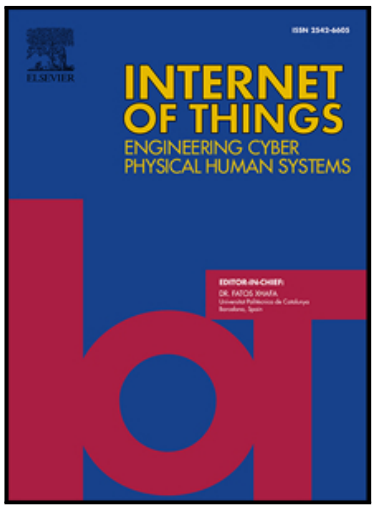

Multi-Ontology Mapping Generative Adversarial Network in Internet of Things for Ontology Alignment

Varun M Tayur , R Suchithra

PII: S2542-6605(22)00098-1
DOI: https://doi.org/10.1016/j.iot.2022.100616
Reference: IOT 100616

To appear in: *Internet of Things*

Received date: 29 July 2022
Revised date: 12 September 2022
Accepted date: 12 September 2022

Please cite this article as: Varun M Tayur , R Suchithra , Multi-Ontology Mapping Generative Adversarial Network in Internet of Things for Ontology Alignment, *Internet of Things* (2022), doi: https://doi.org/10.1016/j.iot.2022.100616

Multi-Ontology Mapping Generative Adversarial Network in Internet of Things for Ontology Alignment

*Varun M Tayur
Department of Computer Science, Jain (deemed to be university), Jayanagar
Bangalore, 560069, India

Suchithra R
Department of Computer Science, Jain (deemed to be university), Jayanagar
Bangalore, 560069, India
**Corresponding author email: varunmtayur@outlook.com*

Declarations

Funding

No funds, grants were received by any of the authors.

Conflict of interest

There is no conflict of interest among the authors.

Data Availability

All data generated or analyzed during this study are included in the manuscript.

Code Availability

Not applicable.

Author's contributions

Varun M Tayur and Suchithra R. contributed to the design and methodology of this study, the assessment of the outcomes and the writing of the manuscript.

Multi-Ontology Mapping Generative Adversarial Network in Internet of Things for Ontology Alignment

Abstract

On the Semantic web, ontologies are thought to be the remedy to data heterogeneity, and correlating ontologies is a highly effective technique. Although the use of representation learning approaches to a variety of applications has showed significant promise, they have had little effect on the issue of ontology matching and classification. In order to establish alignments between two ontologies, this research presents the Multi-Ontology Mapping Generative Adversarial Network in Internet of Things (MOMGANI). For the instance of ontology mapping, we suggest using a two-system representation learning network consisting of a Generator and Discriminator. The Generator applies a probabilistic softmax classifier to the different Name, Label, Comments, Properties, Instance descriptions, concept characteristics, and the neighbourhood concepts for each of the ontology's properties. In order to support the assertions that the Generator has generated, the Discriminator network employs a novel Bidirectional Long Short-Term Memory (Bi-LSTM network) with an Ontology Attention mechanism enhanced by the concept's descriptions. As a result, both systems are in a feedback mechanism where they can learn from one another. The system will produce a set of triples that list all the associated concepts from various ontologies as its final product. Domain experts will review these triples outside of the band to ensure that only true concepts and triples are chosen for the alignment. In comparison to using the ontologies separately, the aligned ontology enables extended querying and inference across related ontologies and domains. Considering metrics like recall, precision, and F-measure, the experimental evaluation was performed utilizing the datasets for classes alignment, property alignment, and instances alignment. The proposed architecture provides a recall, precision, and F-measure of 0.92, 0.99, and 0.83 respectively which reveals that this model outperforms the traditional methods.

Keywords: Generative adversarial network; Ontology alignment; IoT and OntoGenerator and OntoLSTM

1. Introduction

Internet of Things (IoT) is an umbrella term that is composed of diverse set of technologies. This diversity is not just limited to devices, but the underlying semantics and the information produced by them. These devices operate using various protocols which just handle the

semantics of communication. Beyond communication, semantics lies embedded in the knowledge of the operational domain. The device data can be annotated to enable automatic extraction of intelligence; the intelligence of the operational domain is solidified in the Ontologies. An ontology is a set of structural rules that represent the concepts of a domain that can be used to perform logic-based operations for retrieving or inferring new information. Ontologies have with them the knowledge of a particular domain expressed in the form of relational triples (subject, object, predicate). Several rules are expressed within the Ontologies that define how the concepts in the domain are related. Rules are often expressed as a triple in the following format (concept, relation, concept). Often, in complex domains such as Smart City or Smart Home. There are several domains (and hence several ontologies) at play in parallel, hence requiring reasoning/intelligence across several domains and ontologies. As new concepts are formalized for a given domain it leads to significant similarities between ontologies. There is a need to develop accurate and reliable techniques to perform the task of matching and finding similarities automatically. Matching or Mapping concepts across ontologies can be challenging considering that there are fundamental differences in syntax, structure, and properties. Extraction of semantic interconnection between the ontologies is the key challenge for extending the knowledge base across ontologies and potentially related domains. An ontology models the knowledge of the domain by defining the concepts, the properties describing a concept, and the relationships between two concepts. The formalization itself has been fragmented over time due to varying reasons and as evident by there are 797+ ontologies directly or indirectly related to IoT. Often, the same concepts in one Ontology are redefined or defined with different terminology and context. The promise of Linked Data and Open Data is falling apart, as observed in only a mere 4% is machine-interpretable and ready for use to build intelligent systems. Ontologies have been developing decentralized in latest days, creating a number of conflicting ontologies. In order to create a single coherent ontology, it is possible to solve the heterogeneity problem. By embracing the division of the ontology effectiveness into screening, integrating, and repairing sub-tasks where matching is a prerequisite for merging and can be done individually or as part of the matching process tough concerns of the cross-ontology community have been handled.

An adversarial learning framework is used to produce alignments between two ontologies. To produce triples that closely resemble the mapped ontologies is a critical condition for employing an adversarial learning technique. The secondary objective of such a system would be to aid in the development of a knowledge base that could be "full" rather than partial aware of its working environment. The feedback loop in which the Generator-Discriminator network functions enables it to cooperate to generate the ontology alignment. For this purpose, the

MOMGANI algorithm has been proposed where the generator part is called as "OntoGenerator" while the Discriminator is called "OntoLSTM+". The OntoGenerator of MOMGANI uses the embedded concepts of the IoT ontologies to produce the mose probable triples which are asserted for validity within the Discriminator. Further, the generated triples will be categorised using the new OntoLSTM+ discriminator network. To increase the classification score, the discriminator layer employs a relation-attention technique with entity relation descriptions. Based on the discriminator's confidence score and the classified relations, new alignments are repeatedly predicted.

The remainder of the paper is organized as follows. Section 2 discusses the related works in Ontology alignment. Section 3 shows the proposed MOMGANI algorithm. Section 4 depicts the experimental results and analysis of the proposed MOMGANI algorithm and finally, section 5 shows the conclusion and future work of the paper.

2. Related works

Over the past few years, there has been a steady increase of research in IoT and building intelligence in the eco-system with up to 200+ ontologies encoding domain knowledge about various allied areas of IoT like Food, Agriculture, Logistics, Health, etc. The rapid growth of several Ontologies being produced is a big concern since there is a lack of standards or convention, resulting in huge complexity for the extraction of intelligence. As noted by [1] the relevance of Ontology in a field like IoT intelligence from the knowledge associated with it. A quick study of the ontologies reveals similar information concepts being defined/redefined in the Ontologies (ex. oneM2M ontology and SSN both define Sensing concepts). So, this means the task of aligning concepts, relations across ontologies are an important problem to solve if we were to build an intelligent knowledge base. By generating new relations and mapping similar concepts, the real benefit of using a "linked data" objective is realized.

