^{y_t}\phi_t))$
	Hinge		*Logística*
Emparelhadas	$\sum_{t\in T^{(+)}_{train}}\sum_{t'\in T^{(-)}_{train}}[\lambda + \phi_{t'} - \phi_t]_+$		$\sum_{t\in T^{(+)}_{train}}\sum_{t'\in T^{(-)}_{train}}\log(1+\exp(\phi_{t'} - \phi_t))$

Legenda: $[x]_+ = \max(x,0)$, $\lambda \in \mathbb{R}_{\geq 0}$ e $T_{train} = T^{(-)}_{train} \cup T^{(+)}_{train}$.

O aprendizado de modelos de *embedding* envolve a escolha de uma função de custo, a qual geralmente é minimizada por meio do método de Gradiente Descendente Estocástico (ou uma de suas variações). Essa funções consideram os escores dados por um modelo, o valor verdade das triplas, além de restrições (e.g., regularização) associadas aos parâmetros do modelo. Como o conjunto de restrições depende de cada modelo, elas não são apresentadas.

Funções de custo podem ser divididas em pontuais (*pointwise*) ou emparelhadas (*pairwise*) [Mohamed et al. 2019]. Funções de custo pontuais abordam uma tripla por vez. Por exemplo, a função erro quadrático, disposta na Tabela 4.3, mede a diferença quadrada entre o escore ϕ_t e o rótulo $y_t \in \{0,1\}$ de uma tripla de treinamento. Note que y_t é igual a um se a tripla for positiva (pertencer ao grafo de conhecimento) e zero se ela for negativa ou pseudo-negativa. Por sua vez, funções de custo emparelhadas tomam um par contendo uma tripla positiva e uma (pseudo)-negativa. Por exemplo, a função *hinge*, disposta na Tabela 4.3, considera de forma conjunta os escores ϕ_t e $\phi_{t'}$ de uma tripla positiva $t \in T^{(+)}_{train}$ e (pseudo)negativa $t' \in T^{(-)}_{train}$, respectivamente. Por fim, é válido notar que usualmente não são considerados todos os pares de triplas $(t,t') \in T^{(+)}_{train} \times T^{(-)}_{train}$ no cômputo da função de custo, mas sim uma amostra de triplas (pseudo) negativas para cada tripla positiva.

4.7. Considerações Finais

O interesse pela construção, inferência e aplicações de grafos de conhecimento têm florescido nos últimos anos. Esse interesse se deve a diversos fatores, dentre eles, a naturalidade com que conhecimento e informação são dispostos na forma de rede, a abundância de dados heterogêneos, multimodais e multirrelacionais, assim como o surgimento de técnicas que propiciam a construção de bases e grafos de conhecimento de forma cada vez automatizada. Diante disso, neste capítulo foi apresentado de forma introdutória o emprego de técnicas de aprendizado de máquina em tarefas relacionadas a grafos de conhecimento (ver Tabela 4.4). Especialmente, foi apresentado um conjunto de modelos destinados à ta-

refa de complementação, baseados no aprendizado de representações vetoriais para grafos de conhecimento. Dito isto, cita-se oportunidades e desafios de pesquisa em aberto.

Tabela 4.4. Sistemas e modelos de aprendizado de máquina apresentados neste capítulo.

Construção Automatizada	*Sistemas:* • DeepDive [Zhang et al. 2016]. • Fonduer [Wu et al. 2018].
Complementação (Inferência de Fatos)	*Modelos gráficos probabilísticos:* • Redes Lógicas de Markov [Richardson and Domingos 2006] *Modelos de características de grafo:* • *Path Ranking Algorithm* [Lao et al. 2011]. *Modelos de características latentes:* • Distância translacional: TransE [Bordes et al. 2013] e TransH [Wang et al. 2014]. • Correspondência semântica: ANALOGY [Liu et al. 2017], ConvE [Dettmers et al. 2018], JOIE [Hao et al. 2019], MKBE [Pezeshkpour et al. 2018], RESCAL [Nickel et al. 2011], R-GCN [Schlichtkrull et al. 2018] e SimplE [Kazemi and Poole 2018].

