

**BANSILAL RAMNATH AGARWAL CHARITABLE TRUST'S
VISHWAKARMA INSTITUTE OF TECHNOLOGY
PUNE- 411 037**

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)



A Project Report
On
**“Ontology Building for Semantic Search Engine based on Phrase Similarity of Triplet
Extraction”**

Submitted By

- | | | |
|----|----------------|-----------------|
| 1. | Rohit Mujumdar | Gr. No. -131230 |
| 2. | Akanksha Patil | Gr. No. -131629 |
| 3. | Pranjal Patil | Gr. No. -131305 |
| 4. | Poshraj Sharma | Gr. No. -131396 |

Under guidance of
Prof. Dr. Manasi Patwardhan
Department of Computer Engineering,
Vishwakarma Institute of Technology, Pune

2016 -2017
**BANSILAL RAMNATH AGARWAL CHARITABLE TRUST'S
VISHWAKARMA INSTITUTE OF TECHNOLOGY
PUNE-411 037**

(An Autonomous Institute Affiliated to Savitribai Phule Pune University)



CERTIFICATE

Certified that this report for the project “**Ontology Building for Semantic Search Engine based on Phrase Similarity of Triplet Extraction**” is approved by me for submission. Certified further that, to the best of my knowledge, the report represents work carried out by the student as the Major Project as prescribed by the University of Pune in the academic year 2016-17.

Prof. Dr. Manasi Patwardhan
(Guide)

Department of Computer Engineering,
Vishwakarma Institute of Technology, Pune

ACKNOWLEDGEMENT

We would like to gratefully acknowledge the enthusiastic supervision of our project guide Prof. Dr. Manasi Patwardhan for her continuous, valuable guidance, patience, constant care, and kind encouragement throughout the project work that made us to present this project report of our major project in an efficient manner.

We would like to thank all our colleagues, friends, teaching and non-teaching staff members from Computer Engineering Department for creating a pleasant atmosphere during our work at VIT Pune.

Finally, we wish to thank our family members and our friends who have always been very supportive and encouraging.

ABSTRACT

The semantic search engine needs the user to enter a query keyword, with which data, text, images and videos, relevant and suitable to a primary school student is scraped from the internet. This involves web crawling and web scraping. The text data fetched from various sources is cleaned and processed to make it suitable for Triplet extraction. The phrases (relations) from the Triplets are compared for the similarity using Phrase2Vec model, and many similar relations are replaced by a common relation which semantically covers a majority of similar relations. This data is used to construct an ontology for the query word and is stored in an OWL-XML format. Another aim involves extending this ontology for further query searches and using the leaf nodes to perform 1- level depth offline scraping and build the ontology further. The final output involves displaying the definition text, images and videos related to the text and use the ontology to suggest similar keywords and fetch their results.

TABLE OF CONTENTS

Sr. No.	Topic	Pg. No.
1.	Introduction	1
2.	System Architecture	2
	a. System Diagram	3
	b. Use Case Diagram	3
	c. Sequence Diagram	4
	d. Class Diagram	5
	e. Component Diagram	6
	f. UI Screenshots	7
3.	Module Wise Implementation Methodology	12
	1. Base Ontology Module	12
	a. What is an Ontology?	12
	b. The OWL API	12
	c. Components	12
	i. Concept	13
	ii. Individuals	13
	iii. Relations	13
	iv. Annotations	14
	d. Scope of Our Ontology	14
	e. Structure of The Ontology	14
	f. Format of The Ontology	17
	g. Reading/Querying the Ontology	17
	2. Web Scraping Module	18
	3. Triple Extraction Module	19
	4. Relation Similarity Module	21
	a. Word2Vec	21
	b. Phrase2Vec	25
	5. Common Relation Determination Module	27
	6. Ontology Building	29
	a. Storing Triplets	29
	b. Storing Annotations	29
	7. Offline Scraping	30
4.	Accuracy/Limitations	31
	1. Web Scraping Module	31
	2. Triplet Extraction Module	31
	3. Relation Similarity Module	34
	4. Common Relation Determination Module	34
5.	Future Scope	35
	1. Web Scraping Module	35
	2. Triplet Extraction Module	35
	3. Relation Similarity Module	35
	4. Common Relation Determination Module	35
	5. Ontology Building Module and Base Ontology Module	35
	6. Attractive UI	35
6.	Conclusion	36

INTRODUCTION

Normal search engines give us links of the webpages that contain the query searched using the said search engines. However, there aren't many *semantic* search engines in place. A good example of the same is Wolfram Alpha.

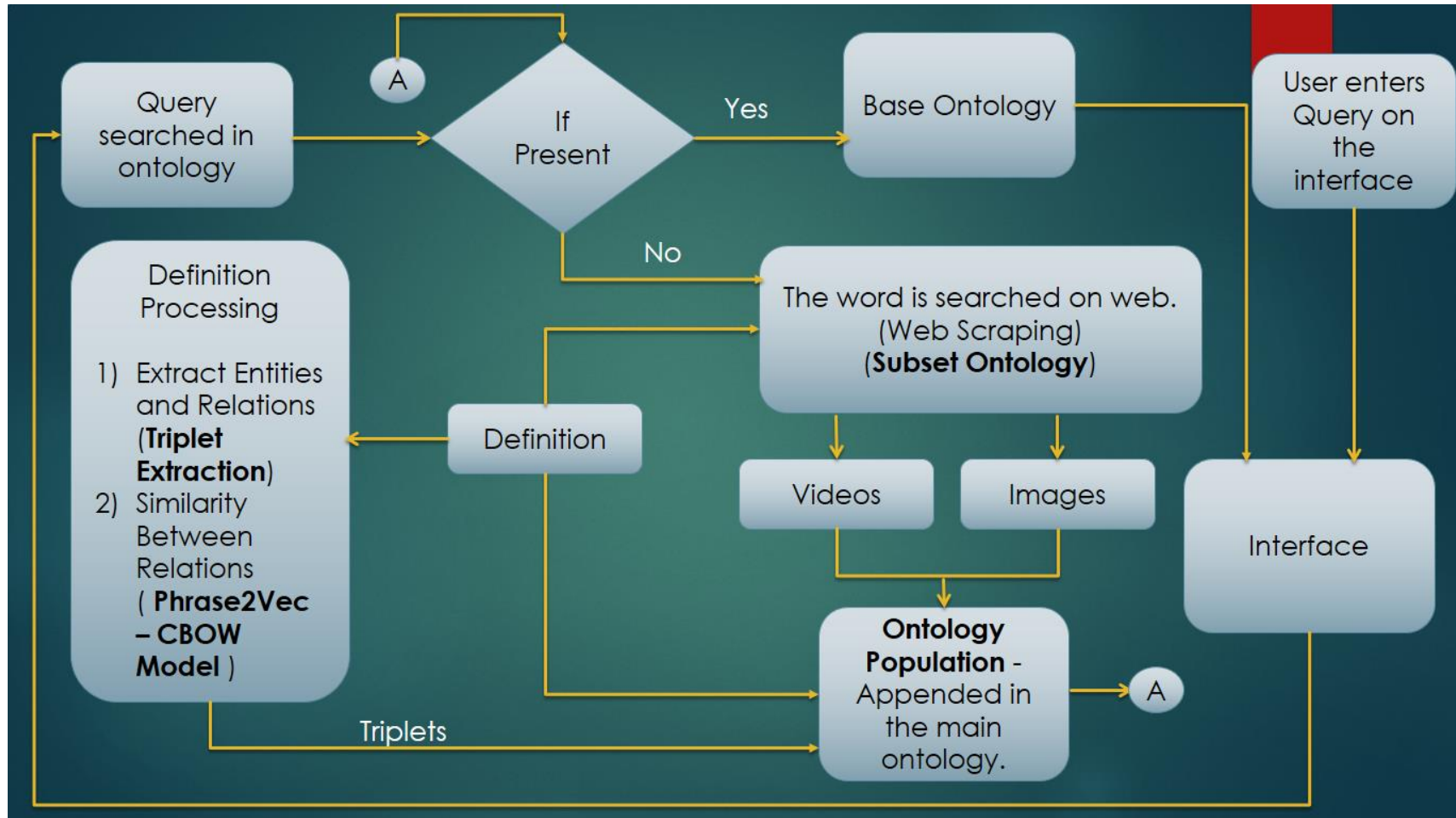
Semantic search is a data searching technique in which a search query aims to not only find keywords, but to determine the intent and contextual meaning of the words a person is using for search. Intent, which comes from the user, explicitly states what he or she is looking for. And context could be understood as everything that surrounds a search and makes this go in either direction, i.e., what gives it meaning. Thus, by understanding and connecting intention and context, search engines are able to understand the different queries, both what motivates and what is expected of them.

