TÉCNICO LISBOA

Neural Methods for Biomedical Synonym Discovery and Concept Alignment

Leonor Clode Silva Jardim Fernandes

Thesis to obtain the Master of Science Degree in

Biomedical Engineering

Supervisors: Prof. Bruno Emanuel Da Graça Martins
Dr. Daniel Pedro de Jesus Faria

Examination Committee

Chairperson: Prof. Ana Luísa Nobre Fred
Supervisor: Prof. Bruno Emanuel Da Graça Martins
Members of the Committee: Dr. André Francisco Martins Lamúrias

October 2021

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

Firstly, I would like to acknowledge my dissertation supervisors Prof. Bruno Martins, Prof. Daniel Faria and Prof. Cátia Pesquita for the constant guidance and patience during the elaboration of this dissertation. Without their support and insight this work would have never been possible and for that, I am deeply grateful.

The last few years in Instituto Superior Técnico would not have been as successful without a handful of people to whom I would also like to express appreciation. Mada and Mila, who since day one proved that Técnico would be more than classes and grades, there's no one better to have lived it *mamma mia* style with. Rosa, whose exchanged phone calls while commuting were in so many moments an unconditional guidance and support. Not forgetting, of course, Pinho, Lopes, Sofia C. and Carlota, I would have not made it without the random dinners, weekends or checking in messages. Sofia G. and Sofia A. for always making time for coffee, and never getting tired of listening whatever the challenge. For all the other friends who were part of the different projects that impacted my Tecnico's years (in AHEAD, NEBM, CPMEBM or MCP) thank you for making me believe that everything can be done and that a burden that one choses is not felt (*Quem corre por gosto não cansa!*).

To my parents and sister I owe the greatest thank you of all, for standing by me and teaching me lessons no one else could. Ritinha, with whom I can always travel back in time, thank you for allowing me to be goofy and keeping my mental sanity in check. To my father, who always believes, whatever the crazy idea or project, thank you for showing me how to dream and that no dream is small or big enough to pursue. To my mother, who understands me better than anyone, thank you for teaching me to never stop fighting, that resilience and persistence are one's greatest qualities.

Last, but certainly not least, to Dani, words cannot describe your importance in this journey. For every late night and early morning insecurity, thank you for being my emotional rock (or my *dead sea*) and for not letting me lose sight of myself.

Abstract

In the biomedical domain, the identification of synonymous concepts is highly challenging, due to vocabulary heterogeneity, lexical variations, and non-uniform coverage across standardized terminologies. This work tackles this particular challenge, arguing that concept alignment can be made through approximate string similarity using deep neural networks. In particular, this work extends recent studies that assessed string-matching methods in non-biomedical fields, i.e. using bi-directional recurrent neural networks or transformer models to encode and match pairs of strings. The models were trained with biomedical data collected from Wikidata, and tested on 15 datasets built from different biomedical ontologies, representing specific domains. The tests assessed aspects such as the influence of positional encodings together with the inputs, the size of the training dataset or the contribution of model fine-tuning with specific in-domain data. The experimental results show that deep neural networks consistently performed better than traditional string similarity approaches, particularly with larger amounts of training data. In most of the tests, models based on Transformers also performed better than models based on recurrent neural networks.

Keywords: Biomedical Concept Alignment, String-Matching, Supervised Machine Learning, Recurrent Neural Networks, Transformer Networks

Resumo

No domínio biomédico, a identificação de conceitos sinónimos é altamente desafiante devido à heterogeneidade de vocabulário, às variações lexicais, e à cobertura não uniforme de terminologias médicas padronizadas. Este trabalho aborda este desafio em particular, argumentando que o alinhamento de conceitos pode ser feito através da semelhança aproximada de *strings* utilizando redes neuronais. Em particular, foram aproveitados estudos recentes que avaliaram métodos de correspondência de *strings* em áreas não biomédicas como, por exemplo, a utilização de redes neuronais recorrentes bidireccionais ou modelos *Transformer* para codificar e combinar pares de conceitos. Em particular, foram treinados modelos com dados biomédicos recolhidos da Wikidata, e testados em 15 conjuntos de dados (*datasets*) construídos a partir de diferentes ontologias biomédicas, representando domínios específicos. Os nossos testes avaliaram aspetos tais como a influência de codificações posicionais enquanto *input* destas redes, o tamanho do *dataset* de teste, e a contribuição do ajuste fino do modelo (*fine-tuning*) com dados específicos de cada domínio. Os resultados experimentais mostram que as redes neuronais tiveram um desempenho consistentemente melhor do que as abordagens tradicionais de semelhança de *strings*, particularmente com maiores quantidades de dados de treino. Na maioria dos testes, os modelos baseados no modelo Transformer também tiveram melhor desempenho do que os modelos baseados em redes neuronais recorrentes.

Keywords: Alinhamento de conceitos biomédicos, Correspondência de *strings*, Aprendizagem supervisionada, Redes Neuronis Recorrentes, Modelos Transformer

Contents

List of Tables

List of Figures

Acronyms

Adam Adaptive Moment Estimation. 6

ANN Artificial Neural Network. 5, 6, 7, 11, 12

bi-GRU bi-directional Gated Recurrent Unit. 14, 15, 16, 19, 20, 21, 37

BiRNN Bidirectional RNNs. 8

CBOW Continuous Bag of Words. 11

CNN Convolutional Neural Networks. 6, 15

EHR Electronic Health Records. 1

FMA Foundational Model of Anatomy Ontology. 24, 25, 26, 28, 29, 30, 32, 34, 35, 51

GRU Gated Recurrent Unit. 7, 8, 15, 19, 28

HDO Human Disease Ontology. 24, 26, 28, 29, 30, 32, 34, 50

HPO Human Phenotype Ontology. 24, 26, 28, 29, 30, 32, 34, 50

IDF Inverse Document Frequency. 10

LSTM Long Short-Term Memory. 7, 8, 15, 16

LTU Licenses To Use. 2

MA Mouse adult gross anatomy. 25, 26, 28, 29, 30, 31, 32, 34, 35, 51

MLP Multi-layer Perceptron. 6

MPO Mammalian Phenotype Ontology. 24, 26, 28, 29, 30, 32, 51

NCBI National Center for Biotechnology Information. 25, 26, 28, 29, 30, 32, 50

NCIT National Cancer Institute Thesaurus. 25, 26, 28, 29, 30, 31, 32, 34, 51

NLP Natural Language Processing. 1, 2, 3, 5, 10, 11, 12, 37

OAEI Ontology Alignment Evaluation Initiative. 25, 26

OLS Ontology Lookup Service. 24, 25, 26

ORDO Orphanet Rare Diseases Ontology Health Records. 24, 26, 28, 29, 30, 32, 33, 50

ReLU Rectified Linear Unit. 20, 21

RNN Recurrent Neural Network. 3, 7, 8, 9, 14, 15, 19, 27, 28, 29, 30, 31, 34

SNOMED CT Systemized Nomenclature of Medicine – Clinical Terms. 25, 26, 28, 29, 30, 32, 50

Uberon Uber-anatomy ontology. 24, 26, 28, 29, 30, 32, 50

UMLS Unified Medical Language System. 17

Chapter 1

Introduction

Terminology standardization is the process of establishing, in a community of experts, a shared common consensus on the use of technical terms and disambiguating their meaning. It is, therefore, a collection of well-organized, well-structured terminological data [1]. Standardisation of biomedical terminology is important in the biomedical and healthcare domain [2]. Standardized terminology can be used to represent medical information in Electronic Health Records (EHR), data retrieval and reuse for evidence-based decision making, as well as to improve communication between stakeholders. The use of structured reporting and standardized medical language has been demonstrated to have a direct impact on patient health and to enhance the use of medical data in secondary activities such as research, public health, and case studies [3]. Moreover, it facilitates interoperability of health systems [4].

Interoperability is defined by the capacity of two or more systems to share health data and utilise it after it has been received [5]. Interoperability between EHR systems and health-care stakeholders not only provides for smoother workflows and less ambiguity, but also enhances data transmission between them. Furthermore, an interoperable environment with information sharing enhances health-care delivery by making the suited data available to the right stakeholders at the right time. As a result, it is possible to obtain a patient-centered, value-driven care that improves health outcomes while lowering costs [6].

The standardisation of biomedical terminology and interoperability of electronic health systems are enabled by controlled vocabularies and ontologies. Controlled vocabularies are expressed by cataloged concepts and terms [7] whilst biomedical ontologies provide a formal definition of biomedical concepts and relationships between them [8]. This type of resources, including well-known examples, such as SNOMED[1] or the International Classification of Diseases (ICD)[2], usually contain synonyms for most terms, with Natural Language Processing (NLP) methods leveraging terminological resources, but they seldom cover all potential synonyms. This hinders tasks such as integrating clinical notes across different authors and domains or identifying new or rare terms that are not presented in standardized terminology [9]. Moreover, there are many ontologies covering overlapping domains, which leads to the same concept being defined in different ontologies with different terms [10]. The non-uniform coverage across subjects or languages motivates the development of methods for automatically performing alignments between multiple existing specialized terminology resources, in order to link together synony-

[1] https://www.snomed.org/

[2] https://www.who.int/standards/classifications/classification-of-diseases

mous concepts across different vocabularies and potentially unlock biomedical knowledge by bridging inaccessible data [11].

Concept alignments have been extensively studied within the context of ontology alignment. Most existing methods correspond to heuristic approaches combining multiple types of similarity metrics [12]. In the clinical-medical domain, similar concepts can be lexically similar (e.g. *dilated RA* and *dilated RV*), but also highly dissimilar (e.g. *cerebrovascular accident* and *stroke*). Thus, using similarity metrics based on matching character sub-sequences can be especially challenging for medical synonym discovery. As an alternative, recent studies have successfully explored deep learning approaches for synonym discovery in various contexts [13, 14, 15]. Furthermore, other NLP tasks, including approximate string similarity, have also relied on deep neural networks in other fields (e.g. the alignment of geographical or organizational names) [16, 17].

Taking inspiration on the aforementioned recent studies, this dissertation will focus on a supervised deep learning approach to perform approximate string matching in the biomedical field. A supervised deep learning algorithm automatically adjusts weights according to a certain training set [18]. Hence, it can contribute to the optimization of the process of concept alignment since this weight adjustment takes into consideration different features of the pairs of concepts present in the training set instead of focusing on a single similarity metric or a hybrid approach between two similarity metrics. This weight adjustment can be specifically relevant in cases where synonymous concepts are lexically dissimilar.

The proposed deep learning approaches will be tested on datasets built from different different biomedical ontologies, representing specific domains. It is, therefore, important to consider and be aware of the vast dimension biomedical terminology encompasses. Many ontologies use different systems and Licenses To Use (LTU) [19], which adds technical and economic barriers to obtain a uniform coverage in biomedical ontologies. Additionally, most biomedical ontologies were developed independently [12] and are often focused in a certain sub-field (e.g. cover only anatomy related terms or only disease related terms). Similarity tendencies are often related to a certain domain, posing difficulties in cross-domain evaluation [20]. For instance, if a neural model is trained with molecular biology related terms, its weights will be adjusted to their similarity predispositions that might not be equally weighted in anatomical terms. Therefore, obtaining a representative trained neural network is an additional challenge when designing deep learning approaches across the biomedical scope. The use of general resources such as Wikidata [3], a large-scale collaborative database, have aided in obtaining a broader coverage of biomedical terms [19].

1.1 Objectives

The main goal is of this dissertation is to perform concept alignment through classification of pairs of terms as synonyms. More specifically, this work presents and extends previously developed methodologies for string-matching, applying it to the biomedical field. Methods to be explored in the context of this extension and further objects of study within this problem are further described below:

1. Extension of deep learning models by developing neural network layers capable of encoding character position (i.e. representing input strings with character position information);

[3] https://www.wikidata.org/

2. Assessment of the influence of the aforementioned positional encoding;

3. Assessment of the impact of the training data size;

4. Assessment of cross-domain contribution (leveraging a generic training dataset to test the models in datasets retrieved from different ontologies and terminologies, across biomedical sub-domains);

5. A general verification if these methods are indeed successful and can contribute to string-matching within the biomedical scope.

1.2 Methodology

Taking inspiration in design science research process model [21] the methodology underlying this work divided itself in five steps: problem awareness; suggestion; development; evaluation and conclusion. In the first stage of this work a general revision was conducted with the goal of understanding why biomedical concept alignment is a challenge and what approaches have already been used to tackle the problem. Moreover it was important to understand how textual information can be represented and the underlying theoretical background in neural networks. Related work revision focused particularly on two main areas: on the one hand on the use of deep learning methods in the biomedical field for similar NLP tasks (e.g. ontology and concept alignment or synonym discovery); and, on the other hand, on state-of-the-art string matching approaches.

The suggested string-matching approach had as a starting point two neural network models: a RNN proposed by Santos et al. [16] and extended by Borges et al. [17] and a transformer model [22] also extended by Borges et al. An extension to both these models is presented by adding character positional encoding, since it was shown to improve NLP classification tasks [23].

The development of the aforementioned models relied on the Python programming language [4], which was chosen due to its popularity and extensive documentation. More specifically, Pytorch and [5] Pytorch-lightning[6], Python deep learning libraries, were used in this study. The datasets, trained model and source code are available in a public github repository[7].

In order to evaluate the proposed neural network in the biomedical field, it was trained with a generic, balanced dataset, with terms and concepts retrieved from Wikidata. The final dataset size was of 1 250 000 pairs of concepts and had no repeating occurrences, hence contributing to a one-shot learning of the neural network models. The methods were tested in 1 intra-domain and 14 cross-domain datasets. The intra-domain dataset was also retrieved from Wikidata using the same methodology and was obtained using a stratified fold with the goal of not having repeated pairs of concepts in the training and validation set. The remaining datasets relied on various sources of ontologies and terminologies, covering different areas of the biomedical scope. Results were compared against individual string similarity metrics and the exact Borges et al. proposed models, in order to validate not only the model as a string-similarity approach but also the value of the positional-encoding extension. Further experiments addressing the impact of the dataset size and considering in-domain training were conducted relying on the same neural

[4] https://www.python.org/
[5] https://pytorch.org/
[6] https://www.pytorchlightning.ai/
[7] https://github.com/LeonorFernandesIST/BiomedicalConceptAlignment.git

network models. The evaluation metrics used throughout the experiments were accuracy, precision, recall and F1 measures.

Finally, conclusions were drawn in regard to the models' success in performing the string-matching task in the biomedical scope.