Pode-se elencar algumas perspectivas de pesquisa relacionadas ao desenvolvimento de modelos baseados em *embedding* para grafos de conhecimento. Primeiramente, é preciso que se avalie em que nível esse tipo de modelo é capaz de vencer a falta de estruturas simbólicas mais formais, como regras e restrições [Trouillon et al. 2019]; mais ainda, como eles se comparam a métodos que fazem uso dessas estruturas, por exemplo, *Probabilistic Soft Logic* [Bach et al. 2017]. Nesse sentido, há indícios de que o desempenho preditivo desses modelos sofra com problemas de generalização quando o grafo modelado é demasiadamente esparso e/ou ruidoso [Pujara et al. 2017]. Isso indica que o emprego conjunto de diferentes abordagens de inferência é potencialmente mais adequado. Relacionado a isso, como embutir ou considerar conhecimento formal no processo de aprendizado de modelos de *embedding* é de interesse; isso pois alguns modelos de características latentes são incapazes de induzir certas regras lógicas (e.g., subsunções) a partir das asserções presentes no grafo [Gutiérrez-Basulto and Schockaert 2018] e serem, portanto, logicamente consistentes.

Também é relevante o desenvolvimento de modelos latentes que considerem um espectro maior de informação, por exemplo, dinâmica temporal, estruturas diversas do grafo e relações de maior aridade. Apesar de usualmente serem consideradas de forma atemporal, as asserções em grafos de conhecimento costumam ser sensíveis ao tempo. Nesse contexto, considerar a dinâmica evolutiva de entidades e relações pode propiciar tanto o desenvolvimento de melhores modelos quanto novas tarefas e aplicações, por exemplo, a predição temporal de ligações [Trivedi et al. 2017]. Além disso, é de interesse produzir modelos que infiram estruturas mais complexas do grafo de conhecimento, por exemplo, caminhos entre entidades, os quais podem ser vistos como relacionamentos de mais alta ordem. Por sua vez, é relevante que sejam desenvolvidas abordagens latentes capazes de lidar com relações de maior aridade uma vez que parte importante das relações

em bases de conhecimento não são binárias [Fatemi et al. 2019a].

Concernente ao aprendizado, é importante o desenvolvimento de metodologias que gerem *embeddings* para novas entidades sem que se precise aprender novamente as representações vetoriais das entidades já presentes no grafo [Wang et al. 2019b]. Isso é importante uma vez que esse retreino é potencialmente impraticável em aplicações reais onde novas entidades surgem diariamente.

Por fim, são relevantes o desenvolvimento de metodologias para construção de bases de conhecimento a partir de informação multimodal. Há uma grande quantidade de informação em imagem, sensorial e em áudio que raramente é integrada a dados textuais em um repositório de conhecimento comum no qual consultas possam ser realizadas [Dong and Rekatsinas 2018]. Nesse contexto, os métodos de aprendizado profundo possivelmente provejam as ferramentas necessárias para integração multimodal de dados.

Agradecimentos

Os autores agradecem ao CNPq, à FAPERJ e ao CENPES/Petrobras pelo financiamento.

Referências

[Bach et al. 2017] Bach, S. H. et al. (2017). Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research (JMLR)*, 18:1–67.

[Baker et al. 1998] Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The berkeley FrameNet project. In *Proceedings of the 36th annual meeting on Association for Computational Linguistics* -. Association for Computational Linguistics.

[Bengio et al. 2013] Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(8):1798–1828.

[Bonatti et al. 2019] Bonatti, P. A. et al. (2019). Knowledge graphs: New directions for knowledge representation on the semantic web (dagstuhl seminar 18371).

[Bordes et al. 2013] Bordes, A. et al. (2013). Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 2787–2795, USA. Curran Associates Inc.

[Bouraoui et al. 2018] Bouraoui, Z., Jameel, S., and Schockaert, S. (2018). Relation induction in word embeddings revisited. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1627–1637, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

[Brodie and Mylopoulos 1986] Brodie, M. L. and Mylopoulos, J. (1986). Knowledge bases vs databases. In *On Knowledge Base Management Systems*, pages 83–86. Springer.

[Dettmers et al. 2018] Dettmers, T. et al. (2018). Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

[Ding et al. 2018] Ding, B. et al. (2018). Improving knowledge graph embedding using simple constraints. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 110–121.

