UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO
DEPARTAMENTO DE ESTATÍSTICA E INFORMÁTICA
PROGRAMA DE PÓS–GRADUAÇÃO EM INFORMÁTICA APLICADA

GUILHERME HENRIQUE SANTOS SOUSA

MÉTODOS BASEADOS EM DEEP LEARNING PARA MAPEAMENTO DE ONTOLOGIAS: ALINHAMENTO DE PROPRIEDADES E CLASSES

RECIFE – PE

2022

GUILHERME HENRIQUE SANTOS SOUSA

MÉTODOS BASEADOS EM DEEP LEARNING PARA MAPEAMENTO DE ONTOLOGIAS: ALINHAMENTO DE PROPRIEDADES E CLASSES

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática Aplicada do Departamento de Estatística e Informática - DEINFO - Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

ORIENTADOR: Rinaldo Lima

CO–ORIENTADOR: Cássia Trojahn dos Santos

RECIFE – PE

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

S725m Sousa, Guilherme Henrique
MÉTODOS BASEADOS EM DEEP LEARNING PARA MAPEAMENTO DE ONTOLOGIAS: ALINHAMENTO DE PROPRIEDADES E CLASSES / Guilherme Henrique Sousa. - 2022.
67 f. : il.

Orientador: Rinaldo Lima.
Coorientadora: Cassia Trojahn dos Santos.
Inclui referências.

Dissertação (Mestrado) - Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, 2023.

1. Property Mapping. 2. Ontology Mapping. 3. Semantic Similarity. 4. Neural Networks. 5. Deep Learning. I. Lima, Rinaldo, orient. II. Santos, Cassia Trojahn dos, coorient. III. Título

CDD 004

GUILHERME HENRIQUE SANTOS SOUSA

MÉTODOS BASEADOS EM DEEP LEARNING PARA MAPEAMENTO DE ONTOLOGIAS: ALINHAMENTO DE PROPRIEDADES E CLASSES

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Informática Aplicada do Departamento de Estatística e Informática - DEINFO - Universidade Federal Rural de Pernambuco, como parte dos requisitos necessários para obtenção do grau de Mestre.

Aprovada em:

BANCA EXAMINADORA

Rinaldo Lima (Orientador)
Universidade Federal Rural de Pernambuco

Filipe Rolin Cordeiro
Universidade Federal Rural de Pernambuco

Renata Vieira
Universidade de Évora

Resumo

Ontologias são estruturas que permitem relacionar conceitos semânticos. Essas informações são frequentemente utilizadas para tarefas textuais que necessitam de um entendimento semântico do texto. Porém, devido a diferentes autores criarem ontologias para diferentes propósitos, podem ocorrer diferenças entre os conceitos dessas ontologias, dificultando a interoperabilidade entre elas. Visando mitigar tal problema, a tarefa de Ontology Matching é aplicada visando alinhar conceitos semelhantes em diferentes ontologias. Uma das tarefas mais importantes em Ontology Matching é o alinhamento de propriedades. Porém, a maioria dos sistemas utilizam um conjunto de regras criadas manualmente com características para calcular a similaridade entre diferentes propriedades. Outro problema reside no fato que devido a dificuldade em definir o conceito de similaridade entre entidades, muitos sistemas precisam comparar todas as entidades na ontologia fonte contra todas as entidades na ontologia alvo. O grande espaço de busca faz com que sistemas sejam incapazes de gerar um alinhamento entre ontologias muito grandes. Neste trabalho, propomos uma arquitetura baseada em redes neurais (Deep Learning) para o alinhamento de propriedades que utiliza embeddings no cálculo de similaridade e um conjunto de regras que melhoram os resultados quando comparado como outros sistemas. Outra contribuição é a utilização de classes top-level para criar uma métrica auxiliar de similaridade.Nessa proposta, avaliamos o uso de classes pré-estabelecidas na redução do espaço de busca ao criar clusters entre as entidades envolvidas. Além disso, apresentamos um experimento onde se aplica a proposta supracitada na tarefa de ontology matching simples integrada a uma arquitetura deep learning. Os resultados experimentais mostram que a solução proposta tem um desempenho competitivo e até supera sistemas do estado da arte quando comparados usando-se o mesmo protocolo experimental.

Palavras-chave: Mapeamento de Propriedades. Mapeamento de Ontologias. Similaridade. Redes Neurais. Aprendizado Profundo

Abstract

Ontologies are structures that allow semantic concepts to be related. They are often used for textual tasks that require a semantic understanding of the text. However, because different authors create ontologies for different purposes, differences may occur between the concepts of these ontologies, making interoperability between them difficult. To mitigate this problem, Ontology Matching is applied to align similar concepts in different ontologies. One of the most important tasks in Ontology Matching is the alignment of properties. However, most systems use a set of manually created rules with features to calculate the similarity between different properties. Another problem lies in the fact that due to the difficulty in defining the concept of similarity between entities, many systems need to compare all entities in the source ontology against all entities in the target ontology. The large search space makes systems unable to generate an alignment between very large ontologies. In this paper, we propose an architecture based on neural networks (Deep Learning) for property alignment that uses embeddings in the similarity calculation and a set of rules that improve the results when compared to other systems. Another contribution is the use of top-level classes to create an auxiliary similarity metric.In this proposal, we evaluate the use of pre-established classes in reducing the search space by creating clusters among the entities involved. Furthermore, we present an experiment where the aforementioned proposal is applied to the task of simple ontology matching integrated with a deep learning architecture. Experimental results show that the proposed solution performs competitively and even outperforms state-of-the-art systems when compared using the same experimental protocol.

Keywords: Property Mapping. Ontology Mapping. Semantic Similarity. Neural Networks. Deep Learning

Lista de Figuras

Lista de tabelas

Lista de Siglas

RNN	*Recurrent Neural Networks*
LSTM	*Long Short-Term Memory*
GNN	*Gated Neural Network*
NLP	*Natural Language Processing*
NER	*Named Entity Recognition*

Sumário

1 Introdução

O avanço da computação possibilitou a criação da internet e com ela uma capacidade inimaginável de comunicação e compartilhamento de conteúdo. A internet possibilitou a transmissão de diversos tipos de conteúdo como vídeos, imagens e texto. Nela, também, empresas encontraram formas de migrar seus negócios para o mundo digital e alcançar uma grande quantidade de pessoas. Porém, com essa grande escala, o processamento manual de dados se tornou algo inimaginável fazendo surgir a necessidade de tornar a internet mais legível para as máquinas.

Iniciativas como a Semantic Web (HENDLER; BERNERS-LEE, 2010) trazem uma série de padrões e recursos para definir o armazenamento e transmissão de dados com anotações semânticas, permitindo que agentes inteligentes possam interpretar e processar esses dados com facilidade. Recursos como RDF (DECKER et al., 2000), OWL (HITZLER et al., 2009) e Schema.org (GUHA et al., 2016) fornecem uma linguagem padronizada para a descrição de elementos e tipos semânticos com o objetivo de adicionar meta-dados ao conteúdo presente na web. As ontologias são um desses recursos e são frequentemente utilizadas para descrever uma hierarquia de entidades contidos em uma estrutura em grafo. Essas entidades são classes que representam conceitos sobre objetos, propriedades que descrevem a relação entre classes e instâncias que são exemplos concretos de classes.

Apesar da existência desses recursos, grande parte das anotações semânticas ainda são produzidas por humanos e mesmo com a padronização podem surgir problemas de interoperabilidade dos dados. Pessoas diferentes podem descrever o mesmo conceito sobre diferentes pontos de vista e para fins específicos. Diversos fatores dificultam a interpretação desse conteúdo como diferentes idiomas ou diferentes granularidades e especificidades na descrição de um conceito. Como esses recursos normalmente são descritos em linguagem natural, problemas como ambiguidade, abreviações e erros ortográficos frequentemente ocorre e dificultam a construção de sistemas que utilizam esses dados para tomar decisões. Para lidar com esses problemas, sistemas que utilizam dados semânticos devem possuir a capacidade de lidar com contexto nos elementos textuais e nas relações entre elementos.

Como solução para esses problemas, sistemas automáticos de alinhamento são desenvolvidos para gerar uma relação de equivalência entre classes e propriedades em diferentes formatos. Uma grande quantidade de sistemas existe para a produção de

alinhamentos entre ontologias, porém grande parte desses sistemas ainda utilizam regras codificadas manualmente para encontrar padrões de alinhamentos e que muitas vezes não são gerais o suficiente para alinhar ontologias de diferentes domínios além de dificultar a generalidade do sistema e exigir grande esforço manual para sua construção. Outro problema é a utilização desses sistemas em larga escala sendo que grandes ontologias contém grande quantidade de elementos na ordem de milhões de entidades e a comparação de todos os elementos entre si gera uma quantidade impraticável de possibilidades que devem ser consideradas. A maioria das técnicas utilizadas para reduzir esse espaço se baseiam em padrões estatísticos e hipóteses de similaridade baseadas em padrões léxicos e sintáticos falhando em agrupar conceitos de forma semântica para essa redução.

O avanço das técnicas em processamento de linguagem natural e aprendizado de máquina permitem que essas ferramentas possuam maior capacidade de compreensão sobre textos em diversos formatos. Assim, sistemas de alinhamento podem atingir alta performance na geração de alinhamentos entre diversos domínios ao serem enriquecidos ou baseados em aprendizado de máquina. Uma das técnicas utilizadas frequentemente na tarefa de alinhamento é a utilização de redes neurais para a criação de modelos que geram representações de conceitos simbólicos de forma que possam ser comparados utilizando vários critérios a fim de reconhecer um maior número de alinhamentos. Essas representações são conhecidas como *embeddings* que são vetores n-dimensionais que podem comprimir a informação contida nas estrutura presentes nas ontologias e permitem que operações algébricas sejam aplicadas nesses dados simbólicos para realizar operações semânticas. Utilizando essa capacidade, modelos de linguagem são construídos de forma a gerar embeddings que agregam as informações semânticas em um texto com a finalidade de auxiliar em diversas tarefas. Para o alinhamento de ontologias, esses podem ser utilizadas para a geração de *embeddings* que permitem a comparação semântica entre entidades presentes nas ontologias. Porém, apesar da capacidade de representação desses modelos, a maioria dos sistemas de alinhamento ainda utilizam essas técnicas como métodos auxiliares e não exploram todo o potencial dessas técnicas.

Um dos maiores problemas encontrados na utilização dessas técnicas é a dificuldade de modelar métricas de similaridade entre entidades. Muitas dessas abordagens utilizam embeddings para comprimir a representação de um grupo de entidades com o intuito de evitar comparações entre entidades que improvavelmente serão alinhados. Apesar da

flexibilidade que os embeddings fornecem, ainda não se é definido qual o conceito geral de similaridade entre conceitos. A maioria dos sistemas desenvolvidos para essa tarefa e para alinhamento são treinados com pares de objetos similares com o intuito de aprender automaticamente quais atributos são mais importantes na definição de similaridade. Porém, essa abordagem requer um grande conjunto de treino rotulado que em muitas vezes é difícil de se obter. Além disso, para tarefas de alinhamento, um sistema que recebe duas entidades como entrada exige uma grande quantidade de comparações para abranger todas as possibilidades. Assim, a definição de um conceito geral de similaridade pode auxiliar na extração de informações contidas em cada entidade para guiar o alinhamento resultando em alinhamentos de melhor qualidade e a redução no espaço de busca.

Uma das tarefas que utilizam o conceito de similaridade é o alinhamento de propriedades a qual é uma área importante para o alinhamento de ontologias visto que algumas hipóteses mostram que a informação descrita por elas pode auxiliar no alinhamento geral de entidades. Uma dessas hipóteses é o princípio da localidade (JIMÉNEZ-RUIZ; GRAU, 2011) o qual descreve que entidades próximas a uma entidade alinhada tendem a ser alinhadas entre ontologias. Apesar da importância dessa tarefa, pouca atenção tem sido dada ao alinhamento de propriedades recentemente e como utilizar os *embeddings* para aprimorar a capacidade de alinhamento desses sistemas.

1.1 Hipótese

Tendo em vista os problemas descritos anteriormente, três hipóteses se mostram promissoras em aprimorar a qualidade dos sistemas nessas áreas. A primeira hipótese é que a utilização de embeddings de palavras e sentenças podem auxiliar no cálculo de similaridade entre entidades e aumentar a capacidade de alinhamento entre propriedades nos sistemas de alinhamento. Esses recursos podem ser adicionados a uma técnica base de alinhamento de propriedades para atingir melhores resultados quando comparado a outros sistemas do estado da arte na mesma categoria.

Outra hipótese é que uma métrica geral de cálculo de similaridade deve ser capaz de distinguir tipos diferentes de entidades com base em uma série de características. Algumas análises no padrão de alinhamento entre entidades indicam que existe uma classificação de alto nível a qual todos os conceitos pertencem e que conceitos de diferentes classes possuem

baixa similaridade. Algumas ontologias definem tipos fundamentais de alto nível como por exemplo a ontologia DOLCE e a utilização desses tipos podem auxiliar na construção de métricas gerais de similaridade. Porém, essas ontologias exibem classes que geralmente agrupam conceitos que humanos frequentemente percebem como elementos diferentes. Partindo da hipótese de que humanos possuem sensibilidade maior ao perceber diferença entre conceitos físicos como seres vivos e objetos, a utilização de um sistema de classes com maior distribuição de conceitos físicos pode auxiliar no cálculo de similaridade entre conceitos.

Além da perspectiva de uma função universal de similaridade, a hipótese de que conceitos diferentes estão agrupados por classes e que conceitos similares pertencem à mesma classe, surge então a possibilidade da utilização dessas classes para reduzir o espaço de busca entre as entidades numa ontologia. Ao assumir que conceitos similares pertencem à mesma classe e que conceitos distintos nunca são similares a conceitos em outras classes, pode-se desconsiderar a comparação entre conceitos de classes distintas reduzindo o número total de comparações. Apesar da redução, o sistema de classes utilizado possui influência na quantidade esperada de comparações, pois um sistema de classificação que agrupa conceitos frequentemente considerados como diferentes faz com que os sistemas de alinhamento tenham que executar a operação de comparação em entidades que poderiam ser filtradas em etapas iniciais.

1.2 Objetivos

Diante das hipóteses descritas, esta pesquisa visa investigar os seguintes tópicos:

- investigar a utilização de embeddings e da extensão de alinhamento para sistemas de alinhamento de propriedades
- propor um conjunto de classes top level derivada de ontologia fundacionais para definir um conceito de similaridade semântica entre conceitos de domínios distintos
- propor uma arquitetura para a classificação de conceitos em classes top level e verificar a possibilidade de redução do espaço de matching em sistemas de alinhamento utilizando esses tipos

1.3 Contribuições

Este trabalho contribui com um dataset rotulado com 6 classes top level, uma arquitetura que apresenta melhores resultados para a classificação de conceitos e a proposta de utilização desse tipo de classificação para redução do espaço de matching para ontology matching.

Além deste capítulo introdutório, o trabalho possui mais 5 capítulos sendo eles Conceitos Básicos onde são revisados os conceitos para entendimento e execução dos experimentos, trabalhos relacionados onde são apresentados os trabalhos a qual esta pesquisa tem fortes influências, o capítulo Método onde são descritos as abordagens e arquiteturas utilizadas, o capítulo de Experimentos onde são descritos os resultados dos experimentos e por fim a conclusão onde são apresentados trabalhos futuros.

2 Conceitos Básicos

Nesta seção são apresentados os conceitos básicos para o entendimento da proposta apresentada. As ontologias são estruturas que descrevem entidades e suas relações. Esses conceitos contém atributos descritos em linguagem natural como rótulos e comentários. Desta forma, técnicas relacionadas ao processamento de linguagem natural são adequadas para o processamento dessas estruturas. As próximas seções descrevem o avanço na área de processamento de linguagem natural e a grande qualidade dos modelos de linguagem atuais. Tais modelos são baseados em Deep Learning e aprendizado de máquina para atingir alta capacidade de representação. Também será revisado as diversas técnicas de aprendizado utilizadas para treinar tais modelos. Em seguida serão apresentados os tipos de representação textual e as arquiteturas utilizadas para o processamento de sequências textuais bem como as diversas aplicações desses modelos na área de processamento de linguagem natural. Por fim, será abordado os conceitos básicos do processamento de ontologias e a tarefa de alinhamento de ontologias.

2.1 Aprendizado de máquina

Aprendizado de máquina (AGGARWAL, 2022) é um tópico relacionado a inteligência artificial e estuda como máquinas podem aprender apenas observando dados sem serem explicitamente programadas para realizar determinada tarefa. Em áreas como processamento de linguagem natural, a quantidade de estruturas possíveis nas linguagens são incontáveis tornando a tarefa de registrar todos os padrões possíveis uma tarefa intangível. Outro problema é que algoritmos escritos manualmente são fixos e visam resolver um problema em específico. Algumas tarefas são extremamente complexas para serem resolvidas por algoritmos manualmente construídos, porém exemplos de soluções dessas tarefas são fáceis de se obter. Desta forma, a utilização de algoritmos que podem aprender com os dados diminuem o esforço humano e complexidade dos problemas a serem resolvidos, pois o pesquisador passa a pensar no que deve ser resolvido e não como.

Vários conceitos são abordados na área de aprendizado de máquina como classificação, extração de features e técnicas de aprendizado os quais serão discutidos nas próximas seções.

2.1.1 Features

Na área de aprendizado de máquina e reconhecimento de padrões as *features* são um conjunto de características mensuráveis utilizadas para descrever um fenômeno (BISHOP, 2007). As *features* são utilizadas por diferentes tipos de algoritmos para a tomada de decisões e geralmente expressam valores numéricos ou categóricos. As features também podem ser representadas como diferentes visões de um mesmo conjunto de dados. Por exemplo, em uma tarefa relacionada a processamento de texto, as *features* podem ser as letras, palavras ou segmentos do texto ou caracteristicas do texto como frequência de palavras, etc. Já em uma tarefa de reconhecimento de padrões em imagens, essas *features* podem ser os pixels na imagem ou alguns objetos reconhecidos utilizando pre-processamento.

A escolha de boas *features* é muito importante para a tarefa desempenhada pois elas definem a capacidade de descrição dos fenômenos. Uma das etapas frequentemente utilizadas antes de utilizar as features é o pré-processamento. Nessa etapa, as *features* passam por processos que visam aumentar a qualidade dos dados como pro exemplo normalização, remoção de valores vazios ou filtrar *features* irrelevantes.

