Proceedings of the of the 17th Conference on Computer Science and Intelligence Systems pp. 733–739
DOI: 10.15439/2022F209
ISSN 2300-5963 ACSIS, Vol. 30

Encoder-Decoder Neural Network with Attention Mechanism for Types Detection in Linked Data

Oussama Hamel, Messaouda Fareh
LRDSI Laboratory, Faculty of Sciences,
University Blida 1,
B.P 270, Route de Soumaa, BLIDA, ALGERIA
Email: {oussamahamel09, farehm}@gmail.com

***Abstract*—With the emergence of use of Linked Data in different application domains, several problems have arisen, such as data incompleteness. Type detection for entities in RDF data is one of the most important tasks in dealing with the incompleteness of Linked Data. In this paper, we propose an approach based on Deep Learning techniques, using an encoder-decoder model with attention mechanism, embedding layer to extract the features of each subject from the RDF triples and the GRU cells to address the problem of vanishing. We use the DBpedia dataset for the training and test phases. Initial test results showed the effectiveness of our model.**

I. Introduction

IN RECENT years, the Linked Open Data (LOD) cloud has been increasing in popularity. As a result of the success of the LOD, many semantic datasets are freely available on the Web in machine-understandable format (primarily RDF (Resource Description Framework)) related to different domains.

With the emergence of the semantic web and Linked Data, several problems related to data uncertainty have emerged, such as imprecision, incompleteness, etc. The main reason for the appearance of these problems lies in the way of building datasets. These were built from incomplete data, heterogeneous formats, semi-structured data, etc. The anomalies cited above pose problems when using so-called uncertain data in reasoning, decision-making, the generation of new knowledge, etc.

According to [1], few approaches use links among datasets, so they can't able to exploit the endless possibilities with the full knowledge of the Semantic Web.

The approaches that exploit data from only one dataset, they stay below what is possible with Linked Data. The reason of this limitations and difficulties of links discovery in Linked Data applications are:

- The datasets are produced, kept or managed by different organizations in different schemas, models, locations, systems and licenses [2]. There is not any "centralized control system," therefore, each publisher decides how to produce, manage and publish a dataset based on its needs and choices;
- The development of several applications which are independent of schema.
- The same real-world entities or relationships are referred with different URIs and names and in different languages, while languages have synonyms and homonyms that make harder that automatic links detection.
- The datasets usually contain complementary information, e.g., consider two datasets about the same domain each modeling a different aspect of the domain. The commonalities between these datasets can be very few and this does not aid automated linking and integration.
- The datasets can contain data that are erroneous, incomplete, out-of-date or conflicting.
- In addition, scalability challenges lie in developing solutions that could exploit the whole LOD as background knowledge by following links autonomously.

To improve the quality of RDF data, we choose to treat incompleteness, more specifically type incompleteness. Indeed, predicting missing types for dataset subjects will provide us with a more complete dataset.

Therefore, the results provided by applications using these datasets will become better. Our solution uses the predicates and objects belonging to the subject to predict its type. With the use of the encoder-decoder model, we will guarantee to extract the semantic relations between predicates and objects. This will improve the accuracy of subject type prediction. The attention mechanism was used to assign high weights to inputs with high importance. In this study, we will work on the DBpedia dataset, applying an approach based on deep learning.

Deep learning techniques have been recently used in many research axes to resolve different types of problems, Artificial intelligence systems use deep learning to solve computational tasks and complex problems quickly [3]. These techniques are very appropriate for dealing with large datasets. They have the ability to analyse and interpret Linked Data, that require efficient and effective tools. So, deep learning techniques are considered to be the most reliable solution that deal with the context of Linked Data, presented by RDF model.

In this paper, we have proposed an encoder-decoder network for multi-labeling. This network incorporates a attention mechanism to model the links between data. Our approach aims to predict missing types for RDF entities using data from their triples.

The remainder of this paper is organized as follows: in Section II, we define some Linked Data concepts. In section III of our paper, we explore the various related works that deal

with the type detection problem. Section IV shows in detail our proposed approach. Section V describes the experimental setup, and Section VI reports the results, followed by a discussion of the results in Section VII. Finally, Section VIII shows the set of perspectives as well as the conclusion of our work.

II. Background and context

In this section, we introduce the main principles of Linked Data, we briefly recall some necessary background knowledge including principles of Linked Data, uncertainty, incompleteness, and links detection. We will also look at some areas where linked data can be used to demonstrate its utility.