Ontology alignment[2,3,4], in general, has been performed using various methods, including rule-based and statistical methods. More recently, neural methods[5,6,7] for incorporating contextual and background information have been applied to ontology alignment as well. In [8] a two-way Gated Recurrent Unit (GRU) neural model is used to align concepts from different ontologies and generate inference rules from external definitions and context information. In [9] the OntoPhil approach uses a lexical and structural matcher to find suitable alignments between the Ontologies. However, all the above approaches determine probable concept matches using word embeddings of the concepts in each ontology, neither of them forms the embeddings for entities and relationships contained in the ontologies. IoT ontologies encode knowledge by capturing domain-level expertise in the form of the knowledge graph

structured relations. Knowledge graph embeddings [10] allow for representation of the structured knowledge from a knowledge graph into concept and relation embeddings. As in [11], producing knowledge graph embeddings for the biomedical ontologies has shown that ontology can be successfully expressed as real-valued vectors associated with concepts and relations. Further, these vectors are used to produce alignments between two ontologies. The authors have projected a IoT-O expressive ontology [12] for the Internet of Things, makes the most efficient use of ontologies that have already been established in particular domains such as sensors, observer, services, quantity sort, units or time. Additionally, theories [13,14] based on attention mechanisms offer dense representations that, while not immediately comprehensible, are nevertheless related to a knowledge graph and hence have connections to other parts. Similarly, the framework for personalized discovery combining fuzzy-based CBR and context ontology [15] is employed to deliver context-based personalized experiences. In addition, the theories use fuzzy set theory [16,17,18] to transposition numerical-type context data received from the external environment in order to create and maintain the case instances on the case-based context ontology. Also, some machine learning models used to select ontologies and the knowledge preference using multimedia communication and their attributes. The literature has proposed a framework called COME-UP Computation Offloading in mobile edge computing with Long-short term memory (LSTM) based user direction prediction. Utilizing the weighted sum method, it selects the top server from the list of candidates and places the incoming work accordingly [27,28]. On the Semantics, ontology is considered to be the remedy to data heterogeneity, and matching ontologies is a highly effective way to deal with the issue. In order to obtain high-quality ontology alignment, ontology meta-matching examines the best weighting to combine different similarity measurements. This is a challenging nonlinear mathematical issue in the ontology matching area. In order to address these problems, a number of findings were analyzed, and it was revealed that generative adversarial networks are a suitable tactic that perform well in terms of maintaining training efficiency and classification accuracy. The SA-GAN algorithm is used to accelerate the speed and achieve good benchmarks in the ontology domain in order to increase the GAN's efficiency. On comparison with the literature[29], the results show that MOMGANI can align various knowledge graphs and can be utilized for both representations training and ontological alignments. The next section depicts the proposed MOMGANI algorithm.

3. Proposed MOMGANI algorithm

In MOMGANI, the structure and existing encoding of the knowledge are used to learn more alignments i.e. the algorithm relies on (a) the concept names, comments, descriptions (b)

relations, and object properties spanning concepts (c) the attributes of concepts. The features of importance in an Ontology are hence fixed to concept names, description, comments, object-properties, immediate super and child classes, and nearby concepts that are being aligned.

In order to calculate similarity scores among concepts from various ontologies, MOMGANI differentiates from [6] in several ways: (a) the generator uses the concepts' names, labels, comments, properties (object), instances, attributes, and the neighbours of nearby concepts in the knowledge graph; (b) the generator also utilises an augmented connection choice algorithm based on entity relations instead of just equivalence relations; and (c) the discriminator employs a novel LSTM+ algorithm.

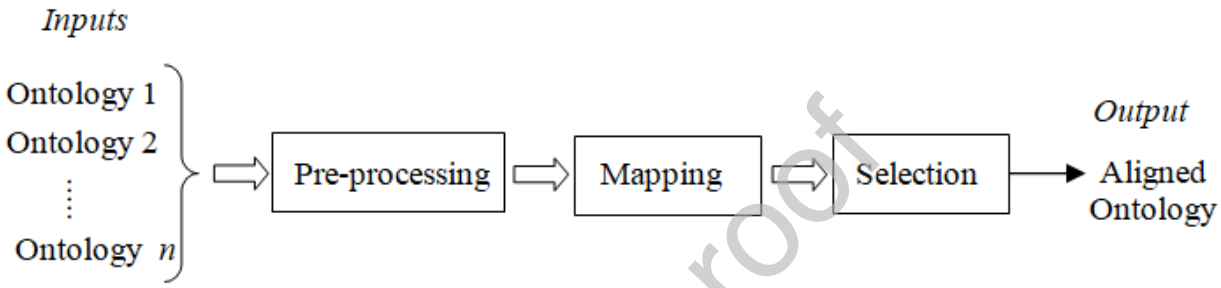

Fig. 1. Architecture of the MOMGANI

Fig. 1 illustrates the steps followed in the execution of this approach. Several Ontologies are input to the algorithm, where the ontologies are pre-processed (normalized) first, later the matching and aligning of the concepts and relations are performed followed by the manual selection / approval by humans to ensure there is no invalid relation being accepted into the aligned ontology. The output Aligned Ontology with the new relations can be utilized dynamically when queries are to be issued across related operating domains. The MOMGANI algorithm is summarized in algorithm 1.

Algorithm 3.1 MOMGANI algorithm

Input: Ontology 1, Ontology 2,…, Ontology n

Output: Alignments of Ontology 1, Ontology 2, …, Ontology n

Steps:

1. Preprocess (Ontology 1, Ontology 2 Ontology n)
2. **do** (TrainMOMGANI (generator, discriminator))
   Generator := OntoGenerator(Ontology <1,2,…,n >)
   Discriminator := OntoLSTM+(Ontology <1,2,…n >)
   until (no new alignments found)
3. Extract top-*N* triples from the discriminator model to produce alignments

4. Selected top-*N* triples are validated (manually)

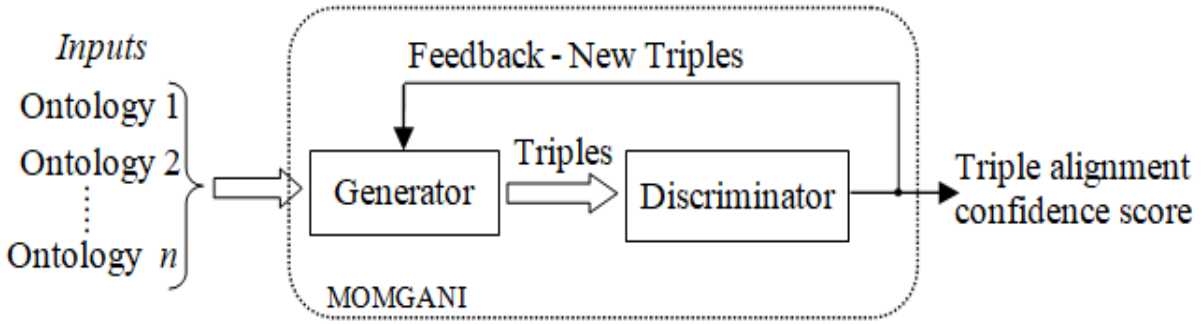

Fig. 2. MOMGANI algorithm high level architecture

3.1. *Multi Ontology mapping Overview*

The Discriminator system feeds in the learned triples and their confidence score, the feedback loop into the Generator allows for continuous learning within the system. The Generator uses a combination of Similarity / Probability and Entity Descriptions (a trivial CNN) to produce triples from the given input Ontologies. The goal of the Generator is to fool the discriminator into believing the given triple is valid. The discriminator is trained independently on the input Ontologies using a Bi-LSTM that utilizes not just the ontology triples for learning but also utilizes the entity descriptions and related data between the concepts. The algorithm completes when no new triples can be aligned. As per the GAN concepts, the loss of the system is computed using the Wasserstein Loss where the loss function is $D(x) - D\big(G(z)\big)$. Here $D(x)$ is the real triple Discriminator's output, $G(z)$ is the output of generator for given noise z, and $D\big(G(z)\big)$ is the output of discriminator for a fake triple. Fig. 2 illustrates the high-level steps followed in the algorithm. Notice that the algorithm is capable of accepting n ontologies. The algorithm terminates when no new alignments can be found.