This project aims at initiating the building such a semantic search engine that would cater to primary school students. The scope of this engine would majorly involve general science upto grade 5.

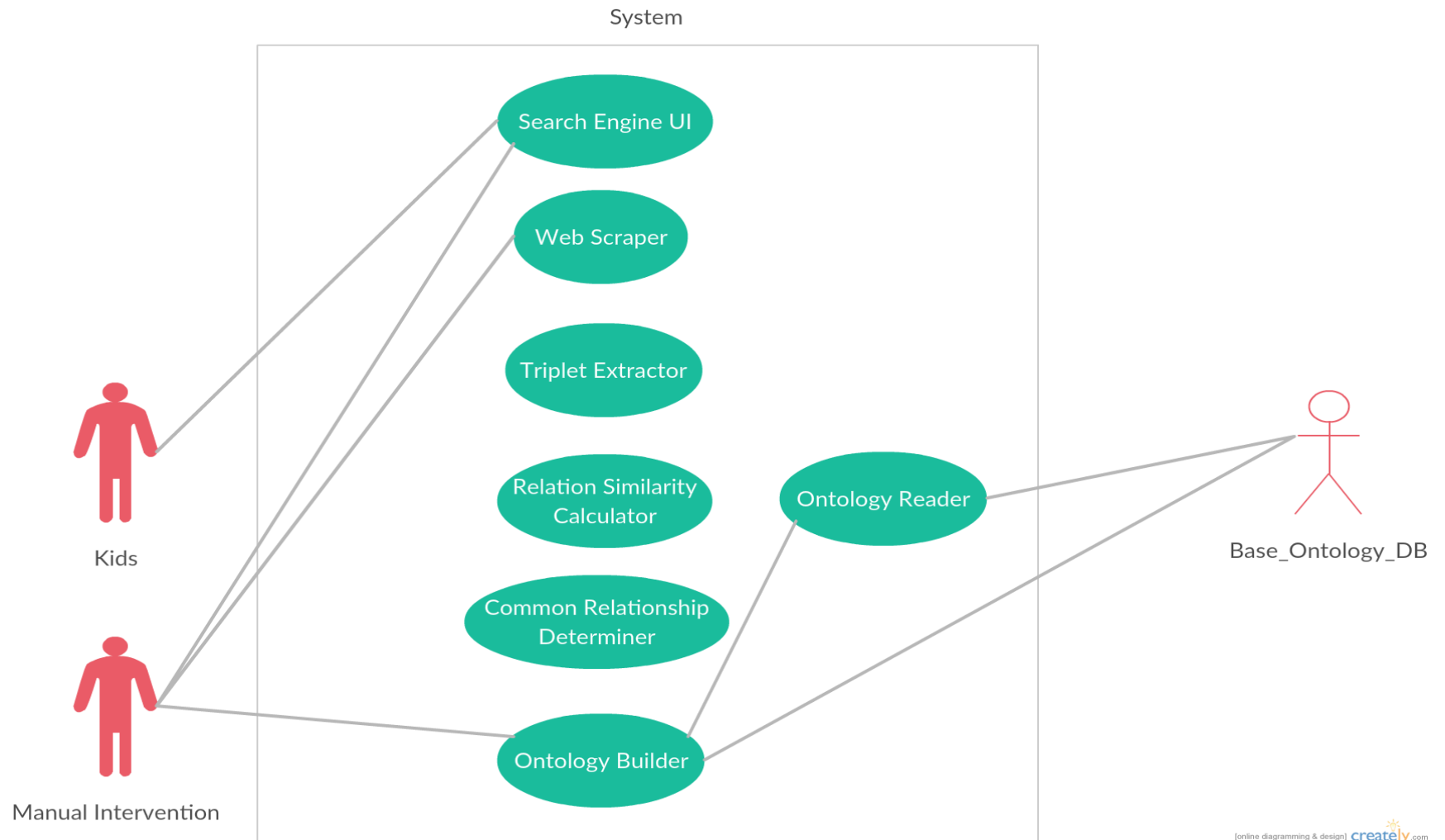
Our semantic search engine relies heavily on the use of ontologies which would form a dynamic knowledge base of semantic search. Newly fetched content would keep getting added to this ontology based on similarity of the content fetched.