2.1.2 Classificação

Uma das áreas mais importantes em aprendizado de máquina é a classificação (KADHIM, 2019). Classificação é a tarefa de atribuir uma classe dentre possíveis categorias a uma exemplo de teste. Os exemplos de teste consistem em um conjunto de atributos que um sistema inteligente utiliza para decidir qual a classe mais adequada para rotular tal exemplo. A classificação é uma das áreas com o objetivo de reconhecer padrões em um conjunto de dados e agrupá-los em um conjunto de acordo com suas características. As modalidades de classificação são: classificação de uma classe, binária ou multi classe. Na classificação de uma classe o classificador é treinado com apenas exemplos pertencentes a uma única classe e deve decidir se novos exemplos pertencem à classe treinada ou não. Esse tipo de abordagem é muito utilizada em casos que os exemplos de uma classe são abundantes e exemplos de anomalias são raros. Na classificação binária o classificador deve decidir se um novo exemplo se encaixa em duas classes pre-estabelecidas. Da mesma forma a classificação multi-classe define inúmeras classes possíveis para um dado exemplo.

A maioria dos classificadores tradicionais dependem de atributos pré-estabelecidos pelo desenvolvedor. Por exemplo, ao utilizar imagens como dados de treino, o desenvolvedor é responsável por processar e extrair os atributos mais importantes como por exemplo número de objetos na imagem ou outras características como linhas e filtros. Exemplos desse tipo de classificadores são Dicision Tree (JURGENSON; MANSOUR, 2018), Gaussian Bayes Network (HECKERMAN; GEIGER, 2013), Random Forest (ZHENG et al., 2022) e Suport Vector Machine (NOBLE, 2006). Porém, a seleção manual de features torna o modelo dependente da implementação e dificulta a generalização para a maioria dos casos. Assim, estratégias modernas como Deep Learning não dependem da extração de features manuais possibilitando com que o modelo aprenda quais features são mais importantes e como extraí-las.

2.2 Redes neurais

As redes neurais são circuitos compostos de neurônios artificiais utilizados para resolver problemas na área de inteligência artificial. As redes neurais processam valores iniciais utilizando uma quantidade de camadas intermediárias. Nesse processo, valores iniciais são multiplicados por uma série de pesos e somados. Após a soma os valores são processados por uma função de ativação e o resultado é utilizado como entrada para camadas posteriores. Um exemplo dessa arquitetura está descrito na Figura 1.

2.2.1 Deep Learning

Deep learning (BENGIO et al., 2021; MATSUO et al., 2022) é uma técnica de aprendizado baseada em modelos neurais. Esses modelos exploram a natureza composicional dos objetos e conceitos naturais. Ao utilizar um grande números de camadas, os modelos podem codificar diferentes estruturas que descrevem características cada vez mais abstratas em vários níveis de profundidade. O uso de Deep Learning se tornou referência para a maioria das tarefas de aprendizado devido a imensa capacidade de representação e a possibilidade de reconhecer padrões complexos nos dados. Por exemplo, redes neurais são capazes de aprender a extrair um conjunto de *features* importantes para a tarefa desempenhada como aprender filtros capazes de reconhecer linhas e arestas assim

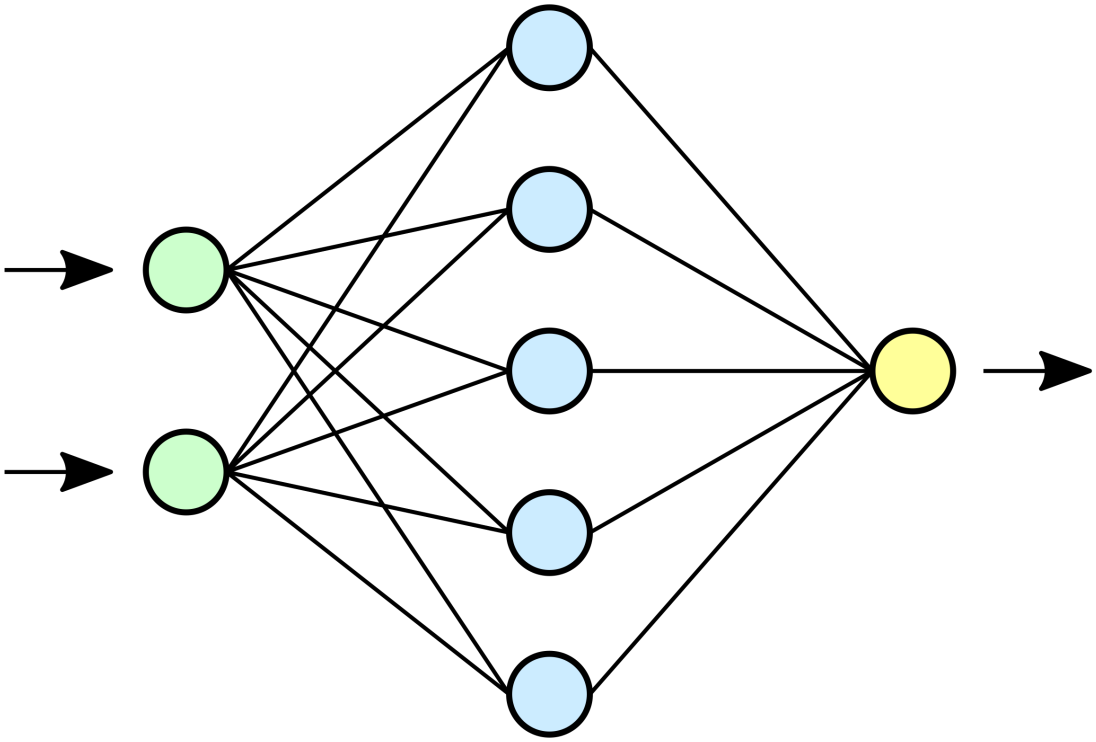

Figura 1 – Redes neurais são compostas de neurônios artificiais que imitam o processo que o cérebro utiliza para processar informações.

como outros padrões observados nos dados de treino. Um exemplo dessa arquitetura está apresentado na Figura[1] 2.

Após os modelos que utilizam Deep Learning atingirem baixas taxas de erro em competições como ImageNet, essas redes se tornaram padrão em solução de diversas tarefas perceptuais e atingem feitos impressionantes. Em 2015 uma rede neural denominada AlphaGo (SILVER et al., 2017) foi o primeiro jogador artificial a ganhar de um campeão mundial no jogo Go. Esse jogo é considerado de extrema dificuldade para inteligências artificiais devido a sua grande quantidade de combinações possíveis. Após esse feito, um novo modelo chamado AlphaZero (SILVER et al., 2018) foi capaz de vencer o estado da arte em vários jogos como xadrez, go e atari sendo treinado apenas ao jogar contra si mesmo.

Na área de processamento de linguagem natural os modelos baseados em Deep Learning demonstram a capacidade de lidar com tarefas cada vez mais complexas. O modelo GPT-3 (BROWN et al., 2020) atingiu resultados competitivos em tarefas distintas. A grande capacidade de entendimento textual possibilitou o modelo a ter desempenhos que

[1] Fonte: <https://www.xenonstack.com/blog/log-analytics-deep-machine-learning>

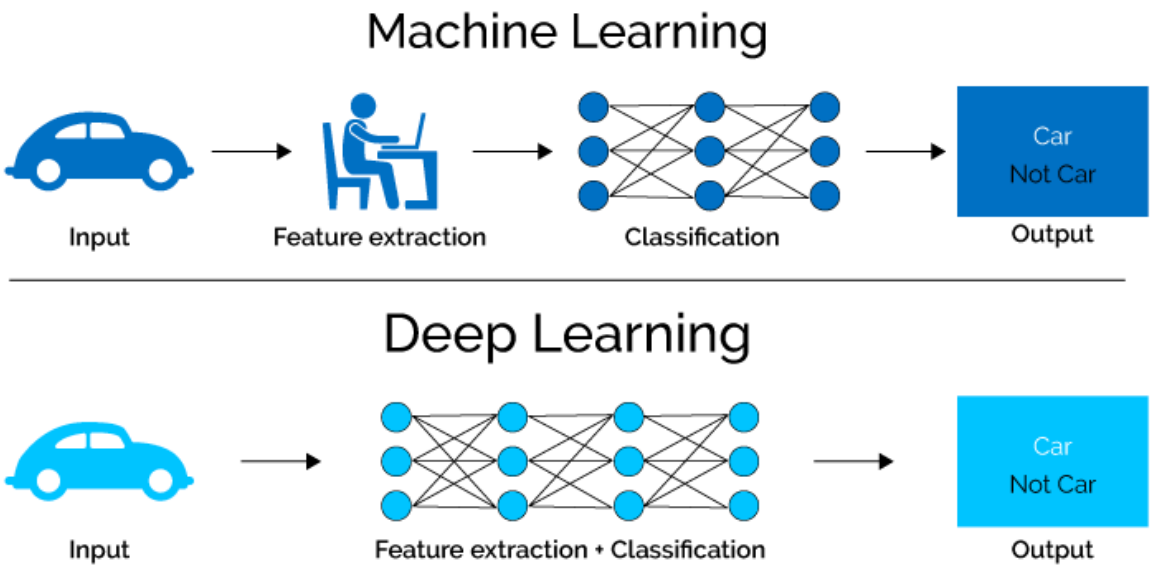

Figura 2 – Arquiteturas de Deep learning podem aprender automaticamente a extrair um conjunto de *features* dos dados de entrada

.

superam modelos do estado da arte em tarefas como sumarização, classificação e tradução. O modelo também é capaz de gerar códigos em diversas linguagens de programação a partir de uma descrição em linguagem natural. O modelo demonstrou também a capacidade de resolver problemas matemáticos descritos textualmente além de gerar textos com fluência próxima a escritos por humanos.

Outra tecnologia com resultados que superam alguns modelos no estado da arte são os modelos baseados em diffusion para geração de imagens. Esses modelos (RAMESH et al., 2022; SAHARIA et al., 2022; ROMBACH et al., 2022) são capazes de gerar imagens realistas a partir de uma descrição textual como exemplificado na Figura 3.

2.3 Tipos de aprendizado

Diferentes tipos de aprendizagem são descritos na literatura. Cada um possui suas características e são melhores adaptadas para uma tarefa específica.

Aprendizado **supervisionado** (SEN et al., 2020) é um processo que um sistema aprende com exemplos rotulados. Geralmente esses exemplos são produzidos por humanos. Alguns modelos como deep learning precisam de uma quantidade alta de exemplos para atingir resultados significantes. É uma das abordagens mais caras porém o modelo é

Figura 3 – Imagens gerada a partir da descrição: "human walking into the center of universe"

treinado diretamente a resolver determinada tarefa e pode encontrar padrões interessantes durante o processo.

Ao contrário do aprendizado supervisionado, o aprendizado **não supervisionado** (KHANUM et al., 2015; CELEBI; AYDIN, 2016; POUYANFAR et al., 2019) não visualiza rótulos durante o treino. Desta forma, o modelo precisa encontrar padrões e relações nos dados. A pesar de não ser supervisionado, esse tipo de aprendizado ainda é direcionado pelo desenvolvedor, pois a tarefa e as métricas de loss devem ser explicitas.

Os tipos de estruturas aprendidas com outros tipos de aprendizados podem ser reaproveitados para outras tarefas. Essa abordagem é conhecida como **transfer learning** (PAN; YANG, 2010; ZHUANG et al., 2021). No transfer learning partes de um modelo que aprendem a extrair atributos dos dados, como por exemplo uma camada de uma rede cnn capaz de reconhecer linhas e arestas, podem ser reaproveitados para tarefas similares. Modelos que utilizam essa abordagem aprendem mais rápido pois não precisam reaprender certos padrões.

2.4 Processamento de linguagem natural

Processamento de linguagem natural é um tópico que abrange uma série de técnicas e estudos sobre o desenvolvimento do entendimento da linguagem natural por autômatos inteligentes. Devido a imensa quantidade de padrões presentes nas linguagens, técnicas

de aprendizado são utilizadas para representar e processar textos. Uma das maiores dificuldades para o processamento de texto é como gerar uma representação que capture os conceitos semânticos e que seja utilizável por máquinas para tarefas como classificação, tradução, etc. Abaixo estão alguns dos tópicos relacionados a essa área e do histórico de avanços das representações e dos modelos.

2.4.1 Representações de palavras

Para que máquinas possam interpretar texto, primeiramente ele deve ser convertido em um formato que permita a realizações de operações semânticas e que possibilite que máquinas realizem tarefas que possuem texto como entrada ou saída. Ao longo do tempo diversas representações são propostas com o objetivo de possibilitar que cada vez mais essas representações modelem mais aspectos das linguagens naturais. Ainda não temos uma descrição formal de grupos que atue diretamente na semântica dos símbolos assim como temos para matemática. Por exemplo, na matemática temos operações como adição que descrevem como combinar dois números e obter outro número correspondente às características dos números combinados. Para palavras isso ainda não foi definido. Não temos um operador que combine duas palavras e retorne uma nova com as características semânticas das palavras combinadas. Para lidar com esse problema, diversas representações intermediárias tentam descrever palavras utilizando elementos em que essas operações são definidas.

2.4.1.1 One hot encoding

Uma das primeiras ideias nesse sentido seria mapear cada palavra para um número específico. Isso possibilitaria que modelos matemáticos já existentes e algoritmos de aprendizado pudessem trabalhar diretamente com o sentido das palavras. Porém, a organização das palavras em um espaço vetorial é uma tarefa difícil pois palavras isoladas não carregam informação suficiente para posicioná-la de forma que as operações matemáticas resultem em resultados coerentes. Outra abordagem seria representar cada palavra como um vetor $\vec{v} \in \mathbb{R}^n$ sendo n a quantidade de elementos em um vocabulário V e todas as dimensões possuem 0 exceto em uma. Apesar de permitir que alguns

algoritmos possam trabalhar com essa representação, ela ainda possui algumas falhas. A primeira é que todas as palavras tem a mesma distância entre si, impossibilitando que palavras relacionadas sejam posicionadas próximas. Outro problema é a esparsidade dessa representação e o grande consumo de memória que aumenta conforme novas palavras são adicionadas.

2.4.1.2 Word embeddings

Seguindo a hipótese distribucional (SAHLGREN, 2008), que diz que palavras com sentidos próximos ocorrem em contextos próximos no texto, é possível desenvolver métodos que atribuam vetores n-dimensionais, conhecidos como word embeddings, para palavras respeitando estruturas como gênero, número, etc. Os métodos mais utilizados para a criação desses vetores são baseados em redes neurais ou decomposição vetorial e se baseiam em atribuir um vetor n-dimensional para cada palavra e ajustar esses vetores de forma que se aproximem de vetores que representam palavras contextualmente próximas e se afastem de palavras distantes. Word2Vec (MIKOLOV et al., 2013) é um modelo baseado em redes neurais capaz de aprender representações eficientes de palavras. Esse modelo mostrou a capacidade capturar relações semânticas interessantes como a capacidade de calcular analogias. Um exemplo dessa estrutura é a capacidade de realizar a operação $\vec{rei} - \vec{v}_{homem} + \vec{v}_{mulher} \approx \vec{v}_{rainha}$. Desta forma, os embeddings demonstram grande potencial em representar conceitos semânticos que possibilitem a realização de operações algébricas.

2.4.2 Representações de sentença

Tendo em vista a característica sequencial dos textos, uma representação única para sentenças é necessária. Sentenças possuem tamanhos diferentes e dificultam a utilização direta em modelos de classificação que recebem como entrada um conjunto fixo de atributos. Uma das estratégias utilizadas para a representação das sentenças é a soma dos vetores $\vec{v}$ representados como one hot encoding. Assim a representação de uma sentença se dá pela fórmula

$$\vec{s} = \sum_{i=0}^{|S|} w_i \tag{2.1}$$

sendo $\vec{s}$ a representação vetorial da sentença S e w a palavra na posição i da sentença. Essa representação também e conhecida como Bag Of Words (ZHANG et al., 2010).

Uma importante métrica derivada dessa representação é a TF-IDF (AIZAWA, 2003). Essa métrica atribui diferentes pesos para palavras que ocorrem frequentemente em diversos documentos e é calculada pela seguinte equação:

$$idf_w = \log \frac{|D|}{df_w} \tag{2.2}$$

$$tfidf = \vec{s} * idf_w \tag{2.3}$$

Outra técnica comum utilizada para gerar representações de sentenças é agregar embeddings pre-treinados das palavras na sentença utilizando soma ou média (MA et al., 2019).

Com os avanços nas GPUs, os modelos baseados em operações vetoriais atingem performance cada vez maior. Existem, contudo, detalhes relacionados ao processamento de sequencias nas GPUs. As GPUs processam vetores de tamanho fixo diferentemente da natureza das sequências que pode variar. Duas técnicas são muito utilizadas: padding e truncation. A técnica de padding consiste em preencher os espaços vazios com algum elemento que geralmente é zero. Já truncation consiste em reduzir as sentenças para um tamanho fixo.

2.5 Modelos de linguagem

Os modelos de linguagem são modelos que aprendem uma distribuição probabilística sobre a probabilidade de ocorrência de sentenças. O modelo deve atribuir alta probabilidade para sentenças coerentes e baixa probabilidade para sentenças incoerentes e que não seguem a estrutura sintática de uma linguagem. Devido ao aprendizado de uma distribuição probabilística, esses modelos são capazes de gerar amostras pertencentes a essa distribuição e apresentam capacidades de gerar textos coerentes.

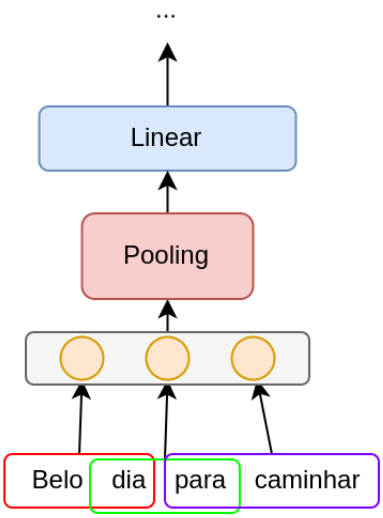

Figura 4 – Exemplo de utilização de arquitetura convolucional para processamento de textos a nível de palavras.

2.5.1 Redes neurais convolucionais

Redes Neurais Convolucionais (CNN) são os modelos do estado da arte para classificação e extração de features em imagens (LECUN et al., 1995). Na área de NLP, as CNNs são muito usadas para o processamento a nível de caracteres (CONNEAU et al., 2017; KIM et al., 2016; JOHNSON; ZHANG, 2016). Nesta área, seu processamento é similar aos modelos baseados em n-grams que geram partições de tamanho n sobre os caracteres no texto. Esses modelos podem ser utilizados para prever estados futuros com base apenas na informação do presente ao se basear na propriedade de Markov que dita que a informação presente é suficiente para a previsão de estados futuros. Estatisticamente é possível prever a distribuição dos caracteres no texto, porém dependências semânticas e contexto podem estar definidas a vários passos no passado, dessa forma esses modelos tem dificuldades em modelar tais dependências.