3.2. *Pre-Processing*

The pre-processing step is very essential for preparing the dataset required for training and validation. The pre-processing begins with the normalization of the input ontologies. All the Ontologies are converted into OWL/RDF format. Using the Apache Jena library, the pre-processing step will extract the required features from the input ontology is in OWL format because it makes the extraction of features from the ontologies easily manageable by this algorithm. The pre-processing stage performs the following actions on the inputs which are as follows:

Stage 1: Standardise the occurrence/usage of punctuation sign, such as the special characters, numbers and spaces of ontology.

Stage 2: Normalize the text by executing the following normalization methods namely lower case conversion, Lemmatisation, *Delete_Links* and *Stop_Words*.

Stage 3: Transforming ontological data into numerical vectors to allow for submission into a neural network, such that

(i) The input Ontology must be transformed into a set of numerical vectors that a model can use as input. Using Apache Jena following features are extracted for each concept in the Ontology label, and labels of its immediate super and child classes, comments, Object-Properties, Instance descriptions, and concept attributes.

(ii) A neural network model cannot take strings of characters as direct input. Hence characters are converted to numbers using word2Vec [19] approach. These embeddings provide a representation vector for each possible concept. The embedding dimension is set to 300 characters. Each provided value is normalized between 0 and 1. De-noising of the data is achieved by, using lowercase on every class label, and by replacing underscores with spaces.

Stage 4: Data Pre-selection to reduce the scope for data explosion. To prevent combinatorial explosion when comparing classes from both ontologies and extracting relations from them, obvious examples need to be removed before passing input data to the neural network model. The two cases are listed below.

Case (i): *Positive examples*: across two different ontologies, some classes have identical names and properties. When this happens, these classes are automatically considered equivalent.

Case (ii): *Trivial negative examples:* To avoid many completely different classes from being selected for comparison distance-based pre-selection is used. If the Levenshtein distance between the labels of two classes is large, then these classes are considered different and this can be overcome by choosing weight to the Levenshtein distance with a factor that is inversely proportional to the length of the labels

: the shorter the labels, the greater the final distance. Let a and b be the labels of two classes that need to be compared. Let L_a and L_b be their respective length. The index i can be computed using the relation.

$$i = \frac{Levenshtien(a,b)}{L_a * L_b} \quad (1)$$

A threshold t_h is defined and if $i > t_h$ the two classes as classified as different.

3.3. *Mapping*

3.3.1. *OntoGenerator*

The set of concepts that are encoded in ontologies X and Y be represented as C_x and C_y respectively. From X and Y the alignment A can be estimated using the relation

$$A = \{(C_x, C_y) \in C_X \times C_Y | C_x \equiv C_y\} \quad (2)$$

For example, the concept Device from reference ontology IOT-O and SAREF resembles the same concept as System in SSN. Hence the system and device can form a pair in A resembled as (system, device). If c_1 and c_2 are from the same ontology, then the condition $c_1 \equiv c_2$ is not satisfied. Therefore the concept c_x from the ontology X should be aligned with almost one concept c_y from the ontology Y and vice-versa. After the estimation of alignment A, the subset of A and A' is considered for training the model. The alignment of ontologies can be considered as the classification model where $q(c_y \mid c_x)$ represents the probability of a concept c_y, such that the concept c_y is to be aligned with the concept c_x. Therefore the probability $q(c_y \mid c_x)$ can be represented as the function of concept embedding similarity.

$$q(c_y \mid c_x) = softmax(sim(c_y, c_x)) = \frac{e^{sim(c_y, c_x)}}{\sum_{j \in Y} e^{sim(c_y, c_x)}} \quad (3)$$

Here *sim* is estimated by the cosine similarity expressed as

$$sim(c_y, c_x) = \frac{\vec{v}(c_x) \cdot \vec{v}(c_x)}{\|\vec{v}(c_x)\|_2 \|\vec{v}(c_y)\|_2} \quad (4)$$

Where the concept embedding for c_x is $\vec{v}(c_x)$. Inspired by [11], a contextualized concept embedding method is adopted that is particularly well suited to knowledge graph alignment. Concept embeddings for the attributes such as Name, Label, Comments, Properties, Instance descriptions, Concept attributes and the neighborhood of nearest concepts in the knowledge graph are generated. The auxiliary information is encoded via attribute triples by the concept attributes, where the attribute triples are represented by $< c, t, v >$. Here c, t and v represents the concept, attribute type, and attribute value respectively. The nearest concept embeddings are aggregated to contextualize the embedding in each concept. This aggregation is achieved by the random walks in the knowledge graph, for the concept c in MOMGANI. The embedding $\vec{v}(c)$ can be estimated by the relation.

$$\vec{v}(c) = \sigma(\vec{v}_0(c) + W_n E_n(c) + W_l E_l(c) + W_c E_c(c) + W_{pd} E_{pd}(c) + W_{inst} E_{inst}(c) + W_a E_a(c) + W_{nc} E_{nc}(c)) \quad (5)$$

Where d and σ is the initial dimension and sigmoid function respectively. Also, $\vec{v}_0(c) \in R^d$. $E_n(c)$ and $E_l(c)$ is the aggregate name and label embedding respectively. $E_c(c)$ is the aggregate encoding of comments of c, $E_{pd}(c)$ is the aggregate embeddings of property descriptions of c, $E_{inst}(c)$ is the aggregate embeddings of instance descriptions of c, $E_a(c)$ is the aggregate embeddings of attributes of c, $E_{nc}(c)$ is the aggregate neighbourhood context embedding of c, while $W_n, W_l, W_c, W_{pd}, W_{inst}, W_a, W_{nc} \in R^{dxd}$ are weight matrices. More specifically,

(i). The aggregate name, label, comments are computed by max-pooling over the embeddings.

(ii). The attribute value a_v and the attribute type a_t are passed through the fully connected sigmoid layer to obtain the attribute embedding a expressed as

$$a = \sigma(W_{at} a_t + W_{av} a_v) \quad (6)$$

Where $W_{at}, W_{av} \in R^{d \times d}$ are weight matrices.

(iii). The initial embedding vector $\vec{v}_0(c)$ for every concept c_i in the neighbourhood of c is averaged to estimate the aggregate neighbourhood context W_n. Neighbourhood of concept c is obtained by k random walks of length l.

Using the cross entropy q, the quality of the predicted alignment is found using the concept embeddings

$$q(c_y \mid c_x) \forall (c_x, c_y) \in C_X \times C_Y : - \sum_{x \in X} \sum_{y \in Y} 1_{[C_x \equiv C_y]} \, log q(c_y \mid c_x) \quad (7)$$

In the training data, the similarity between concepts aligned can be measured using the concept embedding, that represent the small subset of $C_X \times C_Y$. The indicator function $\phi(c_x, c_y)$ for (7) is given by

$$\phi(c_x, c_y) = \begin{cases} 1_{[C_x \equiv C_y]} & if \quad C_x \; is \; aligned \\ \frac{1}{N_{unl}} & if \quad C_x \; is \; unlabeled \end{cases} \quad (8)$$

Where $\frac{1}{N_{unl}}$ and N_{unl} represents the uniform distribution and number of currently unaligned concepts respectively. The alignment classification loss can be estimated using the indicator function ϕ as

$$\mathcal{L}_G = \sum_{x \in X} \sum_{y \in Y} \phi(c_x, c_y) \; logq(c_y \mid c_x) + \sum_{x \in X} \sum_{y \in Y} \phi(OP_x, OP_y) \; logq(OP_y \mid (OP_x and \; (C_x \equiv C_y))) \quad (9)$$

The maximum likelihood alignment between X and Y can be enabled by minimizing $\mathcal{L}_G$that makes the MOMGANI to learn the probability alignment function. For every tuple $< c_y, c_x >$ where $q(c_y \mid c_x)$ is above a threshold the alignment is suitable to be evaluated and considered. The subsets of X and Y therefore also will likely have a probability that q can be max-weighted for matching. OntoGenerator learns valid triples that are generated using the concepts in the set $C_X \cup C_Y$ and by using the feedback from the discriminator. The less likely matching is replaced by a better match for a concept based on q. The replacement is done as alignments are found, newly added concept matches are appended into A'_{i+1}. When nothing new can be matching the algorithm terminates. Algorithm 2 provides the high-level steps followed within the Generator for the estimation of triples.

