Chapter 3

Ontology Based Model for Rumor Initiator Detection in Online Social Networks

This chapter focuses on the first contribution of this thesis, i.e., detecting the initiator of a rumor in OSNs. We provide an introduction and motivation for the proposed approach in Section 3.1. Section 3.2 shows the proposed ontology-based model for OSNs. Section 3.3 shows the use of the proposed ontology model to detect rumor initiators. Section 3.4 demonstrates the usage of the proposed method for a case study of Twitter (now known as X)[1]. Section 3.5 provides a quality assessment of our proposed model for its acceptability. Section 3.6 concludes this chapter with a few future possibilities. Detailed related work is provided in Section 2.2.1.

3.1 Introduction

Various Online Social Networks (OSNs) are running on the Internet today, with Facebook and Twitter being the most popular. These networks are structurally

[1]We have used the name Twitter instead of X throughout this thesis.

identical but semantically and contextually different [30]. For example, a user on Facebook is viewed as a friend of another user and they have a symmetric relationship with each other, that is, if A is a friend of B, then B is also a friend of A. However, on Twitter, there is a follower-and-followee type of relationship which is unidirectional rather than symmetric. In addition, there is no consensus on the guidelines for the development of OSN [31]. Each OSN uses different techniques to represent its interface and functionality. In addition, each OSN has different concepts and terminologies. So, they lack in generality because of their differences in various aspects. In cases where the same rumor spreads over multiple OSNs, traditional methods require separate detection pipelines for each platform due to structural and semantic inconsistencies. Since there is no semantic alignment between various OSN platforms, rumor source detection logic cannot be reused, leading to redundant and increased efforts. To address these problems, we have proposed a three-layered ontology-based model for OSNs. The proposed ontology overcomes this limitation by introducing a platform-independent upper-level ontology that standardizes the core OSN concepts. For example, a *User* is an entity that can perform an *Activity* on a *Post*. Whether it is a "Retweet" on Twitter or a "Share" on Facebook, the action is conceptually similar and is modeled identically at the upper level. Each OSN-specific structure is then mapped to this unified model through a vendor-specific lower ontology, allowing data from different OSNs to be integrated into a single semantic knowledge base. Consequently, a single reasoning framework and query engine (via SPARQL) can identify the rumor initiator across all platforms simultaneously, eliminating redundant effort and ensuring consistency in analysis. This unified, interoperable approach enables scalable, efficient, and reusable detection of rumor sources in heterogeneous OSNs. We selected an ontology to model OSNs for the following reasons.

1. Ontology lays the foundation for the knowledge graph of OSNs which can accommodate real-life social networks with millions of nodes and billions of links and link cycles.

2. Ontology imparts semantics to the data by annotating it with structured metadata, enabling not only syntactic representation but also semantic interpretation of entities and their relationships within a domain. It is useful in scenarios which require the integration and uniform analysis of heterogeneous data sources, such as Online Social Networks (OSNs). This semantic formalization is useful for addressing the challenge of interoperability across multiple OSNs. Each social network typically adopts its own data representation schemes, relationship models, and terminology. For example, while Facebook uses a symmetric 'friendship' model, Twitter relies on an asymmetric 'follow' structure; similarly, actions such as 'sharing' or 'retweeting' may differ in implementation but are functionally analogous. The ontology mitigates these differences by providing one solution for all. Given that users often maintain accounts on several social platforms and may initiate the same piece of information (which can be rumors) across multiple OSNs, a platform-agnostic approach becomes essential. In the absence of such an approach, identifying rumor initiators would require the development of separate detection methods for each OSN. So, opting for a single solution is more convenient than developing algorithms for initiator detection on multiple OSNs separately.

3. Ontology is empowered with reasoning abilities that can be exploited to answer domain-related queries for intended purposes, i.e., finding rumor initiators in OSNs a posteriori, finding all OSNs where rumor is initiated, etc.

3.2 Proposed Ontology-Based Model for Rumor Initiator Detection in OSNs

We propose a three-layered architecture for modeling the OSN concepts used to find the initiator of rumors in the OSN. This three-layer architecture comprises a design-time layer, an integration layer, and a run-time layer. In the design-time layer, the core constructs of the upper-level ontology are defined. Based on this, lower-level

ontologies can be constructed for different OSNs that conform to the upper-level ontology. The lower-level ontology model is populated with OSN data to create an RDF triple store in the integration layer. The runtime layer is responsible for querying on this RDF data of the OSN to get the desired outcomes. The architecture of the proposed work is shown in figure 3.1. The layers are explained in detail in the following subsections.

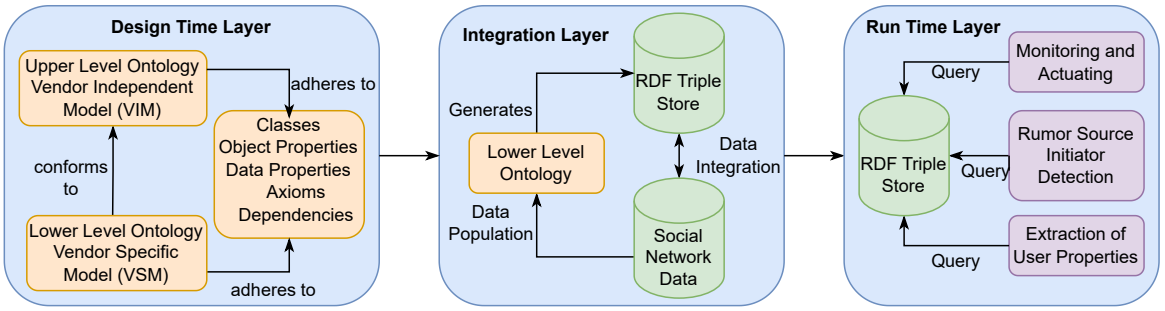

FIGURE 3.1: Three Layered Architecture for Modeling OSN Ontology

3.2.1 Design Time Layer

The design time layer deals with the construction of the ontology model for OSNs. It specifies the system structure, ontology component behavior, data flow, and control flow. Here, we propose an upper-level ontology that contains the domain concepts of OSNs. This is a vendor-independent model (VIM), as it provides abstraction. An upper-level ontology is a model of commonly shared concepts and relationships generally applicable across various domain ontologies like Facebook, Twitter, etc. This ontology employs a core glossary that overarches the terms and associated object descriptions used in domain ontologies. Hence, the purpose is to fit *one size for all* [94]. This ontology model represents various "different vendor, domain-specific" applications. So, they can be used for possible interoperability. Figure 3.2 shows our proposed upper-level ontology obtained from the OntoGraf plugin of Protégé, an open-source ontology editor and framework for building intelligent systems[2].

