AUTOMATED KNOWLEDGE UNDERSTANDING AND RECOGNITION ASSISTANT (AKURA)

Project ID: 17-026

Methodology Report

H.H.N.C.Jayanandana
H.P.N.H.Herath
H.M.S.Piyasundara
R.Rishanthakumar

Bachelor of Science Special (honors) In Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology
Sri Lanka

May 2017

Table of Contents

1. Natural Language Processing and Data Extraction..... 3

2. Ontology Based Information Extraction..... 6

2.1 Meaningful information extraction..... 7

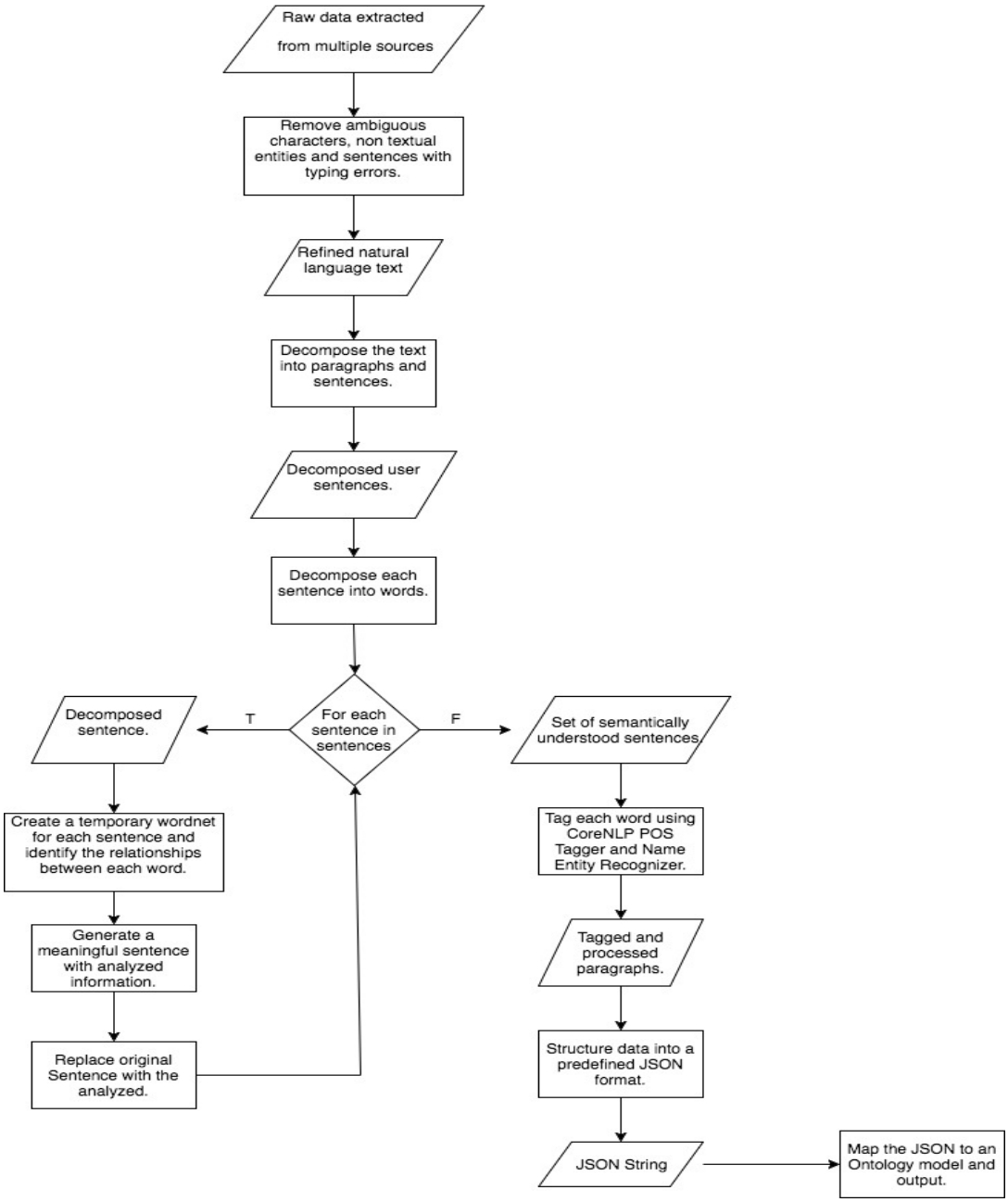
2.2 Theme concept understanding..... 7

3. Ontology Mapping and Integration..... 8

4. Knowledge Retrieval And Representation..... 16

1. Natural Language Processing and Data Extraction

This component will basically focus on the data retrieval from dynamic resources, refine the retrieved data, tokenizing, and semantically understand the data and creation of multiple ontologies. The process followed in order to achieve the expected result will be represented in the following flow diagram.



