IRE MAJOR PROJECT

Ontology Enrichment, Visualization and Validation

Abstract

Ontologies form one of the pivotal layers of the Semantic Web. With the explosion of data on the Internet, it becomes necessary to store this data in the form of knowledge in ontologies. Since ontologies are rarely static and need to evolve with time, the automatic enrichment of ontologies from text becomes an important research field. Since previous work on ontology enrichment fails to take contextual information into consideration whilst extracting concepts and relationships, we propose and implement ***OntoEnricher***, an LSTM-based model to extract concepts and relationships from a text corpus in order to enrich a seed ontology from the Information Security domain. We further propose and implement ***OntoViewer***, a WebVOWL based ontology visualization and validation tool to be used by ontology validators with facilities to accept and reject the extracted relationships. We then crowdsource the ontology validation, calculate validator credibility from Twitter profile using ***TweetCredibility*** and use this credibility metric to score the decisions made by them to write changes to the seed ontology accordingly.

Section 1: Introduction

The purpose of this project is to propose and implement a framework for ontology enrichment, visualization and validation. As part of this pipeline, the first stage involves the creation of ***OntoEnricher***, an LSTM model that is trained on Information Security corpus and uses an integrated path-based and distributional approach to semantically understand the text corpus and extract concepts and relationships from it. The extracted concepts and relationships, along with the seed ontology, are fed to ***OntoViewer***, a WebVOWL-based ontology visualization and validation tool that uses crowdsourcing with credibility metrics for ontology validation. This tool allows the ontology validator to accept/reject the newly extracted concepts and relationships and writes these decisions to a database. ***TweetCredibility,*** a Word2Vec based model, uses the validators' Twitter account to evaluate their credibility by using their posts and followers as feature vectors. This credibility is used to score the accept/reject decisions made by the validator and based on the weighted average of these decisions, the final decision to write these changes is made and saved as a final owl file.

We thus divided our work into the following subprojects as follows:

a) **OntoEnricher**, the LSTM model to understand and extract concepts and relationships from a text corpus (done by Vivek)
b) **OntoViewer**, a WebVOWL based tool to visualize and validate ontologies (done by Harish)
c) **TweetCredibility**, a Word2Vec based model to predict the credibility of a twitter account based on posts and followers (done by Apoorav)

Avani was in charge of doing the literature survey for this project.

Section 2: Website

The website for this project, with the documentation and the explanatory video is hosted at https://ontologyenrichment.bookmark.com/
(note: bookmark is blocked on IIIT Lan, please access using a private connection eg:mobile network)

Section 3: OntoEnricher (Vivek)

The LSTM model to enrich ontologies

3.1 Related Work

The following are the seminal papers on ontology enrichment that we briefly went through:

1. Learning Concept Hierarchies from Text with a Guided Hierarchical Clustering Algorithm
2. Learning concept hierarchies from text corpora using formal concept analysis
3. Gimme' the context: context-driven automatic semantic annotation with C-PANKOW
4. OntoMas: a tutoring system dedicated to ontology matching
5. Using word2vec to Build a Simple Ontology Learning System

A common gap identified in these papers was that they used naive pattern matching and consequently the lack of semantic understanding and retention of concepts over longer phrases/sentences. This becomes extremely important as concepts could be embedded in words, phrases, sentences or even paragraphs which pattern-based approaches are unequipped to deal with.

To overcome these gaps, we decided to implement an integrated path-based and distributional LSTM model. We used the LSTM since it has a memory gate that allows us to keep track of the concept being talked about and the integrated model allows us to semantically understand text using distributional embeddings and capture path-based relationships at the same time. We describe our approach in further detail in the subsequent sections.

To develop our integrated LSTM model we went through the following papers:

1. End-to-End Relation Extraction using LSTMs on Sequences and Tree Structures: We discarded this paper as it focuses on ***relation*** extraction and disregards concepts.