3.3.2. *OntoDiscriminator*

In this section, the discriminator uses a novel Long Short-Term Memory (LSTM) network with a relation attention module that improves the matching score for a triple being evaluated. Full use of supervision information is possible only when multiple valid instances for training the model with emphasis on relations are used. The entity representation of the attention module can be made better by using entity descriptions that give more background information about the entities. This also improves the performance of the model. The neural network architecture of the ontology discriminator (ontoDiscriminator) is depicted in Fig. 3. It consists of three

---

Algorithm 3.2 Algorithm for the estimation of Triples

Input: $S_1 = \{< C_{x1}, C_{x2}, \text{k} >, < C_{x1}, C_{x3}, \text{k} >, \ldots < C_{xm}, C_{x(m-1)}, \text{k} > | C_{xm} \neq C_{x(m+1)}\}$ and $S_2 = \{< C_{y1}, C_{y2}, \text{k} >, < C_{y1}, C_{y3}, \text{k} >, \ldots < C_{yn}, C_{y(n-1)}, \text{k} > | C_{yi} \neq C_{y(i+1)}\}$

Output: New triples combining Ontology 1 and Ontology 2

For $\forall (c_y, c_x) \in S_1 \cup S_2$ **do**

q$(c_y | \, c_x) = softmax(sim(c_x, c_y))$

for $\forall (OP_x, OP_y) \in properties \; (c_x, c_y)$ **do**

q$(OP_y | \, OP_x) = softmax(sim(OP_x, OP_y))$

$SimMap.add(< c_x, c_y, OP_y >, \text{q})$

$SimMap.add(< c_x, c_y, OP_x >, \text{q})$

end for

end for

Output the Triples from *SimMap* minimizing on loss $\mathcal{L}_G$

modules: The OBi-LSTM (Ontology Bi-LSTM), Ontology Attention Module (OAM), and an Entity Description Module (EDM). The Obi-LSTM layer consists of a unique combination of three Bi-LSTM layers arranged to encode the Triples with the end goal of outputting a confidence score on the provided Triple. The Entity Descriptions Module is responsible for augmenting the Obi-LSTM layer with descriptions of the Concepts, Relations. The Ontology Attention Module is composed of an Attention Layer that tracks the Ontology specific properties such as Name, Comments, Description, Properties, etc., and a Softmax Classifier.

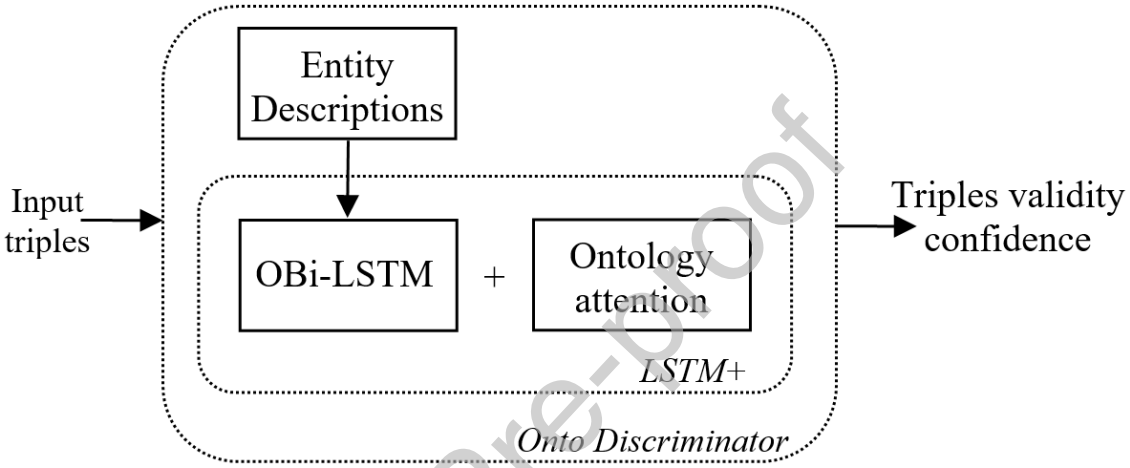

Fig. 3. OntoDiscriminator architecture

3.3.3. *Vector Representation*

Low-dimensional vectors must be estimated from the word tokens since a neural network model is used. In the current method, the words are transformed into numerical vectors by referencing the domain specific pre-trained word embeddings. The word embeddings map every word in the corpus to a n-dimensional vector that can act as a distributed representation of words. It has been proved to be very effective in many NLP tasks. E denotes the trained

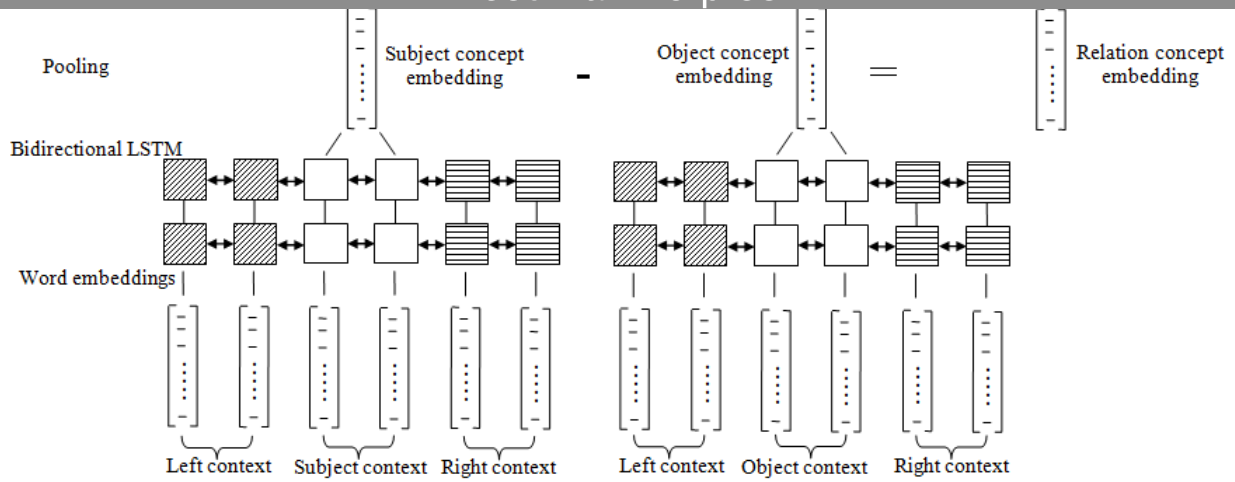

Fig. 4. OBi-LSTM Module

word embeddings using the suggested approach.

3.3.4. *OBi-LSTM (OLSTM)*

This module is used to learn the concepts and their relation of a triple. To provide inputs to this module a Vector Representation of the Subject is required.

The OBi-LSTM model is depicted in Fig. 4. Multi-layer bi-directional LSTM is the core part of the model. To embed text inputs, we used pre-trained word embedding. In the first step, we lookup the head concept h from the triplet (h, r, t) in the training dataset T. Different name variations from ontology is used in place of the concept with probability α for training purposes. To retrieve the sentences from the context corpus we use the head concept embeddings h of length n as keywords. We ensure sentences that have the selected keywords lie together to form phrases. Of the many sentences that match top 10 results are selected for the context $hct_{1\ldots m}$ for concept h. The LSTM network is then used to encode the context sentence hct. The resulting output retains the position to correspond to the concept name in the sentence, this is achieved by multiplying the output of the sequence $cct_{1\ldots m}$ of the LSTM network with a mask. The output is then converted to a normalized vector h by passing through a max-pool layer. The single vector for the tail concept is also estimated using the same procedure. To model the relation between the concept, a vector addition in embedding space is used, which is similar to TransE except that the place of entity embeddings is replaced by LSTM outputs. New triplets (h', r, t') are generated by replacing head, tail concepts with a

---

Algorithm 3.3 Training Obi-LSTM

Input: Training set of triplets $T = \{(h, r, t)\}$, relation L and concept names $C = \{C_{1\ldots n}\}$. Vocabulary V and word

embeddings E. Context corpus $S = \{S_{1\ldots n}\}$.