2.5.2 Redes neurais recorrentes

RNN são redes utilizadas em NLP por sua capacidade de modelar dependências a longo prazo no texto (CHO et al., 2014b; MIKOLOV et al., 2010). Seu funcionamento consiste na utilização de um estado intermediário como memória para gerar representações baseadas em informações no passado. Essas arquiteturas permitem a gerar textos de forma auto regressiva ao utilizar a saída da rede como entrada mantendo o estado anterior.

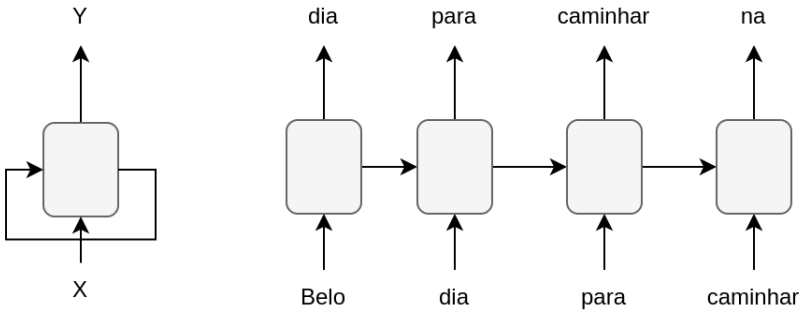

Figura 5 – Arquitetura básica de uma RNN.

Apesar da capacidade de representação, as RNN básicas sofrem com alguns problemas de aprendizado que dificultam a representação de sentenças muito longas. Dois dos comuns problemas são Vanishing Gradient e Gradiend Explosion que acontecem devido a natureza recurrente dessas redes e fazem com que o gradiente diminua ao longo da sentença ou aumente exponencialmente.

Para lidar com esse problema, arquiteturas como LSTM (HOCHREITER; SCHMIDHUBER, 1997) e GRU (CHO et al., 2014a) foram propostas. A arquitetura Long Short-Term Memory (LSTM) utiliza diferentes canais para decidir quais informações devem ser passadas adiante e quais informações memorizar. A LSTM adiciona um gate para decidir quais informações devem ser passadas a diante no texto. Esse sistema atua como um filtro que permite que apenas informações relevantes sejam consideradas para a representação de entrada. Assim o sistema pode representar longas sentenças sem sofrer com os problemas descritos anteriormente. Como a representação dos termos iniciais não recebe informação de termos no final da sequência, duas LSTM podem ser usadas para processar sentidos opostos da mesma sequência e as representações são combinadas para gerar uma única representação. Essa abordagem é conhecida como Bidirectional LSTM.

2.5.3 Attention

O mecanismo de attention (LUONG et al., 2015) tem como objetivo permitir que redes neurais filtrem informações irrelevantes e foquem em informações pertinentes de acordo com o contexto. A abordagem mais utilizada é a multiplicative attention (VASWANI et al., 2017) e consiste em calcular uma média ponderada de um conjunto de valores com base na similaridade entre valores de entrada como descrito na equação 2.4.

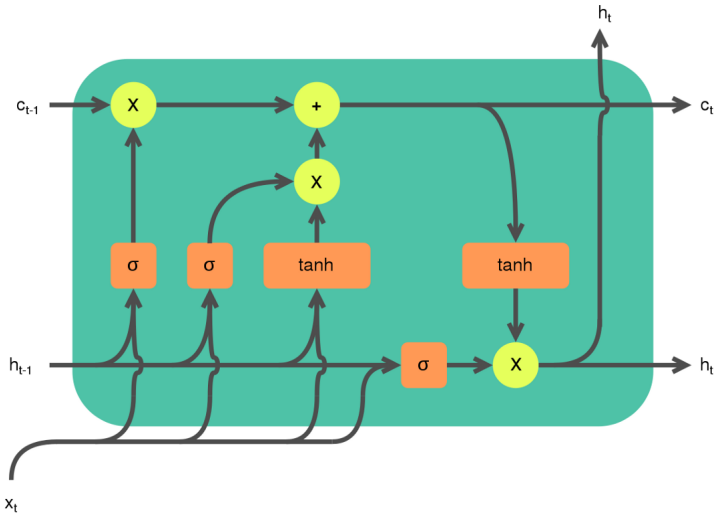

Figura 6 – Arquitetura básica da LSTM

$$attention(Q, K, V) = softargmax(\frac{QK^t}{\sqrt{(d_k}})V \tag{2.4}$$

Uma das implementações de attention utilizadas em NLP é a de self attention. No self attention, a representação dos valores em uma sequência é calculada com base em valores da mesma sequência. Com base na equação 2.4, o calculo do self attention é feito com Q, K e V sendo vetores da mesma sequência.

O mecanismo de attention pode também auxiliar na interpretabilidade da arquitetura pois é possível avaliar quais informações são relevantes em cada entrada. Os pesos calculados pelo mecanismo de attention estão dispostos em uma matriz de attention e podem ser verificados como mostra a Figura 7.

2.5.4 Transformer

Devido a natureza serial das RNNs, o treinamento em um volume grande de informações acaba sendo ineficiente. Como a saída gerada pela rede em um tempo t depende das saídas anteriores, torna-se difícil a paralelização do modelo. Visando aprimorar a performance desse modelo a arquitetura Transformer (VASWANI et al., 2017) foi proposta.

O Transformer foi inicialmente proposto para a tarefa de tradução e é dividida em dois módulos. O primeiro é o encoder que lida com a representação da sequência de entrada e o segundo é o decoder que gera a sentença de saída. O encoder é composto

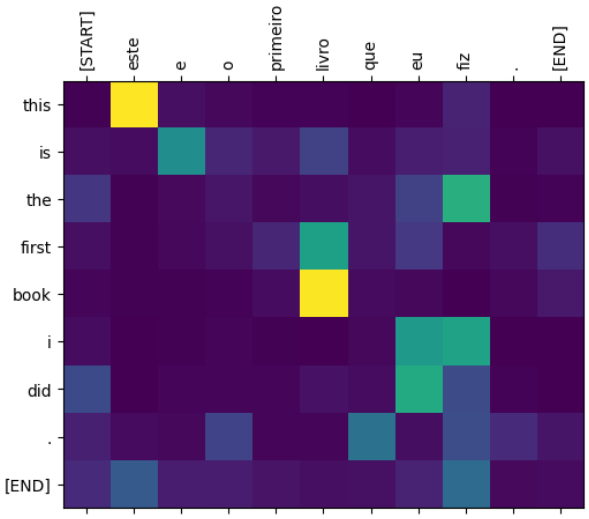

Figura 7 – O mecanismo de attention permite aprender quais palavras são relevantes para a representação de um termo.
https://www.tensorflow.org/text/tutorials/transformer

por vários blocos encoder empilhados. Os blocos encoder são compostos por multi-head attention que são múltiplos blocos de attention em paralelo que são concatenados e apos essa etapa passam por uma feed forward com conexões residuais. O decoder possui uma arquitetura similar, porém adiciona um bloco de attention para combinar a representação gerada pelo encoder e a representação de saída. Como os vetores são processados em paralelo o transformer não diferencia a posição entre os elementos na sequência e é bi direcional por natureza. Desta forma, uma codificação posicional é adicionada aos vetores de entrada para que as representações considerem a posição dos embeddings em relação à sequência. Com essa arquitetura o modelo pode ser treinado de forma paralela. Para inferência, que é a etapa em que o modelo não possui um exemplo de entrada, o decoder é retroalimentado com a saída gerada.

Devido a grande capacidade de representação do módulo encoder, o modelo BERT (DEVLIN et al., 2019) foi proposto. O BERT é constituído apenas de um Transformer Encoder como descrito na Figura 9.

O treinamento do bert é self supervised e consiste em duas tarefas. A primeira é reconstruir a sequência inicial apos a adição de ruídos ou mascaras e a segunda é prever a próxima sentença na sequência. Após esse treino, é esperado que o BERT consiga extrair atributos que geram uma boa representação do texto para sere utilizados em outras tarefas.

Outro modelo com grandes capacidades de processamento é o Gpt-3 (BROWN et al., 2020). Esse modelo é composto basicamente de um transformer decoder, porém,

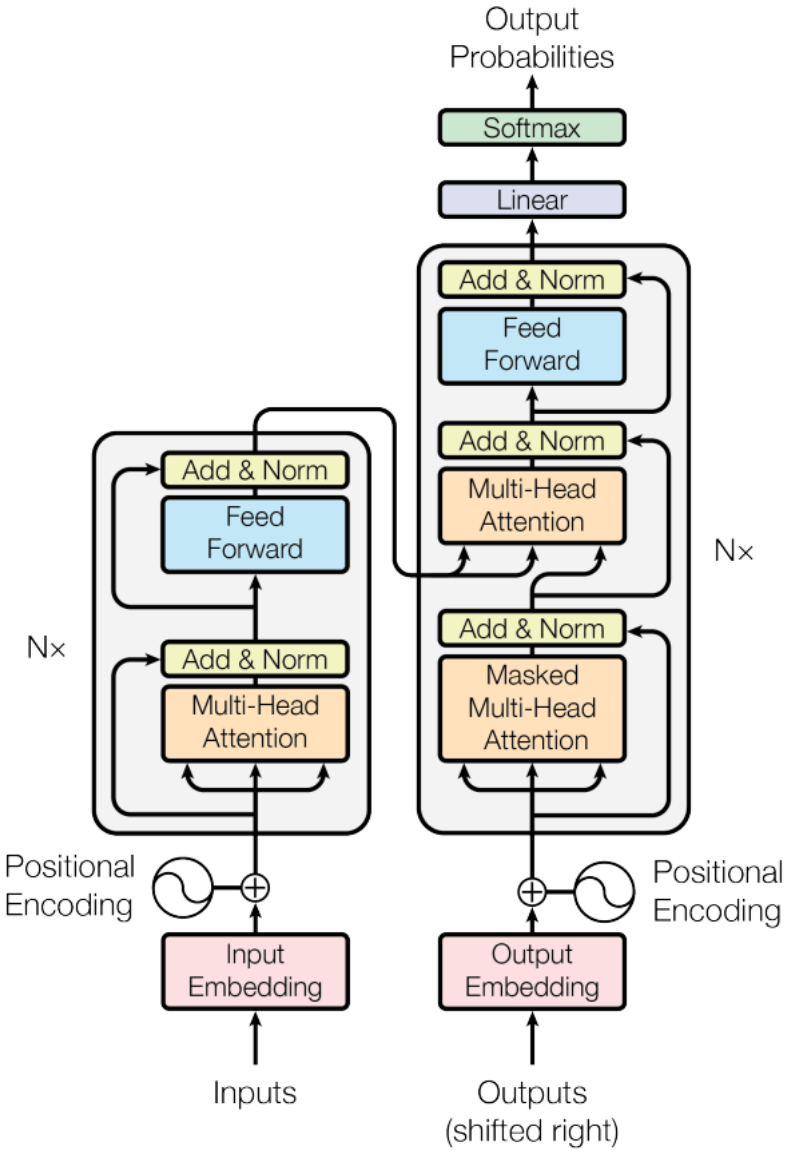

Figura 8 – Arquitetura Transformer

não possui o mecanismo de attention que relaciona os embeddings de entrada com um contexto gerado pelo encoder e se iguala ao encoder. Esse modelo possui uma quantidade enorme de parâmetros e é treinado em um imenso corpo de linguagem natural e com tarefas distintas. Os autores argumentam que devido à exposição a grande quantidade de dados e com tarefas distintas o modelo encontra que o melhor caminho é aprender como realizar few shot learning. Assim, a partir de poucos exemplos demonstrando uma tarefa o modelo consegue, com os mesmos pesos, se adaptar para a nova tarefa e obter um alto grau de generalização. Desta forma, quanto maior o modelo, maior a capacidade de entender os padrões descritos nos exemplos da tarefa e assim o modelo consegue executar

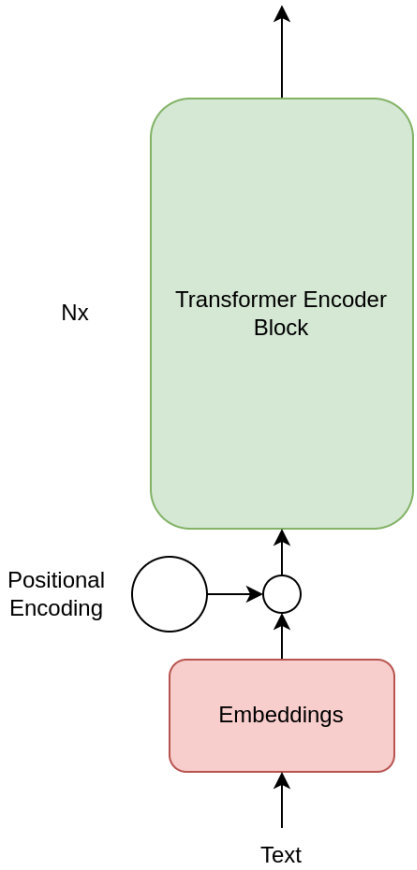

Figura 9 – Arquitetura básica do BERT

diversas tarefas como tradução, classificação e escrita de código com os mesmos pesos.

2.6 Aplicações do processamento de linguagem natural

Tendo em vista a grande capacidade dos modelos de linguagem no estado da arte, algumas aplicações conseguem atingir alto grau de performance e generalidade. Abaixo seguem alguns exemplos de tarefas comuns na área de nlp.

2.6.1 Part of speech tagging

O objetivo da tarefa de POS tagging (STRAKA; STRAKOVÁ, 2017; BOHNET et al., 2018) é encontrar a classe gramatical correta para cada palavra em uma sentença e está relacionada à análise gramatical das palavras. Técnicas iniciais se baseavam em utilizar um banco de dados contendo uma grande quantidade de palavras previamente rotuladas. Contudo, algumas palavras mudam sua classificação de acordo com o contexto.

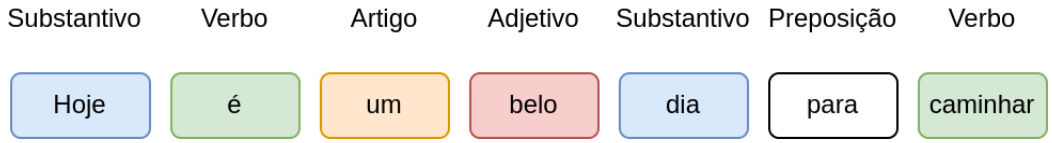

Figura 10 – A frase "Hoje é um belo dia para caminhar"classificada gramaticalmente.

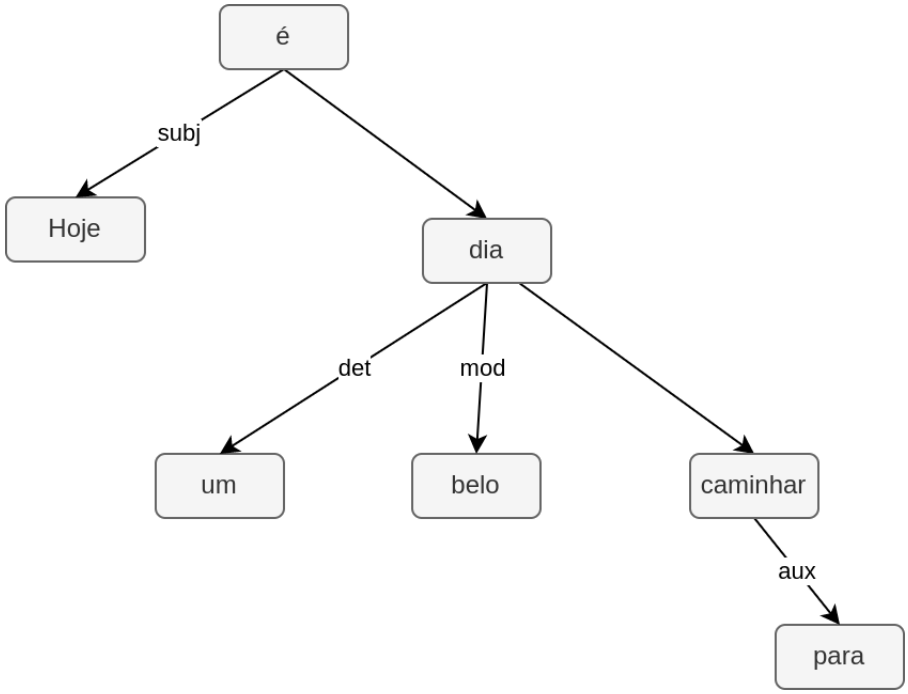

Figura 11 – Exemplo de análise de dependência.

Assim, os modelos com melhores resultados nessa tarefa utilizam uma abordagem neural para reconhecer padrões de utilização de acordo com contexto.

2.6.2 Dependency parsing

A tarefa de dependency parsing (MRINI et al., 2020) está relacionada à análise sintática do texto e possibilita encontrar relações semânticas entre palavras. Substantivos compostos assim como seus adjetivos podem ser agrupados em uma única estrutura utilizando dependency parsing. Essa funcionalidade é utilizada na tarefa de Reconhecimento de entidades nomeadas para agrupar as palavras que compõem o nome de uma única entidade para então serem classificados em uma das classes semânticas e possivelmente relacionados com alguma base de conhecimento. Outras aplicações possíveis para Dependency parsing são extração de relações, busca de informações e question answering.

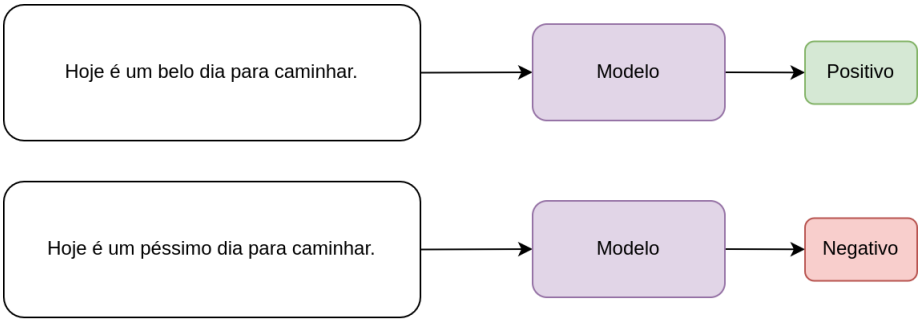

Figura 12 – Exemplo de tarefa de classificação de texto.