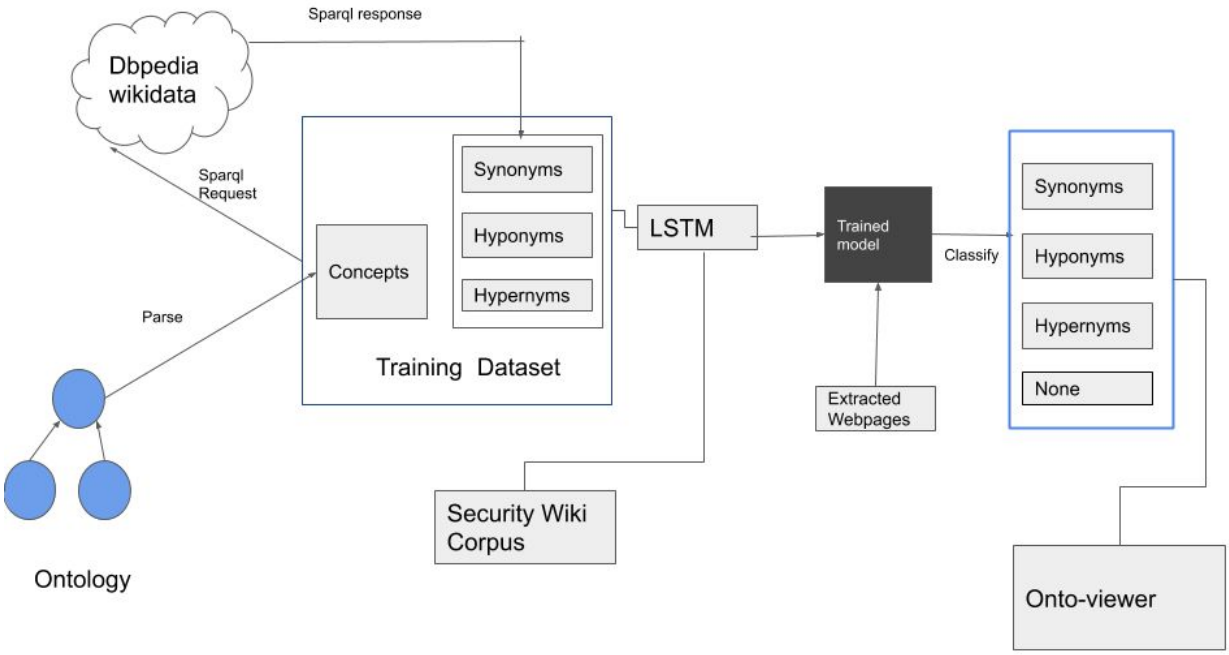
2. Improving Hypernymy Detection with an Integrated Path-based and Distributional Method: We incorporated the integrated model proposed in this paper in our ontology enrichment algorithm.

3.2 Baseline Methodologies tried

We experimented with pre-existing implementations of the ontology enrichment algorithms to find the gaps in these algorithms. For the integrated LSTM model, as mentioned above, we discarded the paper on **End-to-End Relation Extraction** and incorporate the integrated model into our LSTM code.

As part of the first baseline, we implemented a basic LSTM with binary classification and used a toy dataset consisting of ~2500 relations to train our model. For the second baseline, we constructed a larger dataset of ~40000 relations and extended our LSTM model to incorporate multi-class classification, thus allowing multiple relations to be extracted and added to the ontology.

3.3 Architecture



Onto Enricher

1. We parse the Information Security ontology and prepare a list of concepts

2. We extract the hypernyms, hyponyms and synonyms of each of these concepts from online Wiki databases like DBPedia, Wikidata and Wordnet and build a dataset that consists of rows of *(concept, extracted_term, relation_name)* records. Here, *extracted_term* denotes the term extracted from the online databases and *relation_name* is one of [“Synonyms”, “Hypernyms”, “Hyponyms”].
   Note: This is an improvement from Deliverable 2, where we used binary classification which is very constraining.

3. Each of these rows are validated by a domain expert and the row label is retained if and only if the relations belong to the same domain as the seed ontology (in this case, information security) and also share the marked relationship. Otherwise they are marked as ***None.***

4. We also added false negatives to our dataset by extracting noun phrases from the Information Security corpus (apart from the related terms) and marking them as None. This helped us in negatively scoring the model.

5. The dataset after the addition of false negatives comes to ~50K relations.

6. We then proceed to construct our training *corpus* by taking the list of concepts mentioned above and extract articles from the Wikipedia dump that have one of these concepts as their title. These articles together form the training corpus

7. For word embeddings, we train a Word2Vec model on the above constructed corpus. (https://drive.google.com/file/d/1fpN0ROwxMVHbd6URiWqdoAHAcLDS-ln8/view?usp=sharing)

8. The LSTM model uses the tagged training dataset and the corpus to learn the edge weights between every pair of noun phrases in a sentence. Edges that contain a hypernymy/hyponymy/synonymy relationship get higher weights using path-based methods. The concepts get higher weights if they are related to Information Security domain, using distributional (word-embedding) methods. Thus we build an integrated (path-based + distributional) model that gives higher weights only to those relations that contain a hypernymy/hyponymy/synonymy relationship AND are related to the information security domain.