[2]https://protege.stanford.edu

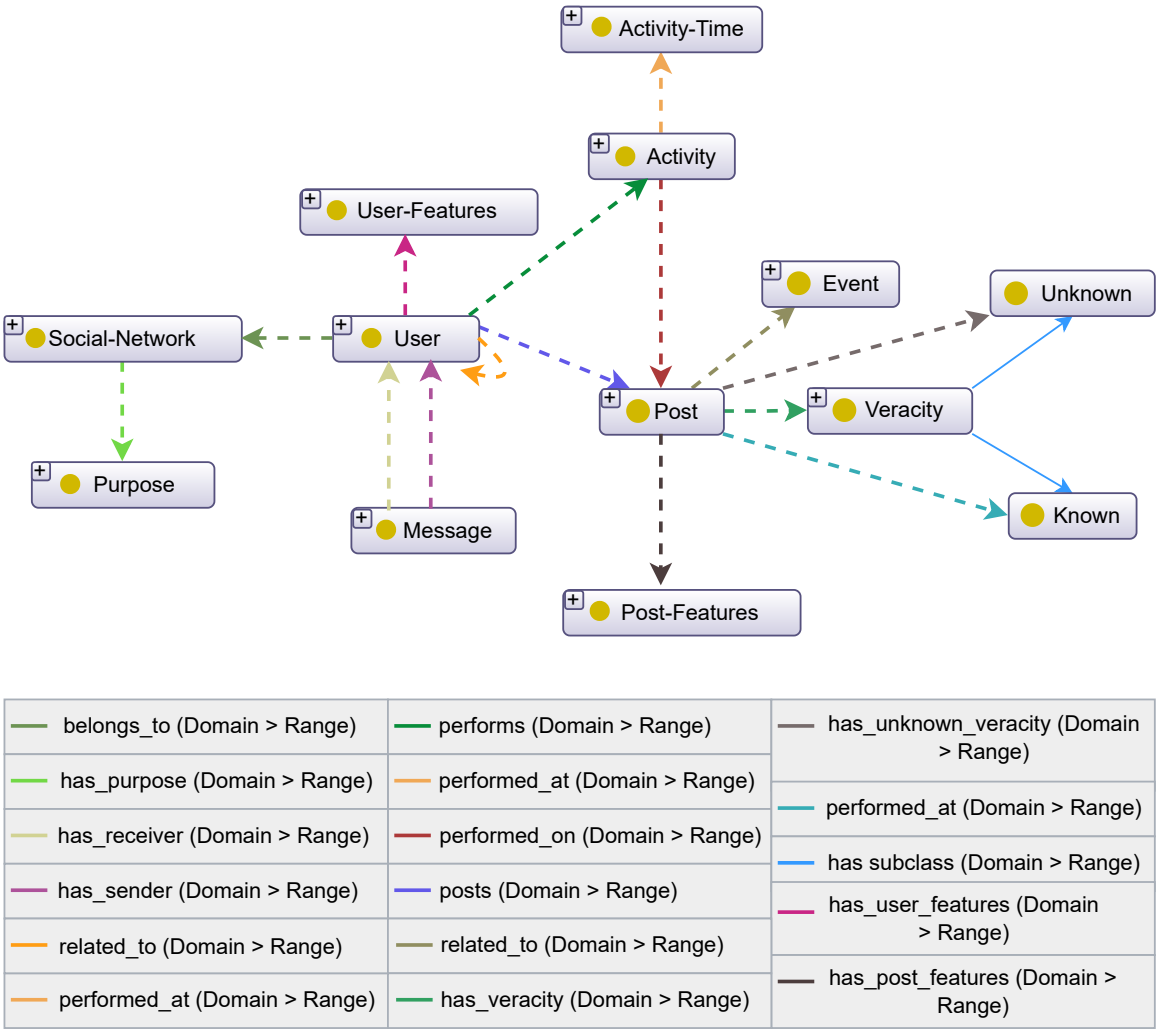

FIGURE 3.2: Upper-Level Ontology Model for OSNs obtained using OntoGraf Plugin of Protégé

The proposed ontology-based model incorporates a set of eleven classes at the upper level, designed to represent the foundational elements of OSNs in a vendor-independent manner. The classes are chosen after a manual analysis of common functionalities and entities across popular OSNs, their structural and semantic characteristics, a review of the academic literature on OSN ontologies, and expert understanding of semantic equivalence and functional roles in social networks. Various OSN platforms may differ in how they define and structure relationships (e.g., Facebook's symmetric friendship vs. Twitter's asymmetric follow model), the core concept of user interaction remains consistent. These platform-specific constructs

are abstracted into generalized concepts through the upper-level ontology.

These classes are connected to each other via properties represented as dashed arrows. Each property is differently colored and mentioned in figure 3.2. The ontology classes are: *Social-Network, Purpose, User, User-Features, Post, Post-Features, Event, Activity, Activity-Time, Veracity (subclasses- known, unknown)* and *Message*. These classes represent the concepts of the OSN domain and are described as follows.

1. *User* - A user is a person who has an account on the OSN site. They can be a user on a single OSN or multiple OSNs and have the role of a friend, follower, or followee to other users depending upon the user-relations of the OSN.

2. *User-Features* - These are the characteristics that describe a user. There can be unique and overlapping properties of users on different OSNs. For example, name and profile picture are common user properties that exist in most of the OSNs while work, education, etc. are user properties that exist in Facebook but not Twitter.

3. *Post* - It is some thought, idea, news, etc. shared by a user in the form of text, image, video or attachments with other users on which some actions take place.

4. *Post-Features* - These are the characteristics that describe a post. For example, type of the post i.e. text, video, links, etc., hashtags included with the post.

5. *Activity* - It is the action performed by the users on the post. For example, like, comment, share etc. These are very important to understand the behavior of users and also of posts within the system.

6. *Activity Time* - This class deals with the temporal aspects of the OSNs. It tells us when an activity is performed, and we can track its propagation.

7. *Event* - This class tells which event a post belongs to, that is, is it a natural calamity, political event, personal life event, etc. Events can be useful in finding the interests of users.

8. *Message* - Besides broadcasting the posts to many users, OSNs also enable one-to-one communication using messaging features. One can directly communicate with others using a secure channel.

9. *Purpose* - This class is useful to find the semantics of some concepts. For example, if the purpose is friendship, then users are in a symmetric relationship with each other; if the purpose is micro-blogging, users are in a directed relationship with each other etc.

10. *Social Network* - This class is built by composing all the classes and has a purpose.

11. *Veracity* - This class represents the status of truthiness, i.e., the veracity of a post. It is further divided into two subclasses- *Known* subclass i.e. veracity status is known to be true or false and *Unknown* subclass i.e. veracity status is yet to be verified.

Each class acts as an abstract container for semantically similar entities on various OSN platforms. Table 3.1 shows each of the generic classes mapped to the native concepts of multiple platforms.