SYSTEM ARCHITECTURE

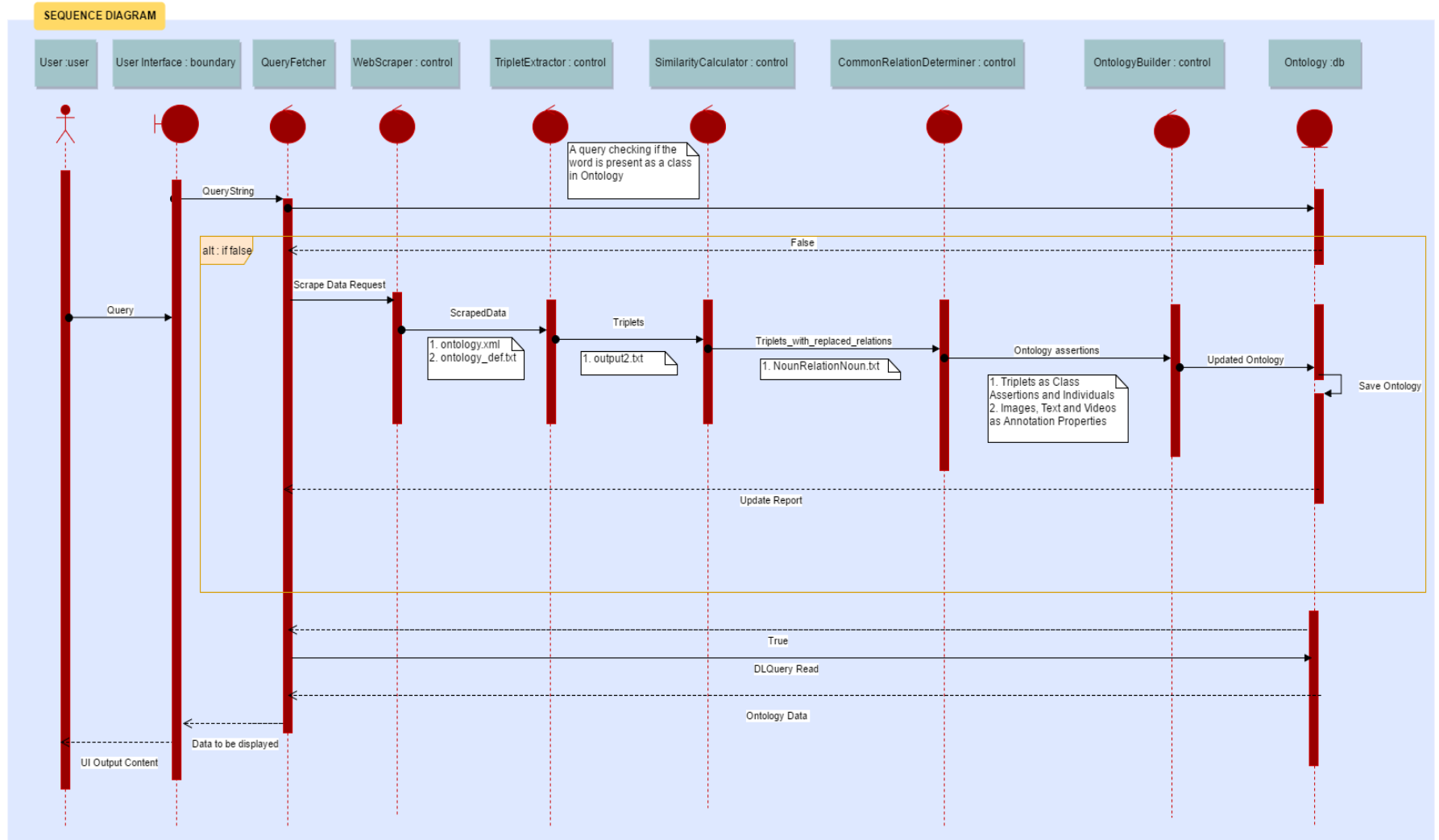
a) SYSTEM DIAGRAM



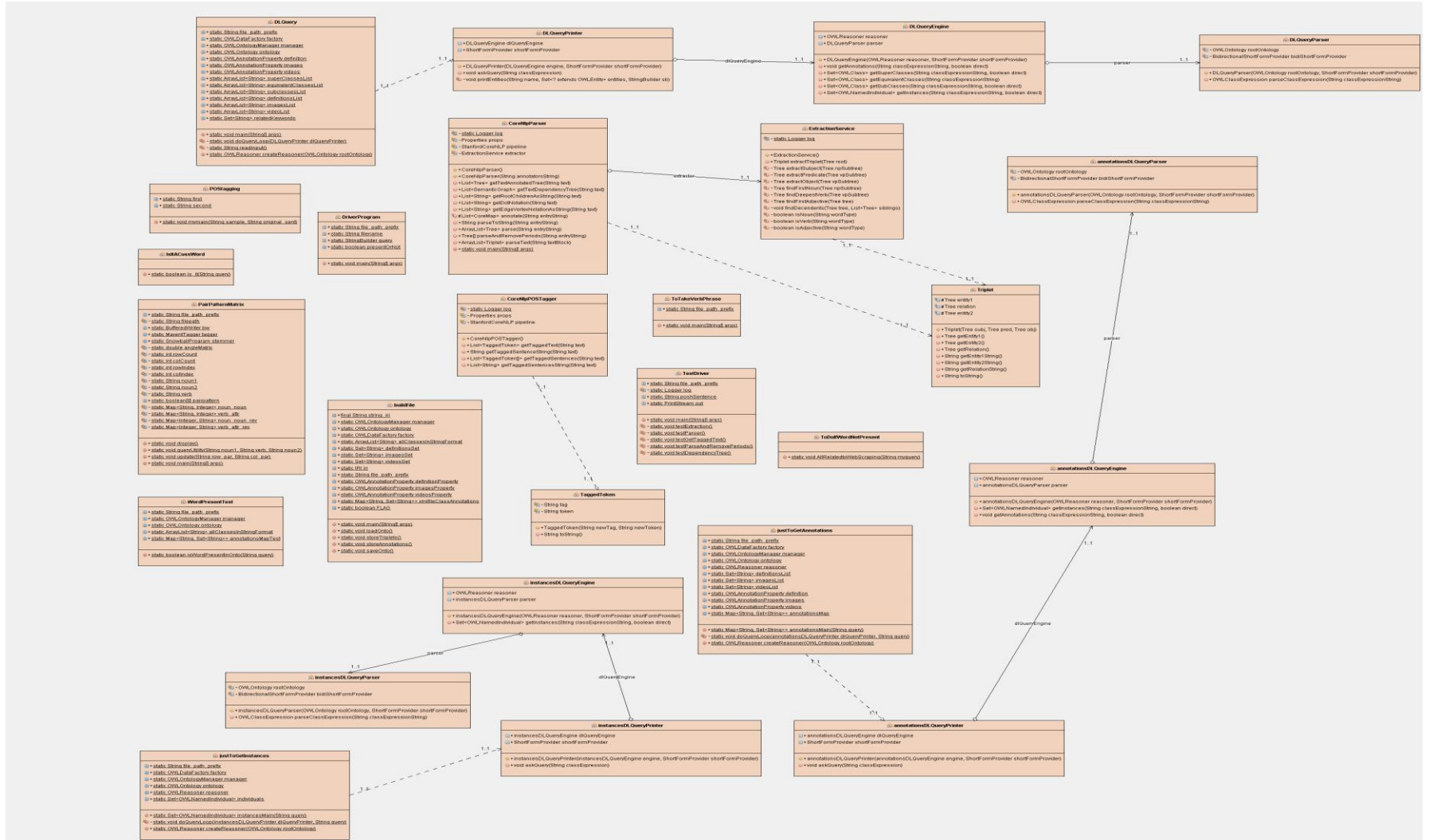
b) USE CASE DIAGRAM



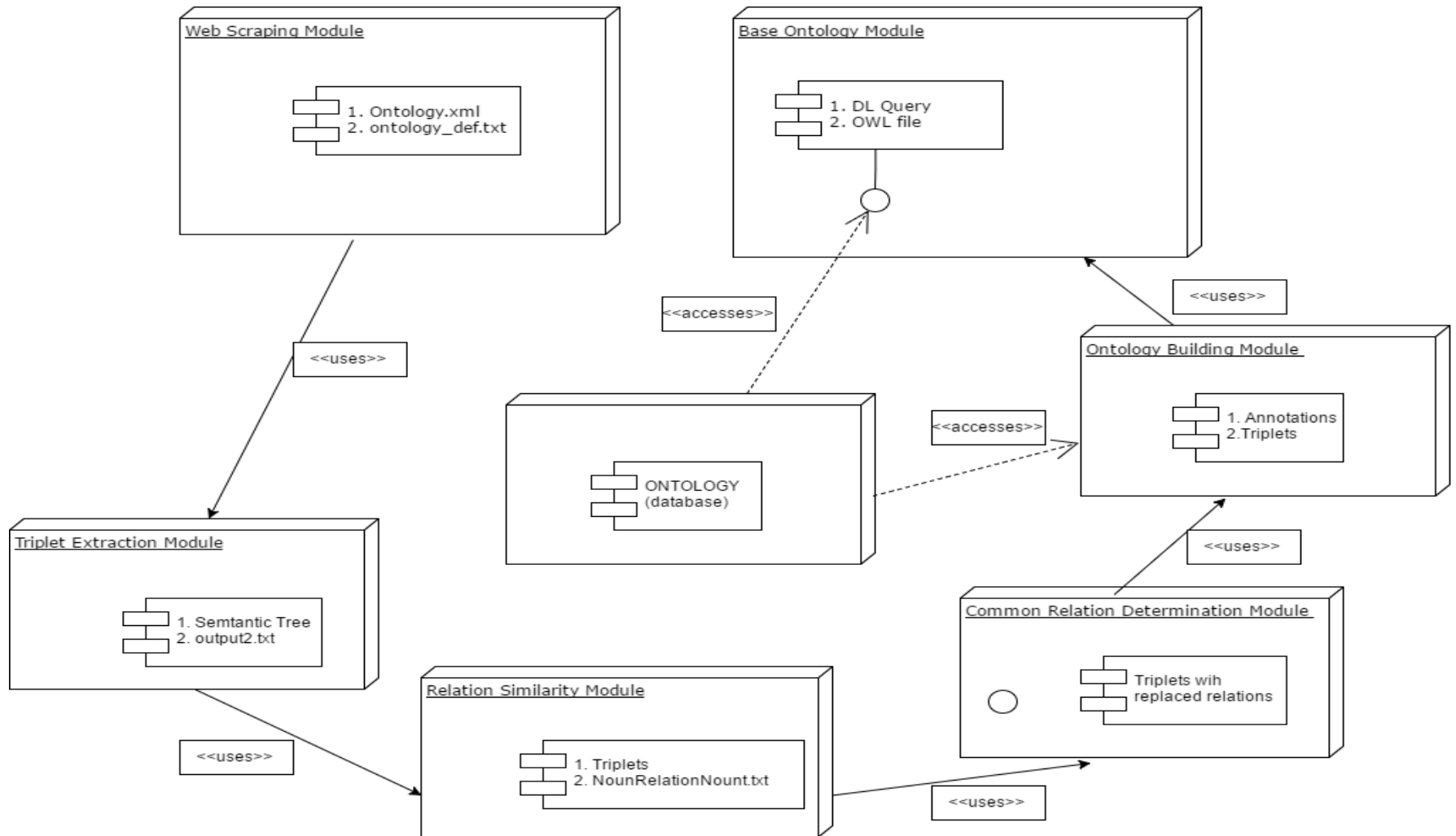
c) SEQUENCE DIAGRAM



d) CLASS DIAGRAM

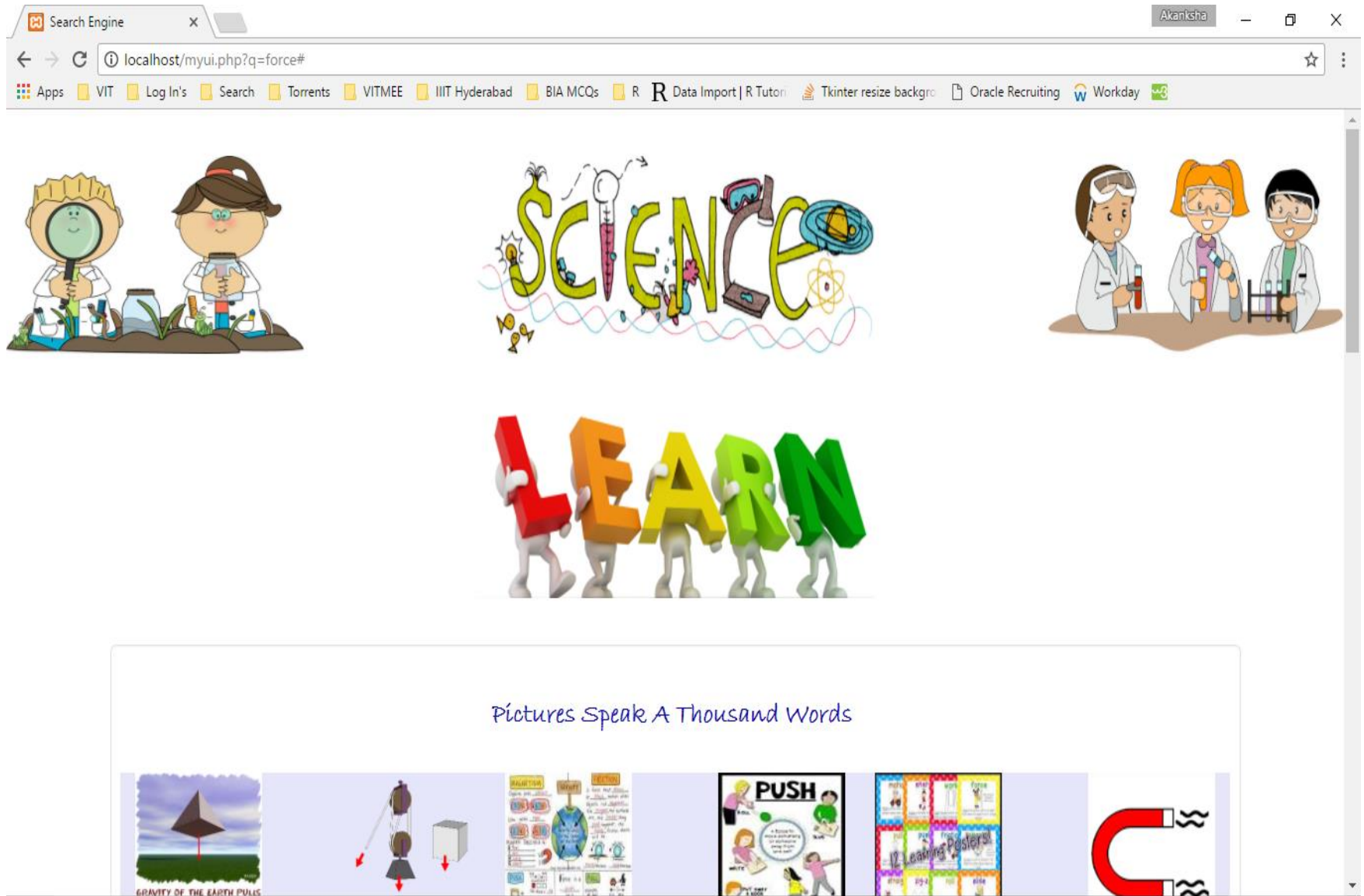


e) COMPONENT DIAGRAM



f) UI SCREENSHOTS





Search Engine x Akanksha

localhost/myui.php?q=force#

Apps VIT Log In's Search Torrents VITMEE IIT Hyderabad BIA MCQs R R Data Import | R Tutori Tkinter resize backgro Oracle Recruiting Workday

Pictures Speak A Thousand Words

The collage consists of 12 small images arranged in two rows of six. The top row includes: 1) A diagram of a sailboat on water with the text 'GRAVITY OF THE EARTH PULSES OBJECTS TOWARDS THE CENTER OF THE PLANET'. 2) A diagram of a pulley system with a weight and a spring scale. 3) A diagram of a magnet with a coil of wire and a battery, labeled 'MAGNETIC FORCE'. 4) A diagram of a person pushing a box with the word 'PUSH'. 5) A grid of colorful cards with various physics terms. 6) A diagram of a magnet with a coil of wire and a battery, labeled 'MAGNETIC FORCE'. The bottom row includes: 1) A diagram of a person pushing a box with the text 'Force, Work and Energy'. 2) A diagram of a pulley system with a weight and a spring scale. 3) A diagram of a magnet with a coil of wire and a battery, labeled 'MAGNETIC FORCE'. 4) A diagram of a person pushing a box with the word 'PUSH'. 5) A grid of colorful cards with various physics terms. 6) A diagram of a magnet with a coil of wire and a battery, labeled 'MAGNETIC FORCE'.

LEARN

Search Engine

localhost/myui.php?q=force#


Apps VIT Log In's Search Torrents VITMEE IIIT Hyderabad BIA MCQs R R Data Import | R Tutor Tkinter resize backgro Oracle Recruiting Workday

FORCE

What is force? In physics force is a push or pull on an object. A force can cause an object to accelerate slow down remain in place or change shape. How to Measure Force? The unit of measure for force is the newton which is abbreviated as "N". One newton is the force needed to accelerate one gram of mass by one centimeter per second squared. Other units of force