Output: Trained LSTM network

loop

for (h, r, t) ∈ SimMap **do**

// lookup head concept names

$C = \{C_{1\ldots n}\} \leftarrow lookup((C, c, h)$

//retrieve context sentences

$C_{1\ldots m}^{cth} \leftarrow retrieve\ (S, \{c_{1\ldots n}\})$

$C_{1\ldots m}^{cth} \leftarrow$ LSTM $(E(C_{1\ldots m}^{ct}))$

$C^h \leftarrow$ pool $(C_{1\ldots m}^{cth})$

//look up tail concept names

$\{C_{1\ldots n}\} \leftarrow$ lookup(C, c, t)

//retrieve context sentences

$C_{1\ldots m}^{ctt} \leftarrow$ retrieve $(S,\ \{C_{1\ldots n}\})$

$C_{1\ldots m}^{ctt} \leftarrow$ LSTM $(E(C_{1\ldots m}^{ctt}))$

$C \leftarrow |C^t - C^h|$

end for

Update network w. r. t. $\Delta[\gamma + d(h - t, r) - d(h' - t', r)]$

end loop

return Trained OBi − LSTM network

---

random concept, for training the model. The margin-based ranking-loss is minimized while the model is trained and the margin-based ranking is given by

$$\Delta[\gamma + d(h - t, r) - d(h' - t', r)] \tag{10}$$

We have used 2-layer bi-directional LSTM network having a 200 dimension and 200 dimensional word embeddings, for the experiments. In the estimation of ranking loss L, we have used the Euclidean norm in distance function $dist$ and margin as $\gamma = 0.2$. To optimize the network Vanilla stochastic gradient descent with a learning rate $l = 1.0$ is used. Due to the size of the dataset, the hyper-parameter values are heuristically chosen.

3.3.5. *Ontology Attention (OA)*

The most essential part of the ontoDiscriminator is the attention mechanism. Fig. 5 illustrates the model of the attention layer. In the attention model, the lower weights are learned for the invalid instances whereas the higher weights are learned for the valid instances [19]. The softmax

classifier then gets the OBi-LSTM output as the input. Relation classification for ontology alignment is treated to be a translation from the entity h to the tail entity t. The relation is expressed as a triplet $r(h,t)$. In [11] the embeddings are expressed as $h + r * t$. Motivated by [20], feature embeddings considered are related by the absolute difference between the feature embeddings of entities defined by

$$v_{relation} = h - t \tag{11}$$

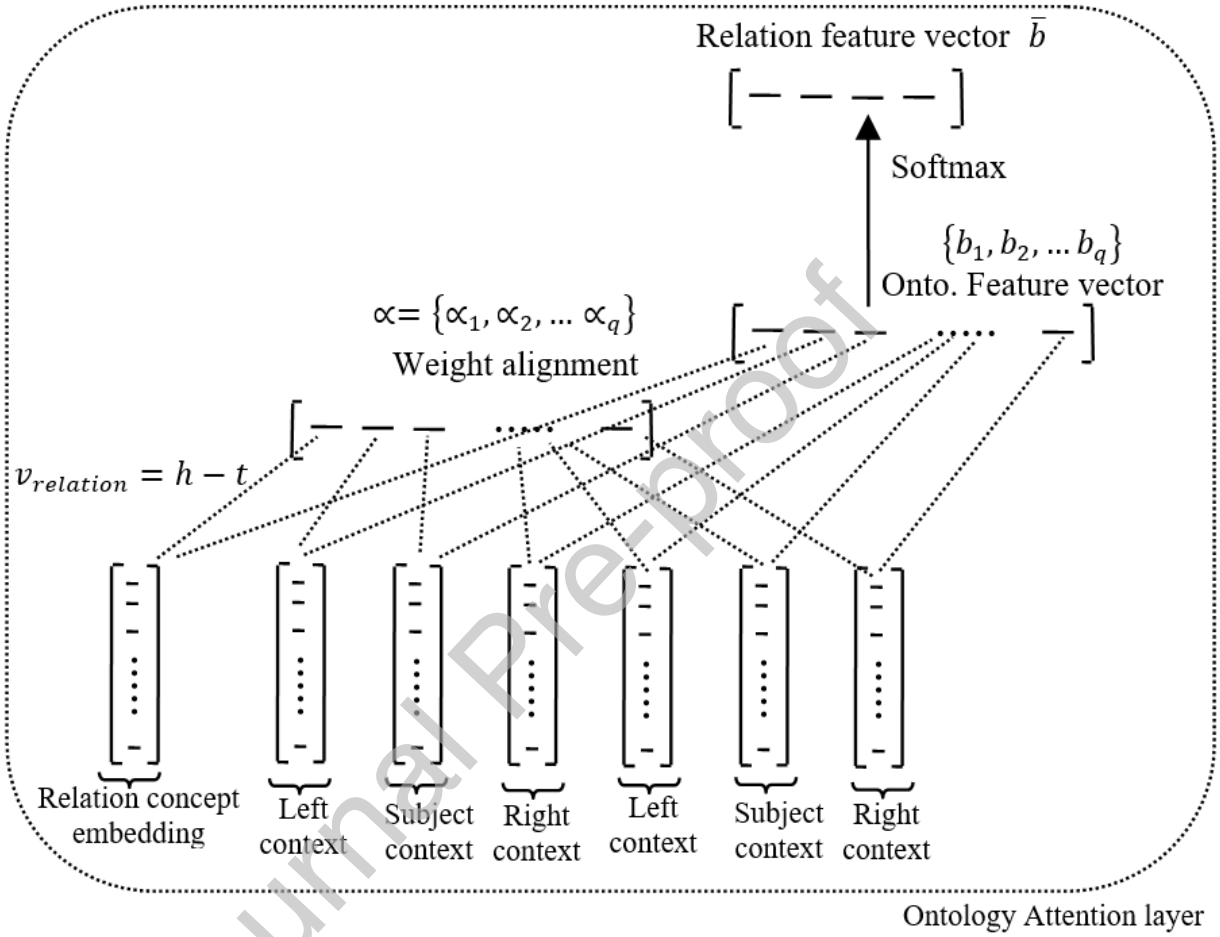

Fig. 5. Ontology Attention Layer

Every instance in the input may or may not have the relation r. If the feature vector has high similarity with $v_{relation}$, then the instance express the relation r. Similarly if the feature vector have a lower similarity with $v_{relation}$, then the instance does not have the relation r. Let $\{b_1, b_2, \dots, b_q\}$ represent feature vectors computed by OBi-LSTM, then $v_{relation} = h - t$ is used to denote the relation between h and t. The similarity or relatedness (attention weight) between $v_{relation}$ and each instance feature vector is given by,

$$\alpha_i = \frac{exp(\omega_i)}{\sum_{j=1}^{q} exp(\omega_b)} \tag{12}$$

$$\omega_i = W_a^T(\tanh[b_i; v_{relation}]) + b_a \quad (13)$$

where $[x_1 : x_2]$ denotes the vertical concatenation of $x_1, x_2,\ 1 \le i \le q$. The intermediate matrix is $W_a \in R^{1\times(3n+k)}$ and the offset value of the instances is $\alpha = \{\alpha_1,\ \alpha_2, \dots,\ \alpha_q\}$. The overall weightage can be expressed as.

$$\bar{b} = \sum_{i=1}^{q} \alpha_i b_i \quad (14)$$

where $\bar{b} \in R^{3n}$

3.3.6. *Entity Descriptions (ED)*

Entity descriptions provide supplementary information about the concept under study. Here another traditional CNN which uses a single max-pooling layer and a convolution layer is used to extract the features from the entity descriptions. The pair that resembles (entity, description) is denoted by

$$D = \{(e_i, b_i)\ i = 1, 2, \dots, |D|\} \quad (15)$$

The vector of e_i and that of words in descriptions are obtained by looking up the word embeddings E. The CNN is used to estimate the vectors of d_i, whose weight matrices are represented by $\widehat{W}_d$. In this approach, the entity vector is chosen to be near to the descriptions, where the error between e_i and d_i is given by the relation.

$$\mathcal{L}_e = \sum_{i=1}^{|D|} \|e_i - d_i\|_2^2 \quad (16)$$

The entities for the attention module can be better represented using the background knowledge extracted from the descriptions that give more information for prediction relation.