A relationship property in an ontology connects two instances or individuals of classes, expressing a meaningful semantic association between them. There are two types of properties. First, object properties that link an individual of one class to an individual of another class. For example- *User* ***performs*** *Activity*, *Post* ***has_veracity*** *Veracity*. Second, data properties that link an individual to a literal data value (e.g., string, number, date). For example, *ActivityTime* ***hasDatetime*** *"2024-10-10T15:00"*. The relationship properties enable reasoning using which one

TABLE 3.1: Generalization Strategy for Upper-Level Ontology Classes

Upper-Level Class	Generalization Strategy
User	Represents any actor: Facebook user, Twitter handle, LinkedIn profile
Post	Abstract form of content: Facebook post, tweet, LinkedIn update
Activity	Any user interaction: like, share, comment, retweet
Activity-Time	Timestamp associated with the activity (e.g., retweet time)
Post-Features	Includes post type (text, video), hashtags, sentiment indicators
User-Features	Profile metadata (e.g., profile picture, verified status, country)
Veracity	Truthfulness of content (true, false, unknown)
Event	Thematic categorization of posts (e.g., political, natural disaster)
Purpose	Defines network semantics (e.g., Facebook = friendship, Twitter = microblogging)
Message	One-to-one communication (DMs, private messages)
Social-Network	The platform itself (Facebook, Twitter, LinkedIn, etc.)

can infer new knowledge based on known relationships. They also support querying using a query language like SPARQL and ensure interoperability. Generalized properties make it possible to model diverse OSN platforms using a shared ontology schema. Each property has a domain and range. The domain specifies the set of classes or individuals to which a property can be applied, while the range specifies the set of values that a property can take. For e.g., *User_1* ***belongs_to*** *Twitter*, here *User_1* which is an individual of class *User* is the domain of the property ***belongs_to*** and *Twitter* which is an individual of class *Social-Network* is the range. The object properties for the upper-level ontology model are described in table 3.2. The table describes each of the properties, its domain and range and a generalized perspective towards that object property.

Table 3.2: Ontology Object Properties and Their Generalization Perspective

Object Property	Domain	Range	Description	Generalization Perspective
belongs_to	User	Social-Network	Links a User or Activity to a specific Social-Network (e.g., Twitter, Facebook)	Supports cross-platform analysis by tagging users with their originating platform
has_post_features	Post	Post-Features	Connects a Post to its features such as hashtags, media type, or sentiment	Normalizes post metadata across OSNs that represent features differently
has_purpose	Social-Network	Purpose	Associates a Social-Network with its intended use (e.g., friendship, microblogging, academic networking)	Encodes platform semantics to inform reasoning about interaction patterns
has_receiver	Message	User	Links a User to another User via a Message, modeling direct communication	Abstracts one-to-one messaging models from different OSNs
has_sender	Message	User	Links a User to another User via a Message, modeling direct communication (DMs, private messages)	Abstracts one-to-one messaging models from different OSNs
has_user_features	User	User-Features	Associates a User with personal or profile attributes like location, verification status, or education	Maps user metadata fields from various platforms to a shared semantic space
has_user_relation	User	User	Describes the social relation between users (e.g., friend, follower, connection)	Generalizes network models—symmetric (Facebook), asymmetric (Twitter), and professional (LinkedIn)
has_veracity	Post	Veracity	Links a Post to its Veracity status (e.g., True, False, Unknown)	Enables reasoning about truthfulness independent of how each OSN flags or moderates content
has_known_veracity	Post	Known	Links a Post to its True-Veracity status	Standardizes how confirmed rumors are represented across platforms
has_unknown_veracity	Post	Unknown	Links a Post to its False-Veracity status	Unifies how uncertainty is flagged across platforms
performed_at	Activity	Activity-Time	Connects an Activity to Activity-Time, enabling time-based reasoning	Unifies timestamp semantics from different OSNs into a single time-aware reasoning structure
performed_on	Activity	Post	Associates an Activity with a Post, indicating which content the activity was directed at	Abstracts content interactions regardless of post types—be it a Tweet, Facebook Post, or LinkedIn Update
performs	User	Activity	Links a User to an Activity (e.g., a user retweets, shares, or likes a post)	Generalizes platform-specific actions like “Share” (Facebook), “Retweet” (Twitter), “Like” (LinkedIn) under one common semantic action model
posts	User	Post	Links a User to content they post	Unifies user-post relationships across OSNs with varied post types
related_to	Post	Event	Associates a Post with an Event, allowing grouping of related content (e.g., all posts about an earthquake)	Facilitates topic-based filtering and clustering across OSNs

Ontology metrics are quantitative measures that are used to assess the characteristics of an ontology. Table 3.3 shows the count of various upper-level ontology constructs. In the table, metric column denotes various ontology constructs and count denotes the number of each ontology construct.

TABLE 3.3: Metrics of Upper-Level Ontology for OSNs

Metric	Count
Class	13
Object Property	16
Data Property	4
Axioms	84
Logical Axiom	52
Declaration Axiom	32

Ontology Complexity- Zhang et al. [95] have proposed metrics to measure the complexity of ontology. They divided these metrics into two distinct categories: one set assesses the overall design complexity of an ontology (referred to as ontology-level metrics), and the other set evaluates the intricacies within the internal structure (referred to as class-level metrics). As these metrics' values increase, so does the need for cognitive resources to understand and maintain the ontology, indicating a corresponding increase in complexity. We have calculated these metrics for our proposed ontology. The value of ontology complexity metrics gives an insight into the complexity of our model. Table 3.4 shows the complexity metrics defined in [95] calculated for our proposed ontology. The complexity metrics of this ontology have low values, indicating that the proposed ontology is less complex. However, as we keep on increasing classes and subclasses, various factors like depth of tree, number of children, etc. will increase, which eventually leads to more complex ontology.

Based on the upper-level ontology, lower-level ontologies are constructed manually over the top of the upper ontology. These are vendor-specific models (VSM) and are less abstract. These ontology models capture more of the vendor's requirements and constraints. A lower-level ontology model is in conformity with the upper ontology model and develops concepts accordingly. However, manual construction of

Table 3.4: Ontology Complexity Calculation

Ontology Level Metrics		Class Level Metrics	
Metric	Metric Value	Metric	Metric Value
Size of vocabulary	29	Average number of children	0.18
Edge node ratio	1.30	Average depth of inheritance	0.09
Tree impurity	5	Average class in-degree	1.18
		Average class out-degree	1.54

lower-level ontology has a cost associated with it, requires human efforts and a priori knowledge. It is a resource-intensive task and involves domain experts, ontology engineers and knowledge engineers who work together to create the relationships, classes, and instances specific to the lower-level domain. Costs can vary significantly depending on the complexity and size of the ontologies being developed, the expertise of the team involved, and the tools and resources used in the process. Human efforts include domain experts who are necessary to ensure that the lower-level ontology accurately represents the concepts and relationships specific to that domain and ontology engineers who are required to apply formal modeling techniques and tools to create the ontology. The efforts can be substantial, particularly for complex domains or when integrating multiple sources of knowledge. In addition, to construct lower-level ontologies, a priori knowledge of the target domain is essential. This includes knowledge of the specific concepts, relationships, and constraints within the domain. Domain experts play a crucial role in providing this knowledge. In addition, familiarity with the upper-level ontology and its structure is necessary to ensure compatibility and alignment with the broader ontology framework.