include the dyne and the pound-force. Force Mass and Acceleration. Force can be figured out if you know the mass and acceleration of an object. This equation comes from Newton's Second Law of Motion: $f = m * a$ Where f = force m = mass and a = acceleration. Forces and Vectors. Force not only has a magnitude (which is what we get in newtons when we use the equation above) but it also has a direction. This makes force a vector. Vectors are shown by an arrow that indicates the direction of the force and a number that indicates the magnitude.

Force, Work and Energy for Kids

Force, Work and Energy




Search Engine x Akanksha

localhost/myui.php?q=force#

Apps VIT Log In's Search Torrents VITMEE IIIT Hyderabad BIA MCQs R R Data Import | R Tutor Tkinter resize backgro Oracle Recruiting Workday




WORK and Energy



Check More Videos Here:

<https://www.youtube.com/watch?v=u0Ko3DbfYZk>

Hey, why don't you check out these as well?



inertia gravity work mass velocity laws of motion friction

MODULE WISE IMPLEMENTATION METHODOLOGY

1. BASE ONTOLOGY MODULE

a) WHAT IS AN ONTOLOGY?

In computer science and information science, an ontology is a formal naming and definition of the types, properties, and interrelationships of the entities that really or fundamentally exist for a particular domain of discourse. It is thus a practical application of philosophical ontology, with a taxonomy.

An ontology defines (specifies) the concepts, relationships, and other distinctions that are relevant for modelling a domain.

Ontology is the technique of defining names, attributes and relationships between items of a particular domain. Domain ontology is the term used to describe the rules and constraints in a particular field and helps to find the associations between terms. Thus ontology can effectively be used to assign meanings to words in the semantic web.

b) THE OWL API

The OWL API is a Java API and reference implementation for creating, manipulating and serializing OWL Ontologies. The latest version of the API is focused towards OWL 2. The OWL API is open source and is available under either the LGPL or Apache Licenses.