3.3.7. *Softmax*

The feature vector $\bar{b}$ is input into the softmax classifier, to estimate the confidence of each reation, given by

$$o = W_s \bar{b} + b_s \quad (17)$$

Where the output of classifier is $o \in R^{n_o}$. The weight matrix is $W_s \in R^{n_o \times 3n}$ and the bias is $b_s \in R^{n_o}$. Let all the parameters be represented as

$$\theta = \left(E, \widehat{W}, PF_1, PF_2, W_a, W_b\right) \quad (18)$$

Let the bag be represented as B. For the i^{th} relation the conditional probability is given by

$$p(r_i|B;\theta) = \frac{exp(O_i)}{\sum_{j=1}^{n_o} exp(o_j)}$$

4. Experimental results and analysis

This section shows the experimental results and analysis of the proposed MOMGANI architecture.

4.1. *Ontology Selection*

In the absence of any reference Ontology for IoT when compared to a very well established domain such as Medicine (such as Unified Medical Language System), the work focuses on producing alignments using three ontologies namely SAREF, oneM2M [21], and SSN [22]. The Smart Applications REFerence (SAREF) ontology is produced by the European Telecommunications Standards Institute (ETSI) and it produces the shared model of consensus that allows for producing machine extractable intelligence and architecting smart applications. SAREF has a core Ontology and built several extensions for the various domains that act as a fundamental block for the development of applications allowing use and reuse of various regions of the ontology-based on their specific requirements. The oneM2M is a leading industry-backed consortium that provides specifications and frameworks that support services and applications like health, public safety, home automation, connected car, and smart grid. It is designed for the extension as deemed fit by organizations and hence it promotes reuse and inter-operations. SSN ontology is produced by the W3C standards group to provide a formalization of describing sensors their descriptions and observation data. It has also enabled more advanced access, expressive representation, and formal analysis of sensor resources. For the lack of having a universal Ontology in the IoT space against which alignments and training can be performed, there is a manual process to select the best candidates produced by the algorithm. A good next step for this algorithm is to produce the candidates for a universal Ontology. Table 1 depicts the Ontology statistics of different alignments with their Classes, Properties, and Instances values.

Table 1. Ontology statistics

Alignment	Classes	Properties	Instances
O1: SAREF	113	94	55
O2: SSN	23	38	2
O3: oneM2M	23	40	0

OntoGenerator is responsible for producing the samples for the Discriminator. The samples are produced using a cross product of all the classes and properties in this format $< O1 : Concept, O1 : Property, O2 : Concept >$. This leads to a huge combinatorial explosion; hence only close matches are selected between the classes and properties. The details of the pre-selection are described in the section Pre-processing.

4.2. *Data*

The experimental evaluation was done using the datasets namely Classes alignment, Property alignment and Instances alignment.

We have used 20% of the dataset for training, 10% of the dataset for validation, and

Table 2. Data used for training, validation and testing for different dataset

		O1: SAREF	O2: SSN	O3: oneM2M
Training	Classes alignment	22	4	4
	Property alignment	18	7	8
	Instances alignment	10	1	0
Training	Classes alignment	11	2	2
	Property alignment	9	4	4
	Instances alignment	6	1	0
Testing	Classes alignment	80	17	17
	Property alignment	67	27	28
	Instances alignment	39	0	0

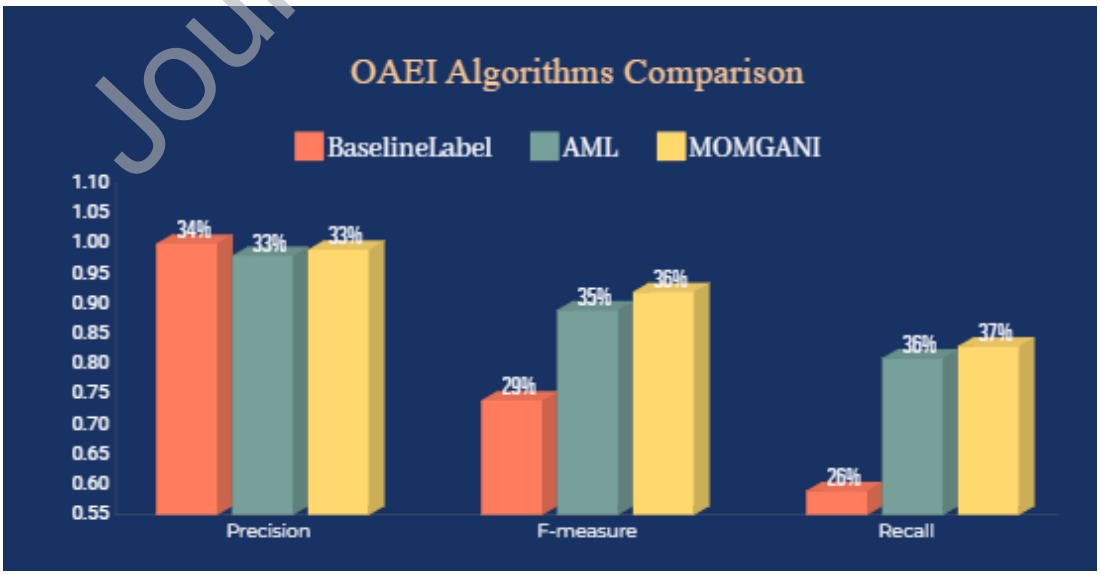

Fig. 6. Performance comparison of MOMGANI with baseline and AML

70% of the dataset for testing as listed in the table 2. We perform the tests as prescribed by the OAEI initiative [23] by evaluating Precision, Recall, and F-Measure. Note that it is not required that the matching system need to match instances of classes. Since there is no universal alignment ontology such as Unified Medical Language System (UMLS) (in the biomedical domain) available for IoT, we are using the MOMGANI algorithm for the Ontology Alignment Evaluation Initiative (OAEI) ontology for comparison purposes. Here are the results of the MOMGANI algorithm for the test cases prescribed in [23]. The comprehensive list of tests is not performed as prescribed by the SEALS framework. That is a future consideration. The reference Ontology used are Ontology 1 [24] and Ontology 2 [25]. The results depicted in table 3 and 4 are only for class alignment, where the graphical comparison is depicted in Fig. 6.

Table 3. OAEI algorithms comparison

Algorithm	Precision	F-measure	Recall
baselineLabel	1	0.74	0.59
AML	0.98	0.89	0.81
MOMGANI	0.99	0.92	0.83

The Precision, F-measure, Recall of the proposed MOMGANI algorithm is estimated as 0.99, 0.92 and 0.83 respectively which is higher than the two hand-coded algorithms baselineLabel and AML (agreement marker light) [26]. The comparison of Precision, F-measure, Recall of the proposed algorithm with the two hand-coded algorithms is depicted in table 3.

Methogology

As discussed in the section related to "OntoGenerator", it considers several parameters while it is generating the triples. Based on the parameters considers on the "generator" side, we weigh in the impact on the overall scores of the algorithm. (1) Name (2) Label (3) Comments (4) Properties (5) Instance descriptions (6) concept attributes and (7) the neighborhood of nearby concepts in the knowledge graph are generated. From the OntoDiscriminator side, we consider the training model augmenting the OBi-LSTM algorithm outputs with Ontology Attention and Entity Descriptions. The Entity Descriptions, Ontology attention, and LSTM layer are trained separately on the Ontology triples independently.

4.2.1. Triple Selection

The output from the MOMGANI algorithm is the potentially valid relation triple. The algorithm continues to train until no more triples are generated and validated. The model obtained after the iteration is the real output that can be used further. The top-100 triples

obtained in the model are presented to the experts for the assertion, once approved they can be used to answer queries when several domains and concepts are in operation.