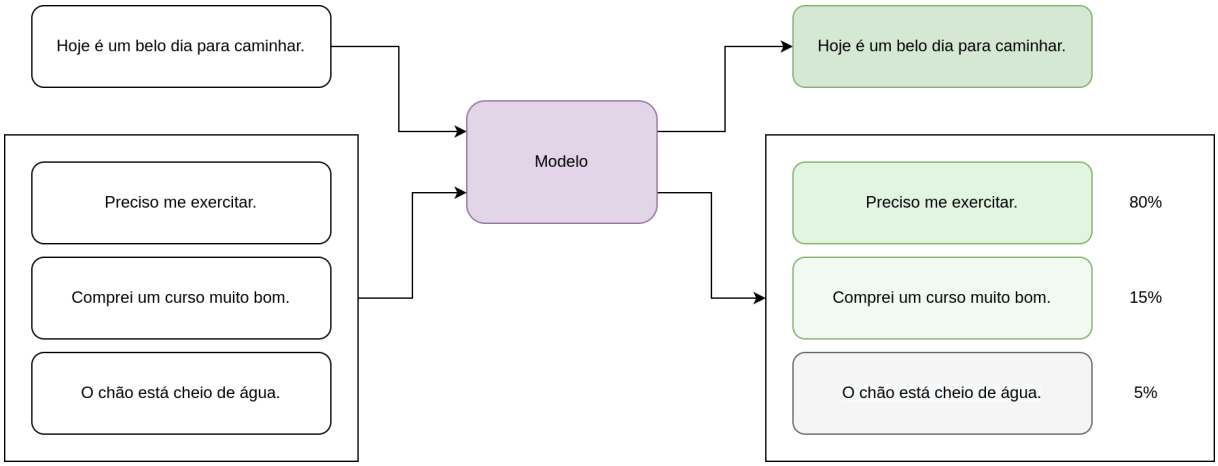

Figura 13 – Um exemplo da tarefa de similaridade entre sentenças.

2.6.3 Classificação de texto

Classificação de texto (YANG et al., 2019; HOWARD; RUDER, 2018) é uma das primeiras tarefas abordadas em nlp. Exemplos dessa tarefa são detecção de SPAM, análise de sentimentos e predição de reviews. Um dos maiores desafios dessa tarefa é como extrair boas features que representam o texto para a classificação.

2.6.4 Similaridade de sentença

Essa tarefa (YANG et al., 2019; LIU et al., 2019) tenta prever o quão similar dois trechos de texto são. Modelos neurais resolvem essa tarefa convertendo os textos de entrada em embeddings que capturam informações semânticas sobre o texto e ao comparar a distância entre esses embeddings pode se obter uma métrica de similaridade. Essa tarefa é utilizada na busca de informações como encontrar documentos que contenham uma frase ou assunto específico.

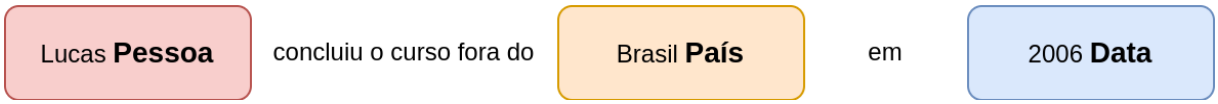

Figura 14 – Um exemplo da tarefa de NER.

2.6.5 Reconhecimento de entidade nomeada

A tarefa de reconhecimento de entidade nomeada (SHAHZAD et al., 2021; WANG et al., 2021b; WANG et al., 2021a) tem como objetivo detectar e classificar entidades em um texto não estruturado. Com essa classificação, o sistema tem uma indicação semântica das características daquela entidade e tem grande similaridade com a tarefa de top level class prediction. Os sistemas projetados para solucionar esse problema geralmente são baseados em várias etapas. A primeira etapa é a segmentação na qual várias palavras são agrupadas para formar o nome de uma única entidade. Para isso técnicas como Dependency Parsing são utilizadas para se agrupar termos sintaticamente relacionados como nomes e adjetivos. A segunda etapa é a classificação em que o trecho agrupado na etapa anterior é classificado em relação a um grupo de classes pre-definidas. Geralmente essa etapa utiliza Ontologias para enriquecer as entidades anotadas com tipos semânticos.

2.7 Ontologias

Ontologias (CHANDRASEKARAN et al., 1999) são estruturas que organizam conceitos, entidades e as relações entre eles. Os elementos de uma ontologia são classes, propriedades e instâncias. As classes representam conceitos abstratos como por exemplo Pessoa, Animal, Número. As classes agrupam indivíduos ou até outras classes. As classes são relacionadas entre si através de propriedades que por sua vez podem ter atributos diversos. As propriedades possuem um domínio que é a origem da relação e um alvo ou range. Relações podem descrever atributos, ações ou relações de causa e efeito entre conceitos. Geralmente as propriedades são descritas com frases verbais. Uma representação particular de uma classe é chamado de instância. Um exemplo de instância é Recife que é uma instancia de cidade. Uma estrutura muito relacionada às ontologias são os Knowledge Graphs (JI et al., 2022) que são específicos para uma aplicação específica enquanto as ontologias tendem a ser gerais de acordo com certo domínio. Ontologias geralmente são utilizadas em tarefas como entity linking (SHEN et al., 2015) e tarefas lógicas como

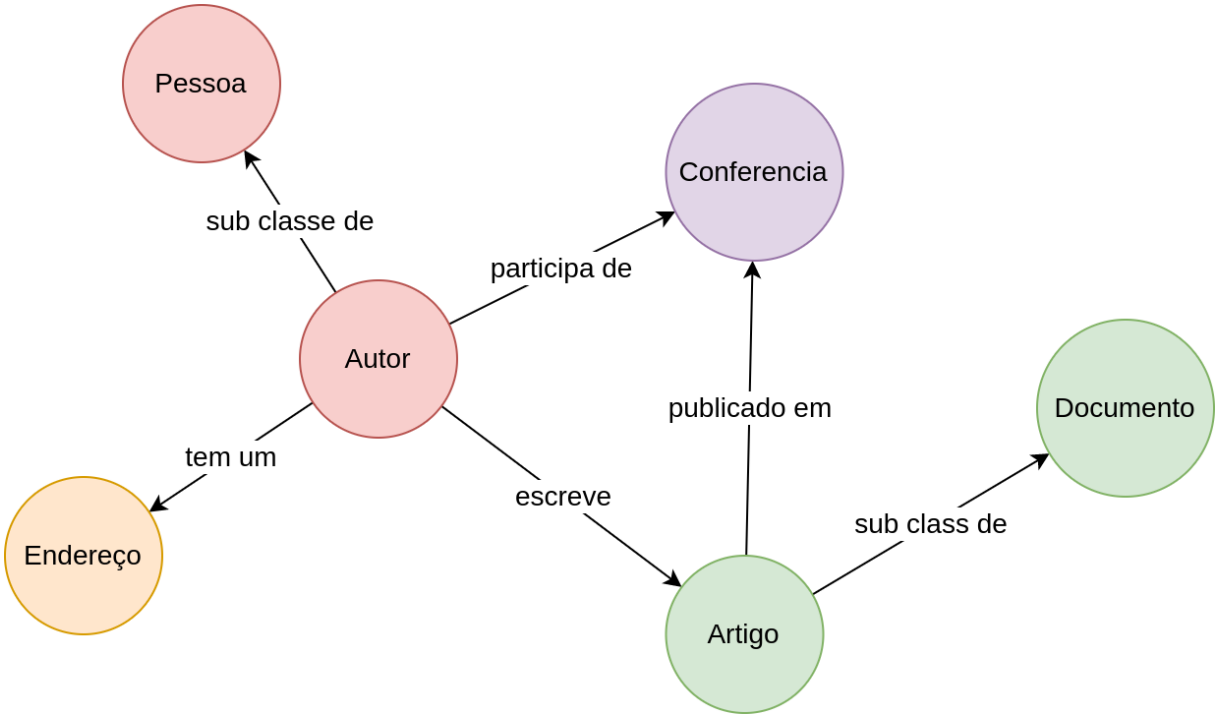

Figura 15 – Exemplo de uma ontologia.

reasoning (HENZE et al., 2004) e dedução.

Ontologias podem ter vários níveis de especificidade. Ontologias que modelam os conceitos fundamentais são conhecidas como ontologias top level (BORGO et al., 2022). Como as ontologias top level modelam conceitos fundamentais, elas fornecem elementos para a organização de conceitos em diversos domínios.

Ontologias podem ser vistas como grafos em que os conceitos são nós e as propriedades são arestas. Os conceitos em uma ontologia são descritos em linguagem natural e podem ter comentários. Da mesma forma, as propriedades podem ter outros atributos e até comentários. Desta forma, o grafo representado por uma ontologia possui atributos complexos. Um dos padrões utilizados para se armazenar grafos semânticos é o RDF (DECKER et al., 2000). O RDF define tipos e relações estruturais padronizadas para se representar conceitos. Porém o RDF não possui capacidade de descrição de conceitos lógicos, assim uma extensão do RDF é o OWL (MCGUINNESS et al., 2004) e sua versão OWL2 (HITZLER et al., 2009). O OWL define uma série de tipos que definem operações lógicas como negação, inversos, e equivalências além de operações sobre conjunto como intersecção, união. A definição OWL conta com predicados que permitem descrever relações orientadas a objetos como subClassOf e subPropertyOf.

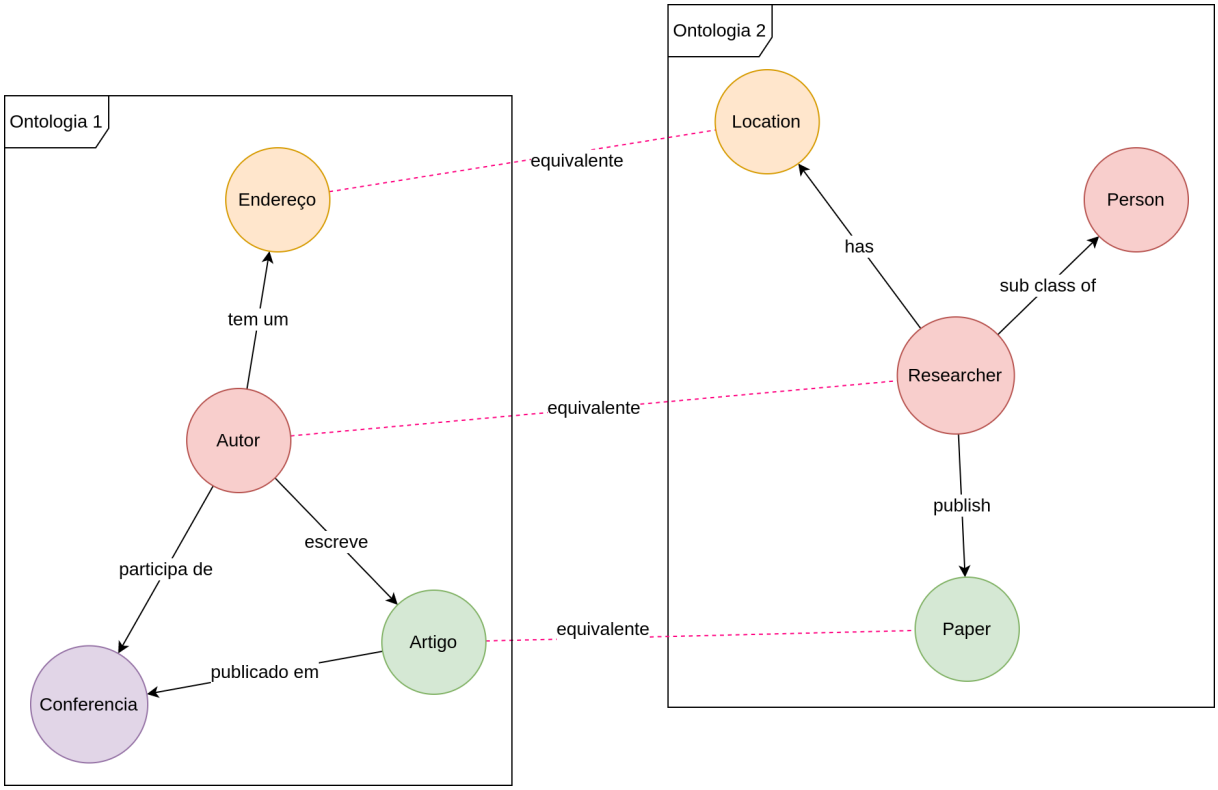

Figura 16 – Exemplo de alinhamento entre ontologias.

2.7.1 Ontology matching

Devido a vários autores terem diversas visões de mundo aplicações distintas para ontologias, pode ocorrer que conceitos expressando o mesmo significado acabem sendo descritos de formas diferentes como utilizando nomes diferentes ou em idiomas distintos, com um conjunto diferente de relacionamentos ou sendo descrito em níveis de abstração distintos. Para solucionar esse problema, sistemas automáticos de alinhamento utilizam um conjunto de regras e recursos como dicionários, enciclopédias ou sinônimos e com um conjunto de regras e métricas comparar conceitos para definir se a similaridade entre eles é alta o suficiente para serem interpretados como o mesmo conceitos. Esta tarefa é conhecida como Ontology Matching (EUZENAT; SHVAIKO, 2013) e um exemplo pode ser observado na Figura 16.

Diversas abordagens existem na literatura como por exemplo a utilização de regras e comparações estatísticas, e outras utilizam deep learning, embeddings e representation learning para aprender representações entre conceitos que possam ser comparáveis.

A tarefa de ontology matching geralmente é dividida entre alinhamento de classes e alinhamento de propriedades. Podem ser observadas as caracteristicas do alinhamento

entre classes em dois exemplos de alinhamentos entre as ontologias CMT e Conference. O alinhamento entre http://cmtChairman e http://conferenceChair demonstra a necessidade de se observar as relações entre as entidades para se encontrar os alinhamentos. Apesar da similaridade entre as palavras charman e chair, a entidade chair pode ser confundida com um objeto (cadeira). Porém, Chairman na ontologia Cmt é sub classe de Person (Pessoa) e Chair na ontologia Conference é sub classe de Committee_member e pode ser utilizado para se desambiguar os termos. Outro exemplo é o alinhamento entre http://cmtSubjectArea e http://conferenceTopic os quais exigem uma análise profunda da estrutura da ontologia e até conhecimento préviou que pode não estar contido nas ontologias.

Já o alinhamento de propriedades, pode utilizar a similaridade entre os domínios para o calculo final de similaridade entre as propriedades. Um exemplo de alinhamento entre propriedades é http://cmtassignedByReviewer e http://conferenceinvited_by no qual os domínios e range são similares porém, nesse caso, uma análise semântica das labels das propriedades são necessárias para se realizar o alinhamento.

3 Trabalhos Relacionados

Neste capítulo serão abordados as pesquisas relacionadas a este trabalho. Na primeira seção serão revistos as métricas de string matching utilizadas para ontology matching. Em seguida, a utilização dessas técnicas no alinhamento de propriedades. Após essa discussão, são apresentados exemplos de sistemas com a maior performance na competição e finalmente os trabalhos fundamentais para o entendimento do desenvolvimento de um tipo de classes e redução do espaço de matching.

3.1 Alinhamento de ontologias

Um dos primeiros trabalhos analisando a performance das métricas de similaridade de texto para ontology matching é (CHEATHAM; HITZLER, 2013). Nesse trabalho diversas métricas foram comparadas entre si na tarefa de ontology matching combinadas com vários tipos de pre-processamentos. Uma importante observação feita no trabalho é que as métricas utilizadas para o alinhamento de classes exibiam performance inferior ao serem utilizadas no alinhamento de propriedades principalmente no dataset Conference, como pode ser observado na Figura 17. Esses resultados indicam que o processamento para propriedades deve seguir uma regra diferente do que o das classes.

Figura 17 – Resultado das métricas de string matching no dataset Conference. O resultado das métricas no alinhamento de propriedades é muito inferior quando comparado às classes.

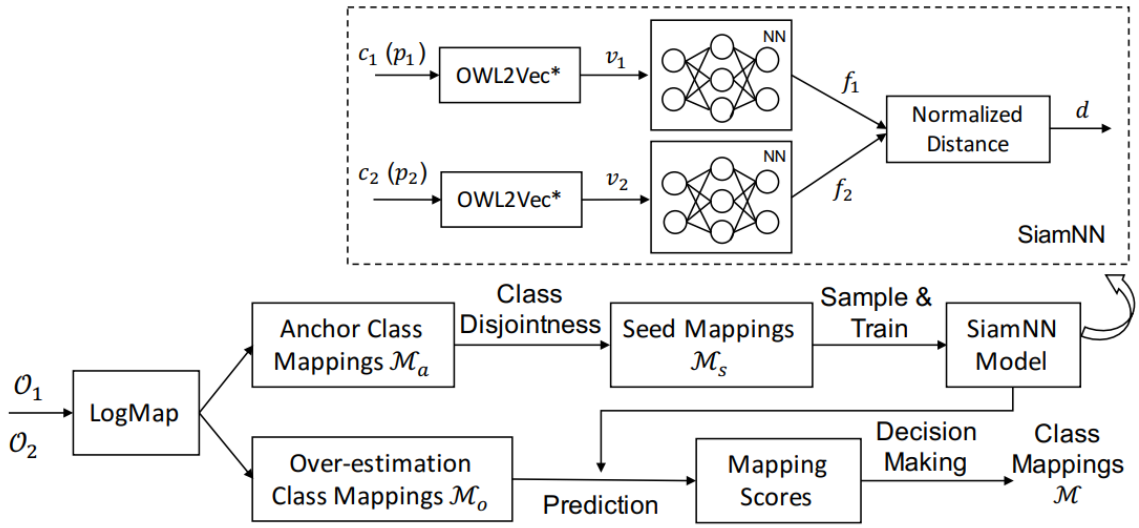

Figura 18 – Arquitetura principal do LogMap.

Seguindo o trabalho descrito anteriormente, o trabalho (CHEATHAM et al., 2014) propõe uma técnica que atinge melhores resultados no alinhamento de propriedades. Essa técnica utiliza as labels do domínio e range da propriedade source e target além dos termos principais no nome da propriedade. O conceito principal é extraído utilizando pos tagging. Em propriedades que contem verbo o conceito principal é o verbo e caso não haja verbos, então o conceito principal é o primeiro nome encontrado junto com seus adjetivos. Assim, a similaridade total é a menor entre a similaridade das labels da propriedade utilizando soft tf-idf e as similaridades de domain e range utilizando tf-idf. Apesar de atingir 100% de precisão na tarefa de alinhamento em conference, o recall do sistema ainda é muito baixo.

O AML (FARIA et al., 2013) é o sistema que atinge os melhores resultados em alinhamento de propriedades na competição OAEI no dataset Conference. Sua grande capacidade de alinhamento vem da base de conhecimento externo e grande quantidade de filtros que o sistema dispõe. O sistema utiliza, também, sinônimos e enciclopédias para auxiliar no cálculo de similaridade. O LogMap (CHEN et al., 2021; JIMÉNEZ-RUIZ; GRAU, 2011) é um dos sistemas da competição com bons resultados que utiliza embeddings para o cálculo de similaridade. Além da utilização dos embeddings o LogMap possui uma etapa de reparação e extenção dos alinhamentos. Uma das hipóteses para a extensão dos alinhamentos é o princípio da localidade. Esse princípio afirma que entidades próximas a um alinhamento correto tendem a ser alinhadas entre as ontologias. A arquitetura geral do LogMap está apresentada na Figura 18.

