2013 27th International Conference on Advanced Information Networking and Applications Workshops

A Machine Learning Algorithm for Searching Vectorised RDF Data

Asaad Sabah Hadi[1], Paul Fergus[2], Chelsea Dobbins[2], Abbas Muhsin Al-Bakry[1]

University of Babylon[1],
College of Information Technology,
Software Department,
Babylon, Hilla, P.O.
Box 4, Iraq.

Applied Computing Research Group[2],
School of Computing and Mathematical Sciences
Liverpool John Moores University,
Byrom Street,
Liverpool, L3 3AF

{Assadsabah, abbasmoh67}@uobabylon.edu.iq; {p.fergus,a.hussain, d.aljumeily}@ljmu.ac.uk, c.m.dobbins@2006.ljmu.ac.uk

Abstract—**The Internet has fundamentally changed the way we collect, access, and deliver information. However, this now means that finding the exact information we need is a significant problem. While search engines can find information based on the keywords we provide, using this technique alone is insufficient for rich information retrieval. Consequently, solutions, which lack the understanding of the syntax and semantics of content, find it difficult to accurately access the information we need. New approaches have been proposed that try to overcome this limitation by utilising Semantic Web and Linked Data techniques. Content is serialised using RDF, and queries executed using SPARQL. This approach requires an exact match between the query structure and the RDF content. While this is an improvement to keyword-based search, there is no support for probabilistic reasoning to show how close a query is to the content being searched. In this paper, we address this limitation by converting RDF content into a matrix of features and treat queries as a classification problem. We have successfully developed a working prototype system to demonstrate the applicability of our approach.**

Keywords –Semantic Web, Linked Data, RDF, SPARQL, Matrix, Vectorisation, Machine Learning, and Classification

I. INTRODUCTION

A consequence of living in the digital age is the abundance of information that is available. In today's society, it is common practice to capture, store, upload and share almost every moment of daily life. Sensors, embedded in everyday objects, are also capable of connecting to the Internet and providing useful information, without user intervention; thus resulting in "information overload". This vast amount of data is growing every day. Providing an intelligent way of searching this data has led to the development of the Semantic Web, Web 3.0 applications and linked data. This new generation of decentralized knowledge management enhances information flow with "machine–processable" metadata [1]. Information, from distributed data sources, can be linked, in order to add more "meaning" to the data. It is this mash-up at the data level, rather than the application level, that has led to the phrase "Web 3.0" being coined [2]. Creating these "links" between objects is fundamental to these applications. This is achieved using the Resource Description Framework (RDF), which provides a means to link data from multiple websites or databases together, and is the basis of Web 3.0 applications [2]. This collection of interrelated datasets can also be referred to as Linked Data [3]. Bizer et al. [4] summarize linked data as being, "simply about using the Web to create typed links between data from different sources". Linked Data provides a way to fuse data, about entities from different sources, together and to crawl the data space, as the data is connected by links [5]. It is this idea that is fundamental to current work, as distributed sources of information are brought together, searched and linked, to access the information we require.

In order to search Linked Data the generated information first needs to be transformed into RDF tuples. This allows items, from different data sources, to be queried and linked together. In order to search the RDF documents the current method used is SPARQL, and tools such as ARC2 (an implementation that allows RDF/XML files to be parsed, serialized and stored [6]). SPARQL [7], is a query language for RDF documents, and at present, is used to search the RDF documents and execute the queries. SPARQL enables values to be pulled from both structured and semi-structured data; it can explore data by querying unknown relationships; complex joins, of disparate databases, are able to be performed, in a single and simple query, and RDF data can be transformed from one vocabulary to another [8].

Linked Data and SPARQL have provided significant improvements over existing search methods, which are designed to process classic Web content (HTML pages that comprise both presentation instructions and data). However, whilst these methods are improving the way in which content can be searched, they do have a considerable drawback. SPARQL queries need to be carefully constructed to match RDF elements – the result returned is either true or false. This approach does not allow for the estimation of how close the query is to the content in the RDF documents. For example, describing the features of a monkey might not be specific enough to identify a Capuchin monkey; however, using a probabilistic approach would be capable of retrieving different types of monkeys, which may contain the Capuchin type. Achieving this with SPARQL alone remains challenging, due to the preciseness of the syntax in the query and the content being searched.

978-0-7695-4952-1/13 $26.00 © 2013 IEEE
DOI 10.1109/WAINA.2013.204

This paper explores this idea further and considers an approach that converts RDF tuples into a matrix representation. This allows us to treat the searching of RDF documents as a classification problem, based on the features defined in a vector object. In other words, using machine learning each search instance is positioned within the density distribution in the matrix. Information is retrieved based on the closeness parameters defined between matrix, instances and search objects (search vector instance).

II. Linked data

Linked Data (also known as the Semantic Web) provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries [9]. It enables intelligent search instead of keyword matching, query answering instead of information retrieval, document exchange between departments via ontology mappings, and definition of views on documents [10]. The heart of the Semantic Web lies in linking data together from different sources. The term Linked Data refers to a set of best practices for publishing and connecting structured data on the Web [4] and is essential in connecting data across the semantic web [11]. Linked Data relies on documents containing data in Resource Description Framework (RDF) format [11], a model for describing resources [12]. The following is a brief overview of this area.

The Linking Open Data community project [36] is the most noticeable example of the implementation of the semantic web. The project's aim is to bootstrap the Web of Data, by identifying existing data sets that are available, and publishing them on the Web [11]. The data sets are distributed as RDF and RDF links are set between data items from different data sources [13]. This project has been incredibly successful. As of September 2011 there were, collectively, 295 data sets, consisting of over 31 billion RDF triples, interlinked by approximately 504 million RDF links [13].