Table 4. Output from the MOMGANI algorithm

SI. No.	OntoGenerator	Precision	Recall	F-measure
1	No Name	0.44	0.34	0.32
2	No Comments, Properties	0.55	0.43	0.39
3	Name	0.73	0.76	0.67
4	Name, Label	0.8	0.79	0.76
5	Name, Label, Comments	0.91	0.8	0.76
6	Name, Label, Comments, Properties	0.93	0.8	0.81
7	Name, Label, Comments, Properties, Instance	0.93	0.88	0.79
8	Name, Label, Comments, Properties, Attributes	0.94	0.87	0.8
9	Name, Label, Comments, Properties, Attributes, Neighbourhood concepts	0.99	0.92	0.83

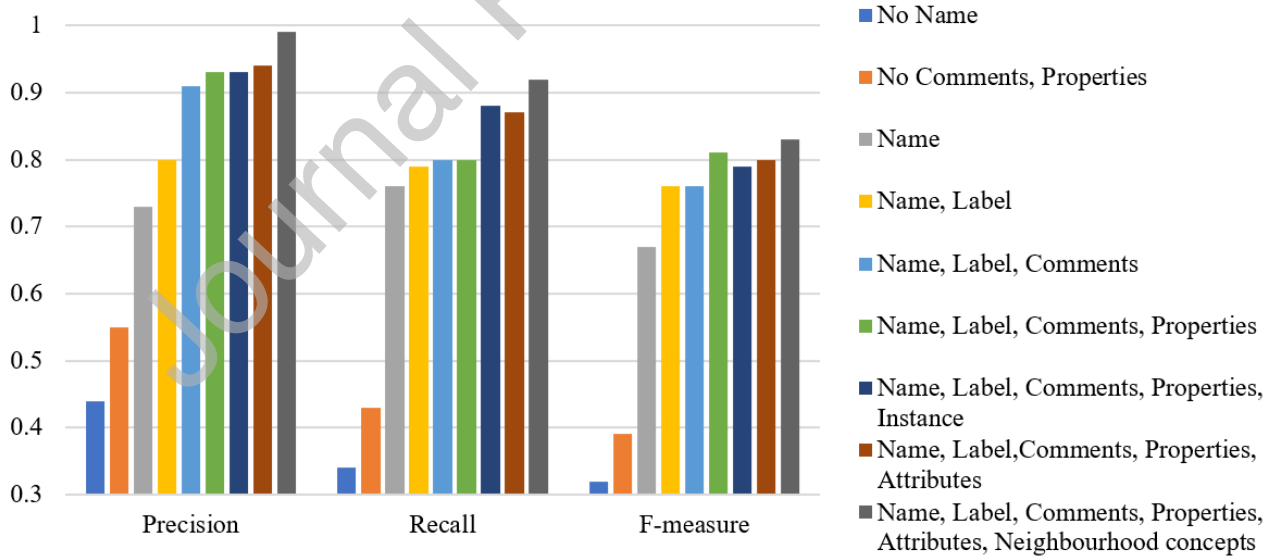

Fig. 7. Performance comparison of MOMGANI with different OntoGenerator

From Figure. 7, it is clearly shown that the findings of the suggested model (MOMGANI) increase the alignment's quality and obtain the greatest values when compared to the most advanced ontology matching systems, demonstrating the ability of the process of iteratively refining the alignment to provide high-quality ontology alignments. Finally, it should be

observed that the system of ontology matching technique may successfully match ontologies with various heterogeneous properties.

4.2.2. *Aligned Ontology*

Fig. 8 illustrates the aligned ontology in action, for a request from a user or a machine where Knowledge Base (KB) can resolve the query as the concepts and relations across various ontologies and related domains are established using MOMGANI. The KB can learn and resolve the queries from across Ontologies provided the axioms are established and rule inference engines are updated to look across the ontologies. The MOMGANI algorithm augments the inference mechanism to look beyond just one ontology and transparently provides replacements for the concepts and the relations.

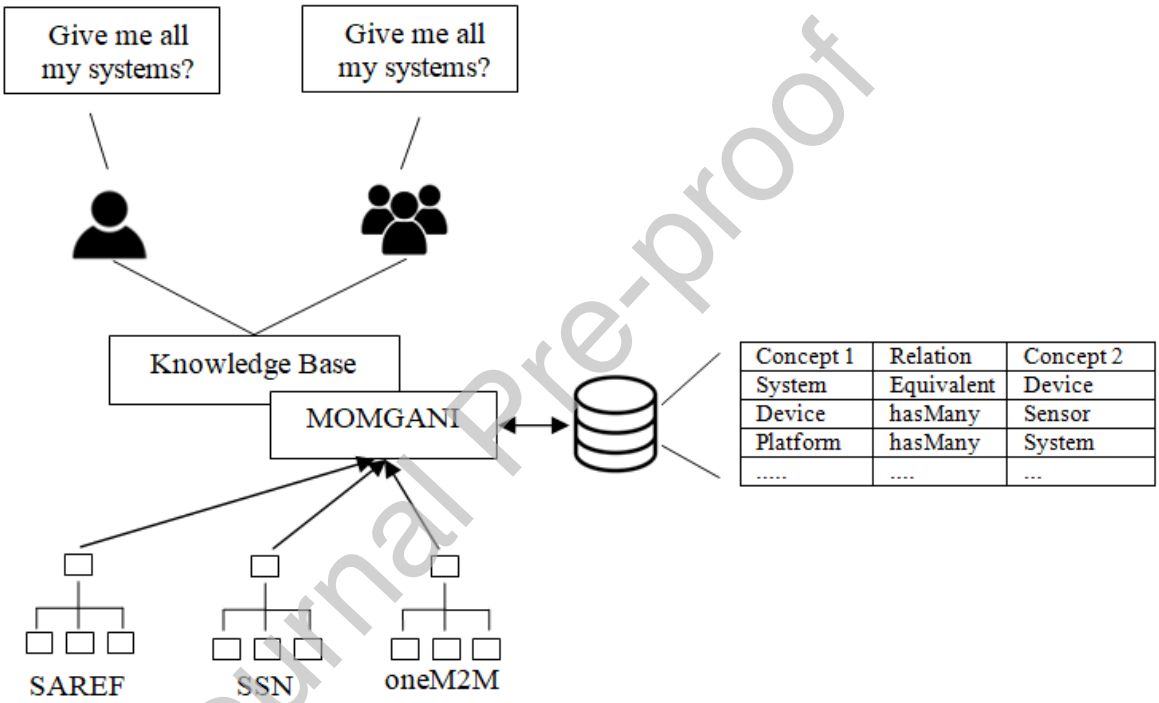

Fig. 8. MOMGANI Aligned Ontology

4.3. *Discussion*

In general, the results show that MOMGANI can align distinct knowledge graphs such that MOMGANI can be used for both representation learning and ontology alignment. Table 4 shows the comparison of MOMGANI results for different OntoGenerator with OLSTM + OA + ED and the graphical comparison is depicted in Fig. 8. The precision, recall, F-measure is maximum for OntoGenerator <Name, Label, Comments, Properties, Attributes, Neighbourhood concets >. The precision, recall, F-measure was estimated as 0.99, 0.92 and 0.83 respectively for this OntoGenerator.

As future work, MOMGANI can consider alignments across several related ontologies. Unlike AML, MOMGANI produces potentially new relations between concepts across ontologies and not just alignment between concepts and properties. A unique combination of a Generator based on a probabilistic classifier and a Discriminator based on a novel LSTM with relation attention assisted by entity descriptions will be able to produce the best possible alignment between ontologies. In the IoT space lack of a generic universal ontology will be a big gap to fill compared to other domains such as biomedicine where UMLS provides a reference ontology for matching and alignments. IOT-O has been selected as the reference ontology in this work because it is the most comprehensive ontology covering several important concepts in the IoT space. SSN covers only sensors and actuators while SAREF and oneM2M cover only certain concepts of IoT domains. An aligned ontology in such a scenario where there is a lot of fragmentation and incompleteness is very important to build an intelligent M2M communication system.