One of the advantages of ontology is interoperability. The upper-level ontology can be reused by different vendors in the same domain with interoperable concepts. Table 3.5 shows the upper ontology concepts mapped to lower ontology concepts for four OSNs- Facebook, Twitter, ResearchGate, and LinkedIn. The table serves as an interoperability reference, linking upper-level ontology concepts to their platform-specific counterparts for the four popular social networks, which can be extended to other OSNs as well. These mappings were manually curated with expert knowledge,

empirical usage patterns, and existing ontological vocabularies. These four OSNs have been considered as they are highly popular among users for different purposes. Facebook is the most popular friendship network with almost 3 billion monthly active users[3]. Twitter is the largest microblogging OSN with 237.8 million monetizable daily active users[4] which is also widely used for the dissemination of news in real time. ResearchGate is a popular OSN among the researchers' community in academics and LinkedIn is the most popular professional social network[5]. These OSNs primarily offer text-based user content, but they also support multimedia content. There are a few concepts that are semantically different. For example, *User* in Facebook is not semantically similar to *User* in Twitter. However, there are few concepts which are semantically equivalent. For example, *Activity-Time, Event, Message* and *Veracity*.

3.2.2 Integration Layer

The integration layer provides a knowledge base by populating the lower ontology model with OSN data. The OSN data can be extracted using various methods like application programming interfaces (APIs) provided by OSNs, web scraping or third-party tools. Data extraction from OSNs requires compliance with the terms of services of each platform to respect user privacy and legal requirements. Some social networks like Twitter and Facebook provide their APIs, i.e., Twitter API[6] and Facebook Graph API[7] respectively. To access the data, one needs to register as a developer and obtain API key or access token. These APIs typically provide the OSNs data in a semi-structured format like JSON or XML. Web scraping is another method of extracting data directly from HTML pages of OSNs. However, there are many OSNs that do not allow web scraping or have strict limitations. In addition, web scraping comes with a lot of legal and ethical considerations. In some cases, third-party tools or libraries may provide simplified interfaces to access social

[3]https://www.statista.com/topics/751/facebook/#topicOverview
[4]https://www.statista.com/topics/737/twitter/#editorsPicks
[5]https://www.statista.com/topics/951/linkedin/#topicOverview
[6]https://developer.twitter.com/en/docs/twitter-api
[7]https://developers.facebook.com/docs/

TABLE 3.5: Mapping of lower ontology model concepts of four OSNs to upper-level ontology concepts to ensure Interoperability

Upper Level Ontology Concepts	Equivalent Concepts in four OSNs that ensures Interoperability			
Social Network	Facebook	Twitter	ResearchGate	LinkedIn
Purpose	Friendship	Microblogging	Academic	Professional
User	Friends(in a symmetric relationship with each other)	Followers/Followees(in a directed relationship with each other)	Followers/Followees(in a directed relationship with each other)	Connections(in a symmetric relationship with each other)
User-Features	Work, Education, Location Contact, BasicInfo, Family/Relations, Life-events, Profile Picture, Name, Date of Birth, Gender	Name, Profile Picture, User Description, Date of Birth	Name, Profile Photo, Skills, RG Score, h-index, reads, Research Interest, citation, contact, Experience, Education, Awards/Achievements	Name, Profile Picture, Education, Contact, Current Position, Education, Skills, Affiliation
Post	Post	Tweet	Project, Research Items, Questions	Post/Article
Post-Features	Post-type (video, text, link, location, emoji), Post-time, hashtags	Tweet-type(148 character text, image, poll, link, video), Post-time, hashtags	Research-Item-Types (articles, chapters, conference papers, data, presentation), Post-time	Post-type(Images, videos, text), Post-time
Activity	Like, Comment, Share	Like, Reply, Retweet	Recommend, Follow, Share, View, Request full text	Like, Comment, Share
Activity-Time	Semantically Equivalent Concept in all Social Networking Sites			
Event	Semantically Equivalent Concept in all Social Networking Sites			
Message	Semantically Equivalent Concept in all Social Networking Sites			
Veracity	Semantically Equivalent Concept in all Social Networking Sites			

media data through APIs or web scraping. For example, Tweepy[8]is used to extract Twitter data. Extraction of OSN's data requires one to consult the platform's official documentation and guidelines before attempting data extraction. The use of APIs or third-party tools has a cost associated with it. In many cases, access to basic data through APIs is free, but there might be restrictions on the number of requests per day or the amount of data one can retrieve. However, some platforms offer premium or enterprise-level APIs that require a fee for more extensive access and features.

The data extracted using any of the above-mentioned methods is populated to the lower ontology model as instances to generate an RDF triple store[9] in the form of triplets and thus creates a knowledge base. An RDF triplet is of the form subject, property and object where the subject is a web resource having some URI, property is also a URI that links the subject to the object, and the object is either a web resource having some URI or hard-coded value. A collection of RDF-triplets combines to form a knowledge graph [96] that represents the linked data. These triples help represent resources and data on the semantic web and extract the desired information from the knowledge base. Due to their URI representation, they are machine-readable.

3.2.3 Run Time Layer

To extract the knowledge, the runtime layer is used to reason the RDF triple-store. This is done using an RDF query language, SPARQL [97]. For finding the rumor initiator in OSNs, we need information regarding the post's veracity, back-tracking a post to its source using temporal features, finding the user characteristics, etc. The information is extracted using SPARQL and further analyzed to find the source of the rumor. The subsequent subsection describes the method for finding rumor initiators in OSNs using the proposed ontology model.

[8]https://www.tweepy.org/
[9]https://www.w3.org/RDF

3.3 Finding the Rumor Initiators in OSNs using Proposed Ontology Model

Rumor initiators are social media users who diffuse information on the network without claiming its truth. Our proposed method to find the rumor initiators assumes that the rumor has already been detected. There are various approaches based on machine learning and deep learning to detect rumors. These approaches use supervised machine learning classifiers, recurrent neural network-based approaches [98–101], convolution neural network-based approaches [102–104], graph convolution network-based approaches [105–107], and many more. There is a vast literature on rumor detection. To gain more insight, one can refer to [108–110]. Deep learning-based methods have been shown to be highly successful with an accuracy as high as 96% in detecting rumors. Since our method is dependent on the rumor detection technique, whatever post is labeled as rumor by these methods will be analyzed to find the rumor initiators by our method. This limits our method in case of a false positive rumor detection as it will also find the source of a post which has been falsely detected as a rumor. Similarly, for cases where rumors have been falsely detected as non-rumors, the initiators are left unidentified.

Let us consider a rumor R being circulated in an OSN by users u_1, u_2, u_3 and u_4 at time t_1, t_2, t_3 and t_4 respectively. A rumor initiator is a user who *forwards* R at time t_1 such that $t_1 < t_2 < t_3 < t_4$. In Section2.2.1, we have seen the previous work done to identify this user. However, the proposed solutions were limited to trees or graphs limited in structures. In real-world scenarios, we do not have such simple structures, but graphs having millions of users with billions of links and link cycles of varying lengths. By populating ontology with instances, we get an RDF triple-store integration layer which can be represented in the form of knowledge graphs. These knowledge graphs are capable of accommodating all the cycles in the networks without any restriction on the network structure. Ontology models have the inherent reasoning power to answer domain-related queries. We can perform

implicit reasoning by running ontology reasoners like Pellet, Hermit, etc. to check for any model inconsistency and explicit reasoning by querying the underlying model. In our approach, the lower ontology model is populated with instances to create a knowledge graph queried for data retrieval using SPARQL[10], which improves the performance of search engines [97].