The OWL API includes the following components:

- An API for OWL 2 and an efficient in-memory reference implementation
- RDF/XML parser and writer
- OWL/XML parser and writer
- OWL Functional Syntax parser and writer
- Turtle parser and writer
- KRSS parser
- OBO Flat file format parser
- Reasoner interfaces for working with reasoners such as FaCT++, HermiT, Pellet and Racer

c) COMPONENTS

An Ontology consists of a number of different components. The names of these components differ between ontologies depending on the ontology language used, philosophical persuasion or background of the authors. Despite this, their core components are largely shared between different ontologies. These components can be separated into two kinds: those that describe the Entities of the domain — here called concepts, individuals and relationships; and those which either enable the use of the ontology or describe the ontology itself.

i) CONCEPT

Concepts, also called Classes, Types or Universals are a core component of most ontologies. A Concept represents a group of different Individuals, that share common characteristics, which may be more or less specific.

For example, (most) humans share certain characteristics, such as related DNA, a set of specific body parts, the ability to speak a complex language. Likewise, all mammals share these characteristics, except for the ability to speak.

Most ontology languages allow the author to define Concepts on the basis of these characteristics; additionally, some languages, such as OWL also allow definition of Concepts extensionally by their membership. For example, the Concept “members of the beatles” could be defined as the set of “John, Paul, George and Ringo”.

One Concept may be a subconcept (also known as subclass, or kind of) another Concept; this means that if the Concept C' is a subconcept of C, then any individual of type C' will also be an individual of type C. It is possible within an ontology to explicitly state that C' is a subconcept of C; in some languages, including OWL it is also possible to infer this.

Concepts may also share relationships with each other; these describe the way individuals of one Concept relate to the individuals of another.

ii) INDIVIDUAL

Individuals also known as instances or particulars are the base unit of an ontology; they are the things that the ontology describes or potentially could describe. Individuals may model concrete objects such people, machines or proteins; they may also model more abstract objects such as this article, a person's job or a function.

Individuals are a formal part of an ontology and are one way of describing the entities of interest. Perhaps more common within bioinformatics is the development of ontologies consisting only of Concepts which are then used to annotate data records directly.

```
<owlx:Individual owlx:name="CentralCoastRegion">
  <owlx:type owlx:name="Region" />
</owlx:Individual>
```

iii) RELATION

Relations in an ontology describe the way in which individuals relate to each other. Relations can normally be expressed directly between individuals (this article has author Phillip Lord) or between

Concepts (an article has author a person); in the latter case, this describes a relationship between all individuals of the Concepts.

Although it is dependent on the ontology language, it is often possible to express different categories of relationships between Concepts. Consider, for example, “person has father person”. This is an existentially quantified relationship; it is the case that every person has a father, and

that this individual is also a person. This can be contrasted from “person is father of person”; this is a universal quantified relationship. It is true that every individual which is father of a person is, themselves, a person; however, it would be wrong to assert that every person is the father of another.

iv) ANNOTATIONS

Annotations are the meta data for an OWL object. They store extra information about that object.

d) SCOPE OF OUR ONTOLOGY

This ontology is specifically made for students from 3rd, 4th and 5th standards for science domain. All the science topics are included in this ontology which makes it easier for the students to learn things.

e) STRUCTURE OF THE ONTOLOGY

All the classes or their subclasses are modeled from the queries that are to be searched or have been searched. If two classes are related they have a common individual which is named after the relation between the two classes. This is a departure from the conventional structure where two individuals are connected by an Object Property and its data is stored in the Data Property. If modelled as a graph this new structure reduces an edge and a node, thus reducing the walk and size of the graph. The data for each class (word in ontology) is stored in the class annotations with the following annotations – definition, images and videos.

Force (http://www.semanticweb.org/hp/ontologies/2016/10/Force) : [D:\pranjal\posh_onto.owl]

File Edit View Reasoner Tools Refactor Window Help

Force (http://www.semanticweb.org/hp/ontologies/2016/10/Force) Search for entity

Active Ontology Entities Classes Object Properties Data Properties Annotation Properties Individuals OWLViz DL Query OntoGraf Ontology Differences SPARQL Query

Class hierarchy: force

- Thing
 - acceleration
 - force
 - friction
 - gravity
 - inertia
 - kinetic_energy
 - laws_of_motion
 - mass
 - momentum
 - motion
 - potential_energy
 - velocity
 - work

OntoGraf:

Search: contains Search Clear

gravity

URI: <http://www.semanticweb.org/hp/ontologies/2016/10/Force#gravity>

Annotations:

videos "<https://www.youtube.com/watch?v=ijRIB6TuMOU>;"

images

"http://www.ducksters.com/science/gravity_solar_system.jpg;<https://s-media-cache-ak0.pinimg.com/736x/77/82/62/778262cc5db99fec9e3c82a78b1941a8.jpg>;"

definition "Gravity is the mysterious force that makes everything fall down towards the Earth. But what is it? It turns out that all objects have gravity. It's just that some objects, like the Earth and the Sun, have a lo..."

No Reasoner set. Select a reasoner from the Reasoner menu Show Inferences

8:44 PM 5/15/2017

Force (http://www.semanticweb.org/hp/ontologies/2016/10/Force) : [D:\pranjal\posh_onto.owl]

File Edit View Reasoner Tools Refactor Window Help

Force (http://www.semanticweb.org/hp/ontologies/2016/10/Force) Search for entity

Active Ontology Entities Classes Object Properties Data Properties Annotation Properties Individuals OWLViz DL Query OntoGraf Ontology Differences SPARQL Query

Class hierarchy: force

- Thing
 - acceleration
 - force
 - friction
 - gravity
 - inertia
 - kinetic_energy
 - laws_of_motion
 - mass
 - momentum
 - motion
 - potential_energy
 - velocity
 - work

OntoGraf:

Search: contains Search Clear

force

URI: http://www.semanticweb.org/hp/ontologies/2016/10/Force#force

Annotations:

definition "What is force?
In physics, force is a push or pull on an object.
A force can cause an object to accelerate, slow down, remain in place, or change shape.
How to Measure Force
The unit of measure for force is the ne...
images
"http://www.physics4kids.com/files/art/motion_gravity1_240.jpg;https://s-media-cache-ak0.pinimg.com/originals/97/9f/a8/979fa8e6d1320b6aa16b85ba31f659d...
videos "https://www.youtube.com/watch?v=u0Ko3DbfYZk;"

Thing

gravity

No Reasoner set. Select a reasoner from the Reasoner menu ☒ Show Inferences

8:36 PM 5/15/2017

f) FORMAT OF THE ONTOLOGY

Ontology is stored in OWL-XML format.

g) READING/QUERYING THE ONTOLOGY

OWL APIs have been used to query the ontology. The query is passed to the reader module which it reads as an OWL class. The data is fetched from the annotations stored for that class.

The classes which are related to this queried class in some way are shown as related keywords for this queried class. The related keywords can themselves act as links for further searches. Thus 1-level depth is used to hunt for related keywords.

2. WEB SCRAPING MODULE

This module comes into picture when the word is not found in the ontology. The word is passed as a query to google custom search for ducksters.com using jsoup library of java. Similarly, we extend the search to the following 7 websites.

- <http://www.ducksters.com/>
- <http://www.physics4kids.com/>
- <http://www.biology4kids.com/>
- <http://www.geography4kids.com/>
- <http://www.chem4kids.com/>
- <http://www.cosmos4kids.com/>
- <https://www.wikipedia.org/>

These resulting links are parsed based on whether the link contains the query or not. If such a link is found, we will parse the text and images present on that webpage using webClient class of htmlUnit. A similar process is repeated for other websites and word preprocessing (replacing spaces by underscores, converting uppercase text to lowercase) is done for each of them.

We create two output files –

Ontology.xml: Contains a semi structured documentation of the data obtained from the webpages. The file has the following format –

```
<?xml version="1.0" encoding="iso-8859-1"?>
<subset_ontology>
<keyword category="class">
<title lang="en">mass
</title>
<text>
...
</text>
<img>
...
</img>
<img>
..
</img>
<vdo>
..
</vdo>
</subset_ontology>
```

This file is parsed to save the text and image-video data obtained as annotations in our ontology.