[Dong et al. 2014] Dong, X. L. et al. (2014). Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *SIGKDD - International Conference on Knowledge Discovery and Data Mining*, pages 601–610.

[Dong and Rekatsinas 2018] Dong, X. L. and Rekatsinas, T. (2018). Data integration and machine learning: A natural synergy. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 1645–1650, New York, NY, USA. ACM.

[Ehrlinger and Wöß 2016] Ehrlinger, L. and Wöß, W. (2016). Towards a definition of knowledge graphs. In *SEMANTiCS (Posters, Demos, SuCCESS)*.

[Fatemi et al. 2019a] Fatemi, B. et al. (2019a). Knowledge hypergraphs: Extending knowledge graphs beyond binary relations. *CoRR*, abs/1906.00137.

[Fatemi et al. 2019b] Fatemi, B., Ravanbakhsh, S., and Poole, D. (2019b). Improved knowledge graph embedding using background taxonomic information. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):3526–3533.

[Gesese and Russa Biswas 2019] Gesese, G. A. and Russa Biswas, H. S. (2019). A comprehensive survey of knowledge graph embeddings with literals: Techniques and applications. In *Workshop on Deep Learning for Knowledge Graphs*.

[Gutiérrez-Basulto and Schockaert 2018] Gutiérrez-Basulto, V. and Schockaert, S. (2018). From knowledge graph embedding to ontology embedding? an analysis of the compatibility between vector space representations and rules. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Sixteenth International Conference, KR*, pages 379–388.

[Hamilton et al. 2017] Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Representation learning on graphs: Methods and applications. *IEEE Data Eng. Bull.*, 40:52–74.

[Hao et al. 2019] Hao, J. et al. (2019). Universal representation learning of knowledge bases by jointly embedding instances and ontological concepts. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD 19*. ACM Press.

[Jia et al. 2016] Jia, Y. et al. (2016). Locally adaptive translation for knowledge graph embedding. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 992–998. AAAI Press.

[Kazemi and Poole 2018] Kazemi, S. M. and Poole, D. (2018). Simple embedding for link prediction in knowledge graphs. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 4289–4300, USA. Curran Associates Inc.

[Kejriwal 2019] Kejriwal, M. (2019). Advanced topic: Knowledge graph completion. In *Domain-Specific Knowledge Graph Construction*, pages 59–74. Springer International Publishing.

[Lao et al. 2011] Lao, N., Mitchell, T., and Cohen, W. W. (2011). Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Processing*, EMNLP '11, pages 529–539, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Lehmann 1992] Lehmann, F. (1992). Semantic networks. *Computers & Mathematics with Applications*, 23(2-5):1–50.

[Lehmann et al. 2015] Lehmann, J. et al. (2015). Dbpedia – a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

[Lenat 1995] Lenat, D. B. (1995). CYC: a large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

[Lin et al. 2015] Lin, Y. et al. (2015). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2181–2187. AAAI Press.

[Liu et al. 2017] Liu, H., Wu, Y., and Yang, Y. (2017). Analogical inference for multi-relational embeddings. In Precup, D. and Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 2168–2178, Sydney, Australia.

[Mohamed et al. 2019] Mohamed, S. et al. (2019). A comprehensive survey of knowledge graph embeddings with literals: Techniques and applications. In *Workshop on Deep Learning for Knowledge Graphs*.

[Nickel et al. 2016] Nickel, M. et al. (2016). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

[Nickel et al. 2011] Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, USA. Omnipress.

[Niu et al. 2011] Niu, F. et al. (2011). Tuffy: Scaling up statistical inference in markov logic networks using an RDBMS. *Proc. VLDB Endow.*, 4(6):373–384.

[Noy et al. 2019] Noy, N. et al. (2019). Industry-scale knowledge graphs: Lessons and challenges. *Queue*, 17(2):20:48–20:75.

[Noy and Mcguinness 2001] Noy, N. F. and Mcguinness, D. L. (2001). Ontology development 101: A guide to creating your first ontology. Technical report, Standford.

[Pan et al. 2018] Pan, J. Z. et al. (2018). Content based fake news detection using knowledge graphs. In *Lecture Notes in Computer Science*, pages 669–683. Springer International Publishing.

[Paulheim 2017] Paulheim, H. (2017). Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web*, 8(3):489–508.