To find the rumor initiators, we use the property path expressions of the ontology. These expressions backtrack on a propagation structure using a specific property of the RDF. For finding the rumor initiator, we consider two cases, when the veracity of the rumor is known and when the veracity of the rumor is unknown. Furthermore, we find in which of the OSNs the same rumor is initiated. This ensures finding an initiator for one OSN and generalizing it for others, thus providing interoperability and removing duplicacy of efforts to find the same rumor initiator on multiple OSNs individually.

3.3.1 When the veracity of the Rumor is Known

When the veracity of the rumor is known, it means that we already know whether the rumor is true or false. Many researchers have attempted to find the veracity of rumors. Rumor veracity can be classified as true, false or unknown. Unknown veracity means that its truthiness cannot be established and finding the post-veracity is still unresolved. Many machine learning-based methods [111–114], and deep learning-based methods [115] have been proposed to determine the rumor veracity. For more information, Varshney and Vishwakarma provide a comprehensive survey on rumor veracity detection [116].

In both cases i.e. true rumor and false rumor, we only have to find out the initiator of the post. For this purpose, we backtrack the *Activity* class of ontology along with over the *Activity-Time* in the upper-level ontology. We find all users who perform some activity on the *rumorous* post using property path expressions

[10]https://www.w3.org/TR/rdf-sparql-query/

and then sort them according to the *Activity-Time*. Thus, we get the users who first initiated the post. This scenario is explained in figure 3.3. In figure 3.3, *User 1* has *Tweeted* a *Rumorous Post* having veracity *True* at time t_1. This rumor is further *Retweeted* by *User 2, User 3* and *User 4* at time t_2, t_3 and t_4 respectively. By backtracking on activity *Retweet*, we can find *User 1* who is the rumor initiator in this case.

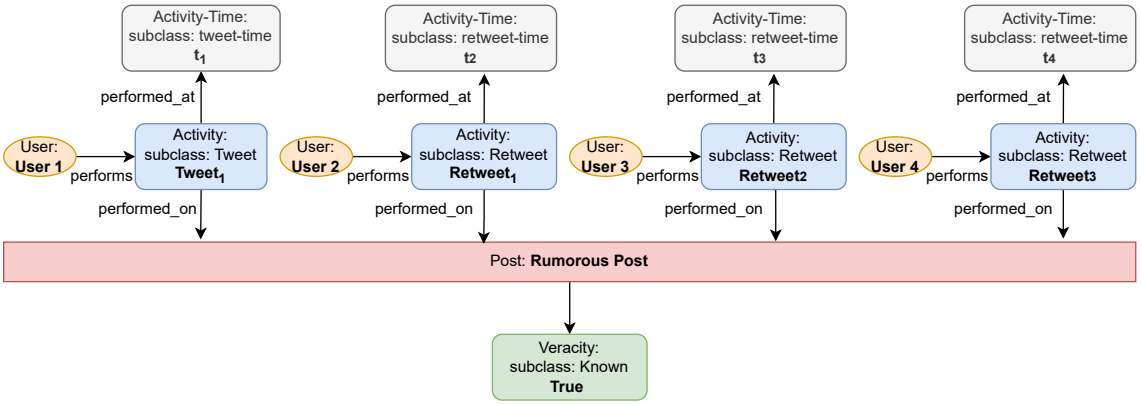

FIGURE 3.3: Propagation of a rumorous post with different activities performed by users at time t_1, t_2, t_3, t_4 where $t_1 < t_2 < t_3 < t_4$

3.3.2 When the veracity of the Rumor is Unknown

When the rumor's veracity is unknown, it means that we do not have any information regarding the post's authenticity. In this case, we have to find the initiator of the rumor and establish their credibility. The initiator can be found using the previous approach of backtracking on the property path; however, we further have to analyze user and post features to find whether a user is credible or post is authentic. For finding the features of a user, we use *User-Features* class of ontology. This class is further divided into *Profile-based* and *Account-based* features (see Fig.3.6). Similarly, for post features, we use *Post-Features* class of ontology which is further specified as *Content-based*, *Propagation-based* and *Multimedia* features. These features are further analyzed to establish the credibility of the user and the authenticity of the post. For example, if the post comes from a verified user who has a genuine profile, we consider the user to be credible and the post as authentic. Establishing

the credibility of the user is outside the scope of this thesis work, and one can refer to [117–119] for more insight. Here, the ontology model is used to extract features that further establish the credibility of users in OSNs. This scenario is explained in figure 3.4. In figure 3.4, *User 1 tweets* a rumor at time t_1. All the *Post-Features* subcategorized as *Content-Based*, *Propagation-Based* and *Multimedia* have been extracted to establish the authenticity of the post. Also, since *User 1* is the initiator, all the *Account-based* and *Profile-based* features of *User 1* have been extracted to establish the credibility of this user. Different OSNs have different user and post-features as specified in Table 3.5, so the lower-level ontology helps to extract the user and post-features for a particular OSN.

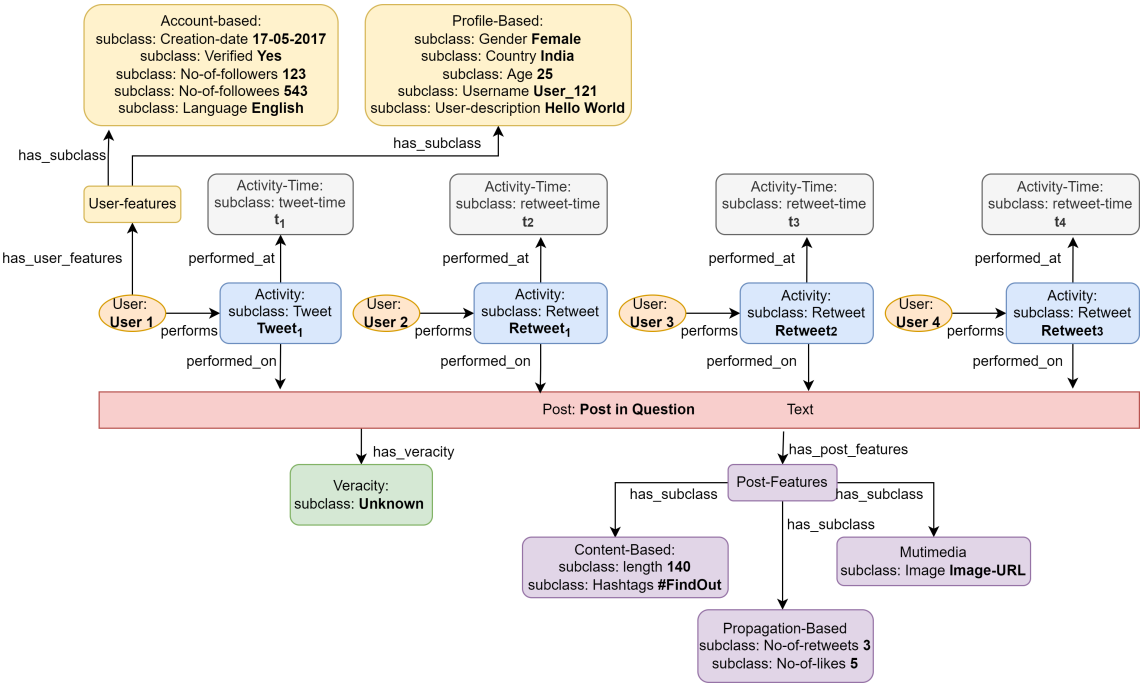

FIGURE 3.4: Propagation of a rumor with different activities performed by users at time t_1, t_2, t_3, t_4 where $t_1 < t_2 < t_3 < t_4$ and extraction of user features and post features