Ontology.def.txt: This file contains the text extracted (henceforth called as definition) which is used for further parsing and extraction of entity-relation-entity triplet.

3. TRIPLET EXTRACTION MODULE

The paragraph of text is separated sentence wise and noun-phrases which would later serve as the two entities bound by a relationship.

The noun phrases which would later serve as two classes (entities) connected by a relationship and the verb phrase which would later serve as relationship is extracted by Triplet Extraction - Semantic Tree.

For example, for the sentence,

“Potential energy is converted into kinetic energy.”, the semantic tree looks like this -

```
(ROOT
(S
(NP (JJ Potential) (NN energy))
(VP (VBZ is)
(VP (VBN converted)
(PP (IN into)
(NP (JJ kinetic) (NN energy))))))
(. )))
```

We start recording the verb phrase from:

1. the beginning of the first verb phrase token (VP) to
2. the beginning of the last noun phrase token (NP).

The extracted verb phrase is stripped of the NLP tokens and the extracted relationship would be ‘is converted to’.

We extract the noun phrases as follows:

1. first Entity:
 - a. the beginning of first noun phrase token (NP)
 - b. the beginning of the first verb phrase token (VP)
2. Second entity
 - a. The beginning of the first noun phrase token (NP) encountered after the extraction of the Relation
 - b. till the end

The extracted entities would be

1. Potential Energy
2. Kinetic Energy

Consider the following paragraph, say it is extracted from the web.

“Coal is required to generate energy. This energy is called as thermal energy. Heat is needed for thermal energy generation. Heat is caused by burning coal. Thermal energy is known as internal kinetic energy. The random movement of molecules produces the internal kinetic energy. Thermal energy is converted to electrical energy. Thus, thermal energy is used for generation of electricity. ”

The triplets formed for this example would be:

1. *is called as;This energy;thermal energy*
2. *is needed for;Heat;thermal energy generation*
3. *is caused by;Heat;burning coal*
4. *is known as;Thermal energy;internal kinetic energy*
5. *is used for;thermal energy;generation of electricity*

All such sentences are transformed into triplets and they are sent to the Pair Pattern Module.

4. RELATION SIMILARITY MODULE

We begin by constructing a symmetric matrix of all the relations obtained from Triplet Extraction. We use the Phrase2Vec Model to find out the similarity between them. The measure of similarity is cosine similarity.

We first understand the Word2Vec model before understanding the implementation of Phrase2Vec model.

a) Word2vec

Word2vec is a group of related models that are used to produce word embeddings. These models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space.

Word2Vec is *not* a single monolithic algorithm. Word2Vec contains two distinct models (CBOW and skip-gram), each with two different training methods (with/without negative sampling). Word2vec is *not* deep learning; both CBOW and skip-gram are "shallow" neural models.

Both architectures describe how the neural network "learns" the underlying word representations for each word. Since learning word representations is essentially unsupervised, you need some way to "create" labels to train the model.

Skip-gram and CBOW are two ways of creating the "task" for the neural network -- you can think of this as the output layer of the neural network, where we create "labels" for the given input (which depends on the architecture).

For both descriptions below, we assume that the current word in a sentence is w_i .

- **CBOW:** The input to the model could be w_{i-2} , w_{i-1} , w_{i+1} , w_{i+2} , the preceding and following words of the current word we are at. The output of the neural network will be w_i . Hence you can think of the task as "*predicting the word given its context*". Note that the number of words we use depends on your setting for the window size.
- **Skip-gram:** The input to the model is w_i , and the output could be w_{i-1} , w_{i-2} , w_{i+1} , w_{i+2} . So, the task here is "*predicting the context given a word*".

Example,

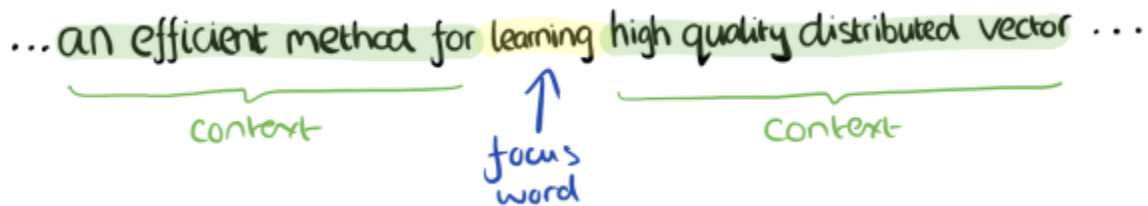
The sentence "Hi fred how was the pizza?" becomes:
Continuous bag of words: 3-grams {"Hi fred how", "fred how was", "how was the", ...}
Skip-gram 1-skip 3-grams: {"Hi fred how", "Hi fred was", "fred how was", "fred how the", ...}

According to Mikolov, Skip-gram works well with small amount of the training data, represents well even rare words or phrases whereas CBOW is several times faster to train than the skip-gram, slightly better accuracy for the frequent words.

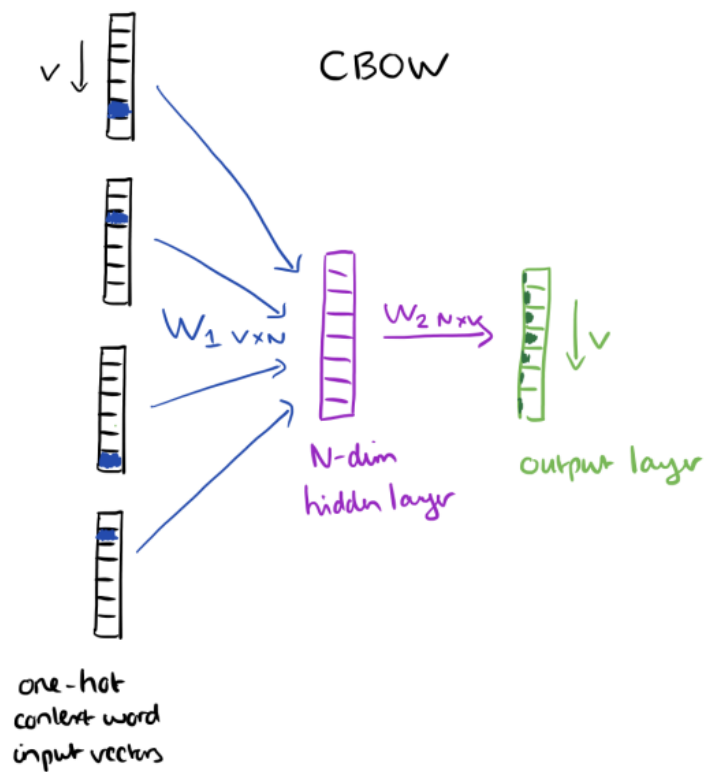
Let's look at the continuous bag-of-words (CBOW) model first.

Consider a piece of prose such as “The recently introduced continuous Skip-gram model is an efficient method for learning high-quality distributed vector representations that capture a large number of precise syntactic and semantic word relationships.”

Imagine a sliding window over the text, that includes the central word currently in focus, together with the four words and precede it, and the four words that follow it:



The context words form the input layer. Each word is encoded in one-hot form, so if the vocabulary size is V these will be V -dimensional vectors with just one of the elements set to one, and the rest all zeros. There is a single hidden layer and an output layer.



The training objective is to maximize the conditional probability of observing the actual output word (the focus word) given the input context words, with regard to the weights. In our example, given the input (“an”, “efficient”, “method”, “for”, “high”, “quality”, “distributed”, “vector”) we want to maximize the probability of getting “learning” as the output.