As shown in the chart, the valuable data extraction will be generated using a series of processing that will be done to the raw data extracted from the APIs. The data that will be extracted from the APIs will depend on the user queries that will be entered into the system. Once the raw data are extracted, The collected data will be refined so that the data will not contain any ambiguous entities such as Images, Non ASCII characters, typing errors and emoticons. Once the paragraphs are all validated, the text will be sent into the understanding mechanism after the text is decomposed into paragraphs and sentences.

Each extracted sentence will be subjected to further tokenization into separate words and then the system will generate a wordnet for the sentence to simulate a real world understanding process. Then after the analysis is done, a sentence with the true meaning of the raw sentence will be generated and then the old sentence will be replaced by the understood sentence. Once all the sentences are being replaced, the same mechanism will be applied to the whole paragraph in order to understand the whole meaning of the paragraph. Then a paragraph with only the meaningful data will be resulted and the resulting paragraph will be sent into the tagging mechanism for POS(Part-of-Speech) and NER(Named Entity Recognition) tagging processes. The POS and NER tags will be identified by using CoreNLP and the tagging process algorithm will be implemented by ourselves.

The tagging process will again subject the processed information into segmentation for easy tagging and for each word, its POS and NER tags will be appended to the end of the word to make it easy to create the ontologies later. After the tagging process is done, the processed text will be mapped into a predefined JSON structure so that it can be used by the other components for other processing purposes. Then the JSON will be mapped into multiple ontologies of XML format and will be passed into the next component.
The main 3 contents of this component will be:

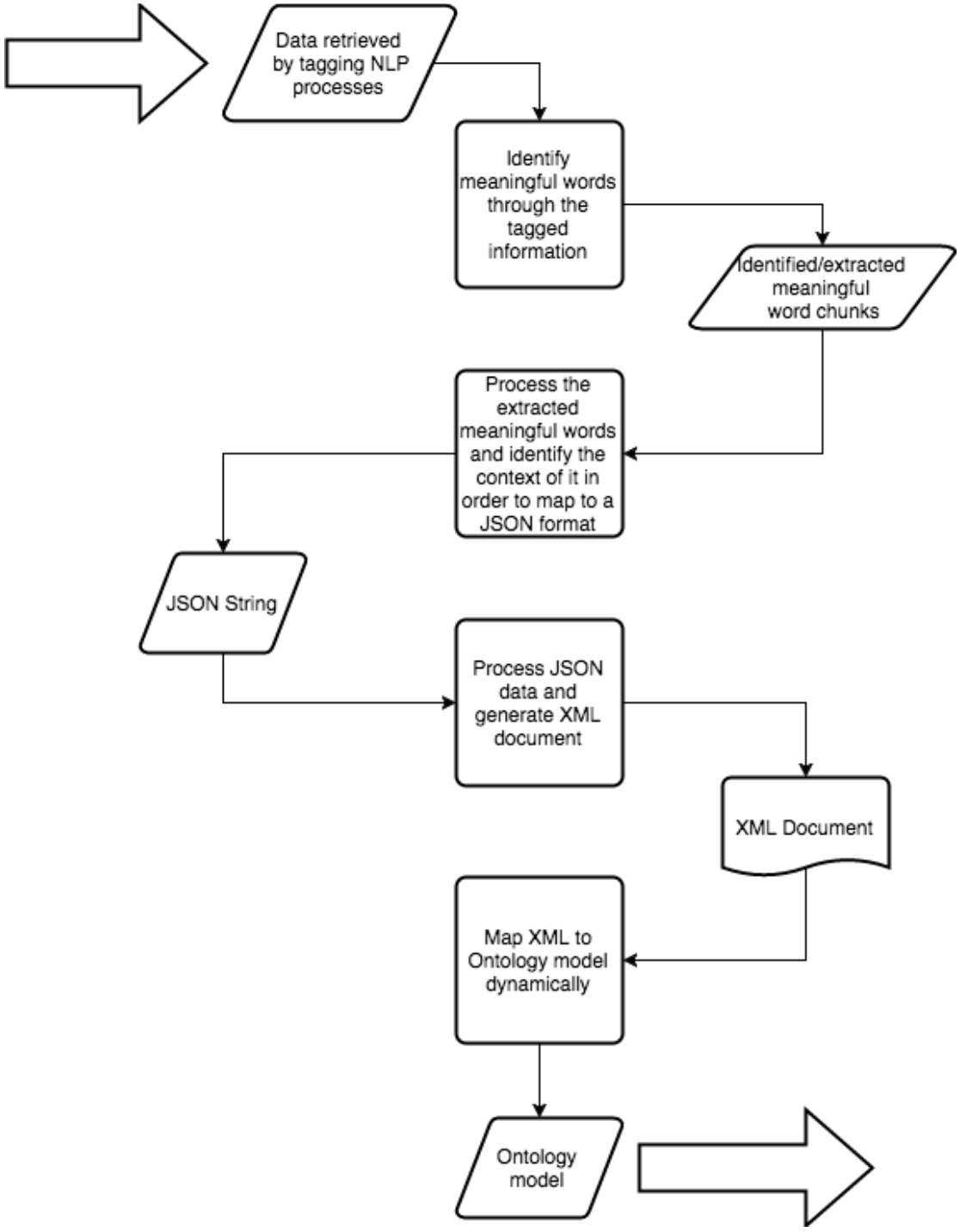
- The data extraction algorithm.
    - This will handle the extraction of data from a multitude of data sources available on the web. The data that will be extracted will depend on the user query.