A. Linked Data

"Linked Data refers to a method of publishing structured data, so that it can be interlinked and become more useful through semantic queries, founded on HTTP, RDF and URIs" [4].

Linked Data is a design principle that presents links between RDF-formatted data published on the web rather than links between documents. This enables machines to explore the web and find other data using the links concept [5].

The various objects in this version of the web are identified by URIs (Uniform Resource Identifier).

Tim Berners-Lee proposed four rules for designing Linked Data, which are explained below:

- Use of URIs to identify objects and concepts.
- Use of the http protocol to allow humans to access sites/data.
- Use of semantic web standards to provide information relevant to URIs.
- Introducing links to other data to give more options when exploring.

B. Uncertainty in Linked Data

Data uncertainty represents the degree of reliability, inaccuracy and imprecision of the data. According to the W3C, uncertainty is either aleatory or epistemic [6].

- Aleatory: characterized by lack of information, incompleteness information, etc. from the world.
- Epistemic: it describes the non-systematic nature of the data (variability, irreducibility) and the natural variability of a system.

C. Incompleteness in Linked Data

Due that there is an overwhelming quantity of heterogeneous data on the web. Integration of data silos provided by the Linked Open Data community can provide information to curate this data and boost the Semantic Web field to its true potential. Nevertheless, even the largest graphs, for example DBpedia, suffer from incompleteness [7]. Linked Data within the enterprise can be plagued with issues of data incompleteness, inconsistency and noise.

Incompleteness in the context of Linked Data can take the form of missing information, incomplete triples, missing links between different resources, etc.

D. Links detection

Published data must be linked to other existing datasets. However, creating links between datasets requires careful analysis, given that the amount of data published is constantly increasing, the links discovery process must be automatic. Consequently, to efficiently build the Web of data, there must be solutions capable of linking data between different datasets of web of data, to detect missing links between data.

The link detection task for Linked Data is seen as a solution for the datasets incompleteness. This task enables us to discover new triples and improves the quality of data delivered to applications and systems using datasets built on Linked Data concepts.

E. Applications of linked data

The Semantic Web and linked data can have a significant impact on a wide range of applications. Here are a few examples:

- Medical field: medical and personal patient data is commonly stored in multiple incompatible systems [8]. This representation may cause issues when integrating the data (incompleteness, errors, etc.), resulting in inaccurate or completely false diagnoses. The use of linked data to represent medical data is viewed as a solution for easing data integration. The incompleteness problem can be solved by using type detection in linked data.
- E-commerce: based on the semantic web, agents collect product data from multiple stores in order to provide the best deals to customers. They can also perform auctions, negotiations, and contract drafting automatically (or semi-automatically).
- Knowledge management in an organization: semantic web techniques are used to provide the possibility of representing knowledge in the form of concepts, allowing leaders to have answers to semantic queries.

III. Related Works

In this part, we will explore a number of related works that deal with type prediction in RDF datasets.

A statistical heuristic link based type prediction mechanism, has been proposed in [9], this work was evaluated on DBpedia.

In [10], the authors propose a supervised hierarchical SVM classification approach for DBpedia by exploiting the contents of Wikipedia articles.

In [11], the authors propose a multi-label classification algorithm based on word embedding such as Word2Vec, FastText and GloVe in order to capture the semantic aspect between entities and relations.

Another approach named Class Assignment Detector proposed by [12] to detect correct and incorrect classes assignments for entities in RDF data by analyzing class characteristics.

The authors in [13] solved the type prediction problem by using the Twitter profiles of RDF entities. The data extracted from these profiles were used as features in training data in order to facilitate the prediction task.

Another approach proposed in [14] uses word embedding and network embedding to predict the infobox types for Wikipedia articles. This information is useful for the type generation procedure for RDF entities.

In the work done in [15], the authors propose a binary classifier using structural data and based on machine learning techniques to predict the types of RDF entities.

The work carried out in [16] consists in proposing an approach which deals with types prediction by text classification. Two classifiers have been proposed to achieve this task.