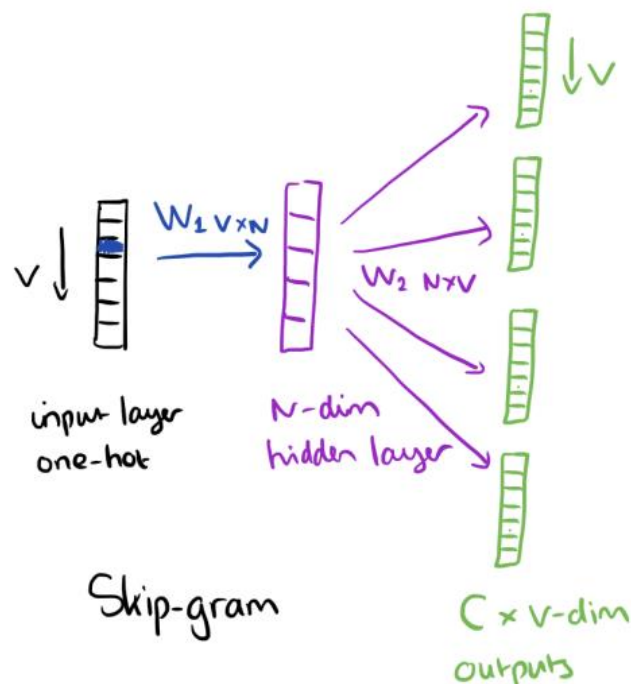
Since our input vectors are one-hot, multiplying an input vector by the weight matrix \mathbf{W}_1 amounts to simply selecting a row from \mathbf{W}_1 .

$$\begin{array}{c} \text{input} \\ 1 \times V \end{array} \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \begin{array}{c} \mathbf{W}_1 \\ V \times N \end{array} \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \end{bmatrix} = \begin{array}{c} \text{hidden} \\ 1 \times N \end{array} \begin{bmatrix} e & f & g & h \end{bmatrix}$$

\mathbf{W}_1

Given C input word vectors, the activation function for the hidden layer \mathbf{h} amounts to simply summing the corresponding ‘hot’ rows in \mathbf{W}_1 , and dividing by C to take their average.

From the hidden layer to the output layer, the second weight matrix \mathbf{W}_2 can be used to compute a score for each word in the vocabulary. The **skip-gram** model is the opposite of the CBOW model. It is constructed with the focus word as the single input vector, and the target context words are now at the output layer:



The activation function for the hidden layer simply amounts to copying the corresponding row from the weights matrix \mathbf{W}_1 (linear) as we saw before. At the output layer, we now output C multinomial distributions instead of just one. The training objective is to minimize the summed prediction error across all context words in the output layer. In our example, the input would be “learning”, and we hope to see (“an”, “efficient”, “method”, “for”, “high”, “quality”, “distributed”, “vector”) at the output layer.

Thus, the output of the Word2vec neural net is a vocabulary in which each item has a vector attached to it, which can be fed into a deep-learning net or simply queried to detect relationships between words.

Our Word2vec model has used pre-trained **Google News** corpus. It includes word vectors for a vocabulary of 3 million words and phrases that they trained on roughly 100 billion words from a Google News dataset. The vector length is 300 features.

b) Phrase2Vec

This is an extension of Word2Vec. We use the Python module, gensim which is designed to automatically extract semantic topics from documents, as efficiently (computer-wise) and painlessly (human-wise) as possible. Gensim is designed to process raw, unstructured digital texts (“*plain text*”). Gensim uses the Google corpus as its training corpus.

A phrase is made up of words a, b, c, d so the phrase would be “a b c d”. Vector of size 300 is fetched for each of the words in the phrase. So, we have an array of arrays that needs to be condensed into a single array (vector) of the same feature size (dimensionality). This is the phrase vector.

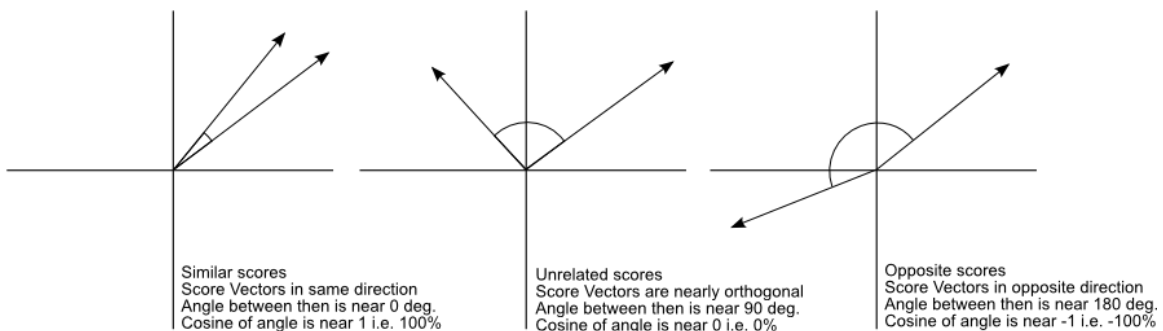
In doing this, the vectors within the set that need to be ignored are taken into consideration. In our case these words are the stop words. The centroid of these vectors is calculated and the phrase vector is obtained. A centroid is the appropriate measure because it incorporates equal contribution of each word that went into forming the semantic meaning of the phrase.

The measure of similarity used is Vector Space Model, i.e. cosine similarity.

The cosine similarity between two vectors (or two documents on the Vector Space) is a measure that calculates the cosine of the angle between them. This metric is a measurement of orientation and not magnitude, it can be seen as a comparison between phrases on a normalized space because we’re the angle between the phrase. What we have to do to build the cosine similarity equation is to solve the equation of the dot product for

$$\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| \cos \theta$$

$$\cos \theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\| \|\vec{b}\|}$$



So, if the similarity between two vectors is closer to one, they are more similar.

Consider the following relations; this is the similarity matrix we have for them.

	is_required_for	is_called_as	is_needed_for	is_caused_by	is_known_as	produces	is_converted_to	used_for
is_required_for	1	0.156399	0.604876	0.183366	0.136432	0.089468	0.138075	0.346722
is_called_as	0.156399	1	0.222255	0.19645	0.369789	0.148613	0.108163	0.312275
is_needed_for	0.604876	0.222255	1	0.15927	0.071422	0.064939	0.129334	0.404929
is_caused_by	0.183366	0.19645	0.15927	1	0.149263	0.218184	0.082721	0.219328
is_known_as	0.136432	0.369789	0.071422	0.149263	1	0.194264	0.126769	0.36233
produces	0.089468	0.148613	0.064939	0.218184	0.194264	1	0.16634	0.213744
is_converted_to	0.138075	0.108163	0.129334	0.082721	0.126769	0.16634	1	0.33524
used_for	0.346722	0.312275	0.404929	0.219328	0.36233	0.213744	0.33524	1

5. COMMON RELATION DETERMINATION MODULE

The average (mean) of each row and thus effectively, each relation's similarity with the others, is calculated. Higher is the mean, higher are the chances of this relation being a representative of other relations.

Thus, the relations above a certain threshold can be considered a base relation and cover the others. Now, all the relations are sorted in descending order of the averages calculated above. For example,

7 "used for"	0.39932101219892502
2 "is needed for"	0.33212820440530777
0 "is required for"	0.33191726543009281
1 "is called as"	0.31424299720674753
4 "is known as"	0.30128367990255356
3 "is caused by"	0.27607280015945435
5 "produces"	0.26194384321570396
6 "is converted to"	0.26083023380488157

We now map the similarities in this descending order

7: (2, 0.40492937), (4, 0.36233044), (0, 0.34672153), (6, 0.33524007), (1, 0.31227478),
(3, 0.21932793), (5, 0.21374387)
2: (0, 0.6048761), (7, 0.40492937), (1, 0.22225514)
0: (2, 0.6048761), (7, 0.34672153)
1: (4, 0.36978871),
4: (1, 0.36978871), (7, 0.36233044)
3: (7, 0.21932793), (5, 0.21818361)
5: (3, 0.21818361), (7, 0.21374387)
6: (7, 0.33524007)

The above map will be processed using following algorithm.

Algorithm:

```
For each Row in above Map, {
    For each element in Row, {
        If (element is not replaced by any base relation) {
            Replace the element by base relation}
        Else {
            If (similarity of previous base relation is less than current
            relation and current base relation is not visited) {
                Replace the element by base relation}
            }
        }
    }
```

All the other relations which can be covered by the current base relation, they get replaced with the base relation for which we have empirically decided the threshold value of similarity, here, say 0.34

Thus,

Relation No.	Relation Name	Replaced?	Replaced by Relation No.	Final Relation Decided
0	is_required_for	Y	2	is_needed_for
1	is_called_as	N	7	used_for
2	is_needed_for	Y	7	used_for
3	is_caused_by	N	7	used_for
4	is_known_as	Y	1	is_called_as
5	Produces	N	3	is_caused_by
6	is_converted_to	N	7	used_for
7	used_for	N	7	used_for

A new file with the updated triplets is generated and is sent to the next module.