- The WordNet generation algorithm.
    - This will be the core algorithm that will make up the whole component. This algorithm will make sure that only the most meaningful and the useful data only will be extracted from the input text, making the output more accurate and reliable.
- The tagging algorithm.

  This algorithm will use CoreNLP framework to detect and identify the properties of each words and for each word, append the tags. After tagging, the algorithm will generate the JSON string that will be used to generate the ontologies.

2. Ontology Based Information Extraction

In this document it describes the main objective of OBIE component and the methodology of achieving the expected functionality. Mainly this component will divide into two sectors, which can be called as meaningful data extraction and theme context understanding.



2.1 Meaningful information extraction

According to the figure 1, OBIE component is depending on the data which is processed and tagged by the NLP process which was performed by the previous component. So that it will be the main input to this component. After retrieving the above-mentioned data it will go through an identifying process where it can identify the meaningful word chunks, which can be used in the future understanding processes. In this process we are using Stanford coreNLP as a tool to extract the words and also to identify the context of single words.

After going through this coreNLP process it will store the word/data chunks which have been identified their context. This will be the main input for the next section of this component where it is going to understand the theme concept.

2.2 Theme concept understanding

As mentioned in the above, this section will mainly focus on understanding the theme concept of the data chunks.

This is going to perform by help of an algorithm where it can identify and map the relationships between the words which can be called as entities and finally it will generate a JSON string by representing the relationships of the entities and the information corresponding to each entity.