1.3 Contributions

The main contributions of this work are the following:

- A method for modelling biomedical concepts and aligning pairs of synonymous concepts within the biomedical scope, using supervised deep learning approximate string matching. The success of the model, which was shown to be better than traditional methods, is useful to the biomedical field since correct concept alignments contribute to synonym discovery and ontology alignment which can enhance healthcare.
- A study on the impact of the training dataset size, which showed that aligning biomedical concepts relying on approximate string matching through deep neural networks are much more successful with larger sets of training data.
- A study on the impact of considering in-domain and all-domain data in training, which showed that not only does in domain training achieve, in general better results, but in cases where there is lack of abundant information (a small dataset) leveraging from other ontologies in the training set is useful (all-domain training). Hence, it calls attention to the importance of not only considering a large generic dataset but also to take into consideration the existence of overlapping domains in biomedical ontologies.

1.4 Thesis Outline

The remainder of this dissertation is organized into the chapters described below.

- Chapter 2 addresses important concepts and related work, focusing on deep learning and textual representation methods as well as state-of-the-art work on biomedical concept alignment and string-matching.
- Chapter 3 details the proposed approach for the concept alignment through approximate string matching. It describes the deep learning models leveraged in this work and the proposed positional encoding extension.
- Chapter 4 describes the experimental evaluation conducted in this work. It details the datasets and evaluation methodology used. Then, it presents and discusses the experimental results obtained in this work.
- Finally, chapter 5 draws the main conclusions regarding this dissertation and presents possible future work proposals.

Chapter 2

Concepts and Related Work

The work presented in this thesis relies on Artificial Neural Networks (ANNs) to perform approximate string-matching and, therefore, align biomedical concepts. Hence, it is important to not only understand the basic ANN architectures, but also how textual information (i.e. the biomedical concepts) can be represented. Sections 2.1 and 2.2 introduce these concepts and models. Moreover, it is also relevant to explore traditional and state-of-the-art approaches on both string-matching and biomedical concept or ontology alignment with the goal of understanding what has been done in similar tasks or fields and leveraging previous successes whilst taking into consideration known limitations. Sections 2.3 and 2.4 overview these approaches and section 2.5 summarizes the related work.

2.1 Introduction to Neural Network Models

ANN are computational approaches inspired on the human nervous system, the biological system that is responsible for locomotion (activating our muscles), processing and interpreting external and internal stimuli, learning and decision-making, among other things [24]. To further understand the architecture behind an ANN it is important to introduce concepts on a biological neural network. The human nervous system's basic unit is the neuron. In a very general way, a neuron receives a stimuli or signal with a certain intensity and, if this intensity reaches a certain threshold, it generates the propagation of the signal - through synapses - to another neuron, therefore forming a network of connected neurons [25].

An ANN is made up of a series of connected nodes that serve as a simplified representation of biological neurons [26]. Each node receives a certain weighted input, whose product, after being processed by a non-scalar function, will be passed as the subsequent output. A representation of synapses is made whenever a certain output is passed on as input to another node. The learning process can be, therefore, defined as the capability of learning weights that will produce the most appropriate response.

Several architecture types of ANN are used for deep-learning. In the following sections, the most common architectures used in NLP will be introduced.

2.1.1 The Perceptron

The simplest neural network, introduced by Rosenblatt [27], is the single-layer perceptron. This single-node network is used for binary classification problems. Mathematically, it can be described as linear

model according to Equation 2.1, where y is output vector, x is the input vector, w the weight vector, b a bias vector and $\phi(\cdot)$ an activation function that returns -1 if its input is negative or 1 otherwise:

$$y = \phi(x \cdot w + b). \tag{2.1}$$

Training the perceptron entails selecting an example from the training dataset and updating each weight in the weight vector w interactively according to Equation 2.2, where w_i is the *ith* element of w, x_i is the *ith* element of x, y is the predicted output, $\hat{y}$ the true label for the data point being processed, and η is the learning rate:

$$w_i = w_i + \eta \cdot (\hat{y} - y) \cdot x_i. \tag{2.2}$$

The training algorithm converging to the correct classification depends on whether η is sufficiently small and if the data is linearly separable (i.e. There is a set of weights that can accurately classify all of the cases in the training set).

2.1.2 Multi-Layer Perceptron

In order to solve more complex problems, the single perceptron was extended to a neural network called a Multi-layer Perceptron (MLP) [28]. In this case there are at least three set of nodes: an input layer; an output layer where the required task is performed; and between them one or more hidden layers, containing computational nodes. The signal propagates feed-forwardly layer by layer until it reaches the output node(s). Mathematically, considering a single hidden layer, the MLP can be defined by Equation 2.3, where x and y are respectively input and output vectors, A and B are matrix's representing the first and second layer weights, a and b are the bias vectors of each corresponding layer and $\phi(\cdot)$ and $\phi'(\cdot)$ are the activation functions of the hidden and output layers:

$$y = \phi(\phi'(x \cdot A + a) \cdot B + b). \tag{2.3}$$

In the final layer of a MLP classifier a softmax function (i.e., a normalized exponential function that produces a probability distribution as an output) is frequently used to train the network while minimizing a given cross-entropy loss. Training the neural network will therefore, similarly to the single layer perceptron, correspond to adapting all weights and bias to their optimal values. This adaptation can be done by applying the back-propagation algorithm in combination with some variation of the gradient descent optimization [29]. The Adaptive Moment Estimation (Adam) technique is an optimization procedure that has been widely utilized to train deep neural networks [30]. Adam uses an exponentially decaying average of past gradients, along with adaptive learning rates for each parameter, to compute parameter updates. In practice, infrequent parameters result in larger updates and frequent parameters in smaller updates.

2.1.3 Convolutional Neural Networks

Convolutional Neural Networkss (CNNs) are complex ANNs in which a filter h, also known as kernel (i.e. a linear transformation followed by an activation function) is applied to a given input sequence [31, 32]. To create a collection of feature maps, each filter conducts a local convolution (i.e. a multiplication of

elements in a matrix followed by a sum) on the sub-sequences of the input. Then, to obtain a single scalar, a global average or max-pooling over time is performed, and the scalars from the h filters are concatenated into the sequence representation vector.

In other words, assuming a sentence as input, the filter will be applied to each instant of a k-word sliding window over the phrase, transforming a window of k words into a dimensional vector d that captures significant features of the words in the window (i.e. the feature map). The pooling operation joins the vectors from different windows into a single d-dimensional vector.

Hence, for a sequence of words $x = x_1, \ldots, x_n$ a convolution layer of width k creates several instances of windows $w_i = [x_i; x_{i+1}; \ldots ; x_{i+k-1}]$. The filter of size k is then applied to each window resulting in m vectors $p_1, \ldots, p_m$. These vectors are mathematically described in Equation 2.4, where $\phi(\cdot)$ is an activation function that is applied element wise and A and a are network parameters:

$$p_i = \phi(w_i \cdot A + a). \tag{2.4}$$

The m vectors can then pass through a max-pooling or and average-pooling layer and a final representation of vector r is obtained. For the case of max-pooling, each element j of r is obtained according to Equation 2.5, where $p_i[j]$ denotes the j-th component of p_i:

$$r[j] = \underset{1 \leq i \leq m}{max}\, p_i[j]. \tag{2.5}$$

The max-pooling operation has the effect of obtaining the most important information across window positions. Each dimension should ideally "specialize" in a specific type of predictor, with the max operation selecting the most essential predictor of each type. On the other hand, the average-pooling operation determines the average value for patches of a feature map, resulting in smoother feature extraction.

2.1.4 Recurrent Neural Networks

RNN are time-dependent complex ANN that compute a hidden state vector h_t at each time step t [33]. As indicated by Equation 2.6, the hidden state is obtained by a non-linear transformation that takes as inputs the prior hidden state $ht - 1$ and the current word input xt:

$$h_t = f(h_{t-1}, xt). \tag{2.6}$$

The simplest RNN is the Elman RNN. At a certain time-step t, the hidden state h_t is a function of the input at the same time step x_t, modified by a weight matrix W. This result is added to its own hidden-state-to-hidden-state matrix U, also known as a transition matrix, and multiplied by the hidden state of the preceding time step h_{t-1}. The weight matrices are essentially filters that determine how much importance should be given to both the present input and the past hidden state [34]. This RNN which is described by Equation 2.7:

$$h_t = \phi(W x_t + U_h h_{t-1}). \tag{2.7}$$

Previous research has shown that Elman RNN has difficulties in modeling long sequences. Some extensions, such as Long Short-Term Memorys (LSTMs) units [35] and Gated Recurrent Units (GRUs) [36], were created to address this limitation. These approaches combine prior states with the current input,

requiring a specific interaction of gating mechanisms.

In GRUs a reset gate r controls how to merge the new input with the existing memory, and an update gate z governs how much of the previous memory is maintained and how much new information is added. Mathematically, these models can be defined by Equations 2.8 to 2.11, where x_t refers to the input vector at a certain time step t, $\odot$ is the Hadamard product (i.e. the entry-wise product of two matrices) and parameters W U and b denote different weight matrices and biases that are adjusted when training the model through back-propagation:

$$z_t = \varphi_g(W_z \cdot x_t + U_z \cdot h_{t-1} + b_z), \tag{2.8}$$

$$r_t = \varphi_g(W_r \cdot x_t + U_r \cdot h_{t-1} + b_r), \tag{2.9}$$

$$\tilde{h_t} = \varphi_h(W_h \cdot x_t + U_h \cdot (r_t \odot h_{t-1}) + b_r), \tag{2.10}$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h_t. \tag{2.11}$$

Due to the gates specifying how much of the input and prior state vectors should be considered, the network learns how to manage long-term dependencies through them.

LSTMs are similar to GRUs but have more parameters (e.g. an extra gate). These networks use various gating mechanisms, including a forget gate f_t, which determines how much of the previous gate will be maintained, an input gate i_t, which determines how much of the proposed gate g_t should be preserved, and an output gate o_t, which determines the output at time t. Mathematically they can be described by Equations 2.12 to 2.17, where x_t, h_t, $\odot$, W, U and b have the same meaning as in the GRUs equations.

$$i_t = \varphi_g(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i), \tag{2.12}$$

$$f_t = \varphi_g(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f), \tag{2.13}$$

$$o_t = \varphi_g(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o), \tag{2.14}$$

$$g_t = \varphi_h(W_g \cdot x_t + U_g \cdot h_{t-1} + b_g), \tag{2.15}$$

$$c_t = f_t \odot c_{t-1} + g_t \odot i_t, \tag{2.16}$$

$$h_t = \varphi_h(c_t) \odot o_t. \tag{2.17}$$

LSTMs have reported to outperform GRUs when more training data is available in tasks requiring modeling longer-distance relations [37].

Another extension of RNNs are Bidirectional RNNss (BiRNNs) which are composed of two RNNs

and read an input sequence in both directions. The forward RNN reads the input from left to right, hence capturing unbounded left side context and the backward RNN reads the input from right to left, therefore capturing the unbounded right side context. The hidden states for each of the RNNs are concatenated according to Equation 2.18 where h_t^f and h_t^b are respectively the forward and backward hidden states and $\oplus$ is the concatenation operator:

$$h_t = h_t^f \oplus h_t^b. \tag{2.18}$$

The output layer can get information from both past and future states at once using this network.

2.1.5 Transformer Models

Transformer models were proposed by Vaswani et al. [22] as a way to use an advanced attention scheme to model input and output dependencies without needing recurrence or convolutions. Scaled dot product attention mechanisms, in which numerous attention heads are applied in parallel, enabling the model to attend to distinct representation sub-spaces at different points, are the foundations of this model.

A Transformer encoder has two sub-layers within each layer. The first sub-layer has a Multi-Head Attention module that aggregates the embeddings (V) of a collection of keys (K) to compute the output embeddings for a set of queries (Q) (Equations 2.19, 2.20 and 2.21). The second sub-layer corresponds to a position-wise fully-connected feed-forward network:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(head_1, ..., head_h)W^O, \tag{2.19}$$

$$head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V), \tag{2.20}$$

$$\text{Attention}(Q, K, V) = \text{Softmax}(\frac{Q \cdot K^T}{\sqrt{d_k}}) \cdot V. \tag{2.21}$$

In the previous expressions, W_i^Q, W_i^K and W_i^V are matrices that linearly project queries, keys and values into the attention space of the ith head, while W^O is a matrix that linearly transforms the concatenation of the outputs of all heads.

Unlike RNNs, this model does not require sequential data to be processed in order.

Wang et al. [38] has recently extended the Transformer model combining it with a RNN. More specifically, the R-Transformer is made up of a series of identical layers stacked on top of each other. Each layer consists of a lower, middle and upper level. The lower level is made up of local recurrent neural networks, which are designed to model local structures in a sequence; the middle level is made up of multi-head attention, which can capture global long-term dependencies; and the upper level is made up of position-wise feedforward networks, which perform a non-linear feature transformation.

2.1.6 Siamese Neural Networks

Siamese neural networks are networks that have an architecture where different parts have their parameters tied, forming two or more identical sub-networks (i.e. in terms of configuration, parameters and weights) [39]. Whenever a parameter update is made in one of the sub-networks it is reflected

throughout the other sub-network(s).

Siamese networks are widely used in tasks that require determining similarity or a link between two similar objects [40, 41, 42, 43]. When the inputs are similar, it makes sense to use a comparable model to handle them, which is why siamese architectures are effective for these tasks. As a result, the networks will have representation vectors with the same semantics, which will make comparing pairs of phrases easier. Since the weights are shared among sub networks, there are fewer parameters to train, resulting in less training data and a lower risk of over-fitting [39].

2.2 Representing Textual Information

This thesis tackles a NLP problem using supervised machine learning. NLP is the study of computer programs that take natural or human language as input [44]. Supervised machine learning is concerned with inferring the parameters of models that accept a certain vector x as input and provide another vector as output, with each position indicating the likelihood of the input belonging to a specific class [45]. When performing a supervised machine learning task in the NLP domain, x will generally encode characteristics of a text such as characters or words. Hence, it is necessary to represent textual information as a vector [46]. A simple and common way to tackle this representation is relying on one-hot-vectors. In this case, given a vocabulary (of words, n-grams or characters) with dimension V, each vocabulary instance is represented as a V-dimensional vector with all values 0 except at the index of the word, n-gram or character represented, which would have the value of 1 [47].

Vector space model are a simple, straightforward technique that uses one-hot vectors to represent textual information in documents as fixed size vectors, by summing the word representations w_i, in a certain document d with N words [48]:

$$d = \sum_{i=1}^{N} w_i. \tag{2.22}$$

This will produce a V-dimensional sparse vector, where a certain word i's frequency in the document is represented by the ith vector element. Based on the idea that uncommon terms are more discriminative, and therefore should be valued higher,the Inverse Document Frequency (IDF) score can, instead, be taken into consideration [48]. This collective-level importance score is calculated by using a weighted average of each word representation w_i in a collection of documents D. This representation is expressed by Equations 2.23 and 2.24:

$$d = \sum_{i=1}^{N} \frac{IDF(i)}{N} \times w_i, \tag{2.23}$$

$$IDF(i) = log(\frac{|D|}{|\{d' \in D | w_i \in d'\}|}), \tag{2.24}$$

where $|D|$ is the number of documents in the collection of documents D and $|\{d' \in D | w_i \in d'\}|$ represents the number of documents that contain the concept w_i.