3.4 Evaluation Mechanisms and Results

To evaluate the performance of the trained model, we take a webpage related to Information Security as input (https://tools.cisco.com/security/center/resources/virus_differences) The next step is to build our testing dataset from this web corpus. This is done as follows:

1. First we implement coreference resolution on the text corpus as pre-processing. This is done to ensure that the entities being referred to in each sentence are noun phrases, making it easier to extract them for the purpose of building the testing dataset.
2. Next, for each paragraph in the text corpus, we extract all the noun phrases in that paragraph using the NLTK NE Grammar parser and perform a combinatorial pair-wise matching of each extracted entity.
3. After doing this for each paragraph, we get a testing dataset. This testing dataset is validated and labelled by a domain expert

We then use the trained LSTM model to predict the labels of the relations in the testing dataset. We filter relations the model identifies as True and pass it as input to the next stage of the pipeline, **OntoViewer.**

We obtained 84% accuracy on a subset of this testing dataset that contained 106 relations.

3.5 Analysis

1. Our model was able to perform significantly better than the state-of-the-art on ontology enrichment. It has higher accuracy, great efficiency (running time: ~2-3 minutes) and requires minimal manual supervision once the trained model is obtained for a particular domain.

2. We observed that concepts (or other concepts similar to them) which hadn't been added in the training corpus (due to lack of validation due to time constraints) were misclassified. This sort of behavior is to be expected, and can be fixed by improving the quality of our dataset.

3. As future work, we would like to experiment on a larger and more diverse training dataset, which we predict should give us better accuracies.

3.4 Code Link :

https://github.com/Remorax/IRE-Major-Project/tree/master/OntoEnricher

Section 4: OntoViewer (Harish Kumar Datla)

OntoViewer is a WebVOWL based tool to visualize and validate ontologies

4.1 Related Work:

Most existing large scale ontology-development projects involve collaborative effort and there are a number of collaborative ontology editors that support this process (e.g. MoKi and OntoWiki). However, these approaches have scalability problem since users who verify ontologies do so since they themselves need them.

Microtask crowdsourcing: Human workers receive remuneration to complete simple, short tasks, is an existing method to obtain contributions by humans at a large scale to overcome the above mentioned shortcomings.

CrowdMaps: crowdaps use microtask crowdsourcing to get human contributions. Crowdmap divides the problem into microtasks which consists of questions for individuals, publishes them on online platforms to get responses and then evaluates them.