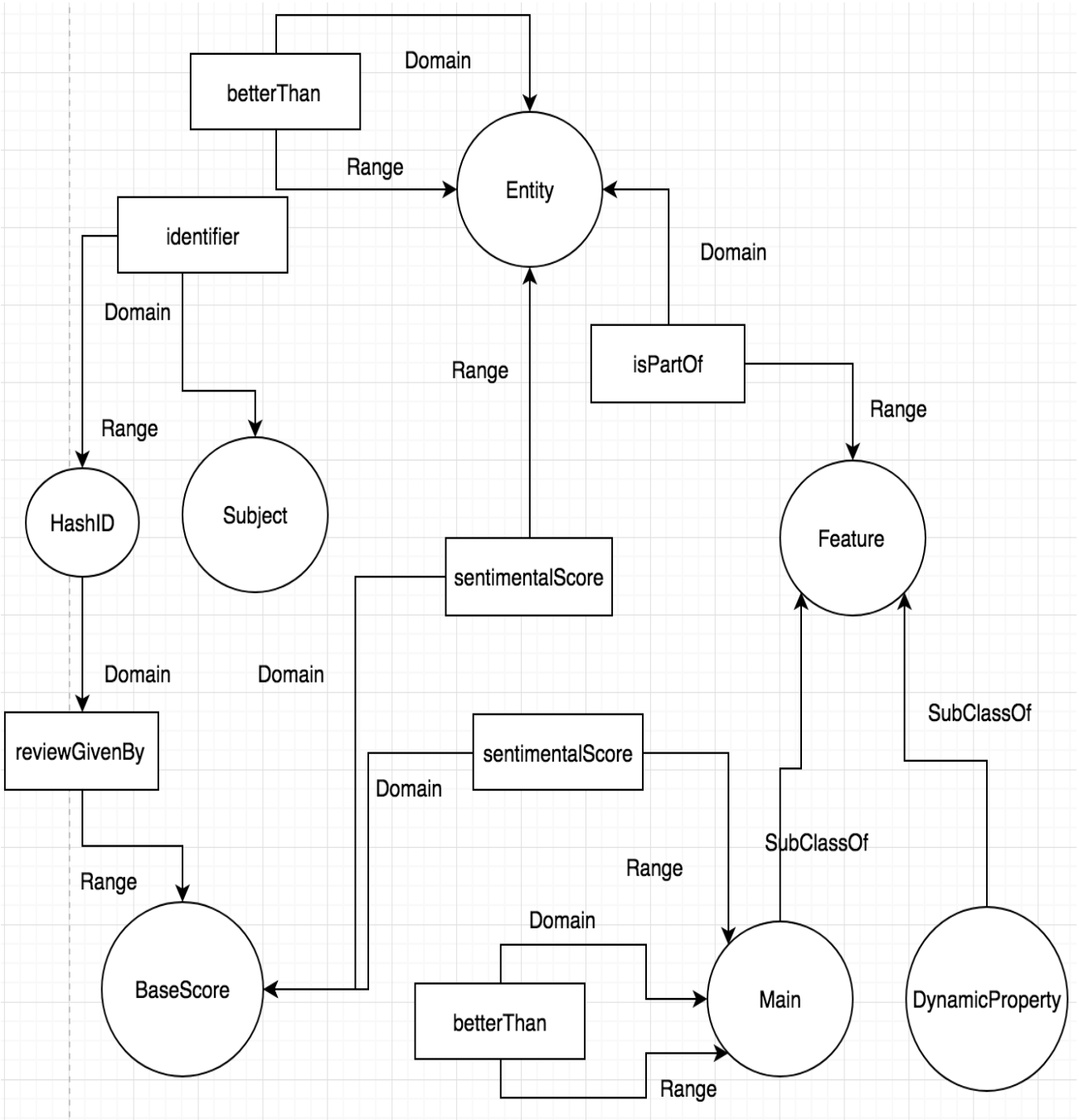
After generating the JSON string we are going to convert this JSON string to a XML document since it will be more readable and strong data representation method which can be used when dealing with ontologies.

As previously stated, XML document will be used as the main input for generating dynamic ontologies. In here we are going to use XSTREAM as a tool which can be helpful in JSON to XML conversions and also Apache JENA as the main ontology mapping tool.

In this process it will mainly focusing on dynamic creation of schema which can be used to map knowledge on to ontologies. Finally this component will output the temporary/disposable ontology models, which will be the main input to the next component, which is called as Ontology based knowledge mapping and integration.