One-hot representations of words are less common nowadays [49]. This representation usually results in vectors that are not efficiently handled due to their large dimension. Moreover, one-hot representation consider all words to be totally independent, regardless of meaning. Hence, in more recent years approaches involve representation of textual information through lower-dimensional dense vectors

[50]. Most of these methods attempt to represent words based on syntactic and semantic information so that words with similar meaning are also closer together in the vector space that represents them [51] (e.g. pain and ache would have similar representations). These methods are referred to as word embeddings and are applied to many NLP tasks (e.g. information extraction and retrieval, text classification and machine translation) [47]. Formally, word embeddings can be defined as a map $f_{we} : \mathbb{N} \rightarrow \mathbb{R}^D$ from a discrete word index to a D dimension real valued vector $\mathbb{N} = \{0, 1, 2, ...\}$ [23].

Many state of the art approaches to perform embeddings rely on neural network models, being one of the most popular the word2vec [52, 53]. This method presents an unsupervised training of word embedding based on the assumption that identical words appear in comparable contexts frequently. It can rely on a skip-gram or a Continuous Bag of Words (CBOW) architecture. The first model focuses on the predicting a context (i.e. surrounding words) given a center word, whereas the CBOW predicts a center word given the context.

Mathematically, the skip-gram aims to, given a sequence of words $w_1, w_2, ..., w_T$, maximize the following average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0} log(p(w_{t+j}|w_t), \tag{2.25}$$

where c is the context size, T the total number of words, and w_t the word at position t. The probability $p(w_{t+j}|w_t$ is mathematically defined by:

$$p(w_O|w_I) = \frac{exp(V(w_O)^T V(w_I))}{\sum_{w=1}^{W} exp(V(w)^T V(w_I))}, \tag{2.26}$$

where $V(w_I)$ and $V(w_O)$ are a vector representations of the input word (center word) and output word (a context word) and W is the vocabulary size.

On the other hand, the CBOW model, given a sequence of words $w_1, w_2, ..., w_T$, is mathematically defined by:

$$\frac{1}{T}\sum_{t=1}^{T} log(p(w_t|w_{t-c}, ..., w_{t-1}, w_{t+1}, ..., w_{t+c}), \tag{2.27}$$

where c also represents the context.

The usage of word and phrase embeddings have been shown to improve methods' performance when they are used as underlying input representations in neural network sentence encoders [54, 55]. Hence, encoding word representations is usually an important step in NLP tasks. Furthermore, when considering one-hot encoding as input representations, studies have efficiently used character based vocabularies instead of word vocabularies, capturing both word orthography arbitrary aspects and possible sub-words or word shape information [56, 57]. This means that this representation may convey more information at a more fundamental level, which is particularly beneficial when dealing with sparse or noisy data [47]. However, due to parallelization, processing text can be computationally costly to model with ANN [58]. This has been addressed by adding position embeddings to the feature level rather than modeling word sequence at an architectural level. The Transformer model [22], which substitutes recurrent and convolution processes with solely attention methods, and the Convolutional Sequence Model [59] have proposed methods to do this. Recently, Wang et al. [23] studied positional embeddings in the

context of sequential word order. A positional embedding is formally defined as a map from a discrete position index to a vector $f_{pe} : \mathbb{N} \rightarrow \mathbb{R}^D$. In this case the final embedding of a word is obtained by summing both the word and position embeddings (Equation 2.28):

$$f(j, pos) = f_{we}(j) + f_{pe}(pos), \tag{2.28}$$

where f_{we} is the word embedding of word j in a certain vocabulary and is in the pos-th in a sentence. Wang et al. showed that in a variety of tasks, including positional encoding at a feature level, consistently obtained better results, which inspired the extension proposed in this dissertation. Moreover, in order to describe the smooth change across consecutive word locations and implicitly capture relative distances between words, Wang et al. expanded word vectors to word functions with position as a variable.

2.3 Approximate String Matching Methods

Approximate string matching [16] (also referred to as string-matching in this work) is a use of NLP to identify whether a pair of character strings represent the same entity or concept. Traditionally, concept alignment methods were done through string similarity metrics. In many state-of-the-art approaches neural networks have performed this task and achieved better results, however, it is still important to understand the traditional metrics as they are an important baseline for the ANNs. In this section, both traditional and state of the art approaches for string similarity are presented.

2.3.1 String similarity Metrics for String Matching

String similarity metrics can be based on character operations, vector-space representations or hybrid methods of the aforementioned metrics.

Character-based operations [60], also called sequence-based or edit distance, take two strings and calculate the edit distance between them using it as quantification for the similarity between two strings. For instance, given to strings s_1 and s_2 the Levenshtein edit distance measure represents the smallest number of changes (i.e., insertions, deletions, or substitutions) required to transform one string into another. The string similarity based on this distance is calculated by Equation 2.29 where $levenshtein(s_1, s_2)$ is the aforementioned distance between strings s_1 and s_2 and $|s_1|$ and $|s_2|$ are the lengths of each string:

$$sim_L(s_1, s_2) = 1 - \frac{levenshtein(s_1, s_2)}{(max\{|s_1|, |s_2|\}}. \tag{2.29}$$

The Jaro metric [61] is another example of character-based operations. This metric is based on the common number of characters and order between the strings, i.e. the difference between their positions should be no more than half the length of the longer string. Character transpositions are also taken into account. The Jaro similarity is given by the following Equation:

$$sim_j(s_1, s_2) = \begin{cases} 0 & if m \equiv 0, \\ \frac{1}{3}(\frac{m}{|s_1|} + \frac{m}{s_2} + \frac{m-t}{m}) & otherwise, \end{cases} \tag{2.30}$$

where m is the number of matching characters, t half the number of transpositions and $|s_1|$ and $|s_2|$ the

lengths of strings s_1 and s_2. Winkler [62] extended this metric by considering a prefix scale that gives higher scores to strings that match from the beginning of the string until a certain prefix length. This extension is defined by:

$$sim_w(s_1, s_2) = sim_j + lp(1 - sim_j), \tag{2.31}$$

where sim_j is the Jaro similarity between strings s_1 and s_2, l is the length of common prefix at the start of the string up to a maximum of 4 characters and p is a constant scaling factor or how much the score is adjusted upwards for having common prefixes. This scaling factor is usually 0.1. In order to further extend this metric, which showed to be problematic in strings that contained multiple words in a different order Christen [63] developed alternative variations. These variations consisted on algorithms where (i) the tokens that make up both strings are sorted before calculating their Jaro-Winkler similarity (the sorted Winkler) or (ii) the similarity is computed across all conceivable token permutations and the maximum value is returned (the permuted Winkler). A more recent character operation metric, the I-Sub measure [64], has been used in many approaches, since it was specially developed for ontology alignment. This measure claims that similarity is determined by both commonalities and differences between the two strings being compared. Equation 2.32 formally defines this metric:

$$sim_{ISub}(s_1, s_2) = comm(s_1, s_2) - diff(s_1, s_2), \tag{2.32}$$

where $comm(s_1, s_2)$ represents the commonalities and $diff(s_1, s_2)$ the differences between the strings. The commonalities $comm(s_1, s_2)$ are calculated by repeatedly finding the longest common sub-string, removing it, searching for the next longest common sub string and removing it, until there are no common sub-strings left. The sum of the common sub-strings lengths of each i iteration is scaled by the length of the original strings $|s_1|$ and $|s_2|$:

$$comm(s_1, s_2) = \frac{2 \cdot \sum_i |maxComSubstring_i|}{|s_1| + |s_2|}. \tag{2.33}$$

The difference between two strings is defined by:

$$diff(s_1, s_2) = \frac{uLen_{s1} * uLen_{s2}}{p + (1 - p) * (uLen_{s1} + uLen_{s2} - uLen_{s1} * uLen_{s2})}, \tag{2.34}$$

where $uLen_{s1}$ and $uLen_{s2}$ are the lengths of unmatched sub-strings left by the last iteration step scaled with the original string's length and p is a factor usually scaled to 0.6.

Vector-space approaches focus on term-based similarity [60] (also known as token-based similarity), whose main characteristic relies on likeliness quantification being associated to an overlap of two token sets. Common approaches rely on computing the cosine [65] (Equation 2.35), Jaccard [66] (Equation 2.36) or Dice [67] (Equation 2.37) similarity measures that are based on character n-grams (i.e. based on consecutive n character sequences, typically with $n = 2$ or $n = 3$). These methods are defined by:

$$sim_{cos}(A, B) = \frac{A \cdot B}{||A||||B|}, \tag{2.35}$$

$$sim_{jaccard}(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{2.36}$$

$$sim_{dice}(x, y) = \frac{2n_t}{n_x + n_y}, \tag{2.37}$$

where in both equations 2.35 and 2.36 A and B are a set of words or n-grams of s_1 and s_2 respectively and in Equation 2.37 n_t is the number of bi-grams found in both strings, n_x the number of bi-grams in string x and n_y the number of bi-grams in string y. The cosine measure is generally used in information retrieval whilst the Jaccard and Dice are more common in string comparison. Skip grams, or bi-grams of non-adjacent letters with gaps of zero, one, or two characters, have also been recommended as a viable alternative to standard n-grams when representing strings to be matched [68].

Hybrid approaches have the ability of being flexible about word order position whilst allowing for small differences in word tokens, combining, therefore, advantages of the aforementioned approaches. Most of these methods rely on a final score that is computed taking into account second-level measures, i.e. sub-measures that are applied to all pairs of word tokens between two strings. For example, in the approach provided by Monge and Elkan the average similarity between the most similar pairs of word tokens is estimated according to a certain sub-measure such as the Jaro Winkler or the Levenshtein similarity. This method is defined by Equation 2.38:

$$sim(s_1, s_2) = \frac{1}{|s_1} \sum_{i=1}^{m} \max_{j=1}^{n} \{sim'(a_i, b_j\}, \tag{2.38}$$

where s_1 is the length of string s_1, $\{a_1, a_2, ..., a_n\}$ is the set of words in s_1, $\{b_1, b_2, ..., b_m\}$ is the set of words in s_2 and $sim'(a_i, b_i)$ is the base similarity metrics applied to words a_i and b_j. When aligning the individual word tokens, the cosine similarity and the Jaccard coefficient are also used in hybrid approaches. In these cases sets of tokens rather than sets of n-grams are considered which consequently softens the metrics by allowing small mismatches (e.g., by applying a threshold over the results of a character-based inner similarity metric) [65, 69, 64].

2.3.2 Neural network models in string-matching

Recently, deep learning approaches have been successfully explored as an alternative to standard string similarity metrics in various domains.

Conneau et al. [70] advanced a generic architecture for determining, from a premise sentence, if a given hypothesis sentence can be inferred. This architecture has also been used for string-matching [16, 17]. In these cases both strings were encoded by a RNN, creating a representation of each vector that were then matched in some way (e.g. by vector concatenation, vector difference, and/or element-wise product), fed into a set of fully-connected layers, and finally processed through a feed-forward layer with a sigmoid activation function, which generates a binary decision.

Santos et al.'s architecture [16] took inspiration on the previously described generic architecture, leveraging two stacked layers of bi-directional Gated Recurrent Units (bi-GRUs) with shortcut connections as the string encoder. This model outputs the final hidden state of the second bi-GRU. Since this is the underlying base of this work, chapter 3 presents further detail on this architecture. Borges et al. [17] proposed several extensions to this model. Instead of considering the final hidden state of the second bi-GRU as the input vector representation the authors considered: the use of max-pooling and

average-polling operations over the hidden states from the second bi-GRU to create the representation of the whole input string; the use of an inter-attention (i.e., alignment) layer, allowing the model to learn to attend and align different pairs of characters between the two input strings. The substitution of the activation functions of the GRU cell with a penalized hyperbolic tangent activation function was also advanced. In this study, the bi-GRU with the max-pooling and average-pooling extension presented better results in most cases.

The string matching problem can also be formulated as retrieval-based ranking problem where, given a string, the goal is to rank a set of other similar strings, with the most similar placed on top. Gan et al. [71], Traylor et al. [72] and Tam et al. [73] all presented deep neural networks to address this problem. Gan et al. [71] proposed a string encoder combined with a ranking component. The string encoder was represented either by a CNN, where final vector representations for the input string were obtained by concatenating results of a max-pooling operation over the outputs of the three convolution layers, or a bi-directional LSTM, where vector representation was defined as the last hidden state of the neural network. The ranking component rated the candidates based on their cosine similarity to the query representation, given the vector representations of the query string and the candidate strings. Equation 2.39 represents the mathematical definition of the loss function used to learn the model parameters:

$$L(\theta) = -log \prod_{Q,D^+} \frac{exp(\gamma R(Q,D^+))}{\sum_{D' \in \mathbf{D}} exp(\gamma R(Q,D'))} \quad with\ R(Q,D) = \frac{y_Q^T \cdot y_D}{||y_Q|| \cdot ||y_D||}, \tag{2.39}$$

where Q represents the query string with a vector representation y_Q, D a candidate string with a vector representation y_D, D^+ the correct target string and D the set of candidate strings to be ranked. The results showed that the CNN-based string encoder outperformed its RNN counterpart, which the authors believe is due to the CNN's greater capacity to catch local patterns in sequential data.

Traylor et al. [72] and Tam et al. [73] also used bi-directional LSTMs to encode pairs of strings, but considered the whole sequence of vectors output by the bi-LSTM for the input strings. In the first case, a CNN with max-pooling was applied to an alignment matrix (obtained by multiplying both string representations) followed by a linear layer to output the final score. The system was trained on triples of strings with a training target of $\sigma(f(s,t)f(s,n))$, where s represents a mention string, t represents an alias of s, and n represents a string that is not an alias of s. The authors constructed five different datasets for the tests, including Wikipedia entities, Wikipedia persons names, patent records, names of music artists, and illness names. In their approach to use a retrieval-based assessment, the previously disclosed model, which included bi-LSTMs and a CNN, outperformed other methods in terms of mean average accuracy scores and hits at top K results, such as simpler neural models or traditional string similarity approaches.