Analysis: After studying the different approaches proposed in the literature for type detection in Linked Data, we can deduce that:

- The majority of works do not take into account the semantic relations between the different components of the triples.
- Use of different techniques for links detection in related works, we cite: supervised hierarchical SVM classification, statistical heuristic, machine learning techniques, text classification and word embedding.
  - The SVM algorithm is not appropriate for large datasets and for high number of features.
  - The extraction of features by the programmer is the major disadvantage of machine learning algorithms. As a result, the data quality suffers. On the other hand, this task is performed automatically by neural networks. Deep learning allows for the extraction of more and significant features and thus produces better results.
  - Using word embedding and statistical heuristics to predict types does not allow for the extraction of more significant features from the inputs and the possible semantic relations between triples. This has a negative impact on the results quality.
- The results obtained can be improved by proposing other solutions.
- Several works test their proposed methods on a subset of DBpedia data, but the tested part is not specified in the research works.
- The extraction of semantic relationships between different types (classes) and resources is not addressed in related works.
- The proposed methods do not assign weights to triples based on their importance during the type detection task.

In order to achieve this goal, we propose our approach based on deep learning in order to explore semantic relationships between the different components of RDF triples. We base our choice on deep learning models' ability to learn from large amounts of data. We added the attention mechanism to give a weighting to inputs according to their importance.

Therefore, the main contributions of this paper are:

- A proposition of embedding model, which exploit the semantic relations between RDF triples.
- A neural network based multi-label classification model for predicting type of resource RDF,
- Using the attention mechanism to improve the quality of model. By assigning weights to triples, semantic relationships between RDF resources and types can be extracted.
- Numerical representation (RDF2Vec) of resources and predicates in the DBpedia dataset.
- The type detection task is approached as a sequence-to-sequence problem, where the inputs and outputs are long sequences.

IV. Materials and method

Before delving into our approach and giving more details, we first start by showing where the missing types problem lies in the ontology proposed by the World Wide Web Consortium (W3C)[1] organization, for modeling uncertain knowledge in the semantic web. This context is presented in Fig. 1. We use deep learning techniques for incompleteness processing.

Detecting links in Linked Data is a solution to identify classes of resources and therefore find new links between data and minimize incompleteness.

The automatic detection of missing types will improve data quality and provide reliable answers to queries launched by the various applications that use Linked Data.

Our solution allows predicting types for resources belonging to RDF datasets based on predicates and object values. It is a model built using deep learning techniques. Google Collaboratory[2] was used for the training phase using the DBpedia dataset.

A. Modeling problem

Our solution consists in treating the link detection problem as a multi-label classification problem.

Multi-label Classification is the task of assigning data points to a set of classes or categories which are not mutually exclusive, meaning that a point can belong simultaneously to different classes. In multi-label classification, the examples are associated with a set of labels Y from a set of disjoint labels L, $Y \subseteq L$.

The inputs of our model represent the predicates $(P1, P2, ..., Pn)$ and objects $(O1, O2, ..., On)$ belonging to a subject S. These inputs are used to predict the output which represents the types of the subject S.

B. Construction steps

The different steps of our model construction are mentioned in Fig. 2.

1) ***DBpedia Dataset****:* In order to train our model, we will use the DBpedia dataset[3]. The latter is built according to the Linked Data principles.

[1]https://www.w3.org/
[2]https://colab.research.google.com/
[3]http://gaia.infor.uva.es/hdt/dbpedia2016-10/dbpedia2016-10.hdt

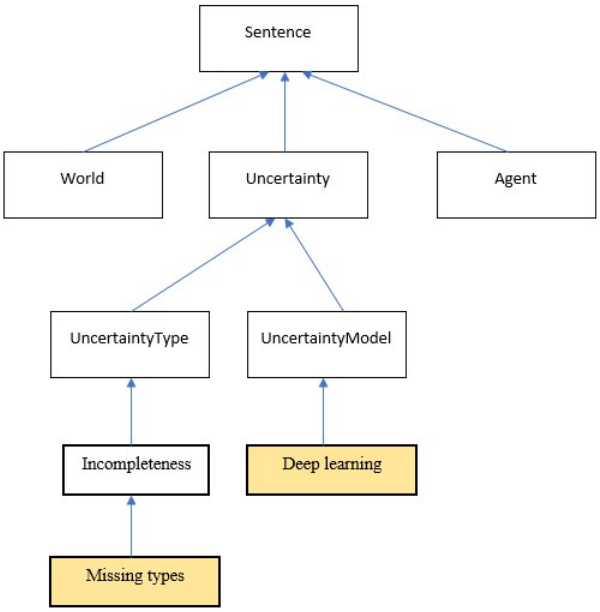

Fig. 1. Types detection in uncertainty ontology

Our dataset is composed of a set of triples (Subject, predicate, object), where the predicate represents the link between the subject and the object.