[Pezeshkpour et al. 2018] Pezeshkpour, P., Chen, L., and Singh, S. (2018). Embedding multimodal relational data for knowledge base completion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

[Pujara et al. 2017] Pujara, J., Augustine, E., and Getoor, L. (2017). Sparsity and noise: Where knowledge graph embeddings fall short. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1751–1756, Copenhagen, Denmark. Association for Computational Linguistics.

[Raedt et al. 2016] Raedt, L. D., Kersting, K., and Natarajan, S. (2016). *Statistical Relational Artificial Intelligence: Logic, Probability, and Computation*. Morgan & Claypool Publishers.

[Ratner et al. 2018] Ratner, A., Ré, C., and Bailis, P. (2018). Research for practice: Knowledge base construction in the machine-learning era. *Commun. ACM*, 61(11):95–97.

[Ratner et al. 2016] Ratner, A. J. et al. (2016). Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, pages 3567–3575.

[Rebele et al. 2016] Rebele, T. et al. (2016). YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames. In *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II*, pages 177–185.

[Richardson and Domingos 2006] Richardson, M. and Domingos, P. (2006). Markov logic networks. *Machine learning*, 62(1-2):107–136.

[Sang et al. 2019] Sang, S. et al. (2019). GrEDeL: A knowledge graph embedding based method for drug discovery from biomedical literatures. *IEEE Access*, 7:8404–8415.

[Schlichtkrull et al. 2018] Schlichtkrull, M. et al. (2018). Modeling relational data with graph convolutional networks. In *The Semantic Web*, pages 593–607, Cham. Springer International Publishing.

[Trivedi et al. 2017] Trivedi, R. et al. (2017). Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3462–3471, International Convention Centre, Sydney, Australia. PMLR.

[Trouillon et al. 2017] Trouillon, T. et al. (2017). Knowledge graph completion via complex tensor factorization. *Journal of Machine Learning Research*, 18(130):1–38.

[Trouillon et al. 2019] Trouillon, T. et al. (2019). On inductive abilities of latent factor models for relational learning. *J. Artif. Int. Res.*, 64(1):21–53.

[van Harmelen et al. 2008] van Harmelen, F., Lifschitz, V., and Porter, B. W., editors (2008). *Handbook of Knowledge Representation*, volume 3 of *Foundations of Artificial Intelligence*. Elsevier.

[Vrandečić and Krötzsch 2014] Vrandečić, D. and Krötzsch, M. (2014). Wikidata. *Communications of the ACM*, 57(10):78–85.

[Vu et al. 2018] Vu, M. H. et al. (2018). Statistical relational learning with unconventional string models. *Frontiers in Robotics and AI*, 5.

[Wang et al. 2019a] Wang, H. et al. (2019a). Exploring high-order user preference on the knowledge graph for recommender systems. *ACM Transactions on Information Systems*, 37(3):1–26.

[Wang et al. 2019b] Wang, P. et al. (2019b). Logic attention based neighborhood aggregation for inductive knowledge graph embedding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:7152–7159.

[Wang et al. 2017] Wang, Q. et al. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

[Wang et al. 2019c] Wang, Y. et al. (2019c). On evaluating embedding models for knowledge base completion. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 104–112, Florence, Italy. Association for Computational Linguistics.

[Wang et al. 2014] Wang, Z. et al. (2014). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press.

[Wu et al. 2018] Wu, S. et al. (2018). Fonduer: Knowledge base construction from richly formatted data. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 1301–1316, New York, NY, USA. ACM.

[Yan et al. 2016] Yan, J. et al. (2016). A retrospective of knowledge graphs. *Frontiers of Computer Science*, 12(1):55–74.

[Yu et al. 2018] Yu, Y. et al. (2018). PreMedKB: an integrated precision medicine knowledgebase for interpreting relationships between diseases, genes, variants and drugs. *Nucleic Acids Research*, 47(D1):D1090–D1101.

[Zhang et al. 2016] Zhang, C. et al. (2016). Extracting databases from dark data with DeepDive. In *Proceedings of the 2016 International Conference on Management of Data - SIGMOD16*. ACM Press.

REALIZATION

EXECUTION

SUPPORT

SILVER SPONSOR

ACADEMIC SUPPORT