In the second case, after the bi-LSTM, Tam et al. [73] proposed a transport plan matrix (the conversion of the encoding of one string to the encoding of the other string) that is multiplied element-wise by a similarity matrix (the inner product of both string representations) and its result is fed to a three layer CNN. Similar to the previous approach, a final linear layer outputs the desired score between the query and the candidate string. In particular the transport plan matrix is obtained through transport problem that is formally described using two probability distributions, p_1 and p_2, as well as a cost matrix C that describes the cost of converting each element in the support of p_1 to each element in the support of p_2. The aim is to create a transport plan matrix that describes how to transfer p_1 to p_2 for with minimal

cost, which can be done using the Sinkhorn iteration method [74]. In this task, the input strings are represented by p_1 and p_2 whilst the character representations generated by the bi-LSTM are h_i and h_j, and the cost matrix C as demonstrated in Equation 2.40:

$$C_{ij} = \max_{i'j'} h_{i'} \cdot h_{j'}^T - h_i \cdot h_j^T. \tag{2.40}$$

Zhao et al. [75] suggested a pre-training strategy for the entity matching problem of identifying which records from two tables correspond to a match. The suggested technique begins by identifying which attribute types exist in a table (e.g., person or organization). As a result, a type identification model was trained using 49 distinct entity types taken from different knowledge sources. For each attribute type, a neural string matching model was pre-trained with entities belonging to these specified types, taking into account numerous entity aliases available in the knowledge bases. Considering a new dataset with fewer training data, the type of attributes existing in the table is recognized first and if that type is known, a model that has been pre-trained with data from that type is fine-tuned using instances from the dataset at hand. Otherwise, fine-tuning is done using a model that has been pre-trained with all type-specific data. The neural networks supporting this work to analyze the matches were a bi-GRU and inter-input attention mechanisms at both the character and word level. The authors used training examples from eight distinct datasets from various areas in their studies. One of the key findings was that fine-tuning from a pre-trained model was consistently poorer than training on domain specific cases from scratch. In terms of fine-tuning, they found that, in general, models that had previously been pre-trained on the union of many variables performed well.

2.4 Biomedical concept and ontology alignment

Given a certain field, an ontology is generally defined as a collection of concepts and categories that demonstrates their features and relationships [76]. Ontology alignment can be defined as the process of corresponding semantically related concepts, classes and properties between two ontologies [77]. When considering ontology matching in the biomedical domain it is important to be aware of the rich lexical component of biomedical vocabulary and consider a variety of annotations per class instead of considering only the primary name of each class within each biomedical ontology. Faria et al. [12] have shown that it is more effective to use all available synonyms for a certain concept and only by doing so can biomedical ontologies from different communities be effectively bridged.

Several attempts have been made to build machine-learning algorithms based on binary classification for ontology matching [78]. These approaches include, among others, classifiers based on decision trees [79], Support Vector Machines [78], and Logistic Regression [80].

Many state of the-art-approaches rely on contextual or external information to aid on identifying medical synonyms or performing biomedical ontology alignment. Wang et al. [13] and Jiang et al. [14] both propose supervised ontology alignment methods through neural networks - a siamese multi-layer percepton with a sigmoid function and a LSTM based method enhanced with a char-embedding technique, respectively. In both cases, the datasets used were from already existing alignment data and results showed that a capability of encoding additional information when available is beneficial.

On the other hand, Scumaster et al. [9] and Kolyvakis et al. [15] present unsupervised learning

methods to tackle these challenges. Schumaster et al. [9] developed a neural network that makes use of contextual information from surrounding text or patient information to build synonym representations and perform the task of synonym discovery. Results from evaluating the neural network on a medical related concept linking dataset were consistent on concluding that performance is better when providing contextualized representations than non contextualized representations. Kolyvakis et al. [15] describe a network based on embedding ontological terms in a high-dimensional Euclidean space to perform ontology alignment, relying on a similarity function whose measurement is higher in the cases where vectors of words that appear in the same sorts of context. The results show that using terminological embeddings to capture semantic similarity can help with ontology alignment.

Although these methods have shown to be effective, there is an underlying struggle on identifying the most suitable and useful sources of background knowledge [12], which in itself has also been a topic of several studies. [81, 82, 83].

Another challenge related to concept alignment is that controlled vocabularies are not accessible in all languages and often lack complete definitions. Rahimi et al. [84] focused on aligning a controlled vocabulary - the Unified Medical Language System (UMLS) to Wikipedia, whose health related articles can contribute with content and multilinguality, through a neural ranking model. With this approach, a broader range of countries are able to adopt the UMLS, whilst also facilitating health information to patient's whose linguistic background differs from the doctor consulted or health system used. This means that for a certain ULMS concept, a neural ranking model will retrieve relevant Wikipedia articles that match that concept and it's multilingual aliases. The results using this neural model showed a substantial improvement of 20% over manual alignment with minimal effort.

2.5 Overview

Biomedical concept alignment is a challenging task where many recent systems have relied on contextual data to facilitate these alignments. However, trustworthy source identification is an issue with these methods. Thus, direct concept alignment between terms is important in this field and string-matching methods can be applied. More specifically, since state-of-the-art approaches mainly rely on deep neural networks that leverage textual processing it makes sense that biomedical concept alignment through approximate string matching would also rely on them.

Chapter 3

Neural Network Models for Biomedical Concept Alignment

This dissertation tackles the string matching problem within the biomedical domain, leveraging from neural network models from previous works. In particular, a RNN proposed by Santos et al. [16] and extended by Borges et al. [17] and a transformer model also proposed by Borges et al. Section 3.1 describes these models. The strategy utilized to improve the models relies on extending them with positional encoding as part of the textual representation mechanism. This extension is described in Section 3.3 and Section 3.4 presents a general summary on the final models. The model implementation relied mostly on *Pytorch Lightning* deep learning library, as mention in section 1.2.

3.1 Neural Network Models General Architecture

3.1.1 Textual Representation as Input

Each neural network model receives as input binary vector sequences that represent the strings to be compared. Firstly, the strings are converted to a unicode canonical normalized format (i.e. a completely decomposed UTF-8 representation with all combining character marks placed in a predetermined order) and padded with a specific symbol that specifies the concept's beginning and end. Each byte represents a single bit set to one, and the normalized strings are subsequently encoded as a sequence of one-hot binary vectors. The binary vectors are sent into either the bi-directional GRU or the transformer model as input, which will consequently produce an embedding for each of the concepts being compared.

3.1.2 Architecture Based on RNN

The neural network proposed by Santos et al. [16] is a siamese RNN. The string encoder, consists of a stack of two bi-GRU. A sequence of real-valued vectors is generated by the first bi-GRU layer and transferred as input to the second bi-GRU. These last outputs form the first bi-GRU are concatenated in each direction by the the second layer, outputting a single embedding for the input. It is important to note that even though each of the concepts entered as input is processed by individual recurrent layers the parameters of these GRUs are shared across the parts of the network that process each input term.

Hence, if $X = [x_1, ..., x_L]$ is a sequence of one-hot vectors corresponding to the byte representations of the characters that compose an input string, with length L, then we can denote by H the sequence of hidden states output by the second bi-GRU, and by s the final representation of the input string, as described by Equations 3.1 and 3.2:

$$H = \text{BiGRU}(= \text{BiGRU}(X)) = (h_i, ..., h_L), \tag{3.1}$$

$$s = h_L. \tag{3.2}$$

After obtaining the two embedding vectors from the two layers of bi-GRUs, they are combined into a single representation by either concatenating them or by calculating the element-wise product the difference between them. This representation is finally fed into two feed forward layers to produce the final output. These layers consist of simple combination of the inputs in addition to a Rectified Linear Unit (ReLU) (i.e. a nonlinear activation function) and a sigmoid activation function.

From the various extensions to this model explained in section 2.3, the current work leverages from addition of max-pooling and average-polling operations over the hidden states from the second bi-GRU since, in most cases it reported better results [17].

3.1.3 Architecture Based On the Transformer Model

Borges et al. [17] also proposed an extension with the transformer model for string matching. More specifically, it extends the R-Transformer model [38] where the the encoder layers use a single and first bi-GRU followed by multi-headed scaled dot product attention mechanism to capture interactions between the two input strings. In order to obtain the vector representation for the input these layers are followed by max-polling and average-pooling operations to aggregate the outputs of the final encoder layer. The remaining layers maintain themselves the same as in the previous architecture (specifically, the two feed-forward layers).

3.2 Positional Encoding

This thesis proposes adding a positional encoding to the aforementioned models as a class of the textual representation encoder so that, given a certain pair of strings, the input representation includes information on each character's position, instead of containing information only on which character it represents (one-hot vector).

A trigonometric position embedding [22] was added to the input representation in which each position embedding is selected as:

$$\begin{aligned} PE_{2k}(\cdot, pos) &= sin(pos/10000^{2k/d_{model}}), \\ PE_{2k+1}(\cdot, pos) &= cos(pos/10000^{2k/d_{model}}). \end{aligned} \tag{3.3}$$

In the previous expressions, pos is the position index, $2k$ and $2k+1$ are the dimension index and d_{model} is the dimension size of embedding.

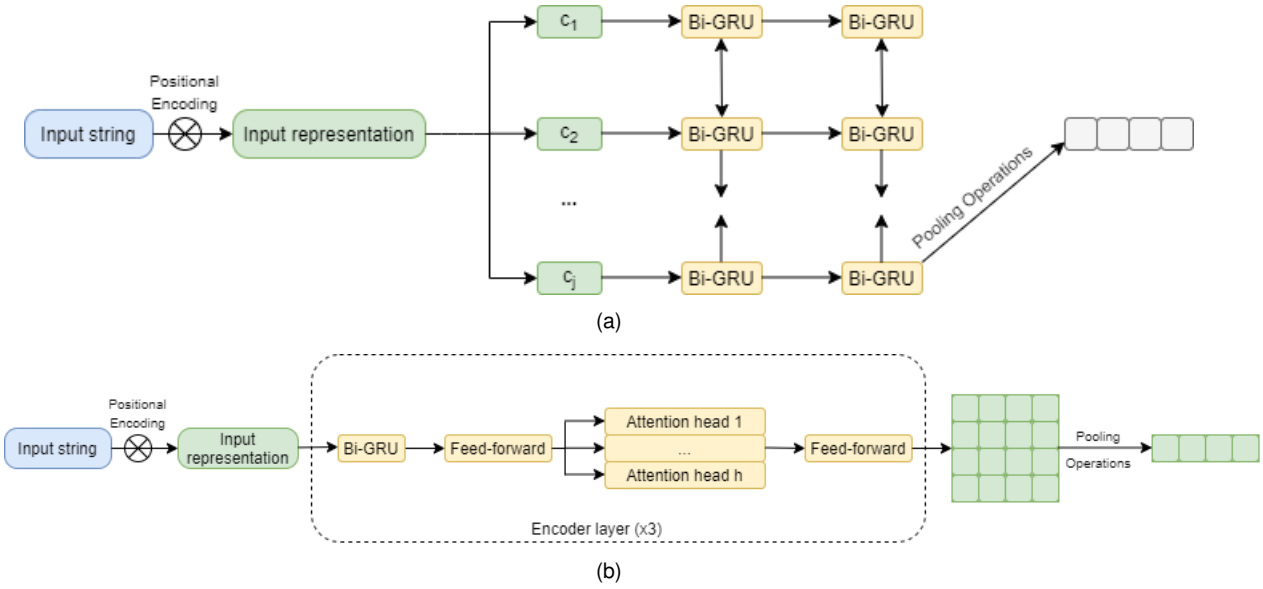

Figure 3.1: (a) The string encoder proposed by Santos et al. [16] with the extensions proposed by Borges et al. [17] (average and max-pooling operations) and the positional encoding propsosed in the present work. (b) Transformer extension proposed by Borges et al. [17] with the positional encoding proposed in the present work.

Hence, each character representation is no longer a byte representing a single bit set to one. It is, instead, that unitary representation added to the positional encoding in each byte (equation 3.4):

$$s_i = 10 * s_i + PE[s_i], \tag{3.4}$$

where s_i is a single byte representation of a character in a string s, and PE is the positional encoding calculated with equation 3.3. The single byte representation is multiplied by 10, so that the represented character is still clearly encoded.

The overall sentence encoder models are instances of the generic approaches illustrated in Figure 3.1. The feed-forward layers after the pooling operations are not represented.

3.3 Summary

The presented string matching technique considers lexical information about concepts, and the model's structure is summarized below.

1. Input layer: input a pair of concepts to the model taking into consideration character representation and positional encoding.
2. Encoder layers: map each character to a low dimensional vector using one of two cases:
   (a) a siamese architecture of bi-GRUs followed by pooling operations;
   (b) a bi-GRU followed by attention heads (from the R-Transformer model).
3. Output layers (linear layers): use the feature vector to classify (using the ReLU and sigmoid activation functions).

Chapter 4

Experimental Evaluation

In order to understand the experimental outcomes it is important to assess the datasets used as well as acknowledge and comprehend the underlying evaluation process, which is the emphasis of sections 4.1 and 4.2. The results and subsequent discussion are the focus of sections 4.3 and 4.4.

4.1 Description of the Datasets

In the course of this work a total of 16 different datasets were used. Each dataset corresponds to a collection of pairs of strings from different biomedical ontologies or collected from biomedical text corpora. In general, a positive instance corresponds to the case where both strings in a pair correspond to the same concept.

The proposed neural networks were trained with a generic balanced dataset featuring instances retrieved from Wikidata[1]. Wikidata is a secondary database (i.e. relies mainly on other resources to develop its content) with an ever-growing number of application cases. It is a large-scaled, multilingual, multidisciplinary, centralized, editable, structured, and linked knowledge-based resource being, therefore valuable for data integration and semantic interoperability among biomedical computer systems [19]. Each concept is associated to a code and taxonomic relationships, such as ”instance of” and ”subclass of,” connect these concepts, allowing data to be classified, categorized, and indexed [85].

In order to obtain a large and generic dataset, English concepts were retrieved as being ”instance of” certain sub-classes that were themselves identified as ”subclass of” or ”part of” the following main classes: *physiological condition, biological component, health science, biology, group or class of chemical substances, zootomy, veterinary medicine, comparative medicine, biomolecular structure, biological region, anatomical entity, biological system, general anatomical term and phenotype*. Positive instances correspond to pairs of concepts that are presented as synonyms in this platform and, therefore correspond to the same concept code (e.g. *induced miscarriage* and *abortion*) whereas negative instances were generated with randomly selected concepts that were not synonyms or generated with the replacement of words in a given list by their antonyms (e.g., concepts with *anterior* replaced by *posterior*). A significant portion of the non-matching pairs are not completely dissimilar, so that the dataset is representative and challenging for automated classification (e.g., *complex global pain syndromes* and *com-*

[1] https://www.Wikidata.org/wiki/

plex regional pain syndromes are non-matching pairs). This process led to the retrieval of 1 293 214 pairs of strings, being– half of them identified as positive synonyms. A stratified fold divided the data into a testing datasets with 1 250 000 instances and a testing dataset with 43 215. No pairs of strings in the training dataset were neither repeated on the same set nor repeated on the testing dataset, which was considered as a validation set from the same domain.

The remaining testing datasets were retrieved from various sources that are presented below.