Our goal is to predict missing types $(T1, T2, \ldots, Tn)$ for subjects Si based on their predicates $(P1, P2, \ldots, Pn)$ and objects $(S1, S2, \ldots, Sn)$ belonging to our dataset. i.e., for a given subject Si, we use all its predicates and objects as inputs to predict its different classes or types. As a result, by inferring new triples, we can have a more complete dataset. (Si, Pi, Oi).

The first step of our approach is dataset reading, choosing the triples concerned by the different learning phases, as well as transforming the format of these triples into a numerical format.

Once our dataset is ready, we proceed to the pre-processing step, which consists of transforming the format of the triples into a format suitable for deep learning models (numerical representation).

Finally, we limit the number of predicates and objects belonging to each subject (inputs) to 205 objects, with the possible types as outputs.

2) ***Pre-treatments****:* Data preprocessing is an important or even crucial step for Machine Learning and Deep Learning. Data quality can directly affect the model learnability.

This step consists of transforming data into a more suitable format that can be used by the model. In this step we transformed each subject, object and predicate into numerical format. The numerical representation of inputs and outputs consists of giving numbers for each subject, predicate and object. Fig. 3 shows an example of this transformation.

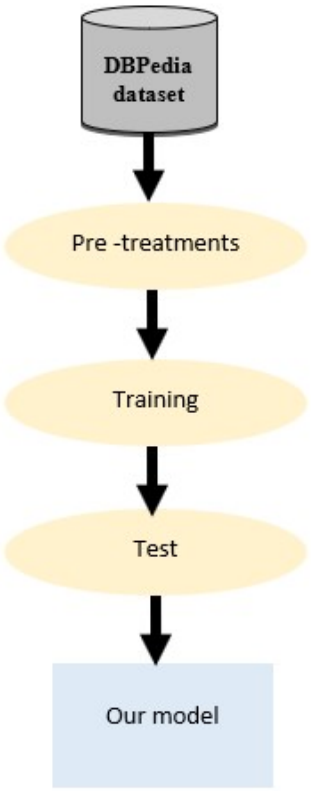

Fig. 2. Steps of our approach to type detection for Linked Data

C. Our model proposal

Our model is an encoder-decoder with attention mechanism which is a neural network design pattern that aids in the generation of an output sequence for every input sequence.

As shown in Fig. 4, the architecture is composed of two parts, encoder and decoder. Each part uses deep neural networks, more precisely gated recurrent units (GRU) in order to handle the sequence inputs of variable length.

The attention mechanism is a technique used in neural networks to focus on certain factors that can influence the