3.3.3 Finding Multiple OSNs where the Source is Registered

In some cases, a rumor initiator can initiate a rumor on more than one OSN to accelerate dissemination, increase visibility, and cover the maximum population of users on OSNs. Finding the initiator in each OSN where they have registered will require individual efforts for each OSN. So, finding the rumor initiator in one OSN and checking where all this user is registered is highly desirable. However, there may be a case where the user registered on multiple platforms has not initiated the rumor on all platforms. By checking where all this user is registered and the circulation of rumor on each OSN, one can initiate a possible rumor counterfeiting mechanism for each of the OSNs and perform an impact assessment of the rumor. Hence, the focus is on *one solution for all*. However, this assumption is limited to publicly available OSN data only, as user registration information is protected by privacy laws. Our ontology-based model supports this interoperability. Each *user* is associated with the *social network* class of ontology. This class has different OSNs as instances (E.g., Facebook, Twitter etc.) and a purpose for OSN (E.g., Friendship, Microblogging, respectively). Upon finding the rumor initiator in one OSN, we can check if the same person is registered on other OSNs or not. Also, if the person is registered on multiple OSNs, we can find out whether she has initiated the rumor on other OSNs also or not. Figure 3.5 explains this scenario. In the figure, *User 1* is registered on both *Facebook* and *Twitter* networks and is the initiator of a *Rumorous Post*, then using interoperable concepts like *User_Name*, we can find if she has initiated rumor on *Twitter* or not. The interoperable concepts of four different OSNs are shown in table 3.5.

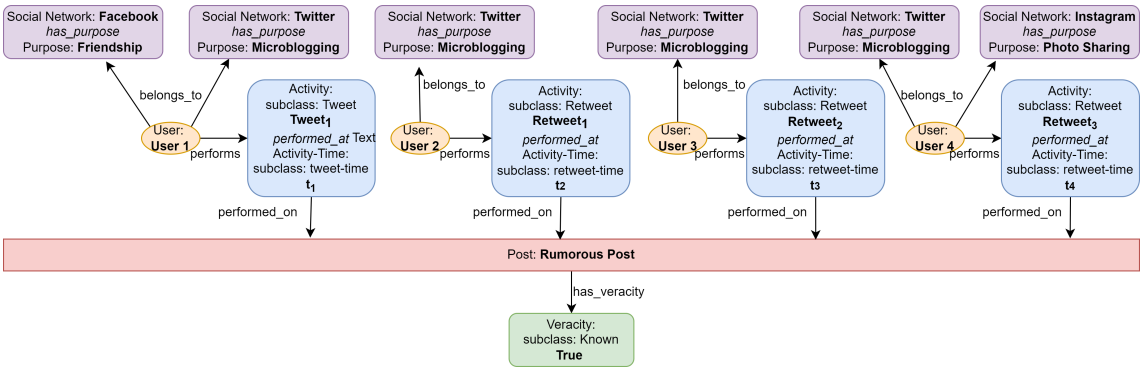

FIGURE 3.5: Propagation of a rumorous post with different activities performed by users at time t_1, t_2, t_3, t_4 where $t_1 < t_2 < t_3 < t_4$ and OSNs where each user is registered

3.4 Case Studies

3.4.1 Case Study 1: Detection of the rumor source initiator on Twitter

To demonstrate our proposed upper-level ontology model for identification of rumor initiators in OSNs, we show a case study of Twitter. Twitter is a popular social microblogging OSN that has millions of users worldwide. Facilitates large-scale information diffusion in real time. It is widely used to spread breaking news about current events, making it a popular breeding ground for spreading rumors [6]. The concepts of Twitter have been mapped to a lower-level ontology that conforms to the proposed upper-level ontology. Figure 3.6 shows the lower ontology model for Twitter derived from the upper ontology model. In the figure, the classes are represented by rectangular boxes with circles, and instances are represented by rectangular boxes with diamonds. The object-properties are represented by dashed colored arrows where the meanings of the colors are mentioned in the figure. Twitter ontology metrics are shown in table 3.6. Here, the metric column represents various ontology constructs like class, object-properties etc. and count denotes the number of each ontology construct.