- Ontology Lookup Service (OLS)[2], a repository for biomedical ontologies that desires to provide a single point of access to latest ontology versions. To date, there are 265 different ontologies. OLS allows term search to be general (gathering all ontologies where the exact term is found) or intra-ontology, where the selection of a certain term provides a list of synonyms present in that ontology. Concepts are identified by a code with the format *ontology code: concept code*, so even though a general search it is possible to identify the same term in various ontologies, these equal terms do not have the same code. The datasets used in this work that had OLS ontologies as sources present pairs of strings retrieved intra-ontology. These ontologies cover disease, anatomical and phenotypic domains and are briefly described below.

    1. Orphanet Rare Diseases Ontology Health Records (ORDO) - a structured, categorized vocabulary for rare diseases that captures links between diseases, genes, and other relevant features. Dedicated to rare disease, it derived form the Orphanet database [3], that is a multilingual resource whose database is populated from literature and validated by international experts. Moreover it links with other terminologies, databases or classifications.
    2. Human Disease Ontology (HDO) - a standardized ontology for human disease that was created with the goal of giving the biomedical community with consistent, reusable, and long-lasting definitions of human disease terminology, phenotypic traits, and related medical vocabulary disease concept.
    3. Foundational Model of Anatomy Ontology (FMA) - a domain ontology for human anatomy that represents a unified corpus of explicit declarative information.
    4. Uber-anatomy ontology (Uberon) - a cross-species anatomy ontology that classifies a wide range of items using standard anatomical criteria including structure, function, and developmental lineage. The ontology offers complete links to taxon-specific anatomical ontologies, allowing functional, phenotypic, and expression data to be integrated.
    5. Human Phenotype Ontology (HPO) - a standardized vocabulary focused on abnormalities' phenotype and clinical features present in human disease.
    6. Mammalian Phenotype Ontology (MPO) - a ontology whose pre-coordinated phenotype concepts, definitions and synonyms describe mammalian phenotypes.

    In all datasets that relied on these OLS ontologies, positive instances were identified as cross-reference concepts (i.e. with the same concept code).

[2] https://www.ebi.ac.uk/ols/index
[3] www.orpha.net

- Ontology Alignment Evaluation Initiative (OAEI)[4], a forum to collect benchmark datasets for ontology matching tools through controlled experiments. Its major purpose is to openly compare systems and algorithms on an equal basis so that anybody is allowed to make informed decisions regarding the best matching techniques. The OAEI organize a yearly evaluation event and provide the publication of tests and results of the event for further analysis. This event is divided in a series of tracks that have been growing over the years, with different foci being added. The ontologies or subsets of ontologies from which datasets used in this work took advantage of are all present in OAEI anatomy tracks and briefly described below.

  1. FMA - a human anatomy ontology already described in the context of the OLS ontologies.
  2. Mouse adult gross anatomy (MA) - a standardized nomenclature for anatomical structures in the postnatal mouse, developed by the Gene Expression Database. Several database services use it to describe gene expression patterns and other biological facts related to mouse anatomy.
  3. National Cancer Institute Thesaurus (NCIT)[5] - a reference terminology that provides structured representation of cancer related concepts, including related diseases, findings, abnormalities for basic and translational research, as well as for clinical care. This ontology is used by a broad variety of public and private partners. In the context of the OAEI anatomy track a subset of the NCIT that contains information about the human anatomy is considered.

  The datasets used in this work derived both from the independent ontologies and from alignments between the FMA and MA ontologies with the human related NCIT subset. In the first case, positive instances correspond to pairs of strings belonging to the same name set (i.e. group of synonyms and main labels of a class). In the second case, positive instances correspond to mapped synonyms in the performed alignment. In both cases, negative instances all have an ISub similarity ≥ 0.7.

- Systemized Nomenclature of Medicine – Clinical Terms (SNOMED CT), a standardized, international, multilingual core set of clinical healthcare terminology that can be used in electronic health records.This terminology focuses on encoding the meanings employed in health information and enhancing patient care by facilitating effective clinical data recording. The set of clinical healthcare terminology includes concepts related to symptoms, procedures, clinical observations, diseases, clinical observations, organisms and other etiologies, body structures, medications, chemicals, equipment and specimens. Each term is associated to a concept code. In the dataset derived from this terminology, positive instances correspond to English concepts with the same SNOMED CT code.

- National Center for Biotechnology Information (NCBI) disease corpus[6], a resource for disease name recognition and normalization. In the datasets derived from this resoursce, positive instances correspond to strings marked as entities in the text with the same SNOMED CT code.

[4] http://oaei.ontologymatching.org/
[5] http://ncit.nci.nih.gov/ncit-browser/
[6] https://www.ncbi.nlm.nih.gov/research/bionlp/data/di sease

Table 4.1: Datasets Description

Dataset	Source	Total	Positive	Dissimilar	
				Positive	Negative
Wikidata (train)	Wikidata	1 250 000	625 000	1.02%	0.00%
Wikidata (test)	Wikidata	43 214	21 607	0.99%	0.00%
ORDO	OLS	116 860	58 430	33.35%	0.10%
HDO	OLS	98 494	49 247	7.55%	0.38%
FMA	OLS	198 306	99 153	2.28%	0.00%
Uberon	OLS	300 322	150 161	7.07%	0.30%
HPO	OLS	350 234	175 117	8.93%	0.14%
MPO	OLS	691 680	345 840	1.44%	0.01%
FMA + NCIT subset - 1	OAEI	26 752	9 229	0.00%	0.00%
FMA + NCIT subset - 2	OAEI	20 000	7 082	0.00%	0.01%
MA + NCIT subset - 1	OAEI	6 705	1 744	0.00%	0.01%
MA + NCIT subset - 2	OAEI	6 000	1 596	0.00%	0.01%
NCIT subset	OAEI	7 592	3 796	0.09%	0.01%
MA	OAEI	768	384	0.00%	0.00%
SNOMED CT	SNOMED CT	3 988	1 994	0.00%	0.00%
NCBI disease entities	NCBI Disease Corpus	15 541	7 770	0.76%	0.00%

All datasets are presented in the Table 4.1 detailing their source, the total number of pairs, the total number of positive instances, the percentage of pairs with matching concepts that are completely dissimilar and the percentage of pairs with non-matching concepts that are completely dissimilar. Non-matching concepts were considered when the pairs had a Jaro-Winkler similarity of 0.

Apart from the OAEI datasets derived from the alignment between FMA or MA with the NCIT subset, all other datasets are balanced (i.e. half of the instances correspond to matching concepts). The four imbalanced datasets have more non-matching than matching concepts. The existence of totally dissimilar matches, occurs with an incidence higher than 1% only in 8 of the datasets and from these, only the ORDO dataset has an incidence superior to 10%. None of the datasets present a high percentage of totally dissimilar non-matches (they are all below 1 percent).

Additionally, it is important to refer that none of the datasets present cross-language pairs. The English language was considered in all cases.

4.2 Evaluation Methodology

The proposed approach was compared to baseline methods over all the testing tests. Specifically, the considered baselines consist of (1) individual string similarity metrics, with a threshold value α tuned for optimal F1 score and (2) the Borges et al. [17] neural network (i.e. the neural networks without the positional encoding component). F1 score tuning was performed by obtaining the threshold value that obtained the best average of all individual dataset scores per α value (between 0.1 and 0.9 with a 0.1

step). All results were measured in terms of accuracy, precision, recall, and the F1 measure.

The accuracy measure tries to convey the overall effectiveness of a classifier, by evaluating the proportion of correct decisions that are returned by the method being evaluated [16]:

$$Accuracy = \frac{TruePositives + TrueNegatives}{TotalNumberofinstances}. \tag{4.1}$$

Precision and recall focus on different per-class quality aspects of a classification system. Precision is described by the number of items accurately assigned to a class divided by the total number of things assigned to that class [16]. In the case of string-matching, a binary classification, precision is defined by the agreement of actual labels with the positive labels predicted by the classifier:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives}. \tag{4.2}$$

Recall is the ratio within a class between correctly assigned items over the total instances of that class [16]. It measures, therefore the effectiveness of a binary classifier to retrieve positive labels (equation 4.3):

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives}. \tag{4.3}$$

F1 score is the harmonic mean of precision and recall, which is important due to the fact that precision can be increased at the expense of recall:

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}. \tag{4.4}$$

In terms of hyper-parameter choices and model training strategies, it is important to notice that there was no hyper-parameter tuning. The tests with models leveraging RNNs used RNN hidden layer size of 60, a hidden layer size of 120 in the dense layer processing the result from the interaction between the string representations, a batch size of 32, and the Adam [30] optimizer with a learning rate of 0.001. The experiments with the R-Transformer encoder considered 3 layers of dimension 512, leveraging 8 attention heads in parallel.

Model training is performed for a maximum of 20 epochs over the training dataset (the second Wikidata dataset), with early stopping being activated when the training loss does not decrease after 3 epochs. Other experiments were conducted in regards to the assessment of the impact of the training dataset size or the contribution of model fine-tuning with specific in-domain data. The in-domain experiments were done with two-fold cross validation, according to a stratified sampling procedure. This means that all the available pairs of concepts in a dataset were split into two subsets, with an equal proportion of positive and negative instances. In each dataset, accuracy, precision, recall and F1 score were calculated by obtaining the sum of true positives, true negatives, false positives and false negatives for both folds and calculating the aforementioned metrics.

4.3 Results

Tables 4.2, 4.3, and 4.4 present the obtained results. Results presented in bold are the best result in terms of that metric for that dataset in the current table. Highlighted results are the best overall neural

Table 4.2: Results with the Levenshtein and Jaro-Winkler metrics

Testing Dataset	Levenshtein ($\alpha = 0.1$)				Jaro-Winkler ($\alpha = 0.1$)			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Wikidata	48.97	**49.53**	97.96	**65.80**	**49.00**	49.49	**98.03**	65.78
ORDO	43.23	46.37	86.47	60.37	**45.56**	**47.68**	**91.13**	**62.61**
SNOMED CT	50.00	50.00	100.00	66.67	50.00	50.00	100.00	66.67
HDO	**48.95**	49.47	97.90	65.73	49.69	**49.85**	**99.39**	**66.40**
NCBI	49.41	49.76	98.82	66.19	**49.63**	**49.81**	**99.25**	**66.33**
HPO	49.29	49.64	98.60	66.03	**49.48**	**49.74**	**98.98**	**66.20**
Uberon	49.75	49.87	99.50	66.44	**49.84**	**49.92**	**99.68**	**66.53**
FMA	49.85	49.93	99.70	66.54	**49.93**	**49.97**	**99.86**	**66.61**
NCIT subset	**49.72**	49.86	**99.45**	66.42	49.70	**49.87**	99.39	66.42
MA	49.61	49.80	99.22	66.32	**49.70**	**49.87**	**99.39**	**66.42**
MPO	49.37	49.68	98.75	66.11	**49.70**	**49.87**	**99.39**	**66.42**
FMA + NCIT subset - 1	**34.49**	34.49	**100.00**	51.29	34.48	**34.51**	99.97	**51.31**
FMA + NCIT subset - 2	**35.41**	35.41	**100.00**	52.30	35.40	**35.43**	99.96	**52.32**
MA + NCIT subset - 2	26.60	26.60	100.00	42.02	26.60	**26.65**	100.00	**42.09**
MA + NCIT subset - 1	26.01	26.01	100.00	41.28	26.01	**26.06**	100.00	**41.34**

network results in terms of that metric for that dataset.

Table 4.2 details the results over each dataset with traditional methods. Although experiments were conducted with 6 traditional string similarity measures, only the results with the Levenshtein and Jaro-Wrinkler metrics are presented in this section since they obtained better overall results and can, therefore, fairly be compared with the proposed neural network results. The remaining results are presented in Appendix A. It is also important to notice that the threshold value α was tuned for optimal F1 scores, and has, in both cases, the value of 0.1. Hence, it is highly likely that most pairs are considered positive instances, which results in the high recall values observed. Table 4.3 presents the results obtained with RNN and R-Transformer model proposed by Borges et al. [17] which correspond to the proposed neural networks without positional encoding. Finally, Table 4.4 presents the results with the proposed neural networks. All neural network models outperform traditional measures in terms of accuracy, precision and F1 score. Focusing on Table 4.3, even though the RNN obtains better results than the R-Transformer model in more than half the datasets, there is not a great difference between the performance of the models. In terms of accuracy it the RNN achieves better results in 10 out of 15 datasets, however, in terms of F1 score it performs better in only 8 out of 15 datasets, which is related to the high recall values obtained in the R-Transformer model. In regards to table 4.4 the proposed R-Transformer outperforms the proposed bi-GRU in most cases.

In order to to evaluate the value of the proposed extension in each neural network, the comparison of the bi-GRU and R-Transformer model with or without positional encoding can be used as an ablation test. In both neural networks, the positional encoding resulted in the models performing better, in general. Particularly, in the RNN the proposed approach outperforms the Borges et al. [17] model in 13 out

Table 4.3: Results with the Borges et al. models without Positional Encoding

Testing Dataset	RNN				R-Transformer			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Wikidata	**89.81**	**89.78**	89.82	**89.77**	89.48	87.61	**91.98**	89.71
ORDO	86.18	90.26	81.09	85.38	**88.49**	**92.69**	**83.55**	**87.84**
SNOMED CT	**83.75**	76.85	**96.67**	**85.59**	83.37	**77.14**	94.84	85.04
HDO	81.39	78.15	87.19	82.36	**86.07**	**82.19**	**92.08**	**86.81**
NCBI	76.10	74.54	79.27	76.77	**76.77**	**74.68**	**80.92**	**77.62**
HPO	**68.40**	**66.67**	73.54	69.86	68.08	65.49	**76.39**	**70.45**
Uberon	67.12	**64.68**	75.43	69.57	**67.98**	64.37	**80.61**	**71.51**
FMA	65.67	62.34	**79.28**	69.71	**70.86**	**67.96**	78.92	**72.96**
NCIT subset	**69.73**	**66.84**	78.27	**72.05**	66.91	62.71	**83.50**	71.55
MA	**63.80**	**62.40**	68.90	65.49	62.11	59.08	**78.02**	**67.19**
MPO	**59.76**	**58.55**	66.87	**62.35**	57.42	56.17	**67.57**	61.26
FMA + NCIT subset - 1	**67.42**	**51.64**	88.97	**65.25**	62.40	47.64	**92.16**	62.72
FMA + NCIT subset - 2	**66.24**	**51.33**	88.80	**64.95**	61.05	47.39	**92.33**	62.54
MA + NCIT subset - 2	**65.85**	**43.49**	93.82	**59.34**	56.90	37.72	**95.90**	53.98
MA + NCIT subset - 1	**66.04**	**43.07**	93.69	**58.95**	57.14	37.31	**96.35**	53.71

of 15 datasets in terms of accuracy, and 11 datasets in terms of F1 score and precision. In terms of recall the results are better in only 6 datasets. Regarding the R-Transformer model, the proposed approach outperforms the Borges et al. [17] in terms of accuracy, F1 measure and precision in 13 out of the 15 datasets. However, in terms of recall it only obtains better results in 2 datasets. Furthermore the proposed R-Transformer model outperforms all three other models in terms of accuracy, precision and F1 measure in 9 out of the 15 datasets. Hence, the combination of the R-Transformer attention mechanism with textual input representation including positional encoding leads to better overall results.