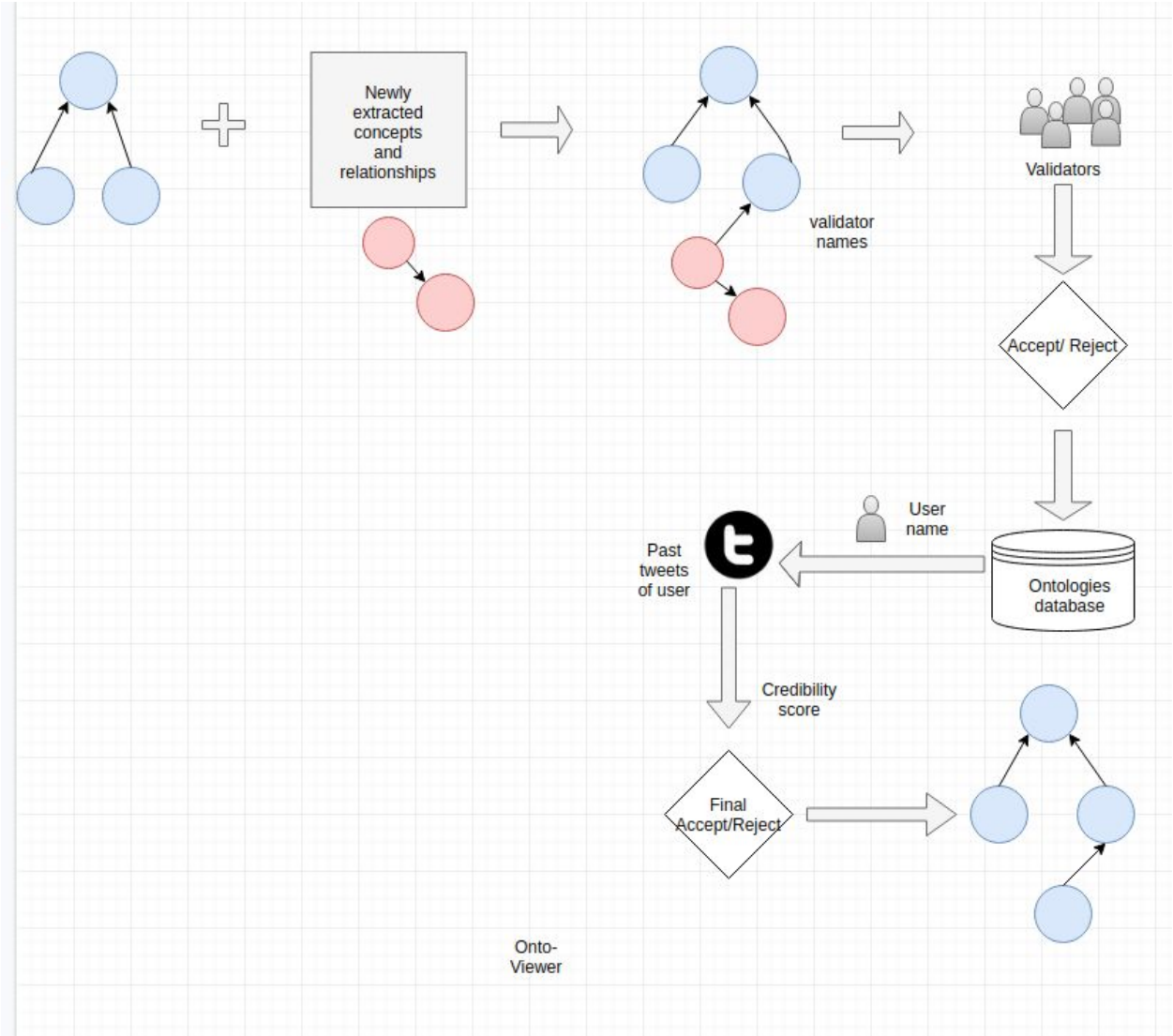
Different types of ontology viewing tools are available like protege, OWLGrED, WEBVOWL are available.

1) Protege is a free and open source ontology editor and framework.
2) Webvowl is a web applications for the interactive visualization of ontologies. It implements the Visual notation for OWL ontologies by giving a graph layout representing the ontology. It has interactive UI, where you can search for relationships and concepts.

We choose to go with webvowl because of its interactive UI and because of its easy availability, independent of any application.

4.2 Architecture :

Newly extracted concepts and relationships are added to the seed ontology file and are reflected on a UI to the user, the user accepts or rejects a decision and these decisions are stored in a database. These past tweets of user are analyzed and is given a credibility score, based on these credibility scores, a final accept or reject decision is taken and is reflected in the new ontology



4.3 Methodology :

1. Users are given a login page to login with a twitter account , the list of ontologies is displayed. The domain expert can choose an ontology to verify.

2. For visualization of ontology, we've built a tool on top of WebVOWl software. WebVOWL allows you to give ontology either in json or owl format and gives you an animation of the subjects, predicates and relationships between them.

3. The conventional webVOWL tool has an upload option, where you can upload an ontology file in json or owl format, but since in our project we need additional requirements, we use flask framework and pass the ontology json contents to the UI for display.

4. We created a flask framework to interact with UI, the json contents are passed to the UI and displayed and accepts the POST requests corresponding to accept/reject decisions.

5. Backend - In the backend part we created db models to store various tables which will be used to create a new ontology based on the accept/reject decisions of the domain expert.

6. In our project we need an accept/reject button for the user to accept or reject a relationship between a subject and a predicate which was not present in the conventional webVOWL tool, so we added a accept/reject button and then we pass on this decisions to the back-end with POST requests.

7. Multiple users can login and accept/reject the relationships and the corresponding decisions are logged in database.

8. Later the tweets of the individual users are analyzed and are given a credibility score for each of the user. These credibility scores are then weighted over the accept/reject decisions over relations and then final decisions are logged in to database.