We populate the lower-ontology model with rumor instances from the Pheme

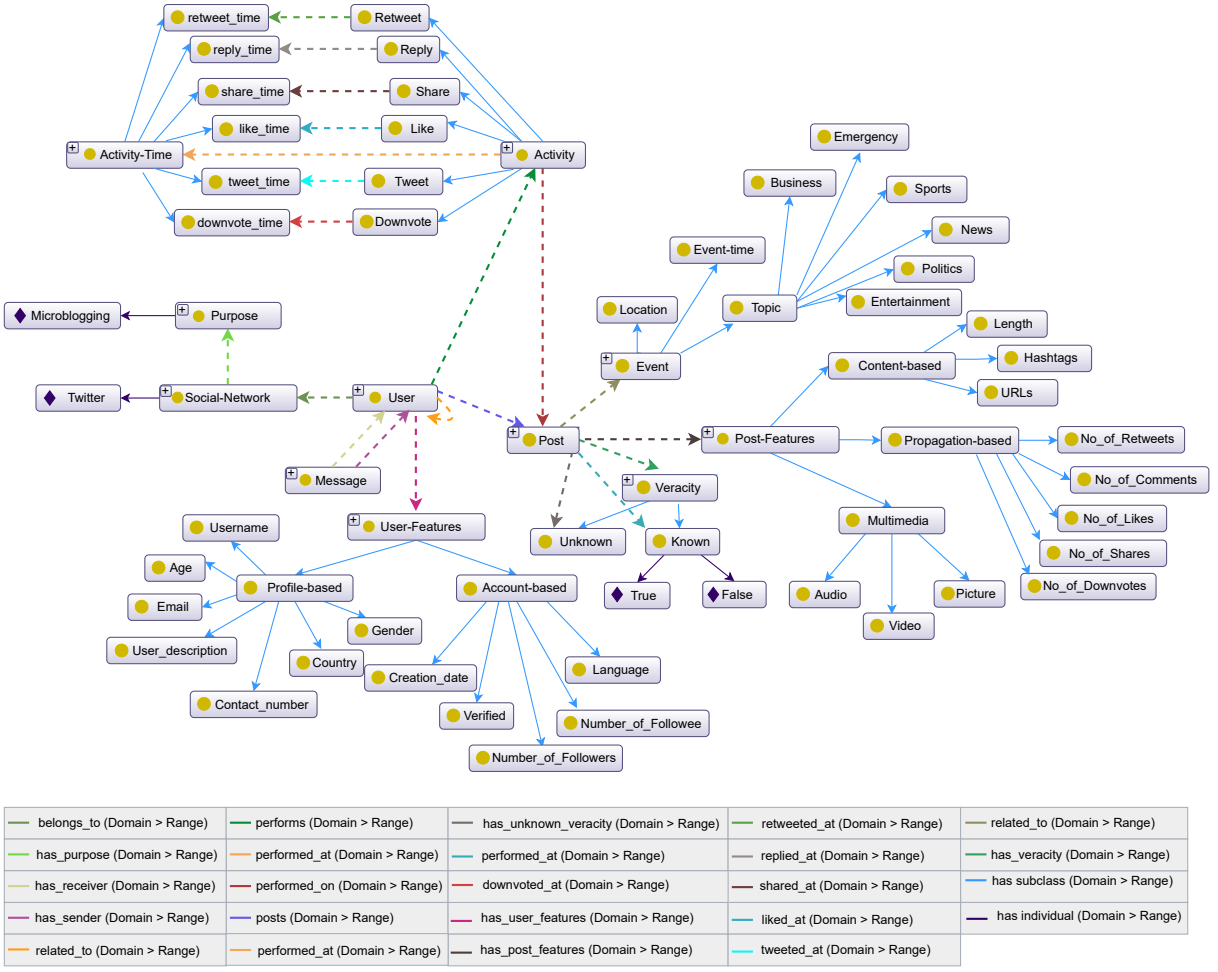

FIGURE 3.6: Lower Level Ontology Model for Twitter obtained using OntoGraf Plugin of Protégé

TABLE 3.6: Metrics of Lower Level Ontology for Twitter

Metric	Count
Class	62
Object Property	59
Data Property	7
Axioms	545
Logical Axiom	386
Declaration Axiom	159

dataset [90]. Pheme dataset comprises a set of Twitter rumors and non-rumors shared amid unfolding news situations. It encompasses rumors associated with nine distinct events, and each rumor is labeled with its corresponding veracity status, classified as true, false, or unverified. The data populated creates a knowledge graph which is queried to find the rumor initiators. The knowledge graph is in the form of RDF triplets. The consistency of the ontology is checked using the HermiT reasoner of Protégé.

In lower level ontology (see figure 3.6), user *Tweet*, *Retweet* or *Share* the post with respective *Tweet-Time*, *Retweet-Time* or *Share-Time*. Using the properties *tweeted_at*, *retweeted_at* and *shared_at*, we can backtrack to the very first user or group of users. The SPARQL query for finding the initiator of a rumorous post on Twitter is as follows:

```
SELECT ?User ?Post ?Avtivity ?ActivityTime WHERE{
        c: Post1 c:has_string ?Post.
        ?User c:performs ?A.
        ?A c:performed_on c:Post1.
        ?A c:has_string ?Activity.
        ?A c:performed_at ?T.
        ?T c:has_datetime ?ActivityTime.}
        ORDER BY ?ActivityTime
```

The result of the query is shown in table 3.7. In the table, the *User* field contains the Twitter User ID of the user, *Post* contains the content of rumor, *Activity* is Tweet and *ActivityTime* is the time at which tweet is tweeted. Here, User ID=”964926744” has tweeted a rumor which is retweeted by other twitter users. Since, the *ActivityTime* of this user is less than other users and he is solely *tweeting*, so this user is considered a rumor initiator in this case.

TABLE 3.7: Query Result

User	Post	Activity	ActivityTime
964926744	"Breaking news: Ghana international and AC Milan star Michael Essien has contracted Ebola, his club has confirmed."	"Tweet"	"Thu Nov 22 20:45:24 +0000 2012"

Similarly, when the veracity of the rumor is unknown, we first find the initiator of the post from the above query and then extract the user features and the post features for further analysis using the SPARQL query.

3.4.2 Case Study 2: Finding Rumor Source Initiator on Multiple OSNs

In this case study, we have created a synthetic RDF dataset comprising instances of 20 users, (*User1* to *User20*). Each user is registered on three OSN platforms- *Facebook*, *Twitter* and *Instagram*. A rumorous post *Rumor* with content *"This is a rumor!!"* is initiated by *User1* on all three platforms simultaneously with a small time difference. *User2* – *User20* are spreading this rumor according to the underlying activity on the respective OSN platforms. We can find the initiator of this post using the same SPARQL query as in Section 3.4.1 which is as follows.

```
SELECT ?User ?Post ?Activity ?ActivityTime
WHERE{
      cs: Rumor cs:has-string ?Post.
      ?User cs:performs ?A.
      ?A cs:performed-on cs:Rumor.
      ?A cs:has-string ?Activity.
      ?A cs:performed_at ?T.
      ?T cs:has_datetime ?ActivityTime.}
ORDER BY ?ActivityTime
LIMIT 1
```

This query results into detecting the initiator of the Rumor post as shown in Table 3.8.

TABLE 3.8: Query Result

User	Post	Activity	ActivityTime
User1	"This is a rumor!!"	"Tweet"	"025-05-23T07:03:06.45"

After finding the initiator, we can check what are the different platforms on which this user is registered and does she initiated the same rumor on other platforms as well using the following SPARQL query.

```
SELECT ?Post ?Platform ?Activity ?ActivityTime
WHERE{
      cs: Rumor cs:has-string ?Post.
      User1 cs:performs ?A.
      ?A cs:performed-on cs:Rumor.
      ?A cs:has-string ?Activity.
      ?A cs:performed-at ?T.
      ?T cs:has_datetime ?ActivityTime.
      User1 cs:belongs-to ?Platform.}
ORDER BY ?ActivityTime
```

This query results all the OSN platforms where User1 (spreading Rumor) is registed as shown in Table 3.9.

TABLE 3.9: Query Result

Post	Platform	Activity	ActivityTime
"This is a rumor!!"	"Twitter"	"Tweet"	"2025-05-23T07:03:06.45"
"This is a rumor!!"	"Facebook"	"Post"	"2025-05-23T07:03:09.24"
"This is a rumor!!"	"Instagram"	"Post"	"2025-05-23T07:03:10.12"

3.5 Proposed Ontology Model Evaluation and Analysis

To validate the design and applicability of the proposed ontology, a comprehensive evaluation has been carried out from both qualitative and quantitative perspectives. This section presents two approaches to assess the effectiveness of the ontology. First, a structured evaluation using the OQuaRE framework is applied to measure its quality in various dimensions such as structural soundness, functional adequacy, maintainability, etc. in Subsection 3.5.1. Second, an analytical comparison is made between the proposed ontology and existing OSN ontologies to highlight its advantages in terms of domain coverage, semantic richness, temporal reasoning, and support for multi-platform interoperability in Subsection 3.5.2. Together, these analyzes demonstrate the suitability of the ontology to support rumor detection and initiator tracking on online social networks.

3.5.1 Ontology Model Evaluation using OQuaRE Framework

The proposed ontology model to find the rumor initiator in OSNs is evaluated for quality and acceptability using the OQuaRE framework (as discussed in Section 2.4). We calculated various metrics for the proposed ontology and using these metrics, calculated the scores for different characteristics and sub-characteristics. Table 3.10 lists the calculated values and scaled values for various OQuaRE metics for the proposed ontology. Table 3.11 lists the characterisctics and sub-characteristics scores.