3.2 Redução do espaço de matching

Devido a alguns sistemas compararem todas as combinações possíveis entre entidades para realizar o alinhamento entre eles, alguns sistemas levam muito tempo para realizarem alinhamentos de ontologias muito granges. O trabalho (JIMÉNEZ-RUIZ et al., 2018) propõe uma técnica para a redução do espaço de alinhamento. Com base na hipótese de que entidades similares compartilham palavras em comum, é possível criar um indice de palavras e em quais entidades ela aparece de forma que possam ser criadas sub tarefas de alinhamento que no total reduz a quantidade total de alinhamentos. Para a divisão das tarefas são testadas duas abordagens: random clusters e a criação de cluster com base no algoritmo StarSpace (WU et al., 2018). Com essa abordagem, sistemas que falharam na competição na track de large bio por ultrapassarem o limite de tempo puderam completar a tarefa em tempo e com bons resultados. Apesar de resultados interessantes, a hipótese de divisão é léxica e pode falhar em encontrar bons candidatos.

O trabalho (JUNIOR et al., 2022) propõe uma arquitetura para a predição de classes top level com base na ontologia DOLCE (BORGO et al., 2022). O trabalho propõe dois datasets para treinar uma arquitetura para predição de classes. Um dos experimentos utiliza um dataset rotulado utilizando as classes da ontologia DOLCE para treinar a arquitetura. O modelo é treinado com a label de um conceito e sua descrição e tem como objetivo prever a classe top level. O modelo utiliza embeddings pre-treinados para representar as labels e aprende embeddings para representar o contexto. A representação é contextualizada utilizando uma LSTM bidirecional. Apesar de demonstrar resultados melhores que os baselines a arquitetura necessita de labels e descrições para a classificação. Porém, muitas ontologias não possuem descrições nas entidades, tornando o modelo pouco representativo para prever esses conceitos em ontologias reais.

4 Proposta

Diante da análise dos problemas descritos e das características encontradas nos trabalhos relacionados, esta proposta visa contribuir com o estado da arte com melhorias nos métodos utilizados previamente. Para isso, são propostas melhorias em 3 cenários relacionados ao alinhamento de ontologias. No primeiro cenário, é proposta uma arquitetura para o alinhamento de propriedades utilizando embeddings e extensão de alinhamentos. O segundo cenário descreve um sistema de classificação de conceitos top level mais adequado para a tarefa de ontology matching assim como a construção de um dataset anotado com essas classes para o treinamento de modelos de classificação. E por fim, será proposto a utilização de classes top level para auxiliar no cálculo de similaridades e reduzir o espaço necessário para o alinhamento entre ontologias.

4.1 Alinhamento de propriedades

Para o alinhamento de propriedades uma extensão no sistema proposto em (CHEATHAM et al., 2014). Alguns dos erros que o sistema comete é causado pelo fato dos métodos puramente léxicos serem incapazes de alinhar conceitos como Location e Place que possuem baixa similaridade léxica porém são geralmente utilizados como sinônimos. Desta forma é proposta uma série de técnicas que melhoram os resultados da arquitetura. A primeira abordagem é utilizar embeddings gerados a partir do Finnish Internet Parsebank (LUOTOLAHTI et al., 2015) utilizando word2vec [1]. Esse método é utilizado caso a similaridade entre os domínios seja 0 e as labels dos domínios sejam compostas de apenas uma palavra. Outra etapa utilizada e conhecida como mapping extension (CHEN et al., 2021) é a inclusão das propriedades inversas em caso de alinhamento. Além dessa abordagem, foi utilizado o modelo pre-treinado [2] MiniLM (WANG et al., 2020) para comparação das labels das propriedades caso a similaridade do domain e range sejam maiores que 95% e a similaridade das labels da propriedade sejam menor que 10% e contenham apenas uma palavra além dos adjetivos serem filtrados do range e palavras

[1] Os embeddings utilizados para o cálculo de similaridade entre as classes pode ser encontrado no endereço <https://turkunlp.org/finnish_nlp.html#parsebank>

[2] E o modelo para similaridade entre as propriedades neste endereço <https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2>

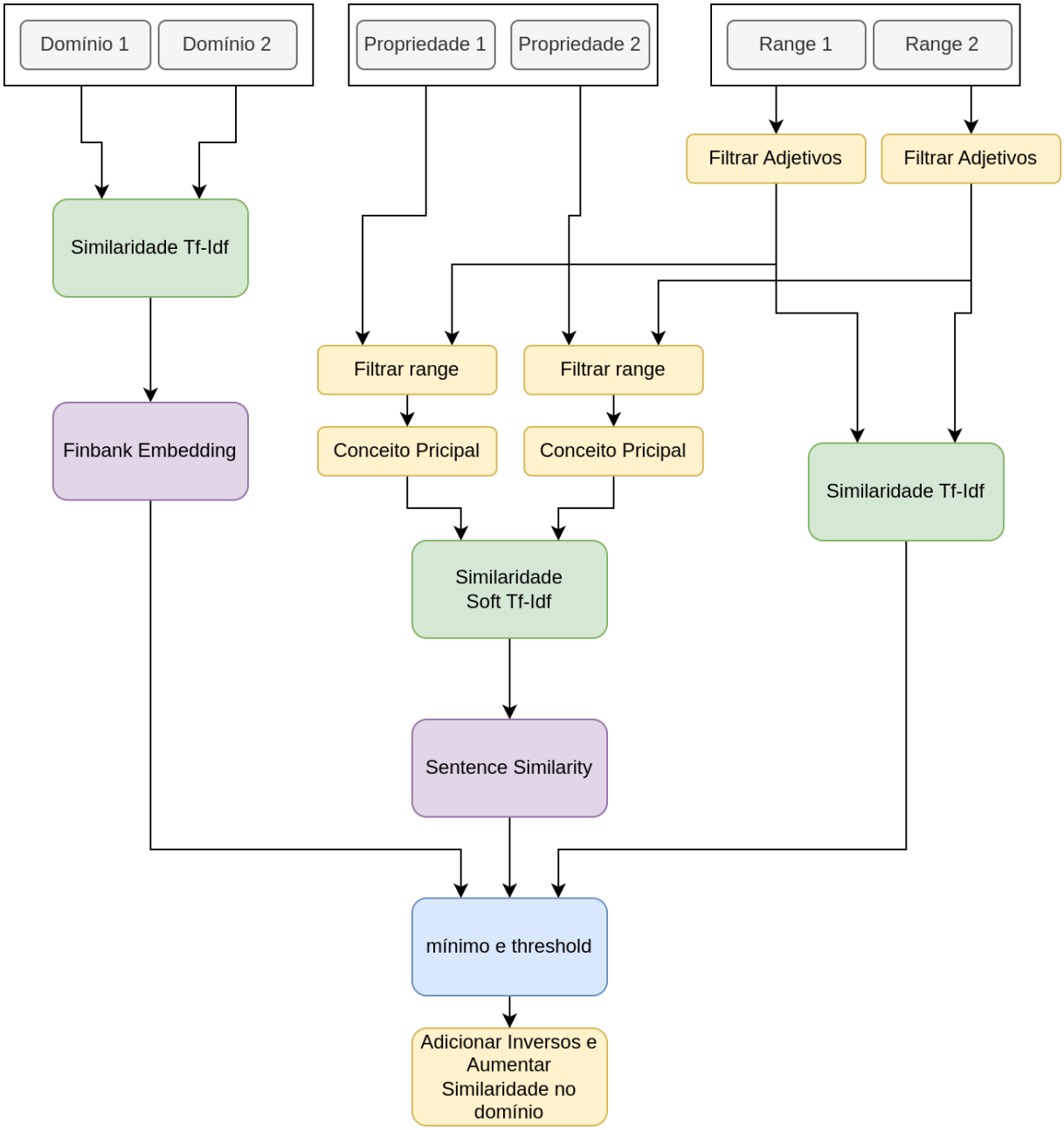

Figura 19 – Caption

similares no final da propriedade e no início do range são removidas. Por fim, a similaridade entre as labels das classes no domínio de propriedades já alinhadas é aumentada, o alinhamento passa por duas etapas e apenas o alinhamento de maior similaridade para cada domínio é mantido.

4.2 Classificação top level

A maioria dos modelos capazes de comparar a similaridade entre conceitos são treinados com pares de conceitos similares com o objetivo de aproximar a similaridade dos conceitos no treino e rejeitar pares não presentes no treino. Essa abordagem faz com que o modelo extraia features importantes para a comparação do modelo de similaridade. Porém, a utilização desse método não auxilia na explicação de quais atributos decidem o que torna dois objetos ou conceitos similares ou diferentes. Além dessa dificuldade, em ontology matching a quantidade de alinhamentos de referência são ordens de grandeza menores que a quantidade de comparações possíveis dificultado a aprendizagem desse conceito. Visando desenvolver uma métrica robusta e explicável de similaridade para ontology matching um importante padrão foi observado. Objetos similares tendem a pertencer a uma classe de alto nível em comum e objetos de diferentes classes nunca são similares. Um exemplo dessa comparação é a diferença entre Pessoa e Livro em que um dos termos é um ser vivo e o outro é um objeto. Um ser vivo não pode ser similar a um ser não vivo. Assim, utilizando um conjunto de classes de alto nível como material e abstrato por exemplo, todos os elementos similares vão estar contidos no mesmo grupo reduzindo a quantidade de comparações necessárias para um sistema de alinhamento e produzindo uma explicação fundamental do conceito de similaridade como ilustrado na Figura 20. Ontologias de top level descrevem esse tipo de classificação e podem ser utilizadas para esse propósito.

4.2.1 Classes propostas

Para gerar grupos mais distribuídos e com maior facilidade de descrição são propostas 6 diferentes classes baseadas nas classes da ontologia DOLCE.

Agent é a classe que agrupa seres vivos, organismos e instituições sejam eles individuais ou representando grupos. Alguns exemplos pertencentes à essa classificação: Pessoa, Animal, Sociedade, Organização e Personagem. Na classe **Event** todos os conceitos relacionados a movimento, ações e eventos são agrupados. Ex: Chuva, Movimento, Salto, Festa, Encontro, Conferência e Explosão. Outra classe é **Place** onde são agrupados lugares, construções, etc. Ex: Construção, Rua, Ponte, Planeta, Rio, Vulcão. **Item** é a classe que agrupa objetos, dispositivos e partes de seres vivos. Ex: Pedra, Carro, Ferro, Metal, Célula, Mão, Areia, Navio, etc. **Virtual** é a classe que descreve abstrações de objetos

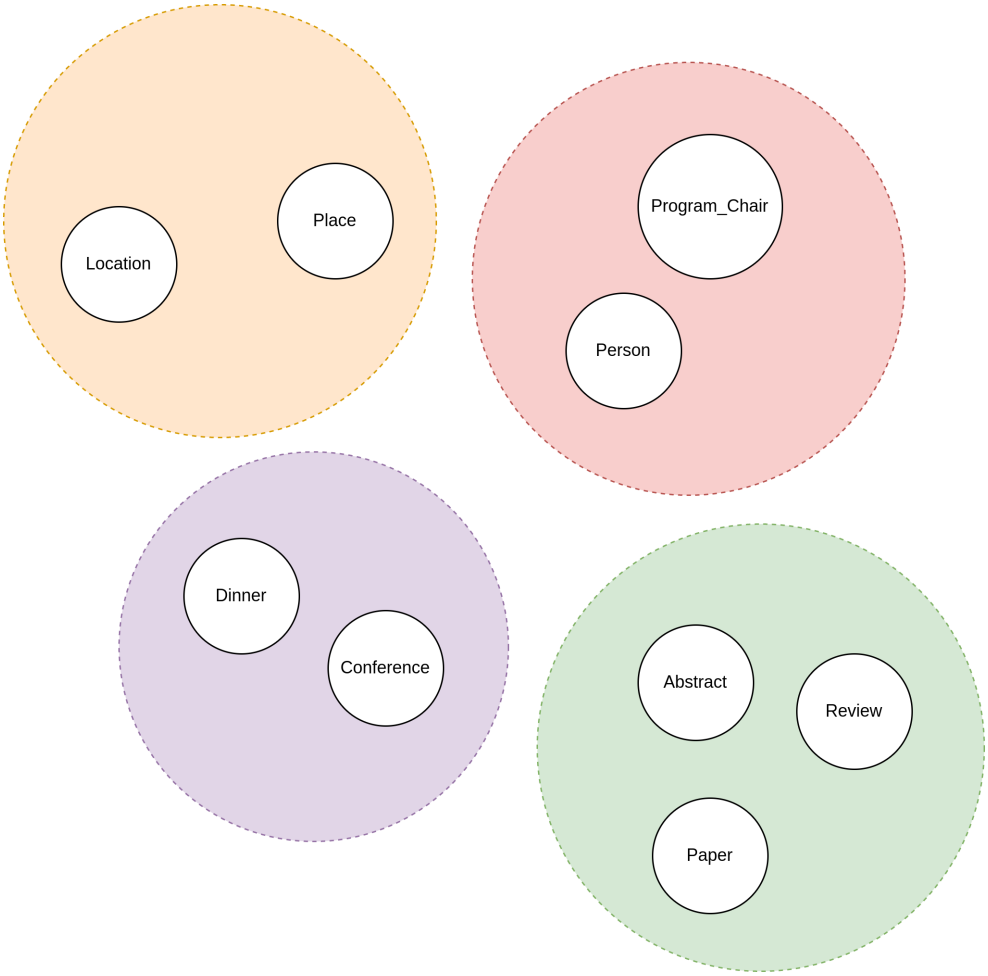

Figura 20 – Agrupar conceitos por tipos reduz a quantidade necessária de comparações já que somente conceitos de um mesmo grupo podem ser similares.

físicos e características físicas como: Imagem, Cubo, Círculo, Número, Linha. **Concept** é a classe mais genérica e abrange todos os outros conceitos que não se classificam nas outras classes. Exemplo: Identidade de Gênero, Subjetividade, Ética, etc.

A aplicação desses tipos podem auxiliar no cálculo de similaridade entre conceitos. Como conceitos similares pertencem à mesma classe top level, se exclui a necessidade de comparação entre diferentes classes. Algumas métrica de similaridade textual podem atribuir um alto valor de similaridade para palavras como Review e Reviewer que são próximas lexicalmente porem distantes semanticamente tendo em vista que Review nesse contexto pode significar um documento ou uma ação e Reviewer é uma pessoa. Já outras palavras como Location e Place que possuem letras diferentes podem ter uma similaridade aumentada por pertencerem a mesma classe Place.

Classe	Quantidade
Agent	1169
Event	2173
Place	2930
Item	1589
Virtual	1535
Concept	752

Tabela 1 – Quantidade de elementos em cada classe no dataset 1.

Classe	Quantidade
Endurant	88410
Perdurant	11683
Situation	7763
Quality	4947
Abstract	4035

Tabela 2 – Quantidade de elementos em cada classe no dataset 2.

4.3 Dataset

Para testar a eficiência das classes em diversas tarefas é necessário treinar um modelo em um dataset rotulado com as classes propostas. Para isso, foram rotuladas as classes do dataset YAGO (TANON et al., 2020) resultando em 10148 exemplos. A quantidade de elementos por clase é apresentada na Tabela 1 e a distribuição das classes rotuladas é apresentada na Figura 21.

O outro dataset utilizado, o qual nos referimos por dataset 2, contém 116838 exemplos com 5 classes cada. As classes são descritas pela ontologia DOLCE sendo elas Endurant, Perdurant, Situation, Quality e Abstract. A quantidade de elementos por cada classe assim como a distribuição das classes estão descritas na Tabela 2 e Figura 22. Os detalhes da construção do dataset estão descritos no artigo (JUNIOR et al., 2022).

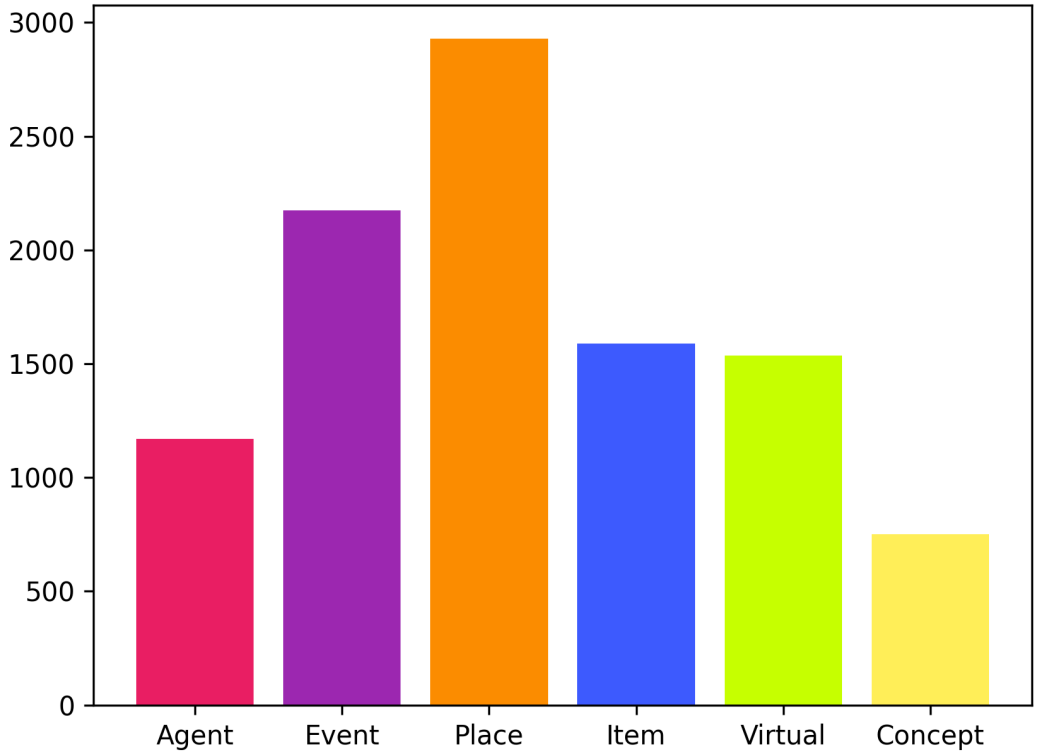

Figura 21 – Distribuição das classes no dataset 1

4.3.1 Arquitetura de classificação

Para que o modelo possa ser utilizado em diversas tarefas, somente a descrição de um conceito é utilizada como entrada. Assim o modelo pode ser utilizado para predizer a classe de qualquer elemento textual. Para realizar esse experimento utilizamos o modelo BERT para gerar embeddings das descrições e uma camada linear para gerar a classe final. Essa arquitetura será comparada com a arquitetura proposta no artigo (JUNIOR et al., 2022)[3] e estão apresentadas na Figura 23. A arquitetura proposta por esse artigo sera referenciada como Modelo 1 e a arquitetura a qual será comparada sera referida como Modelo 2.