The discriminator being the central part of the whole solution is different from all the existing works in many ways, it is an algorithm that is built for alignment of Ontologies as it is fine-tuned to relate concepts and relations not just within a given ontology but against several ontologies. This is possible only when the algorithm can consider several important parts of the ontology for alignments like the name, label, comments, properties, instance descriptions, and context around the class. The learning of the concepts and relations is improved to a large extent by adding the relation-weights and entity descriptions that augment the learning process and they are crucial as they bring in knowledge from across several domains (distant supervision). Training the Generator-Discriminator network is the hard part as convergence is hard to identify. In the first round, the generator is programmed to output the fake triples using the concepts from both the ontologies while the discriminator is trained using both the fake and real concepts triples. In the second round, the generator is programmed to output more fake triples, which is the only input provided to the discriminator for training. This ensures the discriminator can fully realize the fake from the real triples. Such a training optimization ensures MOMGANI can learn the triples correctly when new related ontologies are fed into the algorithm. Also, MOMGANI can produce new triples in the process of alignment of the ontologies that can be used to extract information across varied related domains.

5. Conclusion

In this paper, a MOMGANI algorithm for ontology alignment in the IoT was suggested. Utilizing metrics like recall, precision, and F-measure, the experimental evaluation was carried out utilising the datasets for classes alignment, property alignment, and instances alignment.

where the values for the metrics are 0.92, 0.99, and 0.83, respectively. In order to extend the knowledge base, MOMGANI algorithm moved in the direction of aligning linked ontologies, which would not have been achievable otherwise because it is physically impossible for a human expert to be able to gather knowledge from related and closely related areas. Machines can advance intelligence beyond a single domain of operation because to the capacity to automatically identify aligned ideas and relations. This allows reasoning to expand past domain borders. As some ontologies may not have all of those features defined, some ontology features, such as name, label, etc., must be chosen manually. Instead, some elements must be taught by algorithm or process itself. To enable more precise detection of necessary features to compare when aligning the ontologies, an automatic feature selection approach can be used before the MOMGANI algorithm is executed. The extension of the alignments generated to reach a shared embedding space is another topic of research, allowing ontology alignments to be readily made available for wider application, such as in automated ontology development.

References

1. B. Lima, D. Faria, F. M. Couto, I. F. Cruz and C. Pesquita, Oaei 2020 results for aml and amlc., in *OM@ ISWC*, 2020, pp. 154–160.
2. C. Carlsson, M. Brunelli and J. Mezei, Decision making with a fuzzy ontology, *Soft Computing* **16**(7) (2012) 1143–1152.
3. C. De Maio, G. Fenza, D. Furno, V. Loia and S. Senatore, Owl-fc: an upper ontology for semantic modeling of fuzzy control, *Soft Computing* **16**(7) (2012) 1153–1164.
4. C.-S. Lee, M.-H. Wang, Y.-C. Hsiao and B.-H. Tsai, Ontology-based gfml agent for patent technology requirement evaluation and recommendation, *Soft Computing 23*(2) (2019) 537–556.
5. E. Alkhammash, Formal modelling of owl ontologies-based requirements for the development of safe and secure smart city systems, *Soft Computing* **24**(15) (2020) 11095–11108.
6. G. Bajaj, R. Agarwal, P. Singh, N. Georgantas and V. Issarny, A study of existing ontologies in the iot-domain, *arXiv preprint arXiv: 1707.00112* (2007).
7. G. Ji, K. Liu, S. He and J. Zhao, Distant supervision for relation extraction with sentence-level attention and entity descriptions, in *Proceedings of the AAAI Conference on Artificial Intelligence*, **31**(1) 2017.
8. H.-D. Huang, C.-S. Lee, M.-H. Wang and H.-Y. Kao, It2fs-based ontology with soft-computing mechanism for malware behavior analysis, *Soft Computing* **18**(2) (2014) 267–284.

9. J. Perez, R. Wiksrom, J. Mezei, C. Carlsson and E. Herrera-Viedma, A new consensus model for group decision making using fuzzy ontology, *Soft Computing* **17**(9) (2013) 1617–1627.
10. J. Liu, X.Zhang, Y. Li, J. Wang and H.-J. Kim, Deep learning-based reasoning with multi-ontology for iot applications, *IEEE Access* **7** (2019) 124688–124701.
11. L. Otero-Cerdeira, F. J. Rodriguez-Martinez and A. Gomez-Rodriguez, Definition of an ontology matching algorithm for context integration in smart cities, *Sensors* **14**(12) (2014) 23581–23619.
12. M. B. Alaya, S. Medjiah, T. Monteil and K. Drira, Toward semantic interoperability in onem2m architecture, *IEEE Communications Magazine* **53**(12) (2015) 35–41.
13. M. Compton, P. Barnaghi, L. Bermudez, R. Garcia-Castro, O. Corcho, S. Cox, J. Graysensor network incubator group, *Journal of Web Semantics* **17**(2012) 25–32.
14. M. Palmonari and P. Minervini, Knowledge graph embeddings and explainable ai, *Knowledge Graphs for Explainable Artificial Intelligence: Foundations, Applications and Challenges, IOS Press,, Amsterdam* (2020) 49 – 72.
15. M. Sohn, S. Jeong and H. J. Lee, Case-based context ontology construction using fuzzy set theory for personalized service in a smart home environment, *Soft Computing* **18**(9) (2014) 1715–1728.
16. Ontology1: (http://dbkwik.webdatacommons.org/starwars.wiki.com/.
17. Ontology2: (http://dbkwik.webdatacommons.org/swg.wikia.com/.
18. R. M. Maldonado and S. M. Harabagiu, Bootstrapping adversarial learning of biomedical ontology alignments, in *AMIA Annual Symposium Proceedings*, **2019**, American Medical Informatics Association2019, p. 627.
19. S. Hertling and H. Paulheim, The knowledge graph track at oaei, in *European Semantic Web Conference*, Springer2020, pp. 343–359.
20. T. Mikolov, K. Chen, G. Corrado and J. Dean, Efficient estimation of word representations in vector space, *arXiv preprint arXiv:1301.3781* (2013).
21. V. Rajeswari, M. Kavitha and D.K. Varughese, A weighted graph-oriented ontology matching algorithm for enhancing ontology mapping and alignment in semantic web, *Soft Computing* **23**(18) (2019) 8661-8676.
22. X. et al., An improved multi-objective evolutionary optimization algorithm with inverse model for matching sensor ontologies, *Soft Computing* (2021) 1–14.
23. Z.Zhang, S. Liu, W. Gao, J. Xu, S. Zhu, An enhanced multi-objective evolutionary optimization algorithm with inverse model, *Information Sciences*, **530** (2020) 128-147.

24. X. Xue, Y. Wang and W. Hao, Using moea/d for optimizing ontology alignments, *Soft Computing* **18**(8) (2014) 1589-1601.
25. X. Xue and J. Liu, Optimizing Ontology Alignment Through Compact MOEA/D, International Journal of Pattern Recognition and Artificial Intelligence, **31** (4) (2017) 1759004.
26. X.-m. Yu, W.-z. Feng, H. Wang, Q. Chu and Q. Chen, An attention mechanism and multi-granularity-based bi-lstm model for Chinese q&a system, *Soft Computing* **24**(8) (2020) 5831–5845.
27. Li Xiujuan, M. kagita and R.L.kumar, Machine Learning Techniques for Multi-media Communications in Business Marketing, *Journal of Multiple-Valued Logic and Soft Computing*, **36**(1-3) (2021) 135-150.
28. Zaman, S. K. U., Jehangiri, A. I., Maqsood, T., Umar, A. I., Khan, M. A., Jhanjhi, N. Z., ... & Masud, M, COME-UP: Computation Offloading in Mobile Edge Computing with LSTM Based User Direction Prediction. *Applied Sciences*, **12**(7) (2022) 3312.

Conflict of interest

There is no conflict of interest among the authors.