Figure 3.7 shows the scores for different characteristics and sub-characteristics. The results show the strengths and weaknesses of the ontology model. The global average score of our proposed ontology model is 4.24, which makes the ontology acceptable (as an OQuaRE score of 3 means the ontology is minimally acceptable,

TABLE 3.10: Calculation of OQuaRE metrics

Metric	Calculated Value	Scaled Value
LCOMOnto	2.48	4
WMCOnto	1.90	5
DITOnto	3	4
NACOnto	1.96	5
NOCOnto	1	5
CBOOnto	1.56	5
RFCOnto	2.88	5
NOMOnto	0.95	5
RROnto	0.5678	3
AROnto	2	2
INROnto	0.8225	5
CROnto	0.4516	3
ANOnto	3	3
TMOnto	4	4

and an OQuaRE score of 5 means exceeds expectation). However, significant improvements can be made to structural and functional adequacy having low values of 3.5 and 3.14, respectively. The structural score can be increased by increasing ANOnto and RROnto scores, i.e. annotation richness and properties richness, respectively. Similarly, functional adequacy scores can be increased by increasing ANOnto, RROnto, CROnto, AROnto scores i.e. annotation richness, properties richness, class richness and attribute richness, respectively.

3.5.2 Analytical Comparison of Proposed Ontology with Existing OSN Ontologies

To further establish the utility of the proposed ontology in the context of rumor source detection on OSN, an analytical comparison has been performed against several well-known existing OSN ontologies. This comparison focuses on critical dimensions such as domain specificity, temporal and veracity modeling, support for cross-platform interoperability, reasoning capabilities, and evaluation practice. The analysis highlights how the proposed ontology is structured to address limitations in

TABLE 3.11: OQuaRE Score Calculation

Characteristic	Sub-characteristic	Sub-characteristic score	Characteristic Score
Maintainability	Modularity	5	4.4983
	Reusability	3.16	
	Analyzability	4.66	
	Changeability	4.71	
	Modification Stability	4.80	
	Testability	4.66	
Structural	Formal Relations Support	3	3.5
	Tangledness	4	
	Cohesion	4	
	Redundancy	3	
Reliability	Recoverability	4.5	4.25
	Availability	4	
Operability	Learnability	4.83	4.83
Compatibility	Replaceability	4.75	4.75
Functional Adequacy	Controlled Vocabulary	3	3.14
	Inference	3	
	Consistent Search and Query	3.25	
	Knowledge Acquisition and Representation	3.75	
	Schema and Value Reconciliation	2.5	
	Clustering and Similarity	2.66	
	Indexing and Linking	3.33	
	Guidance and Decision Trees	3.66	
Transferability	Adaptability	4.75	4.75

existing OSN ontologies and offers enhanced semantic expressiveness, query power, and extensibility tailored to rumor source tracking. Table 3.12 summarizes these differences in key aspects.

3.6 Conclusion

This chapter aims to provide a semantic web-based solution for finding the rumor source initiator on OSNs using ontologies, which is the first objective of our thesis. Ontology serves a great purpose in our work as it provides machine intelligence for

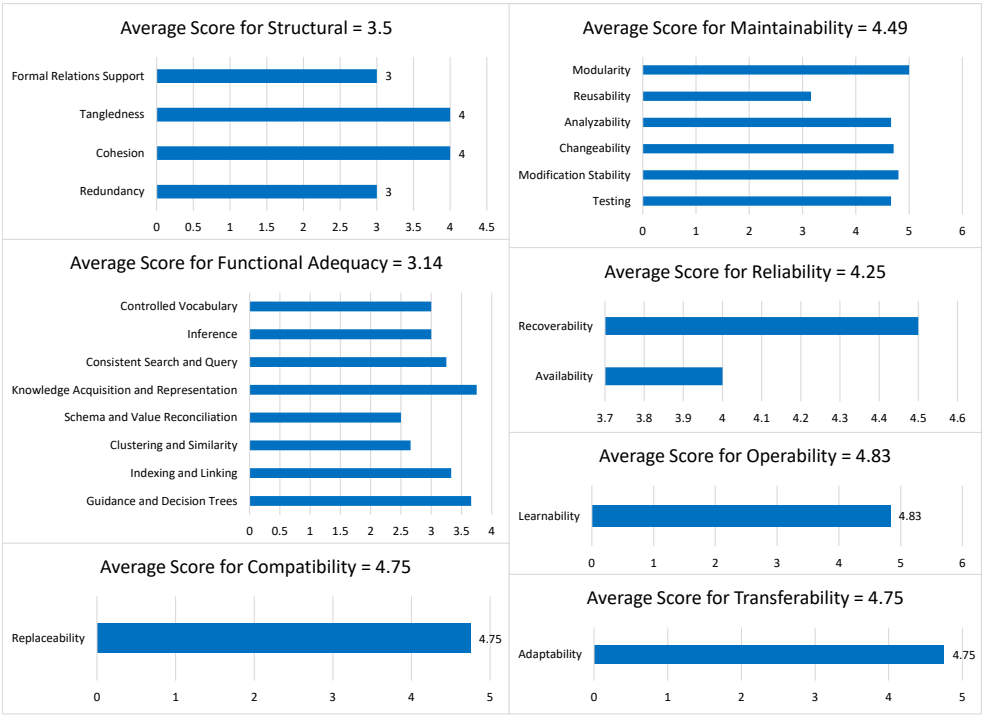

FIGURE 3.7: Ontology evaluation using the OQuaRE framework with global average score for proposed ontology = 4.244

automation, gives semantics and interoperability to the data, and can also be reasoned for. The model proposes an upper-level platform-independent generic ontology that lower-level vendor-specific applications can reuse and queried using SPARQL to find the initiator of posts whose veracity status is questionable. The model is verified using a popular OQuaRE framework for quality assessment and acceptability. The overall ontology score is 4.24, which makes our proposed ontology acceptable. The model introduces the concept of interoperability among different OSNs having semantically equivalent concepts. However, this concept is under-validated in our work. So, future work includes validating the model for different OSNs. In addition, there is scope for improving the structural and functional adequacy scores of the ontology by refining it.

TABLE 3.12: Comparison of Proposed Ontology with Existing OSN Ontologies

Point of Difference	Proposed Ontology	Existing OSN Ontologies
Purpose-Specific	Designed specifically for rumor initiator detection	Focused on P2P sharing of annotations [36], social tagging mechanism and affiliation networks [73], social network of researchers' community [37], security, privacy and access control mechanisms in social networks [38], POI recommendation in location based social networks [76], rumor modeling [39], rumor detection [40]
Key Feature	Supports inference and reasoning	Support representation
Temporal and Veracity Modeling	Explicitly models temporal classes and rumor veracity with *known* and *unknown* subclasses	Less temporal representation and no veracity representation
Temporal based reasoning	Explicitly modeled	No
Multi-platform Interoperability	Allows structured representation of multiple OSNs using an upper-level ontology	Lacks explicit support for cross-platform integration
Architecture	Three-layered	Mostly flat-layered
Structured Quality Evaluation	Yes (using OQuaRE)	No

In the subsequent chapters, we address the second challenging problem related to rumors, i.e., preventing the spread of rumors in the social network. Towards this objective of the thesis, we propose two models based on two rumor countering strategies- blocking rumors and counter-rumor diffusion.