3. Ontology Mapping and Integration

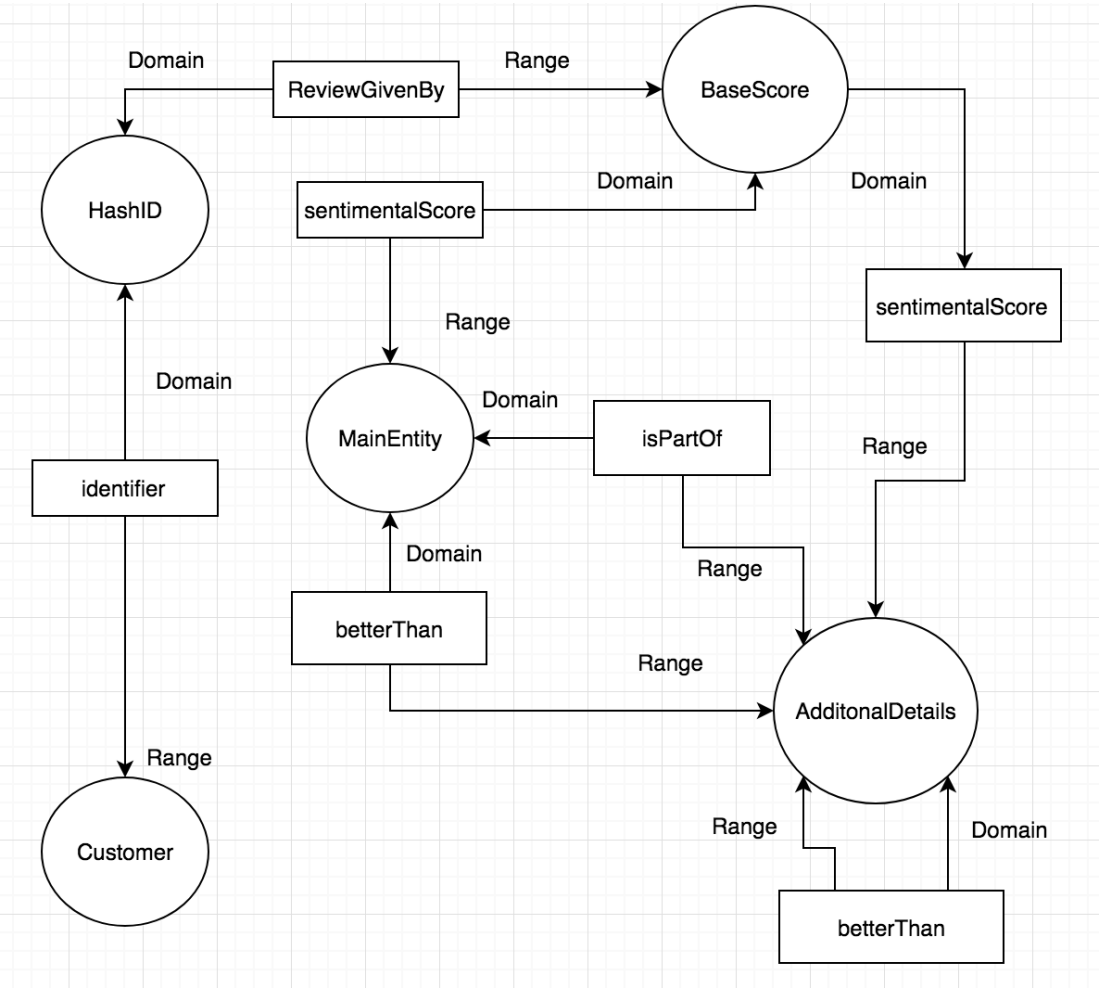
Main Ontology Structure



Class	Description
Entity	This contains the main information of the main entity of the system. In the proof of concept this will contain the main information about the electronic devices.
BaseScore	This contains an integer or a decimal value where it will show the sentimental weight of a comment in a numeric value
HashId	This contains a Hash Key Id to store the information of previous ontologies which were mapped to the system
Subject	This contains information about the person who is responsible for initiating a baseScore value. In the proof of concept product, this will be the reviewer.
MainFeature	Main Feature is the singular occurrence of an instance in the main entity which will have its own set of properties
DynamicProperty	Dynamic Properties will contain multiple occurrences for an instance in the main entity itself.

Table 1 – Main Ontology Class Description

Dynamic Ontology Structure



Class	**Description**
HashId	A unique key to identify the ontology instance
BaseScore	This contains an integer or a decimal value where it will show the sentimental weight of a comment in a numeric value

Customer	This contains information about the person who is responsible for initiating a baseScore value. In the proof of concept product, this will be the reviewer.
MainEntity	This contains the main information of the main entity of the system. In the proof of concept this will contain the main information about the electronic devices.
Additional	This will contain a set of all the additional unrecognized class instances in the ontology.