Fig. 3. An example of the preprocessing process

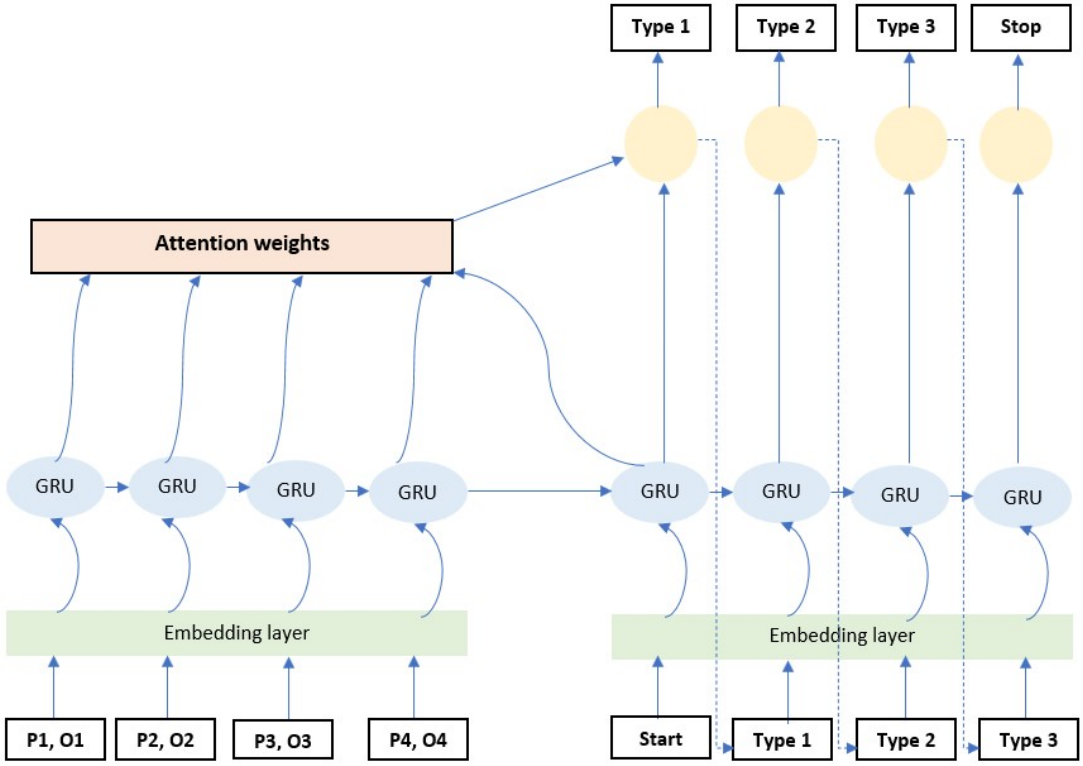

Fig. 4. Architecture of our encoder-decoder model with attention mechanism

model quality. The major contribution of this mechanism is to improve the sequence-to-sequence models performance. The details are mentioned in section IV-C4.

Our model uses as input the different predicates and objects belonging to the subject, and the types of the latter as output. We propose an embedding layer for each of the two components of our model (encoder and decoder).

*1) **Encoder-Decoder**:* The encoder-decoder architecture is used to generate output sequences from input sequences. In our model, we used GRU cells to build each component.

Our encoder is composed of two distinct modules: the Embedding layer, and the generated vector containing the information relating to inputs. This vector is used as the first hidden state of the decoder, in order to guide the decoder in its predictions. In what follows, we will detail how it works.

For our decoder, we distinguish two layers, a GRU layer and a SoftMax layer.

The GRU layer works the same way as the encoder layer with one exception based on Input/Output:

- The decoder takes as initial input the last hidden state generated by the encoder. This hidden state contains the essential information contained in each element of the input sentence.
- Just like the encoder, the decoder has an Embedding layer that generates the Embedding vectors from the numerical representation of each subject.
- To generate a type Ti at a time step TSi, the decoder takes as input: the hidden state, the output generated at the previous time step $TSi-1$, as well as the Embedding vector.

The SoftMax layer predicts a probability distribution over the possible types, and choose the type with the highest probability.

*2) **Embedding layer**:* The embedding layer allow a reduced representation of inputs while keeping the semantic links between the subject's components. This layer enables the more features extraction from data.

*3) **Gated recurrent units**:* Gated recurrent neural networks GRU present a solution to the vanishing gradient problem. They have two gates, one for reset and another for update. They also use a hidden state mechanism, unlike LSTM which use a cell state and 3 gates. We will be able to process long sequences as a result of this.

*4) **Attention mechanism**:* The attention mechanism manages and quantifies the interdependence within input elements (Self-Attention) and between inputs and outputs (General Attention). This mechanism was introduced to solve one of the problems of sequence-to-sequence models, namely their

inability to provide good results when dealing with long sequences. This problem lies at the decoder level where only the last hidden state generated by the encoder is used as a context vector. An attention weight is generated for each input, giving high weights to the most important inputs in the type prediction phase. These values will be used by the decoder for the type prediction based on the results obtained from the SoftMax layer.

V. Experiments

To evaluate the performance of our method, we use the DBpedia in our experimentation, to test the proposed method.

To this end, we use a subset of the dataset from the DBpedia.

A. Dataset

The DBpedia project is a community effort to extract structured information from Wikipedia and to make this information accessible on the Web.

DBpedia is a crowd-sourced community effort to extract structured, multilingual knowledge from the information created in various Wikimedia projects. The DBpedia knowledge bases are extracted from 125 Wikipedia editions. Altogether the DBpedia (2016-10) release consists of 13.1 billion pieces of information (RDF triples) out of which 1.7 billion were extracted from the English edition of Wikipedia, 6.6 billion were extracted from other language editions and 4.8 billion from Wikipedia Commons and Wikidata [17].

In this work, and due to technical constraints (RAM capacity), we used 15292 RDF triples for the different phases. These triples were divided as follows: 60% for the training phase, 20% for the validation phase and 20% for the test phase. the different details are mentioned in the Table I.

B. Hyper parameters

Our model uses the ADAM function as an optimizer and the 'Sparse Categorical Cross Entropy' cost function. The training phase consisted of 81 epochs. The various details are shown in Table II.