O Modelo 2 utiliza labels e descrições informais como entrada com o objetivo de prever em qual classe o exemplo a ser rotulado se encaixa. A arquitetura é baseada em 2 blocos um para codificar as labels e outro para codificar as descrições. O bloco que codifica as labels utiliza embeddings pré-treinados para as palavras e calcula a média dos

[3] Dataset 2 e arquiteturas utilizadas no experimento com referencia ao trabalho (JUNIOR et al., 2022) podem ser encontrados no endereço <https://github.com/BDI-UFRGS/ESWA2021>

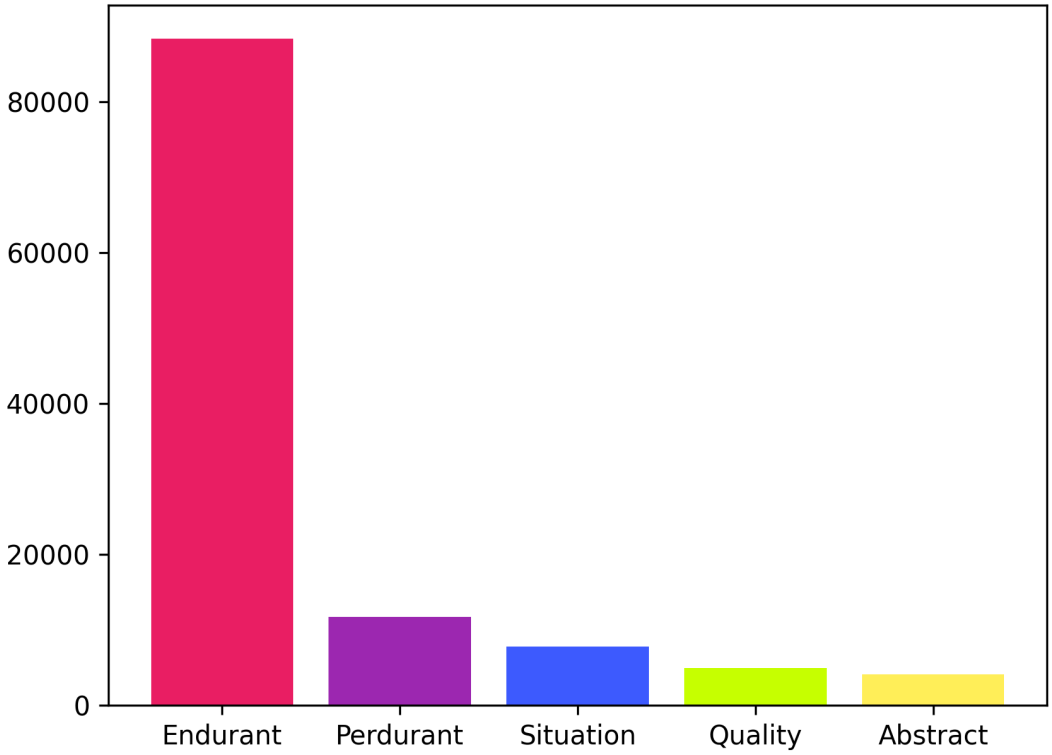

Figura 22 – Distribuição das classes no dataset 2

embeddings para labels de multiplas palavras. O bloco que codifica a descrição utiliza uma camada de embeddings que serão aprendidos e uma LSTM bidirecional para codificar as palavras contidas na descrição. Os embeddings resultantes passam por uma camada Linear e são concatenados para o calculo final das probabilidades através de uma camada linear com softargmax.

O modelo descrito possui uma baixa dimensionalidade e não utiliza mecanismos necessários para construção composicional dos textos como mecanismo de attention e camadas que aprendem representações intermediárias. Além da arquitetura, o uso da label pode não auxiliar na capacidade de distinção do modelo na presença da descrição. Como apresentado na Tabela 3, algumas entidades podem ter a mesma label e dependendo do contexto assumirem classificações distintas, porem a descrição possui um sentido único e é necessária para se desambiguar a classe correta.

Desta forma, é possível descartar as labels e utilizar uma arquitetura que aceita um texto de entrada para a mesma tarefa. Com isso o modelo pode ser utilizado para uma quantidade maior de tarefas como por exemplo prever a classe das labels quando não

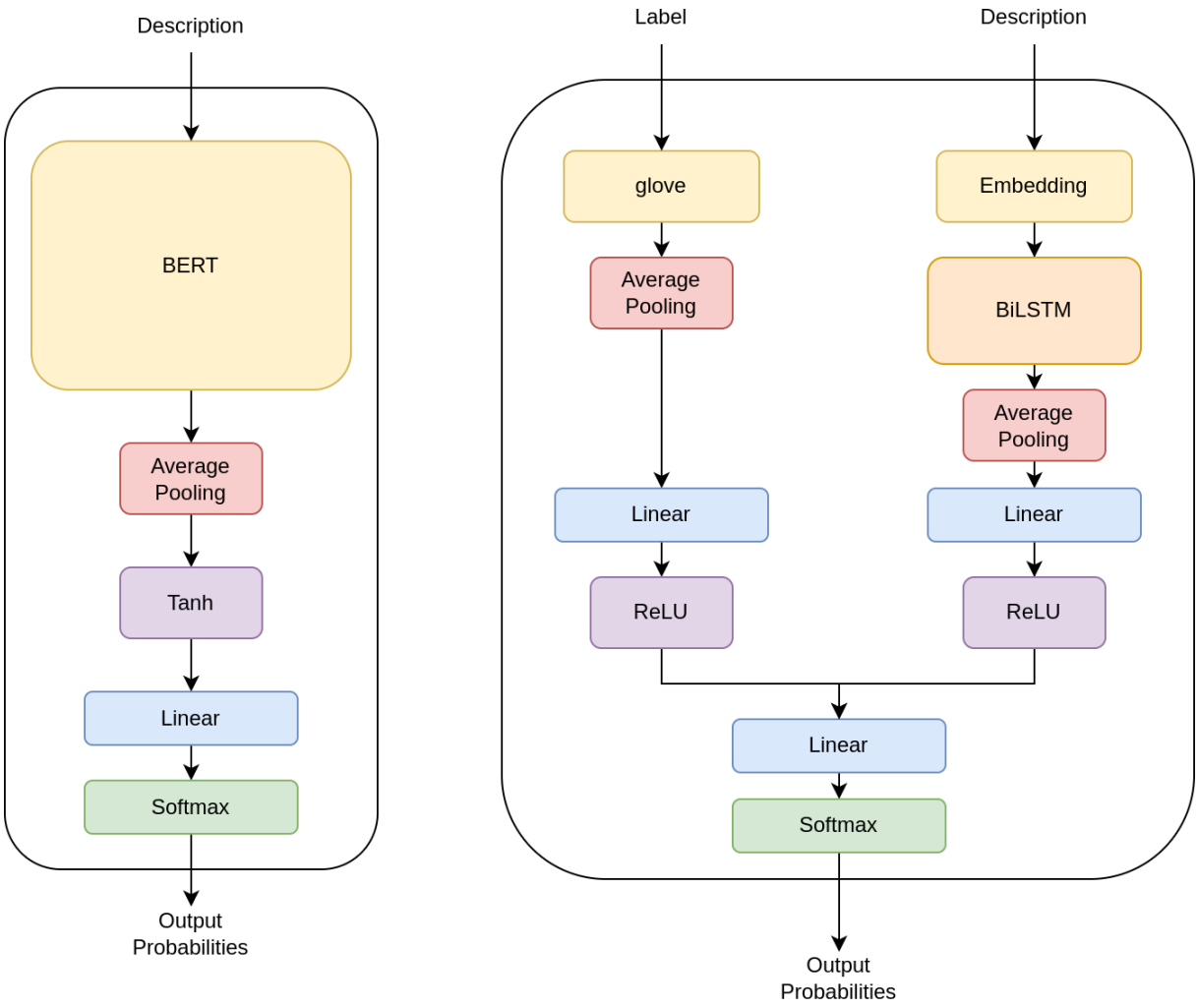

Figura 23 – A esquerda arquitetura proposta e a direita arquitetura de referencia.

rock	Material que consiste em um agregado de minerais como aqueles que constituem a crosta da Terra.	Endurant
rock	Um pedaço de matéria mineral dura e consolidada; ele jogou uma pedra em mim;	Endurant
rock	Um tipo de música originada nos anos 50. Uma mistura de ritmo, blues e musica country.	Perdurant

Tabela 3 – Exemplo de instâncias presentes no Dataset 2.

se possui as descrições ou prever classe das descrições quando não se tem as labels.

Além da escolha de se utilizar apenas um texto como entrada para o modelo, modelos como BERT possuem uma grande quantidade de parâmetros e camadas além de grande capacidade de modelar estruturas da linguagem. Esse tipo de modelo atinge alta performance em diversos tipos de tarefas após fine-tuning. Assim, a arquitetura proposta por esse trabalho utiliza esse modelo para extrair a representação semântica do texto e a utiliza para predizer em qual classe top level tal descrição se encaixa.

4.4 Redução do espaço de matching

Como discutido anteriormente, utilizando a classificação proposta é possível agrupar conceitos em grupos similares separados por cada classe e que não são similares a elementos de outros grupos. Dessa forma, como cada elemento só é comparado com elementos do mesmo grupo, é possível reduzir o espaço de matching. Supondo duas ontologias com x e y entidades respectivamente distribuídas igualmente entre c classes, o número mínimo de comparações pode ser previsto pela Equação 4.1.

$$n_c = \frac{xy}{c} \tag{4.1}$$

Supondo um dataset balanceado e a utilização de c classes, a redução máxima de comparações pode ser calculado com a Equação 4.2. Utilizando as 6 classes a redução máxima esperada para datasets balanceados é de aproximadamente 83% e com 5 classes 80%.

$$r = 1 - \frac{1}{c} \tag{4.2}$$

Tendo como base as classes propostas pela ontologia DOLCE: endurant, perdurant, quality e abstract, percebe-se que apesar da possibilidade de redução no espaço de matching as classes não tendem a gerar uma boa distribuição entre os conceitos que geralmente são interpretados como distintos por exemplo seres vivos e objetos.

5 Experimentos

Neste capítulo será apresentado os resultados e configurações dos experimentos [1] propostos para verificar a validade das hipóteses propostas. O primeiro experimento verifica a performance da arquitetura proposta na tarefa de alinhamento de propriedades sendo avaliado o impacto de cada modificação além da comparação com arquiteturas presentes na competição OEAI no ano de 2021 (POUR et al., 2021). O segundo experimento avalia a performance da arquitetura proposta em comparação com a arquitetura de referência. Nesse experimento as arquiteturas são comparadas no dataset proposto e no dataset de referência. Além disso, o impacto da decisão de utilizar somente a descrição no treino é avaliada junto com a influência na precisão dos modelos em relação a utilização dos tipos. Por ultimo, será avaliado o impacto da utilização dos tipos para reduzir o espaço de alinhamento entre ontologias.

5.1 Alinhamento de propriedades

A arquitetura para alinhamento de propriedades foi testada no dataset Conference. A Tabela 4 contém os resultados da progressão de performance com a adição de cada modificação na arquitetura.

Para esse experimento foi utilizado um threshold de 0.65. As métricas utilizadas são precision, recall e f-measure descritas nas equações 5.1, 5.2 e 5.3 respectivamente. Nas equações, *correto* é o valor de predições corretas geradas pela arquitetura, *tentativas* equivale a $corretos + incorretos$ e *total* é a quantidade de alinhamentos de referência.

$$precision = \frac{correto}{tentativas} \tag{5.1}$$

$$recall = \frac{correto}{total} \tag{5.2}$$

$$f_measure = \frac{2 * (precision * recall)}{precision + recall} \tag{5.3}$$

[1] A implementação utilizada nos experimentos pode ser encontrada em <https://github.com/guihcs/master_degree>

Descrição	Iterações	Precisão	Recall	fm
base	40897	**1.0**	0.28	0.44
similaridade dos embeddings aplicado aos domínios	40897	0.84	0.35	0.49
adição de inversos	40897	0.81	0.39	0.53
similaridade de sentenças aplicada a label das propriedades	40897	0.68	0.41	0.51
remoção de palavras repetidas entre propriedade e target, remoção de adjetivos no target	40897	0.71	0.48	0.57
aumento de confiança entre domínio de propriedades alinhadas	81794	0.73	**0.52**	0.61
restringir um alinhamento para cada domínio	81794	0.83	**0.52**	**0.64**

Tabela 4 – Progressão de performance de acordo com as técnicas.

A arquitetura proposta também foi comparada com os sistemas que participaram da competição OAEI na track Conference no ano de 2021. A configuração do experimento realizada é correspondente aos resultados da modalidade ra1-M2. Resultados comparativos estão apresentados na Tabela 5.

5.2 Classificação top level

Para avaliar a performance da arquitetura proposta, dois experimentos foram realizados. O primeiro no dataset 1 composto pelas labels e comentários dos 10148 exemplos rotulados da ontologia YAGO. Após remover os exemplos sem comentários ou com comentários repetidos o dataset final resulta em 8063 exemplos com 6 classes. As arquiteturas foram comparadas com os classificadores Dicision Tree, Gaussian Bayes Network, Random Forest e Suport Vector Machine. A implementação dos classificadores baseline são retirados da biblioteca sklearn (PEDREGOSA et al., 2011) e utilizados com os parâmetros padrão. As arquiteturas proposta e de referência são baseadas em modelos neurais e foram treinadas em 1 época pois experimentalmente foi a quantidade de melhor desempenho do modelo. Todos os modelos foram avaliados utilizando 10-fold cross

Name	Precision	Recall	F-measure
our	0.83	**0.52**	**0.64**
AML	**1.0**	0.41	0.58
LogMap	0.62	0.28	0.39
GMap	0.56	0.2	0.29
Wikitionary	0.24	0.28	0.26
TOM	0.27	0.24	0.25
ALOD2Vec	0.22	0.3	0.25
LogMapLt	0.24	0.22	0.23
FineTOM	0.24	0.22	0.23
OTMapOnto	0.13	0.48	0.2
edna	0.21	0.11	0.14
StringEquiv	0.07	0.02	0.03

Tabela 5 – Resultado do alinhamento de propriedades em relação aos sistemas que participaram na OAEI 2021

validation e a métrica utilizada é acurácia descrita pela Equação 5.4. O resultado final é a média de todos os folds apresentada na Tabela 6.

$$acc = \frac{correto}{total} \tag{5.4}$$

Name	Acc
DecisionTree	0.25
GaussianBN	0.27
RandomForest	0.40
SVM	0.54
Model2	0.58
Model1	**0.84**

Tabela 6 – Resultados do experimento no dataset 1 com 6 classes

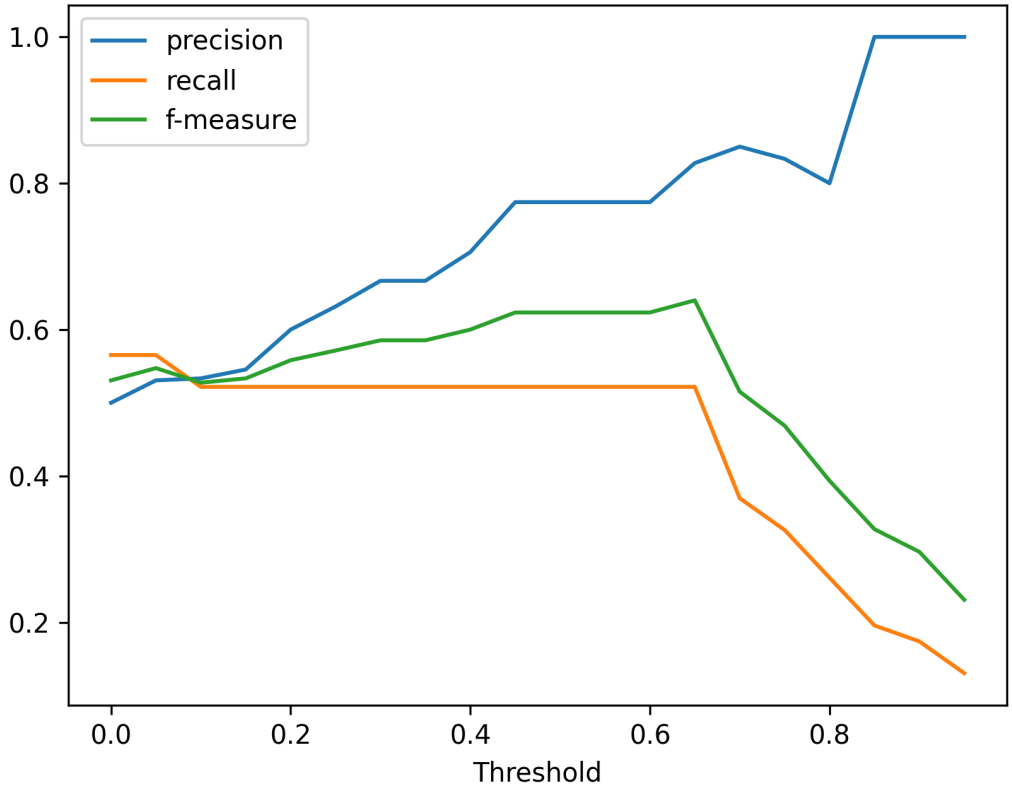

Figura 24 – Comportamento da performance da arquitetura em relação ao threshold.

O segundo experimento foi realizado no dataset 2. Devido ao grande desbalanceamento entre as classes nesse dataset, foi utilizado a técnica de down sampling a qual n exemplos de cada classe são selecionados aleatoriamente sendo n a quantidade de elementos da classe menos representativa. No dataset, a classe menos representativa contém 4035 exemplos resultando em um total de 20175 exemplos com 5 classes. Devido à característica aleatória de seleção dos exemplos, os resultados podem variar a cada nova execução, porém é esperado que a diferença entre os resultados dos classificadores seja proporcional. O experimento foi realizado nas mesmas configurações do dataset 1 e os resultados estão apresentados na Tabela 7.

Para verificar o impacto da utilização apenas da descrição modelo proposto um teste foi realizado as labels de cada dataset após o treino nas descrições. O resultado do teste está descrito na tabela 8. Os resultados mostram boa performance na predição das labels, porém entidades que variam de acordo com o contexto não possuem informação suficiente para que o modelo decida qual a classe referente.

E por ultimo, foi avaliado o impacto da utilização dos tipos top level no alinhamento

Name	Acc
SVM	0.22
GaussianBN	0.24
RandomForest	0.26
DecisionTree	0.34
Model2	0.38
Model1	**0.56**

Tabela 7 – Resultados do experimento no dataset 2 com 5 classes

Dataset	Acc
Dataset 1	0.79
Dataset 2	0.75

Tabela 8 – Resultados na redução do espaço de matching em propriedades

entre classes utilizando a distância de levenshtein em um intervalo de thresholds. Os thresholds variam entre 0 e 0.95 com intervalo de 0.05. Os resultados para precisão estão apresentados na Figura 25, os resultados para recall na Figura 26 e para F-Measure na Figura 27.