Table 2 – Dynamic Ontology Class Description

Training Data Set for Supervised Learning

Step 1 - Determining the unique features in each label

In the primary ontology, every class name will be a label and in order to provide a training data set for the classifier this component is going to implement under supervised learning, it should be decided the unique features of each and every entity class in the main ontology structure. This is basically the set of strict rules we have defined in our static OWL ontology. The unique features are provided in the table below.

Class	Identified Features (+ depicts will or will not)
Entity	Domain of isPartOf property + betterThan property + sentimentalScore property

BaseScore	Always be an Integer or decimal
MainFeature	Range of isPartOf property + betterThan property + sentimentalScore property
DynamicProperty	Range of isPartOf property

Table 3 – Unique features Identified for each class

This can be depicted in a csv in the following format in the training dataset

<label name>
<betterThan property> (1=Domain 2=Range 0= None)
< isPartOf property> (1=Domain 2=Range 0= None)
<sentimentalScore property> (1=Domain 2=Range 0=None)
<word embedding score>

Step 2 – Training the model

Generate a set of use cases and determine the input fields of the training model data. Word Embedding Score will be taken by using an existing dataset available in the google library published word2vec. Create an input data structure and Insert the said data in the use cases into a CSV and train the model using the said csv dataset with the use of the library Deep Learning 4J.

Ontology Integration Process

Step 1 – Decide whether the Dynamic Ontology should be mapped

In order to prevent duplicate instances in the main ontology, before merging a set of dynamic ontologies, it should be checked whether the said ontologies have been already merged or not. In order to achieve this, a hashID is stored in the main ontology as a property of Subject class instances which will contain a unique ID for each ontology model it is being merged which is provided by the data processing and understanding engine in the AKURA Framework. This should be checked using a simple SPARQL query for the existence of the hash key and if such a key exists the engine should decide not to continue with the integration process. If the hash key doesn't match any keys in the main ontology, then it will continue to the next step. This process should be efficient and less process consuming.

Step 3– Direct Mapping of Known Classes

In the structure of the dynamic ontology which needs to be mapped to the main ontology there exists a few class instances which can directly be mapped to the primary ontology structure as explained previously in table 3.

Step 4 – Reading the ontology and extracting the required data into a Java Object

By using the Jena OWL API, traverse through the entire dynamic ontology xml file provided as a file stream input and traverse the entire ontology and extract the instances in the ontology to a dynamic java object model. In this scenario, for the instances that needs to be classified and mapped to the primary ontology model, it is required to get only the required information to the format required by the deep learning classifier to run its algorithm. Therefore, by using the information of known classes in the given dynamic ontology, required information is extracted from each unknown entity instance to make it run it through the classifier and successfully label the unknown entities.

Step 5 – Ontology mapping using deep learning

In this step, each and every unknown instance extracted from step 3 is passed through a trained model which uses a deep learning classification algorithm with 1 base layer and 1 hidden layers. Deep Learning 4J or DL4J Library is used for this classification. After the objects go through the classification, a confidence score will be emitted by the classifier for each instance allowing the application to automatically decide, to which static ontology class the unknown entity belongs to.