9. After that finally a new ontology file with the final decisions is created reflecting the final decision..

4.4 Datasets

We tested the Pizza Ontology (https://protege.stanford.edu/ontologies/pizza/pizza.owl**)** on our tool.

4.5 Evaluation Mechanisms and Results:

Decisions of an expert value more than decision of a non-expert in a certain area , to test this

1) we created dummy Twitter accounts and tweeted tweets related to pizza and security.

2) These tweets are then factored and a credibility score is created for each of the user and then a weighted mean of the decisions is taken place and the final decision is arrived.

3) These decisions are then reflected in the ontology and then we check the final ontology file on webvowls online tool.

The final decisions are inline with the experts decisions.

4.6 Analysis:

- Crowdsourcing mechanisms are not scalable at large and there are various problems related to quality, but however since many subject experts use various social media platforms to write about their area of expertise, we can use their past tweets to analyze their expertise and invite them to review the ontology. This way we can be assured of the quality of the crowdsourced ontology.
- This can also be scaled as we have the feature that allows multiple logins of twitter accounts.

4.7 Code Link:

https://github.com/Remorax/IRE-Major-Project/tree/master/OntoViewer

-

Section 5: Tweet Credibility (Apoorav Singh)

We propose a model to predict the credibility of a twitter account by assigning a credibility score based on the analysis of the tweets posted by the account and the accounts in the following list.

5.1 Related Work

Semantic approach for tweet categorisation

In this paper, they propose a semantic approach of improving the accuracy of Tweet Categorization based on feature expansion from external Knowledge Bases (KBs) and the use of eXtended WordNet Domain as a classifier. In particular, they propose a deep enrichment strategy to extend tweets with additional features by exploiting the concepts present in the semantic graph structures of the KBs. Then, the supervised categorization relies only on the ontological knowledge and classifier training is not required. Empirical results indicate that this enriched representation of text items can substantially improve the Tweet Categorization performance.

Link:
https://reader.elsevier.com/reader/sd/pii/S1877050918312432?token=9B8A8FF9DFBFDEDDD1739663A26270CAC4CC2C83BEFB05468259CB700CC097121695D6305D7492FF582E9542506E528C

Tweet topic categorisation using entity knowledge base and topic enhanced word embedding

In this paper, they present TweetSift, an efficient and effective real time tweet topic classifier. TweetSift exploits external tweet-specific entity knowledge to provide more topical context for a tweet, and integrates them with topic enhanced word embeddings for topic classification. A tweet also provides a set of metadata that we can use to enrich its topical context. For example, there are mentions, URLs, hashtags, named entities, user handle, user name, and profile description. TweetSift makes use of these data by trying to find complete or partial matches between these entities and entries in an entity knowledge base (Entity KB) built from knowledge sources specific for social media.

Link:
https://www.researchgate.net/publication/309471104_TweetSift_Tweet_Topic_Classification_Based_on_Entity_Knowledge_Base_and_Topic_Enhanced_Word_Embedding

We did not adapt the approaches used in these papers because we need to assign a credibility score to the users which requires us to use regression model. But the models proposed above perform topic classification which won't give continuous scores as outputs. Thus, we switch to word2vec for the purpose of scoring.

5.2 Baseline Methodologies Used

Data Extraction**:-** We collect the data from Twitter Streaming API. We use python library Tweepy to connect to Twitter streaming API and downloading the data. In order to access the API, we need to get 4 pieces of information from twitter: API key, API secret, Access token and Access Token secret. The data thus obtained is in JSON format. Each entry/row of JSON dump will have fields, like lang, utc_offset, trends, text, etc.

Multinomial Naive Bayes model for classification :-

The Naive Bayes classifier classifies the given novel tweet into categories/topics by training our model on suitably chosen corpus. In the Naive Bayes classifier, the posterior probability is much simpler to work with because of the independence assumption inherent in the Naive Bayes Model. The multinomial Naive Bayes model is different from the Standard Naive Bayes because it defines the distribution of each feature as a multinomial expression. The proposed multinomial Naive Bayes model provides an accuracy ~72%..

Feedforward Neural Network Model :-

The neural net consists of layers of neurons. The outputs from neurons are passed to the next layer of neurons, which use them as inputs of their own function, and produce further outputs. Our proposed feedforward neural net model is a supervised learning model**,** where the neural net is given labelled inputs. The labelled examples, are then used to infer generalizable rules which can be applied to unlabeled cases. We use softmax function in the final output layer to assign a label/topic to the novel tweet fed into the trained model.