Analisando o gráfico referente à precisão é possível observar um aumento na precisão em thresholds mais baixos. Esse aumento ocorre devido aos tipos rejeitarem alinhamentos como Review e Reviewer os quais tem alta similaridade léxica porém não são entidades similares visto que Reviewer é um Agent e Review pode ser considerado como Virtual ou Item. Devido a baixa granularidade dos tipos (ex. não diferenciar entre humanos e outros animais) a precisão no alinhamento ainda se mantém próxima a precisão obtida com o uso de métricas léxicas.

Em relação ao recall, é possível perceber que com threshold 0, o qual deveria incluir todos os alinhamentos, o modelo não atinge os 100%. Isso ocorre devido a incapacidade do modelo distinguir certas entidades devido a falta de contexto ou mais exemplos de treino.

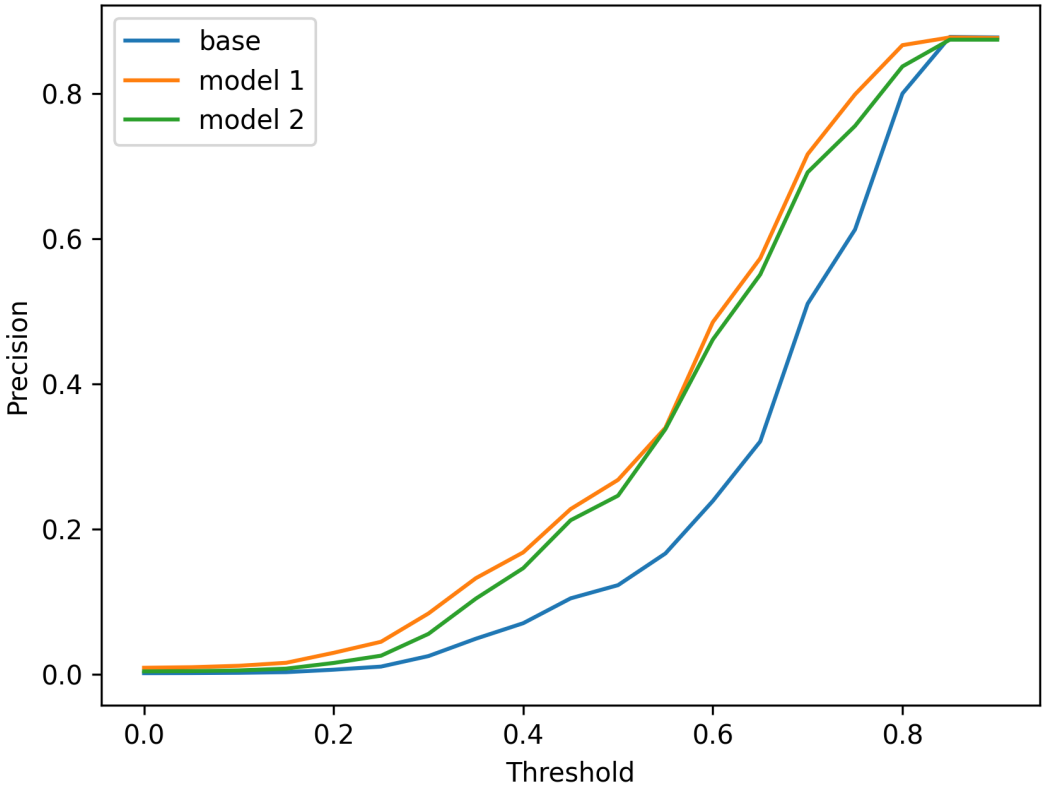

Figura 25 – Precisão

5.3 Redução do espaço de matching

Para avaliar a capacidade descritiva das classes propostas, a arquitetura proposta foi avaliada na redução de comparações em classes e propriedades do dataset Conference. Como a quantidade de comentários nesse dataset é desprezível, o modelo 2 não pode ser executado. A comparação foi feita utilizando a similaridade de Levenshtein com threshold de 0.85. Os resultados esperados são um aumento teórico em precisão mantendo o mesmo recall e com um numero menor de comparações. O resultado para a avaliação nas classes está disposto na Tabela 9 e propriedades na Tabela 10.

Name	Iterations	Reduction	Precision	Recall	F-measure
Base	148853	0%	0.88	0.56	0.68
Model1 dataset 1	**25304**	83%	0.88	0.55	0.68
Model1 datast 2	48616	67%	0.88	0.55	0.68

Tabela 9 – Resultados na redução do espaço de matching em classes

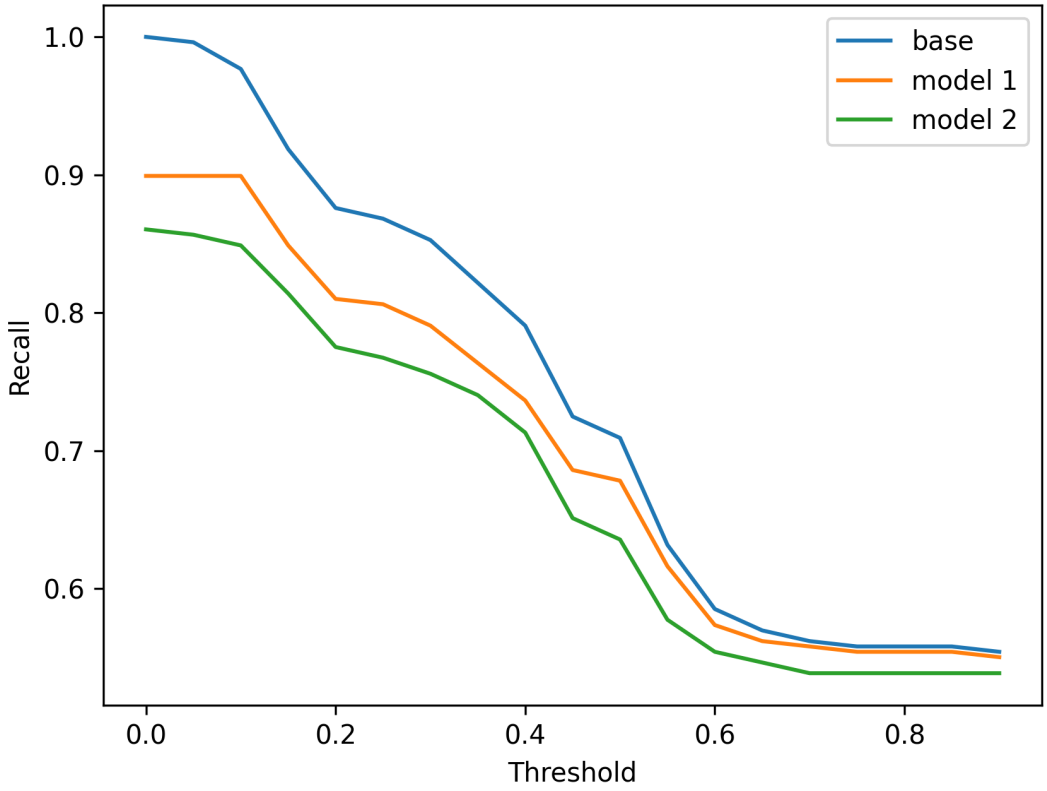

Figura 26 – Recall

Name	Iterations	Reduction	Precision	Recall	F-measure
Base	81794	0%	0.83	0.52	0.64
Model1 dataset1	**13414**	83%	0.86	0.41	0.56
Model1 datast2	15802	81%	0.86	0.41	0.56

Tabela 10 – Resultados na redução do espaço de matching em propriedades

Os resultados experimentais indicam que o grupo de classes propostos tende a atingir a quantidade de redução próxima ao máximo esperado mantendo os valores de precision, recall e f-measure próximos aos valores base. Uma das causas da performance inferior das classes derivadas da DOLCE é a baixa distribuição entre os conceitos. Aparentemente existe uma sensitividade maior na distinção entre objetos físicos do que conceitos abstratos. Desta forma, a classificação proposta atinge um maior grau de separabilidade ao distinguir melhor conceitos físicos como seres vivos e objetos.

Outra observação é a queda de performance quando a técnica é aplicada às

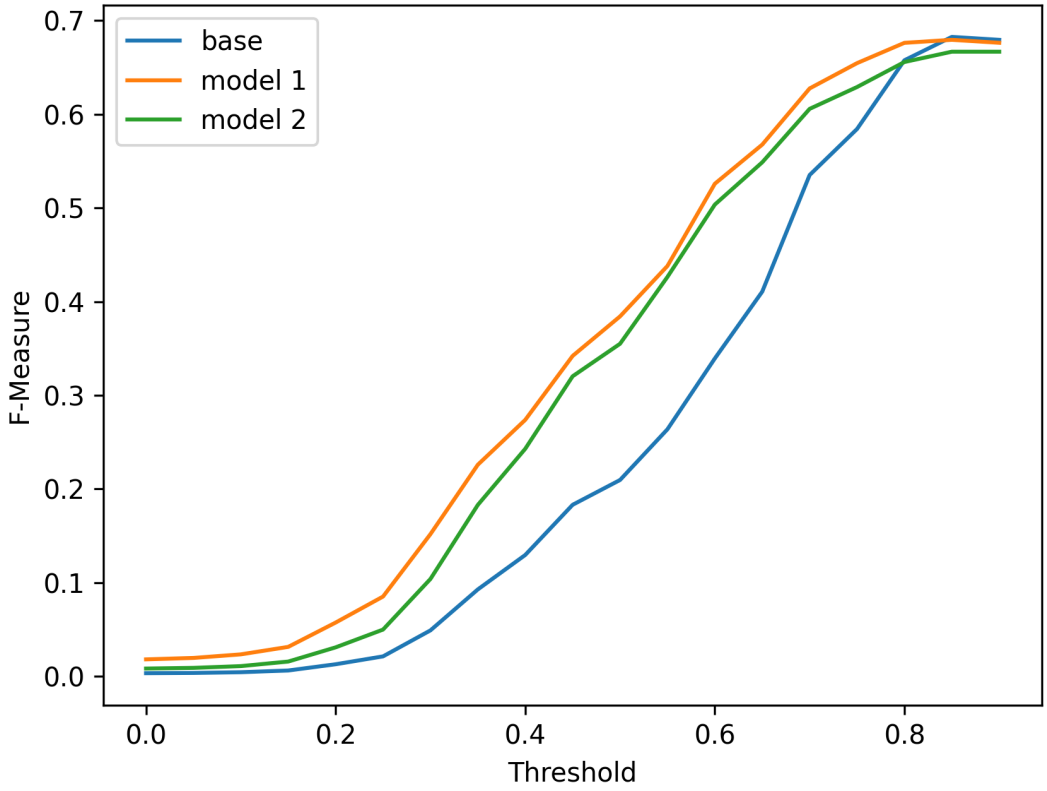

Figura 27 – F-Measure

propriedades quando comparada às classes. Essa queda na performance ocorre devido a imperfeições na acurácia do modelo utilizado. Desta forma, entidades que deveriam pertencer à mesma classe são agrupadas em classes diferentes impossibilitando a comparação e aplicação das técnicas propostas nas seções anteriores.

6 Conclusão e Trabalhos Futuros

O trabalho realizado apresentou propostas para melhoria no estado da arte. Os resultados indicam que as arquiteturas propostas para o alinhamento de propriedades e predição de classes top level superam a performance das arquiteturas comparadas. Além desses resultados, a utilização de classes top level para a redução no espaço de matching se mostrou uma técnica eficiente e que pode ser utilizada como etapa inicial de alinhamento em diversos sistemas. Apesar dessa boa redução, as classes DOLCE tendem a agrupar conceitos físicos em uma única classe. Porém, existe uma tendência a ter uma maior sensibilidade na distinção entre conceitos físicos do que conceitos abstratos. Dessa forma, a proposta de classes que acompanhem essa expectativa podem auxiliar ainda mais no cálculo de similaridades. Os resultados experimentais indicam que esse tipo de classificação permite reduzir boa parte dos alinhamentos possíveis mantendo as métricas de precisão e recall próximas ao resultado original. Apesar dessas melhorias algumas direções ainda podem ser exploradas com o intuito de melhorar ainda mais os resultados. Nas próximas seções são apresentados trabalhos futuros.

Apesar da melhora nos resultados apresentado pela arquitetura, trabalhos futuros são possíveis. Um deles é a utilização de embeddings para calculo de similaridade descartando a necessidade dos embeddings gerados pelas métricas tf-idf. Essas métricas coletam informação global sobre os dados para gerar vetores para os indivíduos. Além de consumir grande quantidade de recurso para ontologias grandes, a métrica deve ser recalculada para cada ontologia. Outro problema é a hipótese de similaridade baseada em estatística assumida por esses tipos de métricas. Essa hipótese assume que termos infrequentes presentes em dois conceitos tendem a ser alinhados. Esse tipo de hipótese não traz um entendimento do conceito de similaridade e pode não se adequar a todos os casos.

Outro ponto de possível melhoria é a pesquisa sobre modelos de alinhamento puramente neurais. Assim, se exclui a necessidade de implementar regras manualmente e permitindo que o modelo decida quais regras utilizar com base no contexto e informações presentes.

O dataset utilizado contém poucos exemplos e não possui um balanceamento uniforme entre as classes. Expandir o dataset pode ajudar melhorar a representação do modelo e assim obter melhores resultados. Outra pesquisa importante é verificar o

quão intuitivas são as classes e descrever quais atributos geralmente são utilizados para classificar os conceitos.

Como expandir a granularidade dos tipos e como utilizar a métrica a fim de aumentar o recall. Um exemplo da utilização dessas classes para uma avaliação de similaridade refinada é a comparação entre Membro de Organização e Membro de Conferência. Apesar da classificação geral atribuir Agent para os dois, a relação entre as palavras membro - organização e membro - conferência são diferentes pois organização pode ser classificado como Agent e conferência como Event. Assim, a análise fina dos conceitos na mesma frase podem auxiliar na construção de uma similaridade ainda mais significante.

Apesar do bom resultado em prever a classificação das labels das entidades, alguns erros do classificador como classificar Silver Supporter como Item mostra que a utilização de um contexto é necessária para se atingir uma acurácia maior. Nesse contexto, Silver Supporter é considerado um Agent e se refere a um patrocinador. Essa informação pode estar presente no contexto de uma ontologia e uma arquitetura capaz de utilizar esse contexto pode obter melhores resultados.

Referências

AGGARWAL, C. C. **Machine Learning for Text, Second Edition**. Springer, 2022. ISBN 978-3-030-96622-5. Disponível em: <https://doi.org/10.1007/978-3-030-96623-2>.

AIZAWA, A. N. An information-theoretic perspective of tf-idf measures. **Inf. Process. Manag.**, v. 39, n. 1, p. 45–65, 2003. Disponível em: <https://doi.org/10.1016/S0306-4573(02)00021-3>.

BENGIO, Y.; LECUN, Y.; HINTON, G. E. Deep learning for AI. **Commun. ACM**, v. 64, n. 7, p. 58–65, 2021. Disponível em: <https://doi.org/10.1145/3448250>.

BISHOP, C. M. **Pattern recognition and machine learning, 5th Edition**. Springer, 2007. (Information science and statistics). ISBN 9780387310732. Disponível em: <https://www.worldcat.org/oclc/71008143>.

BOHNET, B.; MCDONALD, R. T.; SIMÕES, G.; ANDOR, D.; PITLER, E.; MAYNEZ, J. Morphosyntactic tagging with a meta-bilstm model over context sensitive token encodings. In: GUREVYCH, I.; MIYAO, Y. (Ed.). **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers**. Association for Computational Linguistics, 2018. p. 2642–2652. Disponível em: <https://aclanthology.org/P18-1246/>.

BORGO, S.; FERRARIO, R.; GANGEMI, A.; GUARINO, N.; MASOLO, C.; PORELLO, D.; SANFILIPPO, E. M.; VIEU, L. DOLCE: A descriptive ontology for linguistic and cognitive engineering. **Appl. Ontology**, v. 17, n. 1, p. 45–69, 2022. Disponível em: <https://doi.org/10.3233/AO-210259>.

BROWN, T. B.; MANN, B.; RYDER, N.; SUBBIAH, M.; KAPLAN, J.; DHARIWAL, P.; NEELAKANTAN, A.; SHYAM, P.; SASTRY, G.; ASKELL, A.; AGARWAL, S.; HERBERT-VOSS, A.; KRUEGER, G.; HENIGHAN, T.; CHILD, R.; RAMESH, A.; ZIEGLER, D. M.; WU, J.; WINTER, C.; HESSE, C.; CHEN, M.; SIGLER, E.; LITWIN, M.; GRAY, S.; CHESS, B.; CLARK, J.; BERNER, C.; MCCANDLISH, S.; RADFORD, A.; SUTSKEVER, I.; AMODEI, D. Language models are few-shot learners. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. [s.n.], 2020. Disponível em: <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html>.

CELEBI, M. E.; AYDIN, K. **Unsupervised learning algorithms**. [S.l.]: Springer, 2016.

CHANDRASEKARAN, B.; JOSEPHSON, J. R.; BENJAMINS, V. R. What are ontologies, and why do we need them? **IEEE Intell. Syst.**, v. 14, n. 1, p. 20–26, 1999. Disponível em: <https://doi.org/10.1109/5254.747902>.

CHEATHAM, M.; HITZLER, P. String similarity metrics for ontology alignment. In: SPRINGER. **International semantic web conference**. [S.l.], 2013. p. 294–309.

CHEATHAM, M.; HITZLER, P. et al. The properties of property alignment. In: **OM**. [S.l.: s.n.], 2014. p. 13–24.

CHEN, J.; JIMÉNEZ-RUIZ, E.; HORROCKS, I.; ANTONYRAJAH, D.; HADIAN, A.; LEE, J. Augmenting ontology alignment by semantic embedding and distant supervision. In: VERBORGH, R.; HOSE, K.; PAULHEIM, H.; CHAMPIN, P.; MALESHKOVA, M.; CORCHO, Ó.; RISTOSKI, P.; ALAM, M. (Ed.). **The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings**. Springer, 2021. (Lecture Notes in Computer Science, v. 12731), p. 392–408. Disponível em: <https://doi.org/10.1007/978-3-030-77385-4_23>.

CHO, K.; MERRIENBOER, B. van; BAHDANAU, D.; BENGIO, Y. On the properties of neural machine translation: Encoder-decoder approaches. In: WU, D.; CARPUAT, M.; CARRERAS, X.; VECCHI, E. M. (Ed.). **Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014**. Association for Computational Linguistics, 2014. p. 103–111. Disponível em: <https://aclanthology.org/W14-4012/>.