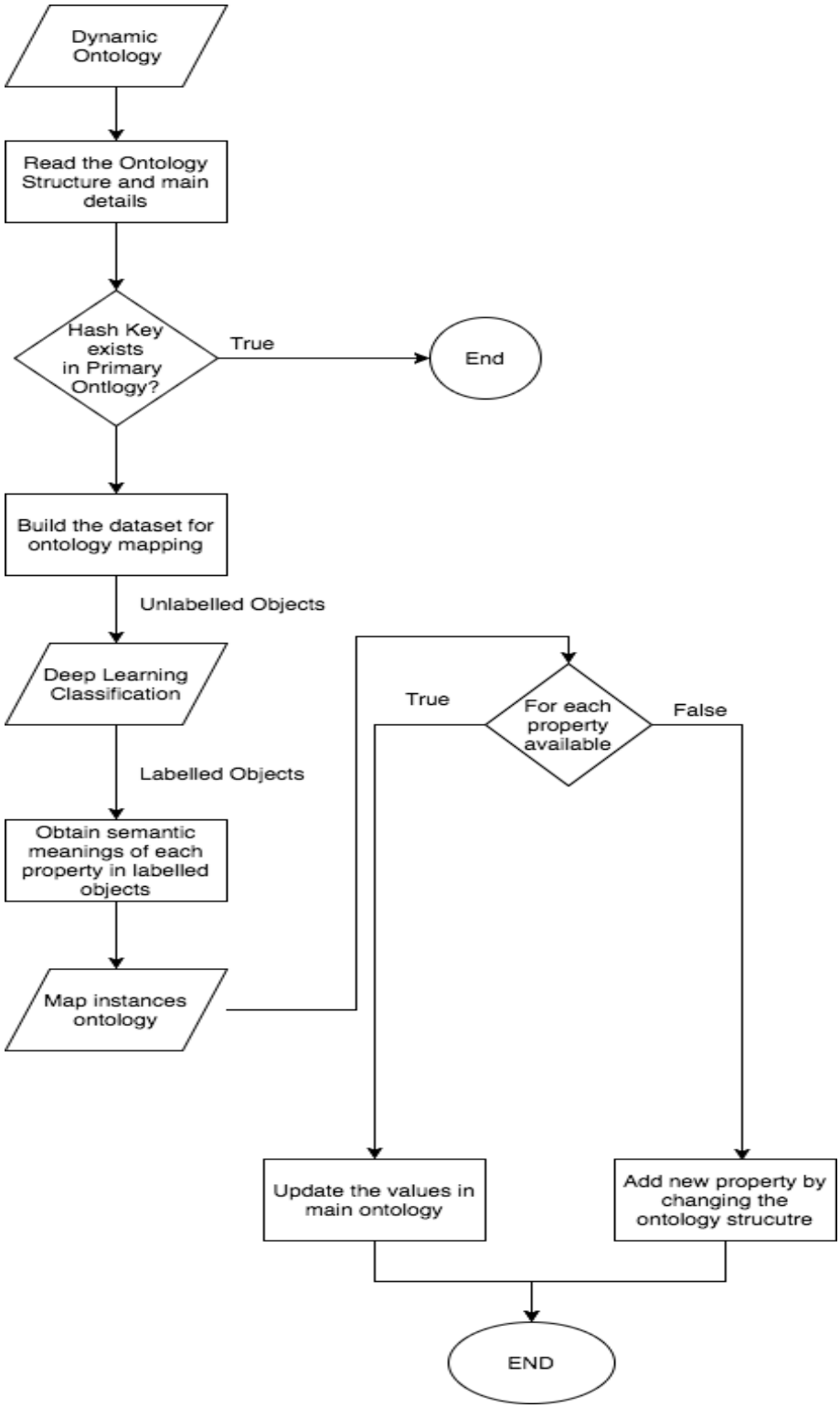
Step 6 – Setting up semantic properties

With unknown ontology classes mapped to the relevant counterparts in the static ontology, next step will be to identify the properties of each class and get the semantic values of the said properties so that duplicate data will not get added in the main ontology. This will be done using word2vec library provided with Deep Learning 4J Library.