VI. Results

To evaluate our method for type prediction in Linked Data, we use the standard evaluation measures: precision, recall and F-measure. In multi label classification, these criteria are defined in the following.

The metrics evaluate the multi-label classification system's performance, on each test example separately by comparing the predicted labels with the gold standard labels for each test example. We focus on 3 major example-based metrics, as defined in [18] [19]:

$$Precision = \frac{1}{p}\sum_{i=1}^{p}\left(\frac{TP}{TP+FP}\right) \quad (1)$$

$$Recall = \frac{1}{p}\sum_{i=1}^{p}\left(\frac{TP}{TP+FN}\right) \quad (2)$$

$$F-measure = \frac{2.Precision.Recall}{Precision+Recall} \quad (3)$$

Where p is the number of instances in the test set. The true positives (TP) is defined as the labels that are identical to the gold standard labels, false positives (FP) as labels that are not true positives, and false negatives (FN) as the gold standard labels that were missed in the prediction results.

Table III illustrates the different results in two cases: Encoder-decoder model with Attention Mechanism (AM), witch represents our solution, and Encoder-decoder model without attention mechanism.

Histogram in Fig. 5 outline the evaluation results using the standard evaluation measures defined in Table III.

During the training phase, we obtained a cost function value of 0.0217 for our model with attention mechanism and 0.0937 for the model without attention mechanism.

After calculating the recall, precision, and F-measure values, we discovered that our model outperformed the encoder-decoder model without an attention mechanism.

TABLE I
The number of records for each step

Dataset	DBpedia
Training (60%)	9174
Validation (20%)	3059
Test (20%)	3059
Total (100%)	15292

TABLE II
Hyper parameters values

Hyper parameter	Definition
Optimization function	Function ADAM
Loss function	Sparse Categorical Cross Entropy
Number of GRU Nodes	1024
Batch Size	64
Embedding size	256

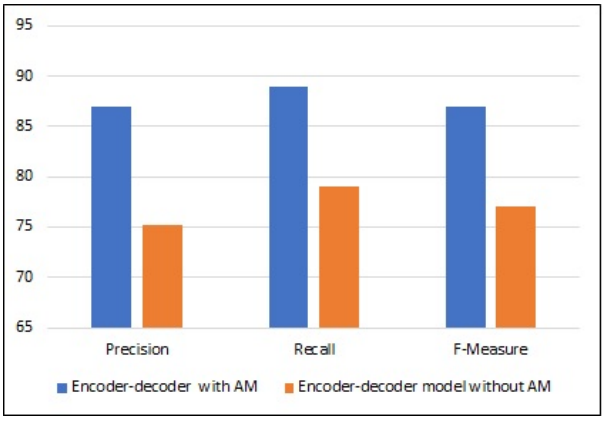

Fig. 5. Histogram of evaluation results

TABLE III
RESULTS OF TYPE PREDICTIONS WITH OUR MODEL AND A SIMPLE ENCODER-DECODER MODEL

Model	Precision	Recall	F-Measure
Encoder-decoder model with AM	86.92	89.00	87.95
Encoder-decoder model without AM	75.16	79.02	77.04

VII. DISCUSSION

According to the presented results in Table III and Figure 5, we note that our modeling presents a good values for used evaluation metrics, comparing with encoder-decoder model without attention mechanism, witch is presented by 86.92% as precision value, 89.00% as Recall value and 87.95% as F-Measure value.

The results clearly demonstrated the significance of employing the attention mechanism. It enables the assignment of weights to inputs based on their importance. This increases the accuracy of output prediction. GRU cells positively influenced the results quality by making the features extraction task from inputs more efficient.

The results for the encoder-decoder model without an attention mechanism are less impressive because no attention value is assigned to the inputs to guide the type prediction process.

These results can be improved by running the model training phase on more powerful machines and by using larger datasets.

VIII. CONCLUSION

The semantic web community has been researching links detection and enrichment of Linked Data for last years. The volume of the available Linked Data on the web has been increasing considerably, along with the existence of erroneous, incomplete RDF data, kept this area active.

Links detection is a new research area of the Semantic Web which studies the problem of finding semantically related entities lying in different knowledge bases.

One of the most important challenges in Linked Data cloud is predicting the missing links between the entities, which is necessary to facilitate the inter-connectivity of datasets in the LOD cloud, in order to enhance and enrich the information that is known about them. Moreover, in the LOD cloud, information about the same entities is available in multiple datasets in different forms.