CHO, K.; MERRIENBOER, B. van; GÜLÇEHRE, Ç.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: MOSCHITTI, A.; PANG, B.; DAELEMANS, W. (Ed.). **Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL**. ACL, 2014. p. 1724–1734. Disponível em: <https://doi.org/10.3115/v1/d14-1179>.

CONNEAU, A.; SCHWENK, H.; BARRAULT, L.; LECUN, Y. Very deep convolutional networks for text classification. In: LAPATA, M.; BLUNSOM, P.; KOLLER, A. (Ed.). **Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers**. Association for Computational Linguistics, 2017. p. 1107–1116. Disponível em: <https://doi.org/10.18653/v1/e17-1104>.

DECKER, S.; MELNIK, S.; HARMELEN, F. van; FENSEL, D.; KLEIN, M. C. A.; BROEKSTRA, J.; ERDMANN, M.; HORROCKS, I. The semantic web: The roles of XML and RDF. **IEEE Internet Comput.**, v. 4, n. 5, p. 63–74, 2000. Disponível em: <https://doi.org/10.1109/4236.877487>.

DEVLIN, J.; CHANG, M.; LEE, K.; TOUTANOVA, K. BERT: pre-training of deep bidirectional transformers for language understanding. In: BURSTEIN, J.; DORAN, C.; SOLORIO, T. (Ed.). **Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)**. Association for Computational Linguistics, 2019. p. 4171–4186. Disponível em: <https://doi.org/10.18653/v1/n19-1423>.

EUZENAT, J.; SHVAIKO, P. **Ontology Matching, Second Edition**. [S.l.]: Springer, 2013. ISBN 978-3-642-38720-3.

FARIA, D.; PESQUITA, C.; SANTOS, E.; PALMONARI, M.; CRUZ, I. F.; COUTO, F. M. The agreementmakerlight ontology matching system. In: MEERSMAN, R.; PANETTO, H.; DILLON, T. S.; EDER, J.; BELLAHSENE, Z.; RITTER, N.; LEENHEER, P. D.; DOU, D. (Ed.). **On the Move to Meaningful Internet Systems: OTM 2013 Conferences - Confederated International Conferences: CoopIS, DOA-Trusted**

Cloud, and ODBASE 2013, Graz, Austria, September 9-13, 2013. Proceedings. Springer, 2013. (Lecture Notes in Computer Science, v. 8185), p. 527–541. Disponível em: <https://doi.org/10.1007/978-3-642-41030-7_38>.

GUHA, R. V.; BRICKLEY, D.; MACBETH, S. Schema.org: evolution of structured data on the web. **Commun. ACM**, v. 59, n. 2, p. 44–51, 2016. Disponível em: <https://doi.org/10.1145/2844544>.

HECKERMAN, D.; GEIGER, D. Learning bayesian networks: A unification for discrete and gaussian domains. **CoRR**, abs/1302.4957, 2013. Disponível em: <http://arxiv.org/abs/1302.4957>.

HENDLER, J.; BERNERS-LEE, T. From the semantic web to social machines: A research challenge for AI on the world wide web. **Artif. Intell.**, v. 174, n. 2, p. 156–161, 2010. Disponível em: <https://doi.org/10.1016/j.artint.2009.11.010>.

HENZE, N.; DOLOG, P.; NEJDL, W. Reasoning and ontologies for personalized e-learning in the semantic web. **J. Educ. Technol. Soc.**, v. 7, n. 4, p. 82–97, 2004. Disponível em: <https://www.j-ets.net/ETS/journals/7_4/10.pdf>.

HITZLER, P.; KRÖTZSCH, M.; PARSIA, B.; PATEL-SCHNEIDER, P. F.; RUDOLPH, S. et al. Owl 2 web ontology language primer. **W3C recommendation**, v. 27, n. 1, p. 123, 2009.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HOWARD, J.; RUDER, S. Universal language model fine-tuning for text classification. In: GUREVYCH, I.; MIYAO, Y. (Ed.). **Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers**. Association for Computational Linguistics, 2018. p. 328–339. Disponível em: <https://aclanthology.org/P18-1031/>.

JI, S.; PAN, S.; CAMBRIA, E.; MARTTINEN, P.; YU, P. S. A survey on knowledge graphs: Representation, acquisition, and applications. **IEEE Trans. Neural Networks Learn. Syst.**, v. 33, n. 2, p. 494–514, 2022. Disponível em: <https://doi.org/10.1109/TNNLS.2021.3070843>.

JIMÉNEZ-RUIZ, E.; AGIBETOV, A.; SAMWALD, M.; CROSS, V. Breaking-down the ontology alignment task with a lexical index and neural embeddings. **CoRR**, abs/1805.12402, 2018. Disponível em: <http://arxiv.org/abs/1805.12402>.

JIMÉNEZ-RUIZ, E.; GRAU, B. C. Logmap: Logic-based and scalable ontology matching. In: AROYO, L.; WELTY, C.; ALANI, H.; TAYLOR, J.; BERNSTEIN, A.; KAGAL, L.; NOY, N. F.; BLOMQVIST, E. (Ed.). **The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I**. Springer, 2011. (Lecture Notes in Computer Science, v. 7031), p. 273–288. Disponível em: <https://doi.org/10.1007/978-3-642-25073-6_18>.

JOHNSON, R.; ZHANG, T. Convolutional neural networks for text categorization: Shallow word-level vs. deep character-level. **arXiv preprint arXiv:1609.00718**, 2016.

JUNIOR, A. G. L.; CARBONERA, J. L.; SCHIMDT, D.; ABEL, M. Predicting the top-level ontological concepts of domain entities using word embeddings, informal definitions, and deep learning. **Expert Syst. Appl.**, v. 203, p. 117291, 2022. Disponível em: <https://doi.org/10.1016/j.eswa.2022.117291>.

JURGENSON, T.; MANSOUR, Y. Learning decision trees with stochastic linear classifiers. In: JANOOS, F.; MOHRI, M.; SRIDHARAN, K. (Ed.). **Algorithmic Learning Theory, ALT 2018, 7-9 April 2018, Lanzarote, Canary Islands, Spain**. PMLR, 2018. (Proceedings of Machine Learning Research, v. 83), p. 489–528. Disponível em: <http://proceedings.mlr.press/v83/jurgenson18a.html>.

KADHIM, A. I. Survey on supervised machine learning techniques for automatic text classification. **Artif. Intell. Rev.**, v. 52, n. 1, p. 273–292, 2019. Disponível em: <https://doi.org/10.1007/s10462-018-09677-1>.

KHANUM, M.; MAHBOOB, T.; IMTIAZ, W.; GHAFOOR, H. A.; SEHAR, R. A survey on unsupervised machine learning algorithms for automation, classification and maintenance. **International Journal of Computer Applications**, Citeseer, v. 119, n. 13, 2015.

KIM, Y.; JERNITE, Y.; SONTAG, D.; RUSH, A. M. Character-aware neural language models. In: **Thirtieth AAAI conference on artificial intelligence**. [S.l.: s.n.], 2016.

LECUN, Y.; BENGIO, Y. et al. Convolutional networks for images, speech, and time series. **The handbook of brain theory and neural networks**, Cambridge, MA USA, v. 3361, n. 10, p. 1995, 1995.

LIU, X.; HE, P.; CHEN, W.; GAO, J. Improving multi-task deep neural networks via knowledge distillation for natural language understanding. **CoRR**, abs/1904.09482, 2019. Disponível em: <http://arxiv.org/abs/1904.09482>.

LUONG, T.; PHAM, H.; MANNING, C. D. Effective approaches to attention-based neural machine translation. In: MÀRQUEZ, L.; CALLISON-BURCH, C.; SU, J.; PIGHIN, D.; MARTON, Y. (Ed.). **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015**. The Association for Computational Linguistics, 2015. p. 1412–1421. Disponível em: <https://doi.org/10.18653/v1/d15-1166>.

LUOTOLAHTI, J.; KANERVA, J.; LAIPPALA, V.; PYYSALO, S.; GINTER, F. Towards universal web parsebanks. In: **Proceedings of the third international conference on Dependency Linguistics (Depling 2015)**. [S.l.: s.n.], 2015. p. 211–220.

MA, X.; WANG, Z.; NG, P.; NALLAPATI, R.; XIANG, B. Universal text representation from bert: an empirical study. **arXiv preprint arXiv:1910.07973**, 2019.

MATSUO, Y.; LECUN, Y.; SAHANI, M.; PRECUP, D.; SILVER, D.; SUGIYAMA, M.; UCHIBE, E.; MORIMOTO, J. Deep learning, reinforcement learning, and world models. **Neural Networks**, v. 152, p. 267–275, 2022. Disponível em: <https://doi.org/10.1016/j.neunet.2022.03.037>.

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

MIKOLOV, T.; KARAFIÁT, M.; BURGET, L.; CERNOCKÝ, J.; KHUDANPUR, S. Recurrent neural network based language model. In: KOBAYASHI, T.; HIROSE, K.; NAKAMURA, S. (Ed.). **INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010**. ISCA, 2010. p. 1045–1048. Disponível em: <http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html>.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: BURGES, C. J. C.; BOTTOU, L.; GHAHRAMANI, Z.; WEINBERGER, K. Q. (Ed.). **Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States**. [s.n.], 2013. p. 3111–3119. Disponível em: <https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html>.

MRINI, K.; DERNONCOURT, F.; TRAN, Q. H.; BUI, T.; CHANG, W.; NAKASHOLE, N. Rethinking self-attention: Towards interpretability in neural parsing. In: COHN, T.; HE, Y.; LIU, Y. (Ed.). **Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020**. Association for Computational Linguistics, 2020. (Findings of ACL, EMNLP 2020), p. 731–742. Disponível em: <https://doi.org/10.18653/v1/2020.findings-emnlp.65>.

NOBLE, W. S. What is a support vector machine? **Nature biotechnology**, Nature Publishing Group, v. 24, n. 12, p. 1565–1567, 2006.

PAN, S. J.; YANG, Q. A survey on transfer learning. **IEEE Trans. Knowl. Data Eng.**, v. 22, n. 10, p. 1345–1359, 2010. Disponível em: <https://doi.org/10.1109/TKDE.2009.191>.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

POUR, M. A. N.; ALGERGAWY, A.; AMARDEILH, F.; AMINI, R.; FALLATAH, O.; FARIA, D.; FUNDULAKI, I.; HARROW, I.; HERTLING, S.; HITZLER, P.; HUSCHKA, M.; IBANESCU, L.; JIMÉNEZ-RUIZ, E.; KARAM, N.; LAADHAR, A.; LAMBRIX, P.; LI, H.; LI, Y.; MICHEL, F.; NASR, E.; PAULHEIM, H.; PESQUITA, C.; PORTISCH, J.; ROUSSEY, C.; SAVETA, T.; SHVAIKO, P.; SPLENDIANI, A.; TROJAHN, C.; VATASCINOVÁ, J.; YAMAN, B.; ZAMAZAL, O.; ZHOU, L. Results of the ontology alignment evaluation initiative 2021. In: SHVAIKO, P.; EUZENAT, J.; JIMÉNEZ-RUIZ, E.; HASSANZADEH, O.; TROJAHN, C. (Ed.). **Proceedings of the 16th International Workshop on Ontology Matching co-located with the 20th International Semantic Web Conference (ISWC 2021), Virtual conference, October 25, 2021**. CEUR-WS.org, 2021. (CEUR Workshop Proceedings, v. 3063), p. 62–108. Disponível em: <http://ceur-ws.org/Vol-3063/oaei21_paper0.pdf>.

POUYANFAR, S.; SADIQ, S.; YAN, Y.; TIAN, H.; TAO, Y.; REYES, M. E. P.; SHYU, M.; CHEN, S.; IYENGAR, S. S. A survey on deep learning: Algorithms, techniques, and

applications. **ACM Comput. Surv.**, v. 51, n. 5, p. 92:1–92:36, 2019. Disponível em: <https://doi.org/10.1145/3234150>.

RAMESH, A.; DHARIWAL, P.; NICHOL, A.; CHU, C.; CHEN, M. Hierarchical text-conditional image generation with CLIP latents. **CoRR**, abs/2204.06125, 2022. Disponível em: <https://doi.org/10.48550/arXiv.2204.06125>.

ROMBACH, R.; BLATTMANN, A.; LORENZ, D.; ESSER, P.; OMMER, B. High-resolution image synthesis with latent diffusion models. In: **IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022**. IEEE, 2022. p. 10674–10685. Disponível em: <https://doi.org/10.1109/CVPR52688.2022.01042>.

SAHARIA, C.; CHAN, W.; SAXENA, S.; LI, L.; WHANG, J.; DENTON, E.; GHASEMIPOUR, S. K. S.; AYAN, B. K.; MAHDAVI, S. S.; LOPES, R. G.; SALIMANS, T.; HO, J.; FLEET, D. J.; NOROUZI, M. Photorealistic text-to-image diffusion models with deep language understanding. **CoRR**, abs/2205.11487, 2022. Disponível em: <https://doi.org/10.48550/arXiv.2205.11487>.

SAHLGREN, M. The distributional hypothesis. **Italian Journal of Disability Studies**, v. 20, p. 33–53, 2008.

SEN, P. C.; HAJRA, M.; GHOSH, M. Supervised classification algorithms in machine learning: A survey and review. In: **Emerging technology in modelling and graphics**. [S.l.]: Springer, 2020. p. 99–111.

SHAHZAD, M.; AMIN, A.; ESTEVES, D.; NGOMO, A. N. Inferner: an attentive model leveraging the sentence-level information for named entity recognition in microblogs. In: BELL, E.; KESHTKAR, F. (Ed.). **Proceedings of the Thirty-Fourth International Florida Artificial Intelligence Research Society Conference, North Miami Beach, Florida, USA, May 17-19, 2021**. [s.n.], 2021. Disponível em: <https://doi.org/10.32473/flairs.v34i1.128538>.

SHEN, W.; WANG, J.; HAN, J. Entity linking with a knowledge base: Issues, techniques, and solutions. **IEEE Trans. Knowl. Data Eng.**, v. 27, n. 2, p. 443–460, 2015. Disponível em: <https://doi.org/10.1109/TKDE.2014.2327028>.

SILVER, D.; HUBERT, T.; SCHRITTWIESER, J.; ANTONOGLOU, I.; LAI, M.; GUEZ, A.; LANCTOT, M.; SIFRE, L.; KUMARAN, D.; GRAEPEL, T. et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. **Science**, American Association for the Advancement of Science, v. 362, n. 6419, p. 1140–1144, 2018.

SILVER, D.; SCHRITTWIESER, J.; SIMONYAN, K.; ANTONOGLOU, I.; HUANG, A.; GUEZ, A.; HUBERT, T.; BAKER, L.; LAI, M.; BOLTON, A.; CHEN, Y.; LILLICRAP, T. P.; HUI, F.; SIFRE, L.; DRIESSCHE, G. van den; GRAEPEL, T.; HASSABIS, D. Mastering the game of go without human knowledge. **Nat.**, v. 550, n. 7676, p. 354–359, 2017. Disponível em: <https://doi.org/10.1038/nature24270>.

STRAKA, M.; STRAKOVÁ, J. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with udpipe. In: HAJIC, J.; ZEMAN, D. (Ed.). **Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies,**

Vancouver, Canada, August 3-4, 2017. Association for Computational Linguistics, 2017. p. 88–99. Disponível em: <https://doi.org/10.18653/v1/K17-3009>.

TANON, T. P.; WEIKUM, G.; SUCHANEK, F. M. YAGO 4: A reason-able knowledge base. In: HARTH, A.; KIRRANE, S.; NGOMO, A. N.; PAULHEIM, H.; RULA, A.; GENTILE, A. L.; HAASE, P.; COCHEZ, M. (Ed.). **The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings**. Springer, 2020. (Lecture Notes in Computer Science, v. 12123), p. 583–596. Disponível em: <https://doi.org/10.1007/978-3-030-49461-2_34>.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L.; POLOSUKHIN, I. Attention is all you need. In: GUYON, I.; LUXBURG, U. von; BENGIO, S.; WALLACH, H. M.; FERGUS, R.; VISHWANATHAN, S. V. N.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA**. [s.n.], 2017. p. 5998–6008. Disponível em: <https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>.

WANG, W.; WEI, F.; DONG, L.; BAO, H.; YANG, N.; ZHOU, M. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). **Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual**. [s.n.], 2020. Disponível em: <https://proceedings.neurips.cc/paper/2020/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>.

WANG, X.; JIANG, Y.; BACH, N.; WANG, T.; HUANG, Z.; HUANG, F.; TU, K. Automated concatenation of embeddings for structured prediction. In: ZONG, C.; XIA, F.; LI, W.; NAVIGLI, R. (Ed.). **Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021**. Association for Computational Linguistics, 2021. p. 2643–2660. Disponível em: <https://doi.org/10.18653/v1/2021.acl-long.206>.

_____. Improving named entity recognition by external context retrieving and cooperative learning. In: ZONG, C.; XIA, F.; LI, W.; NAVIGLI, R. (Ed.). **Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021**. Association for Computational Linguistics, 2021. p. 1800–1812. Disponível em: <https://doi.org/10.18653/v1/2021.acl-long.142>.

WU, L. Y.; FISCH, A.; CHOPRA, S.; ADAMS, K.; BORDES, A.; WESTON, J. Starspace: Embed all the things! In: MCILRAITH, S. A.; WEINBERGER, K. Q. (Ed.). **Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018**. AAAI Press, 2018. p.

5569–5577. Disponível em: <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16998>.

YANG, Z.; DAI, Z.; YANG, Y.; CARBONELL, J. G.; SALAKHUTDINOV, R.; LE, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. In: WALLACH, H. M.; LAROCHELLE, H.; BEYGELZIMER, A.; D'ALCHÉ-BUC, F.; FOX, E. B.; GARNETT, R. (Ed.). **Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada**. [s.n.], 2019. p. 5754–5764. Disponível em: <https://proceedings.neurips.cc/paper/2019/hash/dc6a7e655d7e5840e66733e9ee67cc69-Abstract.html>.

ZHANG, Y.; JIN, R.; ZHOU, Z. Understanding bag-of-words model: a statistical framework. **Int. J. Mach. Learn. Cybern.**, v. 1, n. 1-4, p. 43–52, 2010. Disponível em: <https://doi.org/10.1007/s13042-010-0001-0>.

ZHENG, J.; LIU, Y.; GE, Z. Dynamic ensemble selection based improved random forests for fault classification in industrial processes. **IFAC J. Syst. Control.**, v. 20, p. 100189, 2022. Disponível em: <https://doi.org/10.1016/j.ifacsc.2022.100189>.

ZHUANG, F.; QI, Z.; DUAN, K.; XI, D.; ZHU, Y.; ZHU, H.; XIONG, H.; HE, Q. A comprehensive survey on transfer learning. **Proc. IEEE**, v. 109, n. 1, p. 43–76, 2021. Disponível em: <https://doi.org/10.1109/JPROC.2020.3004555>.