The 4 imbalanced datasets (FMA + NCIT subset 1 & 2 and MA + NCIT subset 1 & 2) correspond to the worst accuracy and F1 scores in the R-Transformer model. Additionally, they correspond to the cases where the proposed RNN outperforms the proposed R-Transformer model. It is also interesting to notice that the recall is always high (above 85 %) in these cases whilst precision is rather low (below 54%). These imbalanced classes not only have more negative instances than positive, but are also datasets where negative instances in the pairs of strings presented a high similarity between them (ISub $\geq$ 0.7). Thus, contributing to the identification of false positives.

Datasets with pairs retrieved from disease related ontologies (ORDO, HDO, NCBI) or general health terms (SNOMED CT) obtain better results than datasets related to phenotypes and anatomy terms. This can be related to the classes from which the Wikidata training set was retrieved (i.e., anatomy and phenotype terms are underrepresented in relation to disease related terms).

Another interesting note, is that the percentage of totally dissimilar pairs that are synonyms does not seem to have an influence on the results. The ORDO presents the highest percentage of totally dissimilar matches (33.35%) and is also the testing dataset with the highest scores (excluding the Wikidata

Table 4.4: Results with the proposed RNN and R-Transformer models

Testing Dataset	Proposed RNN				Proposed R-Transformer			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Wikidata	89.87	89.48	90.34	89.87	**93.00**	**92.33**	**93.80**	**93.03**
ORDO	86.82	90.14	82.65	86.19	**90.15**	**93.23**	**86.55**	**89.73**
SNOMED CT	82.95	76.24	**95.88**	84.91	**84.34**	**79.57**	92.55	**85.54**
HDO	81.76	78.59	87.32	82.67	**84.41**	**82.02**	**88.16**	**84.92**
NCBI	75.48	73.43	79.62	76.34	**76.46**	**74.36**	**80.73**	**77.34**
HPO	68.97	67.72	72.48	69.94	**73.69**	**73.57**	**73.95**	**73.68**
Uberon	68.28	65.73	76.39	70.58	**71.77**	**69.24**	**78.38**	**73.45**
FMA	71.94	71.94	**75.97**	72.95	**74.29**	**75.90**	71.20	**73.39**
NCIT subset	69.83	67.43	76.73	71.73	**70.03**	**66.68**	**80.02**	**72.69**
MA	63.93	62.72	67.62	65.02	**67.32**	**65.21**	**74.22**	**69.42**
MPO	65.82	66.28	64.39	65.23	**68.33**	**69.62**	**65.09**	**67.18**
FMA + NCIT subset - 1	**69.48**	**53.58**	85.95	**65.93**	66.05	50.46	**86.57**	63.66
FMA + NCIT subset - 2	**68.84**	**53.81**	86.07	**66.11**	64.94	50.38	**86.15**	63.45
MA + NCIT subset - 2	**68.42**	**45.21**	91.22	**60.36**	63.90	41.77	**92.52**	57.44
MA + NCIT subset - 1	**68.75**	**44.77**	90.88	**59.88**	64.51	41.76	**92.08**	57.34

validation set). Furthermore, the datasets that don't present any dissimilar matches do not perform necessarily better or worse (e.g. comparing the SNOMED CT and MA + NCIT subset - 1 datasets' results one can infer that it does not have a direct influence).

In order to observe correctly and wrongly classified pairs, appendix B presents examples for the R-Transformer model with positional encoding (the model that overall obtained better results). Since these examples are not a signification portion of the whole datasets (due to their large size) they are merely demonstrative.

4.3.1 Impact of training dataset size results

With the goal of assessing the impact of the training dataset size, experiments were conducted in which the size of the training dataset was reduced (using stratified folds) and evaluated the effect on the accuracy and F1 score evaluation metrics, for the following testing datasets: Wikidata, which is considered from the same domain; ORDO and MA + NCIT subset - 1 which obtained better and worse results in the previous tests, respectively. This impact was measured by varying the dataset size between 1 250 000, 125 000, 12 500 and 1 250 instances. Figure 4.1 illustrates the results for this set of experiments.

Both the ORDO and Wikidata datasets show that a greater amount of training data leads to better results, independently of the model used. The R-Transformer model continues to outperform the RNN in most cases, although the difference between the two models' scores is smaller with smaller training sets. Results on the ORDO dataset show the least influence of varying the training set size, obtaining for smaller training sets (12 500 and 1250) better results than the Wikidata, which was considered a

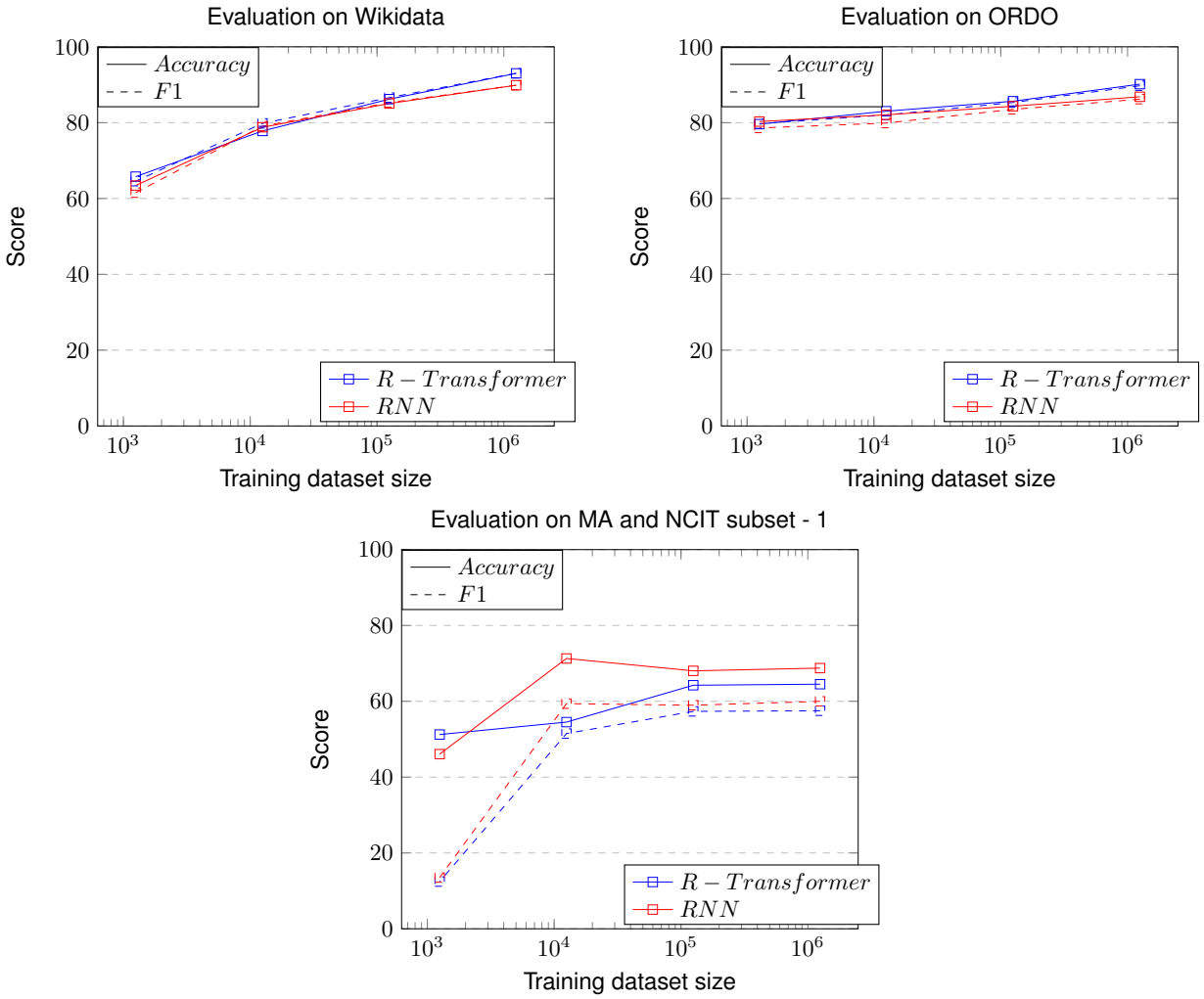

Figure 4.1: Results changing the training dataset size

validation set. Once again, influence on representation of disease related terms is hypothesized to have an influence. On the other hand, experiments with the MA + NCIT subset - 1 were different than expected. Although, in the R-Transformer model the pattern maintains itself (higher scores for bigger training sets), for the RNN model, the 12 500 size dataset presented the better scores. Moreover, in this case the difference between the datasets scores is bigger when using smaller training sets (with the R-Transformer model outperforming the RNN model in the extreme of the smallest training set).

For a small training generic dataset (1 250 pairs of strings) the results are rather discouraging, obtaining in both MA + NCIT subset - 1 and Wikidata worst scores than some of the traditional approaches. In particular for the MA + NCIT subset - 1, the F1 score is extremely low, being less that 20% for both models. The imbalanced dataset might contribute to this (e.g. precision being extremely low due to the identification of many false positives). Hence, the proposed models are considered a good alternative when using a generic training dataset, if it is big and representative enough of the biomedical domain.

Table 4.5: In-domain or in-ontology results for the R-Transformer model

Testing Dataset	Fine-tuning				Training with all ontologies			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
ORDO	**95.87**	**97.19**	**94.47**	**95.81**	92.22	96.47	87.66	91.85
SNOMED CT	**98.76**	**97.56**	**100.00**	**98.77**	91.02	88.44	94.49	91.36
HDO	**91.80**	**95.04**	**88.22**	**91.50**	89.96	91.63	87.94	89.74
NCBI	**88.45**	**97.39**	**79.04**	**87.26**	82.90	84.24	80.75	82.45
HPO	88.94	93.60	85.28	89.25	**89.33**	**94.35**	**85.74**	**89.84**
Uberon	82.38	**89.70**	73.15	80.59	**87.59**	86.60	**88.95**	**87.76**
FMA	**97.04**	96.37	**96.37**	**97.06**	94.67	96.72	92.47	94.55
NCIT subset	81.04	**86.44**	73.61	79.51	**85.45**	87.60	**82.59**	**85.02**
MA	74.80	79.43	67.66	73.07	**89.55**	**90.58**	**89.50**	**90.03**
MPO	**90.22**	**96.30**	**83.66**	**89.54**	89.24	93.71	81.10	88.29
FMA + NCIT subset - 1	**87.63**	**86.51**	76.17	**81.01**	84.38	73.96	**84.46**	78.87
FMA + NCIT subset - 2	**86.40**	**84.88**	75.15	**79.72**	83.62	73.44	**84.28**	78.49
MA + NCIT subset - 2	**90.35**	**82.10**	74.02	77.85	87.02	69.91	**90.00**	**78.69**
MA + NCIT subset - 1	**89.82**	**83.21**	76.04	**79.46**	86.93	69.04	**78.62**	73.52

4.3.2 Fine-tuning and All-Domain Results

The neural model that performed better in most cases (the R-Transformer model) was chosen to design an additional set of experiments to evaluate the effect of how the training domain (i.e. the Wikidata training dataset) affected performance when evaluating the results in same-domain settings. On the one hand, fine-tuning experiments were conducted, where the training set included data from the ontology or domain being tested. In particular, fine-tuning experiments had the models trained with the full Wikidata training dataset, fine-tuned with each of the folds independently and tested on the opposite fold (not included on the fine-tuning training). It was opted to fine-tune the pre-trained model instead of training it from scratch every time since these are time consuming and resource intensive processes. On the other hand, an experiment where a single 2-fold training was executed with the training dataset including data from all ontologies at once was also conducted. Specifically, each fold contained the Wikidata training dataset in its totality and a fold of each dataset which was tested individually on the other fold of each dataset and vice-versa. Hence, in each test it was ensured that the dataset being evaluated was not present at train time. Since the Wikidata testing set was used as reference for the same domain as the training dataset (whilst the remaining datasets were considered cross-domain experiments), I expected that the scores obtained in both cases to become more similar to the ones in the Wikidata dataset. Table 4.5 presents the obtained results for these tasks, where it can observed that in both cases the scores obtained were better in the 4 evaluation metrics for all datasets. For the R-Transformer model, accuracy scores were all above 80% and F1 scores all above 70%.

Fine-tuning involves initializing the deep learning process with weights of the pre-trained model, and training it with the new data. The model is, therefore, adjusting its weights to the new data. In the

case of imbalanced datasets this can be extremely important, since it can perform a class re-weighting (i.e. to take into account asymmetry of cost error directly during the training of the classifier). In my experiments, the results for the imbalanced datasets all improved significantly, increasing at least 21% in terms of accuracy and 16% in terms of F1 score. These datasets also presented better results in the fine-tuned models than when training with all ontologies, as expected. Seeing that there is less adjustment to the imbalanced classes, recall maintains itself higher in the model trained from scratch. It is also interesting to notice that the datasets which improve less with fine-tuning are the ORDO, in terms of accuracy, and the *Mouse Ontology*, in terms of F1 score. In the first case it may be due to the fact that the model already obtained high scores without fine-tuning, and hence the rare disease ontology was probably already well represented in the model. In the second case, it is important to refer that the mouse ontology is the smallest dataset, with only 768 pairs of strings. Consequently, when fine-tuning the model with each fold of 384 pairs, the available data might not be enough to adjust the weights significantly to the domain.

Concerning the results obtained from the model trained with a dataset including data from all ontologies at once, it is significant to note that these domains and ontologies are not only being added to the training set but also enlarging it significantly (2 193 272 instances in each fold). As shown previously, the size of the training dataset also influences the outcome and, therefore training with a dataset that is approximately 1.75 times larger than the original training dataset contributes to the generally better obtained scores. Furthermore, even though it is ensured that the testing dataset is not present in the training set in each fold, due to different ontologies covering the same domains, their might be pairs of strings that are present in more than one testing dataset and, therefore an exact pair being tested can be present in the training set. In particular, this experiment showed that the ORDO dataset presented worse results than with initial training; the rest of disease or general medical terms related datasets all improved in comparison to initial training, but performed worse than fine-tuning the model. Most balanced datasets with anatomy or phenotype related terms obtained improved scores. Assuming that, as mentioned before, the initial Wikidata training dataset was underrepresented in terms of anatomy of phenotype related concepts, then these results support this idea. These datasets benefit from each other being in the training dataset and the percentage of anatomical and phenotype representation increases. Disease or generic related datasets perform better with fine-tuning because the initialized weights already benefited them and are then adjusted in-domain, not needing each other to perform better.