Step 7 – Updating the Primary Ontology

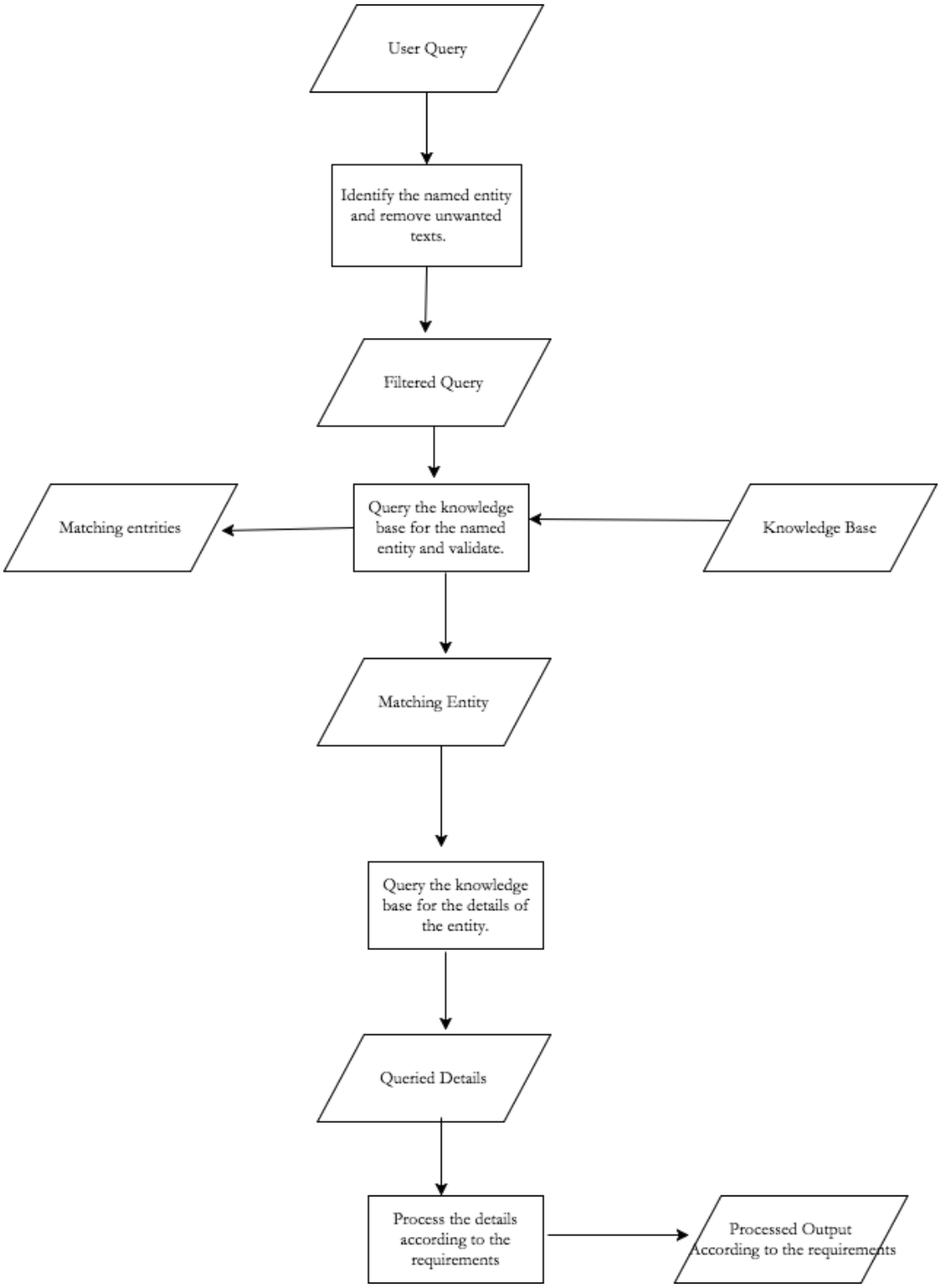
Now with all the unknown instances in the dynamic ontology already labelled and mapped with the labels of existing primary ontology, every instance will be added to the primary ontology according to the label they were given and on update, it will further verify duplicate instances and update accordingly. The primary ontology structure will evolve in this process as all the properties for a given instance in the dynamic ontology should be updated in the primary ontology as well. Due to this, the structure of Data Properties in the ontology entity classes would change with every ontology mapping.

Process Chart



4. Knowledge Retrieval And Representation

This component will be implemented targeting a specific domain where it will retrieve the knowledge from the knowledge base and represent it in a human understandable manner. The following diagram will show the flow of this component implementation



As shown in the figure, user entered query will be filtered in order to get correct results. According to our domain specific application name of an electronic product should be identified. So as a first step unwanted texts addition to the product name should be removed and then it should validated with the existing product names in the knowledge base.

To query the knowledge base ontology traversal API should be written using Jena API and SPARQL. Using this API the relevant entity names will be retrieved and compared with the user queried item and if it is exactly same as what is represented in the knowledge base, it will be directly queried for the details of the entity or otherwise list of related named entities will be displayed for the user to select.

According to the selected entity name, other details of the entity will be queried using the traversal API and it will be fetched according to the requirement. Using this queried data, relevant modifications will be done in order to represent it in a human understandable manner. For example in our review application, cumulative scores should be calculated, comparisons statements should be build etc.

This component will contain:

1. Front-end web application.
   Where the user can interact and the summarized view of the reviews will be displayed. This will be implemented using Angular 2 which is a front-end web application framework.

2. Backend REST API.
   Where the services will be implemented and exposed as a API for the required functionalities. This will be implemented using Java as a programming language, Jena API tool and SPARQL for ontology querying.

These two will be hosted separately.