When you feed the bag of words representation of data as input to the neural network, the model yielded a better accuracy and in addition, we get a final score corresponding to each domain in the output of neural network. Thus, using feedforward neural network was used.

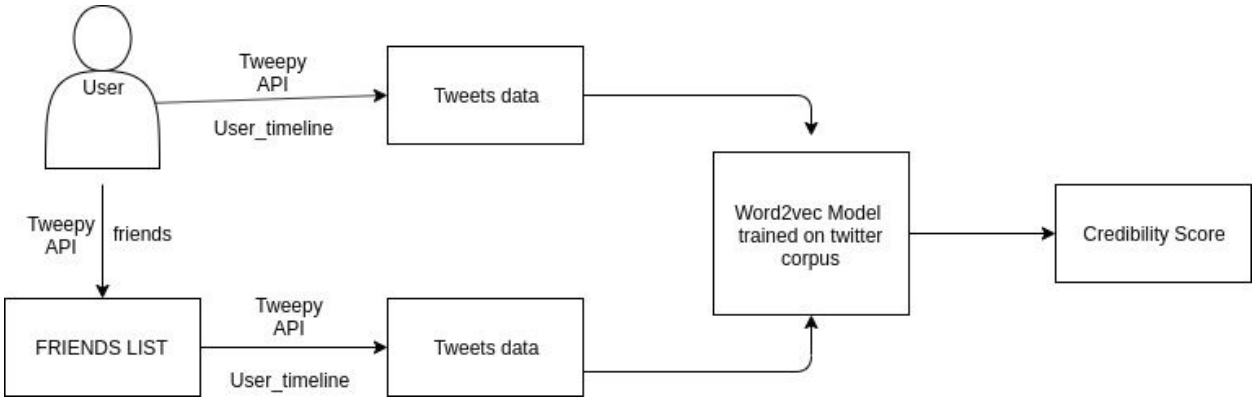
The proposed neural network model provides an accuracy ~75%.

Word2vec :-

The classification algorithms were not sufficient to give us a score for each profile. We needed scores over a continuous interval, and since there are no standard regression datasets available for our purpose, we used an unsupervised algorithm that uses a

Word2Vec model pretrained on Twitter corpus (word2vec_twitter_model.bin) to calculate the word embeddings and take similarity score of these embeddings with security domain. We then take the mean of this score over the length of the text corpus and output the final score.

5.3 Architecture



1. We extract tweets text data from tweepy Twitter API using the target user account handle.

2. Additionally, we also extract the target user's friends list using API.friends and extract the top most popular tweets of the friends accounts.

3. Both the data above is combined and sent to the word2vec model which is trained on the twitter corpus. (https://drive.google.com/file/d/10B7cvx3xN7Ef_FxwIO8sigd1J1Ibe6Lu/view?usp=drivesdk)

4. The word2vec model outputs a credibility score analysing the tweets from the target user and the friends of the user account.

5. The score is a weighted sum of the k-most popular tweets of that user's friends and that person's tweets where greater weights (10x) are given to the tweets of the friends for empirical reasons.

5.4 Evaluation Mechanism and Results

The supervised learning models (Feedforward neural net, LSTM, multinomial Naive Bayes) for topic classification were trained on data extracted from tweepy Twitter API. The data extracted was split into train and test sets and the model was tested on test set to output the accuracy.

For evaluating the word2vec model, we extract data from Twitter API which includes, tweets of the target user account, top popular tweets of the accounts in the following list of target user. The model is fed with above constructed data and it outputs a credibility score for the target user account. The scores generated can be easily verified further.

On a trial run of the model on cyber security domain, the model yields a score of 0.31 for account @dominos_india and a score of 1.98 for the account @saltcontrol (Bio:- The #1 secure enterprise #communications company within the #Cybersecurity 500. A software solution that enables absolute privacy in #mobile communications.). Thus, the model assigns higher score to the target account with higher credibility in the cyber security domain.

5.5 Analysis

The model outputs a credibility score for the target user account. The score provided by the model for accounts related to information security is significantly higher than the score provided for accounts which are not. Thus, the model successfully analyses the tweets of the target account and the following list accounts.

5.6 Code Link:

https://github.com/Remorax/IRE-Major-Project/tree/master/TweetCredibility