4.4 Discussion

This thesis presents extensions of the neural string matching methods developed by Santos et al. [16] and Borges et al. [17], augmenting the proposed architecture to include positional embeddings and assessing performance in cross-domain settings (the main goal for biomedical concept alignment), in-domain settings and when varying the amount of training data. The proposed models were tested on different datasets, covering several biomedical ontologies and domains (i.e., disease, anatomy and phenotype related ontologies). A comparison was performed between the proposed neural models and classical string similarity metrics, where the proposed neural network models consistently outperformed

the traditional techniques. The influence of positional encoding was also studied through a direct comparison between the neural network models with and without this component. Findings showed that, in most cases the inclusion of positional embeddings was beneficial. Between the two proposed models, the R-Transformer model achieved better results than the one based on RNNs except for the 4 imbalanced datasets (out of 15 testing datasets). In regards to the impact of the training dataset size, the experimental outcome demonstrated that neural network models are a good alternative to traditional measures in the case of the training set being large enough. Finally, in-domain and all-domain analysis revealed that when training data included in-ontology terms the models accomplished better outcomes.

One of the great challenges in aligning biomedical concepts is that they do not follow a lexical similarity pattern. The use of deep neural networks is capable of identifying highly dissimilar concepts as synonyms (e.g. *ovocytes* and *egg* from the NCIT subset) but also highly similar concepts as not synonyms (e.g. *larynx muscle* and *pharynx muscle* from the MA dataset). The addition of a positional encoding in the biomedical domain can better tune this classification by taking into consideration character position, and therefore adjusting weights according to it. For instance it can consider a sequence of characters in the beginning, end or middle of a concept to be more or less important (e.g. the identification of a letter or number in the end of the concept can mean it is different than the other concept like in the HPO dataset terms *digit* and *digit i 1* that were only correctly classified as negative instances by the models with positional encoding; or the identification of a suffix in a word that can be irrelevant like in the *propanoic acid* and *propoic acid* concepts from the same dataset that were also only correctly classified as synonyms in the models with positional encoding).

On the other hand, it is also interesting to look at cases where none of the models correctly classified certain pairs of strings. For example, in the FMA dataset, the pair of concepts *hippocampus proper* and *hippocampus major* were always classified as negative instances when, in reality they are synonyms whilst in the HDO dataset *gastric liposarcoma* and *pediatric liposarcoma* were always classified as synonyms when they are not. Although these are randomly selected examples and are not representative of all datasets' instances where the classifiers fail it is interesting to note that the first word seems to have less importance than the second word on the concepts. In this case, character positional might have even reinforced this word position importance. These examples highlight the issue of biomedical lexical complexity and diversity.

Another challenge related to aligning biomedical concepts is the ontology diversity regarding the biomedical sub-domains coverage. This issue was highlighted throughout my experiments. Specifically, it was noted that disease and general medical terminology ontologies performed better than anatomy and phenotype related ontologies. However, these anatomy or phenotype related ontologies did not necessarily present a higher percentage of dissimilar synonyms. As mentioned in the introduction, similarity tendencies are often related to a certain domain [20], which poses an additional difficulty in supervised classification of synonyms (i.e. the training set has to represent all biomedical sub-domains in a balanced way). Even though, the Wikidata training was obtained from various classes, I hypothesize anatomy related sub-classes are less represented in the training set. This is in accordance with the fact that anatomy datasets leveraged each others' presence in the training set in the all-domain experiments. The assessment using an all-domain training set is also interesting to mention in the aspect of overlapping domains. Due to many source ontologies covering the same domain, the same pair of terms can be present in different datasets (e.g. *ear lobule* and *lobule of pinna* are synonymous concepts present

as a pair in both the MA and FMA datasets). Hence, it seems to be extremely challenging to find a large and generic dataset that not only includes all relevant biomedical classes but that also contains them proportionally so that it is applicable to all ontologies without the need to rely on all-domain training.

Chapter 5

Conclusion

In this dissertation, a deep learning approach to aid in biomedical concept alignment was presented. It specifically addresses this problem using approximate sting marching, with the objective of classifying the type of relationship (synonyms or not) between two terms. The present chapter provides a summary of the work's key contributions and suggests future research direction in biomedical concept alignment using deep learning approaches.

5.1 Contributions

Throughout this thesis and the underlying work behind it, it was clear that aligning biomedical concepts is a challenging task. Tackling this problem as an approximate string matching task and leveraging deep neural networks showed to be an efficient method to model biomedical concepts.

In particular, two main neural network model architectures were studied in the context of this work. The R-Transformer model obtained, in most cases, better results then the bi-GRU. Related work in NLP tasks have increasingly relied on this architecture to model textual information. Hence, the achieved results are mostly concordant with recent studies. Moreover, the inclusion of positional encoding embedding in the neural networks' encoder layers also showed to improve results, in most cases. Therefore, the proposed extension enhances model performance in aligning biomedical concepts.

When studying the impact of the training dataset size it was observed that approximate string matching through the deep neural networks perform better with larger sets of training data. Nevertheless these models presented a limitation on smaller training sets (specifically in the order of 1 000 pairs of concepts), which lead in some cases to worst classifications than traditional approaches.

Regarding the datasets used in the present work, it is important to notice that they covered a wide range of biomedical sub-domains and had a variety of source ontologies and terminologies. It was possible to observe that results differed within domains and datasets, which shows the overcoming difficulty in obtaining a single resource which covers the vast biomedical scope. At the same time, this shown difficulty also highlights the importance of obtaining models that efficiently align concepts, so that the consideration of the scope as a whole is possible information is, in fact, interoperable between systems and people.

Furthermore, studies related to the impact of in-domain and all-domain training achieved better re-

sults. Specifically in cases where there is lack of abundant information (a small dataset) leveraging from other ontologies in the training set was found useful. This is can be related to the fact that many ontologies were developed independently and cover overlapping domains. However, the usage of a generic datasets is useful to perform concept alignment even when the ontology is not known, or to leverage overcoming ontology-matching challenges. All things considered, a large training dataset that considers and represents the most biomedical categories it can should be aimed for. Wikidata, still seems to be a good option for this data retrieval since it is a large-scale collaborative ontological medical database.

In conclusion, using the proposed approach, with a large and generic training set can provide improvements and contributions in identifying biomedical synonyms. Enhancing this identification is useful to aid in other tasks such as ontology alignment or synonym discovery which improves standardization of biomedical terminology and information interoperability (e.g. synonymous information identification between several authors and contexts), contributing therefore to the improvement of healthcare.

5.2 Future Work

Building on the results reported on this thesis, there are several interesting directions for further work in exploring neural architecture improvements, textual representation methodologies and taking into consideration the vast biomedical data scope.

A straightforward future experiment would be to retrieve a larger amount of data from Wikidata for the training dataset, including a larger amount of classes in order to obtain more anatomy and phenotype related terms. However, it is also important to notice that these models are time and resource consuming, so there should be a balance between enlarging the dataset and computational effort. Fine-tuning pre-trained models has showed to be a successful alternative for this balance (e.g. using the already trained model and adding more data).

In regards to the textual representation of the input concepts other extensions could also be considered. Maintaining the positional encoding, it would be interesting to generalize the word and character embeddings as continuous functions over a variable (position) instead of being defined as independent vectors. Recently Wang et al. [23] demonstrated this to be more efficient. Another textual representation extension would be to consider Wasserstein distance regularized sequence representation, proposed by Yu et al. [86].

Keeping in line with recent advances in natural language processing, other extensions to the Transformer model might be beneficial to the task in hand (e.g. considering the residual attention layer Transformer proposed by He et al. [87]).

Other interesting experiment directions would be testing cross-language datasets, therefore contributing to the multilinguality challenge [84] mentioned in Chapter 2. For this, Wikidata could still be used for concept retrieval since it is a multilingual resource. However, it would be possible that more resources had to be considered in order to cover a greater amount of languages (e.g. the term *lower leg* does not have correspondence in Portuguese in Wikidata).

Finally, taking into account state-of-the-art approaches in biomedical concept and ontology alignment, it would also be interesting to include contextual information or definitions. In this case the datasets would have to be reviewed to include such information, taking into consideration that context should be

given by reliable resources. Moreover, Wikidata would also not be adequate to be the sole source of the generic dataset (e.g., a pre-processing could be performed by matching Wikidata terms with wikipedia definitions could be made to form the dataset).

Bibliography

[1] F. Vezzani and G. M. Di Nunzio, "On the formal standardization of terminology resources: The case study of trimed," in *Proceedings of The 12th Language Resources and Evaluation Conference*, pp. 4903–4910, 2020.

[2] J. C. Andrews, F. Bogliatto, H. W. Lawson, and J. Bornstein, "Speaking the same language: using standardized terminology," 2016.

[3] A. Awaysheh, J. Wilcke, F. Elvinger, L. Rees, W. Fan, and K. Zimmerman, "A review of medical terminology standards and structured reporting," *Journal of Veterinary Diagnostic Investigation*, vol. 30, no. 1, pp. 17–25, 2018.

[4] J. Ingenerf, J. Reiner, and B. Seik, "Standardized terminological services enabling semantic interoperability between distributed and heterogeneous systems," *International journal of medical informatics*, vol. 64, no. 2-3, pp. 223–240, 2001.

[5] A. Geraci, *IEEE standard computer dictionary: Compilation of IEEE standard computer glossaries*. IEEE Press, 1991.

[6] S. V. Jardim, "The electronic health record and its contribution to healthcare information systems interoperability," *Procedia technology*, vol. 9, pp. 940–948, 2013.

[7] T. B. Patrick, H. K. Monga, M. C. Sievert, J. H. Hall, and D. R. Longo, "Evaluation of controlled vocabulary resources for development of a consumer entry vocabulary for diabetes," *Journal of medical Internet research*, vol. 3, no. 3, p. e24, 2001.

[8] B. Aldosari, A. Alanazi, and M. S. Househ, "Pitfalls of ontology in medicine," *Studies in health technology and informatics*, 2017.

[9] E. Schumacher and M. Dredze, "Learning unsupervised contextual representations for medical synonym discovery," *JAMIA open*, vol. 2, no. 4, pp. 538–546, 2019.

[10] N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M.-A. Storey, C. G. Chute, *et al.*, "Bioportal: ontologies and integrated data resources at the click of a mouse," *Nucleic acids research*, vol. 37, no. suppl_2, pp. W170–W173, 2009.

[11] J. Euzenat, P. Shvaiko, *et al.*, *Ontology matching*, vol. 18. Springer, 2007.

[12] D. Faria, C. Pesquita, I. Mott, C. Martins, F. M. Couto, and I. F. Cruz, "Tackling the challenges of matching biomedical ontologies," *Journal of Biomedical Semantics*, vol. 9, no. 1, pp. 1–19, 2018.

[13] L. L. Wang, C. Bhagavatula, M. Neumann, K. Lo, C. Wilhelm, and W. Ammar, "Ontology alignment in the biomedical domain using entity definitions and context," *arXiv preprint arXiv:1806.07976*, 2018.

[14] C. Jiang and X. Xue, "Matching biomedical ontologies with long short-term memory networks," in *Proceding of the IEEE international conference on bioinformatics and biomedicine*, 2020.

[15] P. Kolyvakis, A. Kalousis, B. Smith, and D. Kiritsis, "Biomedical ontology alignment: an approach based on representation learning," *Journal of Biomedical Semantics*, vol. 9, no. 1, pp. 1–20, 2018.

[16] R. Santos, P. Murrieta-Flores, P. Calado, and B. Martins, "Toponym matching through deep neural networks," *International Journal of Geographical Information Science*, vol. 32, no. 2, pp. 324–348, 2018.

[17] L. Borges and B. Martins, "Evaluating neural methods for approximate string matching and duplicate detection," tech. rep., Instituto Superior Técnico, 2019.

[18] M. Mohri, A. Rostamizadeh, and A. Talwalkar, "Foundations of machine learning.[sl]," 2012.

[19] H. Turki, T. Shafee, M. A. H. Taieb, M. B. Aouicha, D. Vrandečić, D. Das, and H. Hamdi, "Wikidata: A large-scale collaborative ontological medical database," *Journal of biomedical informatics*, vol. 99, p. 103292, 2019.

[20] D. Lopresti and G. Wilfong, "Cross-domain approximate string matching," in *In Procedings of the 6th International Symposium on String Processing and Information Retrieval*, 1999.

[21] J. R. Venable, J. Pries-Heje, and R. L. Baskerville, "Choosing a design science research methodology," 2017.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceding of the Annual Meeting on Advances in neural information processing systems*, 2017.

[23] B. Wang, D. Zhao, C. Lioma, Q. Li, P. Zhang, and J. G. Simonsen, "Encoding word order in complex embeddings," *arXiv preprint arXiv:1912.12333*, 2019.

[24] P. Simmons and D. Young, *Nerve cells and animal behaviour*. Cambridge University Press, 2010.

[25] E. Kandel, J. Schwartz, and T. Hessell, "Propagated signaling: the action potential," in *Principles of neuroscience*, pp. 150–175, McGraw Hill, New York, 2000.

[26] S. C. Kleene, *Representation of events in nerve nets and finite automata*. Princeton University Press, 2016.

[27] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[28] S. Haykin, *Neural networks: a comprehensive foundation*. Prentice Hall PTR, 1994.

[29] Y. Goldberg, "A primer on neural network models for natural language processing," *Journal of Artificial Intelligence Research*, vol. 57, pp. 345–420, 2016.

[30] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[31] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, “A survey of the recent architectures of deep convolutional neural networks,” *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

[32] L. Borges, B. Martins, and P. Calado, “Combining similarity features and deep representation learning for stance detection in the context of checking fake news,” *Journal of Data and Information Quality (JDIQ)*, vol. 11, no. 3, pp. 1–26, 2019.

[33] H. Apaydin, H. Feizi, M. T. Sattari, M. S. Colak, S. Shamshirband, and K.-W. Chau, “Comparative analysis of recurrent neural network architectures for reservoir inflow forecasting,” *Water*, vol. 12, no. 5, p. 1500, 2020.

[34] J. L. Elman, “Finding structure in time,” *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[35] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[36] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv preprint arXiv:1412.3555*, 2014.

[37] W. Yin, K. Kann, M. Yu, and H. Schütze, “Comparative study of cnn and rnn for natural language processing,” *arXiv preprint arXiv:1702.01923*, 2017.

[38] Z. Wang, Y. Ma, Z. Liu, and J. Tang, “R-transformer: Recurrent neural network enhanced transformer,” *arXiv preprint arXiv:1907.05572*, 2019.

[39] T. Ranasinghe, C. Orăsan, and R. Mitkov, “Semantic textual similarity with siamese neural networks,” in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pp. 1004–1011, 2019.

[40] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, “Signature verification using a “siamese” time delay neural network,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 04, pp. 669–688, 1993.