6. ONTOLOGY BUILDING

Using OWL API is used to read and modify the OWL/XML file and thus effectively he ontology.

Following rules are used for building the ontology

For every N1-R-N2 line fetched from the file obtained from the previous module

Where

N1 = entity 1 in the triplet

N2 = entity 2 in the triplet

R = Relationship between N1 and N2

There are two parts:

a. Storing Triplets

If N1 is already present as a class in ontology:

If N2 is already present as a class in ontology:

If R is a relation already present between N1 and N2:

Do nothing;

Else:

Create new relationship between them

Else if N2 not an existing class:

Create new class N2 (subclass of Thing)

If R is a relation already present between N1 and N2:

Do nothing;

Else:

Create new relationship between them;

b. Storing Annotations

Since hitherto non-existing classes have already been created in the previous steps, we just need to check if the scraped annotations are not being repeated.

The ontology.xml file obtained in the Web Scraping module is parsed and scraped data is obtained. Only new annotations are added as Annotation properties to the classes N1 and N2. The three Annotation Properties are definitionProperty, imagesProperty, videosProperty.

7. OFFLINE SCRAPING

A proposed offline module would hunt for:

- a. Those nodes/classes of the ontology for which annotations are do not exist.
- b. Arbitrary nodes/classes which may not have many relations.

This would enable the ontology to keep building/extending itself.

ACCURACY/LIMITATIONS

1. WEB SCRAPING MODULE

- a. Since web crawling using Apache Nutch was unsuccessful, data extraction from the web has been limited to Web Scraping from some selected websites.
- b. Each website has its own DOM structure, webpage parsing isn't generalized. Further, a different preprocessing method needs to be applied to data extracted from every website.
- c. There are not many websites that cater to kids specifically. And not all websites may contain data for every possible keyword entered.

2. TRIPLET EXTRACTION MODULE

Verb Phrase extraction does not give good results for complicated sentences.

Examples:

- a) Simple Sentences (with only one finite verb) results are the best since one finite verb is used and the verb phrase contains that finite verb.

- i. *Potential Energy is converted to Kinetic Energy.*

(ROOT
(S
(NP (NNP Potential) (NNP Energy))
(VP (VBZ is)
(VP (VBN converted)
(PP (TO to)
(NP (NNP Kinetic) (NNP Energy))))))
(.)))

So here the verb phrase is: *is converted to*

- ii. *Rotation of turbines causes electricity generation.*

(ROOT
(S
(NP
(NP (NNP Rotation))
(PP (IN of)
(NP (NNS turbines))))
(VP (VBZ causes)
(NP (NN electricity) (NN generation)))
(.)))

So here the verb phrase is: *causes*

- b) Simple Sentences (other than ones above) pose problems as there can be verbs in the infinitive form and the relationship might get lost because of the rule

i. *As pollution grows, ways to combat it has grown too.*

```
(ROOT
(S
(SBAR (IN As)
(S
(NP (NN pollution))
(VP (VBZ grows))))
(, .)
(NP (NNS ways)
(S
(VP (TO to)
(VP (VB combat)
(NP (PRP it))))))
(VP (VBZ has)
(VP (VBN grown)
(ADVP (RB too))))
(. )))
```

So here the verb phrase is not extracted and the relationship is lost.

ii. *Soil is loose material which lies on top of the land.*

```
(ROOT
(S
(NP (NNP Soil))
(VP (VBZ is)
(NP
(NP (JJ loose) (NN material))
(SBAR
(WHNP (WDT which))
(S
(VP (VBZ lies)
(PP (IN on)
(NP
(NP (NN top))
(PP (IN of)
(NP (DT the) (NN land))))))))))
(. )))
```

So here the verb phrase is not extracted and the relationship is lost.

Whereas, the Triplet could have been soil-lies-on –the-top-of-the-land

- c) Compound sentences contain two verb phrases and potentially two triplets, but we always lose out on one, and even then, the Triplet may not be what we wanted.

i. *Acids are said to be acidic and bases are said to be basic.*

```
(ROOT
  (S
    (S
      (NP (NNS Acids))
      (VP (VBP are)
        (VP (VBN said)
          (S
            (VP (TO to)
              (VP (VB be)
                (ADJP (JJ acidic)))))))
        (CC and)
        (S
          (NP (NNS base))
          (VP (VBP are)
            (VP (VBN said)
              (S
                (VP (TO to)
                  (VP (VB be)
                    (ADJP (JJ basic)))))))
            (. )))
    )
  )
  (. )))
```

So here the verb phrase is: *are said to be acidic and*, which is not correct but we have recorded only till the innermost NP which gives us the wrong noun phrase.

Complex Sentences are too complex and varied to devise a grammatically sound, generalized and consistent rule to extract verb phrases.

3. RELATION SIMILARITY MODULE

Consider the relation similarity example above:

- a. The similarity calculation takes the average of the word representations. So if there is a phrase “is needed for” and “needed for is” it gives the same phrase vector since order of the words doesn’t matter while calculating the mean. This is a compromise on the semantic meaning of the phrase.
- b. “is needed for” and “is required for” have a very close semantic meaning, so the similarity of 0.6 is acceptable.
- c. “is known as” and “is called as” might appear to be very close but can mean different things semantically if the object/subject they are referring to differ in types. The similarity calculation reflects such nuances in semantic difference.
- d. “is known as” and “used for” are definitely not semantically close than “is known as” and “is called as” and yet the similarity of the former is much more (0.1 more) than the latter. Such anomalies are a limitation of the similarity calculation.

4. COMMON RELATION DETERMINATION MODULE

As the name suggests, this module replaces non-unique relations with common and more generalized relations using cosine similarity. We limit the number of relations covered by the base relation using a threshold similarity value.

We checked the results for the following threshold similarity values:

0.3, 0.34 and 0.4. For value 0.3, the base relation was found to cover maximum of relations, some of which were not even semantically related to the base relation (by observation). For value 0.4, many of the relations which found to be semantically related to the base relation, were dropped. By observation, we decided to use 0.34 as the threshold value for similarity.

FUTURE SCOPE

1. WEB SCRAPING MODULE

- a. Implementation of web crawlers to extract data from all over the internet as opposed to specific websites.
- b. Adopting a general preprocessing strategy to include all DOM tree structures.

2. TRIPLET EXTRACTION MODULE

Largest scope for improvement

- a. Grammatically sound method for relation extraction
- b. Coming up with a rule as general as possible so that all kinds of sentences are included

3. RELATION SIMILARITY MODULE

- a. Eliminating all abnormalities encountered in relation similarity.
- b. Adopting a better option than centroid method to come up with the verb phrase.

4. COMMON RELATION DETERMINATION MODULE

A more informed decision on deciding the threshold for replacing relations.

5. ONTOLOGY BUILDING MODULE AND BASE ONTOLOGY MODULE

- a. Semantic building of the ontology
- b. Adopting the conventional structure of ontology where data extracted is stored in Data Assertion Properties and Relations between them as Object Properties without compromising on the ill effects of the large size.
- c. Merging of two ontologies semantically

6. ATTRACTIVE UI

No search engine is complete without an aesthetic UI. Since we concentrated more on the back end part of this project, the UI can be further enhanced by adding animations and icons/symbols for kids and choosing the colours/theme, fonts and the site navigation wisely. This would definitely bring efficiency and subjective satisfaction to the user experience

CONCLUSION

We have proposed an implementation of building an ontology which would form the base of a semantic search engine. The crux of building this ontology lies in Triplet Extraction and finding out similarity between the relations of these triplets. The results obtained for the methods implemented by us are quite satisfactory but can surely be improved. The semantic search engine can be further refined by incorporating the suggestions mentioned in the future scope article of this report. The final aim is to make learning for kids enjoyable and have a strong back end for the search engine.