Links detection aims to deal with the issue of missing data. The completeness of the data simplifies decision-making and task performance in any field of application, including medical diagnostics, e-commerce and ecological prediction.

In this paper, we have proposed a promising new approach dealing with the type detection problem in Linked Data. We have treated this problem as a multi-label classification task using an encoder-decoder model with attention mechanism, and we have obtained very good results. However, in the future, we want to validate our approach by testing it on different datasets and comparing it with the results of related works. We intend to propose a method for dealing with all possible semantic links. We also want to to use NLP techniques on textual objects and to train our model on large datasets.

REFERENCES

[1] P. Ristoski and H. Paulheim, "Semantic web in data mining and knowledge discovery: A comprehensive survey," *Journal of Web Semantics*, vol. 36, pp. 1–22, 2016.

[2] M. Mountantonakis and Y. Tzitzikas, "Large-scale semantic integration of linked data: A survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–40, 2019.

[3] R. A. Fiorini, "Computational intelligence from autonomous system to super-smart society and beyond," *International Journal of Software Science and Computational Intelligence (IJSSCI)*, vol. 12, no. 3, pp. 1–13, 2020.

[4] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data: The story so far," in *Semantic services, interoperability and web applications: emerging concepts*, pp. 205–227, IGI global, 2011.

[5] T. Berners-Lee, "Linked data - design issues." http://www.w3.org/DesignIssues/LinkedData.html, 2006. Accessed: 2022-05-09.

[6] K. J. Laskey and K. B. Laskey, "Uncertainty reasoning for the world wide web: Report on the urw3-xg incubator group.," *URSW*, vol. 8, pp. 108–116, 2008.

[7] X. Sumba and J. Ortiz, "Between the interaction of graph neural networks and semantic web," in *Proceedings of the 2019 NeurIPS Workshop on Graph Representation Learning*, 2019.

[8] C. Wilcox, S. Djahel, and V. Giagos, "Identifying the main causes of medical data incompleteness in the smart healthcare era," in *2021 International Symposium on Networks, Computers and Communications (ISNCC)*, pp. 1–6, IEEE, 2021.

[9] H. Paulheim and C. Bizer, "Type inference on noisy rdf data," in *International semantic web conference*, pp. 510–525, Springer, 2013.

[10] T. Kliegr and O. Zamazal, "Lhd 2.0: A text mining approach to typing entities in knowledge graphs," *Journal of Web Semantics*, vol. 39, pp. 47–61, 2016.

[11] R. Biswas, R. Sofronova, M. Alam, and H. Sack, "Entity type prediction in knowledge graphs using embeddings," *arXiv preprint arXiv:2004.13702*, 2020.

[12] M. Barati, Q. Bai, and Q. Liu, "An entropy-based class assignment detection approach for rdf data," in *Pacific rim international conference on artificial intelligence*, pp. 412–420, Springer, 2018.

[13] Y. Nechaev, F. Corcoglioniti, and C. Giuliano, "Type prediction combining linked open data and social media," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pp. 1033–1042, 2018.

[14] R. Biswas, R. Türker, F. B. Moghaddam, M. Koutraki, and H. Sack, "Wikipedia infobox type prediction using embeddings.," in *DL4KGS@ ESWC*, pp. 46–55, 2018.

[15] N. Mihindukulasooriya and M. Rico, "Type prediction of rdf knowledge graphs using binary classifiers with structural data," in *International Conference on Web Engineering*, pp. 279–287, Springer, 2018.

[16] X. Zhang, E. Lin, and S. Pi, "Predicting object types in linked data by text classification," in *2017 Fifth International Conference on Advanced Cloud and Big Data (CBD)*, pp. 391–396, IEEE, 2017.

[17] H. Jin, C. Li, J. Zhang, L. Hou, J. Li, and P. Zhang, "Xlore2: large-scale cross-lingual knowledge graph construction and application," *Data Intelligence*, vol. 1, no. 1, pp. 77–98, 2019.

[18] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE transactions on knowledge and data engineering*, vol. 26, no. 8, pp. 1819–1837, 2013.

[19] J. Du, Q. Chen, Y. Peng, Y. Xiang, C. Tao, and Z. Lu, "Ml-net: multi-label classification of biomedical texts with deep neural networks," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1279–1285, 2019.