[41] P. Neculoiu, M. Versteegh, and M. Rotaru, “Learning text similarity with siamese recurrent networks,” in *Proceedings of the 1st Workshop on Representation Learning for NLP*, pp. 148–157, 2016.

[42] G. Koch, R. Zemel, R. Salakhutdinov, *et al.*, “Siamese neural networks for one-shot image recognition,” in *ICML deep learning workshop*, vol. 2, Lille, 2015.

[43] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *Procedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition)*, 2005.

[44] K. B. Cohen and D. Demner-Fushman, *Biomedical natural language processing*, vol. 11. John Benjamins Publishing Company, 2014.

[45] T. Jiang, J. L. Gradus, and A. J. Rosellini, “Supervised machine learning: a brief primer,” *Behavior Therapy*, vol. 51, no. 5, pp. 675–687, 2020.

[46] S. M. Weiss, N. Indurkhya, T. Zhang, and F. J. Damerau, “From textual information to numerical vectors,” in *Text Mining*, pp. 15–46, Springer, 2005.

[47] D.-H. Pham and A.-C. Le, “Exploiting multiple word embeddings and one-hot character vectors for aspect-based sentiment analysis,” *International Journal of Approximate Reasoning*, vol. 103, pp. 1–10, 2018.

[48] G. Salton, A. Wong, and C.-S. Yang, “A vector space model for automatic indexing,” *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[49] A. Mnih and G. E. Hinton, “A scalable hierarchical distributed language model,” *Advances in neural information processing systems*, vol. 21, pp. 1081–1088, 2008.

[50] M. Sahlgren, A. Holst, and P. Kanerva, “Permutations as a means to encode order in word space,” in *The 30th Annual Meeting of the Cognitive Science Society (CogSci’08), 23-26 July 2008, Washington DC, USA*, 2008.

[51] J. Daniel and H. M. James, “Speech and language processing,” 2000.

[52] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Advances in neural information processing systems*, pp. 3111–3119, 2013.

[53] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[54] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng, “Parsing with compositional vector grammars,” in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 455–465, 2013.

[55] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, “Recursive deep models for semantic compositionality over a sentiment treebank,” in *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

[56] C. Dos Santos and M. Gatti, “Deep convolutional neural networks for sentiment analysis of short texts,” in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 69–78, 2014.

[57] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” *Advances in neural information processing systems*, vol. 28, pp. 649–657, 2015.

[58] R. Socher, C. C.-Y. Lin, A. Y. Ng, and C. D. Manning, “Parsing natural scenes and natural language with recursive neural networks,” in *ICML*, 2011.

[59] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, “Convolutional sequence to sequence learning,” in *International Conference on Machine Learning*, pp. 1243–1252, PMLR, 2017.

[60] D. D. Prasetya, A. P. Wibawa, and T. Hirashima, "The performance of text similarity algorithms," *International Journal of Advances in Intelligent Informatics*, vol. 4, no. 1, pp. 63–69, 2018.

[61] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 414–420, 1989.

[62] W. E. Winkler, "String comparator metrics and enhanced decision rules in the fellegi-sunter model of record linkage.," 1990.

[63] P. Christen, "A comparison of personal name matching: Techniques and practical issues," in *Procedings of workshop at IEEE International Conference on Data*, 2006.

[64] G. Stoilos, G. Stamou, and S. Kollias, "A string metric for ontology alignment," in *Procedings of the International Semantic Web Conference*, 2005.

[65] W. W. Cohen, P. Ravikumar, S. E. Fienberg, *et al.*, "A comparison of string distance metrics for name-matching tasks.," in *Proceedings of the 2003 International Conference on Information Integration on the Web*, 2003.

[66] P. Jaccard, "The distribution of the flora in the alpine zone. 1," *New phytologist*, vol. 11, no. 2, pp. 37–50, 1912.

[67] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.

[68] H. Keskustalo, A. Pirkola, K. Visala, E. Leppänen, and K. Järvelin, "Non-adjacent digrams improve matching of cross-lingual spelling variants," in *International symposium on string processing and information retrieval*, pp. 252–265, Springer, 2003.

[69] E. Moreau, F. Yvon, and O. Cappé, "Robust similarity measures for named entities matching," in *COLING 2008*, pp. 593–600, ACL, 2008.

[70] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," *arXiv preprint arXiv:1705.02364*, 2017.

[71] Z. Gan, P. Singh, A. Joshi, X. He, J. Chen, J. Gao, and L. Deng, "Character-level deep conflation for business data analytics," in *Proceding of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2017.

[72] A. Traylor, N. Monath, R. Das, and A. McCallum, "Learning string alignments for entity aliases.," in *Proceding of the Workshop on Automated Knowledge Base Construction*, 2017.

[73] D. Tam, N. Monath, A. Kobren, A. Traylor, R. Das, and A. McCallum, "Optimal transport-based alignment of learned character representations for string similarity," *arXiv preprint arXiv:1907.10165*, 2019.

[74] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, pp. 2292–2300, 2013.

[75] C. Zhao and Y. He, "Auto-em: End-to-end fuzzy entity-matching using pre-trained deep models and transfer learning," in *The World Wide Web Conference*, pp. 2413–2424, 2019.

[76] N. Guarino, D. Oberle, and S. Staab, "What is an ontology?," in *Handbook on ontologies*, pp. 1–17, Springer, 2009.

[77] V. Iyer, A. Agarwal, and H. Kumar, "Veealign: a supervised deep learning approach to ontology alignment.," in *OM@ ISWC*, pp. 216–224, 2020.

[78] M. Mao, Y. Peng, and M. Spring, "Ontology mapping: as a binary classification problem," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 9, pp. 1010–1025, 2011.

[79] S. Amrouch, S. Mostefai, and M. Fahad, "Decision trees in automatic ontology matching," *International Journal of Metadata, Semantics and Ontologies*, vol. 11, no. 3, pp. 180–190, 2016.

[80] N. Alboukaey and A. Joukhadar, "Ontology matching as regression problem.," *Journal of Digital Information Management*, vol. 16, no. 1, 2018.

[81] D. Faria, C. Pesquita, E. Santos, I. F. Cruz, and F. M. Couto, "Automatic background knowledge selection for matching biomedical ontologies," *PloS one*, vol. 9, no. 11, p. e111226, 2014.

[82] A. Gross, M. Hartung, T. Kirsten, and E. Rahm, "Mapping composition for matching large life science ontologies.," in *Proceding of the 2nd International Conference on Biomedical Ontology*, 2011.

[83] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. J. Goldberg, K. Eilbeck, A. Ireland, C. J. Mungall, *et al.*, "The obo foundry: coordinated evolution of ontologies to support biomedical data integration," *Nature biotechnology*, vol. 25, no. 11, pp. 1251–1255, 2007.

[84] A. Rahimi, T. Baldwin, and K. Verspoor, "Wikiumls: Aligning umls to wikipedia via cross-lingual neural ranking," *arXiv preprint arXiv:2005.01281*, 2020.

[85] B. Smith, W. Ceusters, B. Klagges, J. Köhler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, A. L. Rector, and C. Rosse, "Relations in biomedical ontologies," *Genome biology*, vol. 6, no. 5, pp. 1–15, 2005.

[86] W. Yu, C. Xu, J. Xu, L. Pang, X. Gao, X. Wang, and J.-R. Wen, "Wasserstein distance regularized sequence representation for text matching in asymmetrical domains," *arXiv preprint arXiv:2010.07717*, 2020.

[87] R. He, A. Ravula, B. Kanagal, and J. Ainslie, "Realformer: Transformer likes residual attention," *arXiv preprint arXiv:2012.11747*, 2020.

Appendix A

Traditional String Metric Measure Results

Table A.1: Results with the Cosine Similarity and Jaccard metrics

Testing Dataset	Cosine Similarity ($\alpha = 0.1$)				Jaccard ($\alpha = 0.1$)			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Wikidata	23.83	50.74	47.68	49.16	39.22	45.87	78.46	57.89
ORDO	15.41	24.04	30.82	27.01	47.46	51.05	94.92	66.39
SNOMED	43.23	46.52	86.46	60.49	6.29	14.09	12.59	13.30
HDO	29.85	39.00	59.70	47.18	43.75	51.65	87.50	64.96
NCBI	26.37	45.11	52.75	48.63	46.69	52.97	93.37	67.60
HPO	35.47	42.10	70.96	52.85	41.29	51.32	82.60	63.31
Uberon	37.05	43.53	74.10	54.84	41.04	50.26	82.09	62.35
FMA	40.49	44.76	80.98	57.66	33.33	54.50	66.65	59.97
Human Ontology	31.47	40.58	62.93	49.34	40.73	49.56	81.45	61.62
MouseOntology	29.43	37.79	58.85	46.03	44.92	51.57	89.84	65.53
MPO	38.90	43.94	77.81	56.16	38.41	47.27	76.83	58.53
FMA and NCI - 1	31.15	35.03	90.32	50.49	13.27	18.78	38.46	25.23
FMA and NCI - 2	32.01	35.99	90.40	51.48	13.53	19.36	38.20	25.69
MA and NCI - 2	25.80	27.68	96.99	43.07	5.85	8.40	21.99	12.16
MA and NCI - 1	25.16	27.04	96.73	42.27	5.68	8.17	21.85	11.90

Table A.2: Results with the Monge-Elkan and I-Sub metrics

Testing Dataset	Monge-Elkan ($\alpha = 0.1$)				I-Sub Measure ($\alpha = 0.1$)			
	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
Wikidata	13.87	56.37	27.75	37.19	33.56	41.95	67.13	51.64
ORDO	9.46	88.48	18.93	31.19	20.74	29.33	41.47	34.36
SNOMED	3.13	88.03	6.27	11.70	50.00	50.05	100.00	66.71
HDO	4.67	51.97	9.34	15.84	33.11	39.87	66.22	49.77
NCBI	3.40	22.19	6.81	10.42	31.45	39.64	62.90	48.63
HPO	2.09	70.87	4.17	7.88	36.19	42.01	72.39	53.16
Uberon	1.24	41.14	2.47	4.67	35.36	41.62	70.73	52.40
FMA	0.54	95.62	1.08	2.13	42.73	46.10	85.47	59.89
Human Ontology	3.99	48.02	7.98	13.69	37.95	43.21	75.90	55.06
MouseOntology	3.78	44.62	7.55	12.92	34.64	40.92	69.27	51.45
MPO	1.84	86.24	3.67	7.04	40.24	44.60	80.48	57.39
FMA and NCI - 1	11.73	64.90	34.01	44.63	34.89	35.21	98.53	51.88
FMA and NCI - 2	12.06	65.73	34.04	44.86	33.96	34.26	98.45	50.83
MA and NCI - 2	17.23	68.30	64.79	66.50	26.40	26.53	99.25	41.87
MA and NCI - 1	16.97	65.89	65.25	65.57	25.83	25.95	99.31	41.15

Appendix B

Example of Classified Strings

In this appendix example of classified strings for each dataset using the transformer model are presented.

Table B.1: Examples of classified strings for the Transformer model

Dataset	True Positive	True Negative	False Positive	False Negative
Wikidata	amyotrophic lateral sclerosis-parkinsonism/dementia complex type 1	thiamine	levoglutamide	kirz
	guam disease	tiamina	n-acetamide	kir2dl6
ORDO	net of the small intestine	rare genetic movement disorder	alternating hemiplegia of childhood	classic glycine encephalopathy
	net of the small intestine	rare genetic coagulation disorder	benign nocturnal alternating hemiplegia of childhood	neonatal glycine encephalopathy
SNOMED CT	inspectionaction	dihydroergocristine	malignant melanoma of skin of external auditory canal	family lutjanidae snapper
	inspection action	dihydroergotamine	malignant neoplasm of skin of external auditory meatus	family lutjanidaesnapper
HDO	schimmelpenning syndrome	malignant neoplasm of gallbladder	periumbilic abdominal lump	antenatal deep vein thrombosis unspecified (disorder)
	nevus sebaceus of jadassohn	malignant neoplasm of heart	periumbilic abdominal mass	deep phlebothrombosis postpartum with delivery
NCBI	genetic disease	severe neonatal jaundice	mesothelioma	breast or ovarian cancer
	inherited human disorder	severe acroparesthesia	pseudoglioma	sporadic breast cancers
HPO	recurrent venous thrombosis	decreased anteroposterior diameter of lumbar vertebral bodies	regulation of consumption behavior	longitudinal fissure
	recurrent deep vein thrombosis	cervical vertebral bodies with decreased anteroposterior diameter	regulation of eating behavior	longitudinal fissure of the cerebrum
Uberon	space of thoracic compartment	ethmoidal bone sinus	mesenchyme of palatoquadrate arch	textus muscularis of cardiac muscle
	thoracic cavity	ethmoid bone	mesenchyme of upper jaw	textus muscularis

Table B.1: Examples of classified strings for the Transformer model

Dataset	True Positive	True Negative	False Positive	False Negative
FMA	hepatovenous segment ix	compact bone of diaphysis of distal phalanx of thumb	intervertebral foramen of sixth thoracic vertebra	trunk of mid segment of anterior interventricular branch of left coronary artery
	couinaud hepatic segment ix	compact bone of diaphysis of middle phalanx of left middle finger	intervertebral foramen of fifth thoracic vertebra	trunk of mid zone of anterior interventricular branch of left coronary artery
NCIT subset	cerebral arteries	collagen fibril	external plantar artery	cd4 plus t lymphocyte
	cerebral artery	collagen fiber	external mammary artery	t4 cells
MA	liver parenchyma	stomach secretion	left vagus x nerve trunk	vestibular membrane
	hepatic parenchyma	stomach region	vagus x nerve	reissner's membrane
MPO	mucous membrane of principal bronchus	regulation of interleukin 5 secretion	abnormal clear layer morphology	negative regulation of smoothened signalling pathway in ventral spinal cord patterning
	main bronchus organ mucosa	regulation of cellular secretion	abnormal dej morphology	negative regulation of hedgehog signaling pathway involved in ventral spinal cord patterning
FMA and NCIT subset	interstitial cell leydig	epithelium gingiva	cauda spermatozoon	somatotropin
	interstitial cell of testis	epithelium	spermatozoon	somatotropic hormone
FMA and NCIT subset - 2	musculus rectus abdominis	articulatio radiocarpalis	lobe of lung	uroepithelium
	rectus abdominis	articulation	middle lobe of lung	urothelium
MA and NCIT subset - 2	thymus trabecula	foot	hand connective tissue	subcutaneous adipose tissue
	thymic trabecula	foot bone	connective tissue	subcutaneous tissue
MA and NCIT subset - 1	tonsil capsule	respiratory system mucosa	smooth muscle tissue	zygomatic bone
	tonsillar capsule	respiratory system lung	arterial system smooth muscle tissue	zygoma