One such application that has come out of the Linking Open Data community project has been DBpedia [14]. This project "focuses on the task of converting Wikipedia content into structured knowledge, such that Semantic Web techniques can be employed against it". DBpedia has been very successful, with 4.7 billion interlinked RDF triples residing [15]. This project has also been extended with the implementation of DBpedia Mobile [15]. The mobile version "allows users to access information about DBpedia resources located in their physical vicinity, from where they can explore links to other resources on the Semantic Web" [15]. This work is of particular interest because of its success in linking data from varied resources together and that data is presented that is in the same proximity as the user.

In contrast, SPITFIRE, takes the idea of the semantic web further by "integrating Internet–connected sensors into the Semantic Web of Things". In this context, providing, "a "machine-understandable" description of sensors and the data they produce" [16]. This work is of significant importance because, when building rich information sources, incorporating as much data from the physical environment as well is vital. However, sensor data tends to be ambiguous; therefore overcoming this challenge is a big step into integrating data from the environment into rich information stores. If sensor data can be "understood" then incorporating this data would produce richer information; thus enabling "smarter" searches to also be performed on the data.

III. Machine Learning

The use of computer algorithms and visualization techniques are considered fundamental to support the analysis of datasets, commonly referred to as Big Data [17]. More recently, such techniques have been used extensively within the medical domain. One example of this is the Common Spatial Patterns (CSP) algorithm. This was proposed by Woon et al. and has been successfully used to study Alzheimer's [18]. In other studies, Latchoumane et al., analyse EEG (electroencephalogram) signals using Multi-way Array Decomposition (MAD). This is a supervised learning process for evaluating multidimensional and multivariate data like EEG [19].

Multi-Layer Perceptrons (MLP) and Probabilistic Neural Networks (PNNs) have featured widely in research to process and analyse medical datasets. MLPs are feed-forward networks that work with back-propagation learning rules. PNNs are similar to MLPs, in this way, and consist of three layers; an input layer, radial basis layer, and a competitive layer. This type of feed-forward network operates using the Parzen's Probabilistic Density Function (PDF). In terms of overall performance, PNN networks perform slightly better than PML networks [20].

The primary goal of such algorithms is to extract meaning from potentially huge amounts of data. Features, associated with particular data, such as datasets that contain data about neurodegenerative diseases, are characterized. This has led to a great deal of work in feature extraction, within datasets. One example of this is the Discrete Cosine Transform (DCT) algorithm that decreases the number of features and the computation time when processing signals. DCT is used to calculate the trapped zone, under the curve, in special bands [21].

Similar algorithms have been used to predict heart disease using Decision Trees, Naïve Bayes and Neural Networks. The results show that, using the lift chart for prediction and non-prediction, the Naïve Bayes algorithm predicted more heart disease patients than both the Neural Network and Decision Tree approaches [22]. Using data collected from patients suffering with Alzheimer's, Joshi et al., were able to identify the various stages of Alzheimer's. This was achieved using neural networks, multilayer perceptrons, including the coactive neuro-fuzzy inference system (CANFIS) and Genetic Algorithms [23]. The results showed that CANFIS produced the best classification accuracy result (99.55%) as compared to C4.5 (a decision tree algorithm).

Other algorithms, such as dissimilarity based classification techniques, have proven to be very useful for analysing datasets. For example, algorithms, such as the k-nearest neighbour classifier (k-NN), and Linear and Quadratic normal density based classifiers, have been extensively used to classify seismic signals. Nonetheless, the results have shown that Bayesian (normal density based) classifiers outperform the k-NN classifier, when a large number of prototypes are provided.

Within the medical domain, dealing with big datasets is not unusual. For example, in pharmocogentics 5Tb files are often used. However, dealing with a file this big is still a significant challenge. Therefore, a great deal can be learnt from the research efforts carried out on medical dataset analysis.

IV. Framework Architecture

Building on the advances made in the Semantic Web, and Machine Learning, the algorithm posited in this paper has been designed to facilitate information extraction from semantic metadata. The information is transformed into a matrix of object instances, with associated features to enable probabilistic searches. The metadata serializations provide rich semantic data structures that describe information. The algorithm is domain agnostic and is generic enough to work on metadata structures that describe different information. This is a key feature within the approach that mitigates the need to fully understand a domain before queries can be constructed. The approach treats the search of data as a classification problem. In other words, the features of the query are described rather than the query itself. Figure 1 describes the process, and below a more detailed description of each stage is presented.

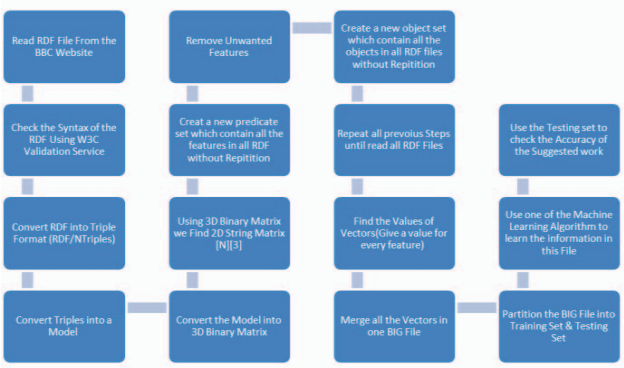

Fig 1. Platform System Design

The process begins by reading all data sources required within the final matrix and loading them into memory. Each of the metadata sources are validated to ensure they conform to the required metadata specification, for example, if RDF is used, then each source would have to validate according to the rules for constructing RDF documents. During the pre-processing stage data is converted into tuples consisting of a subject-predicate-object instance. Once pre-processing has been completed each of the tuples are loaded into a metadata model ready for post data processing. This allows each of the unique features to be extracted. This result is a vector instance that forms part of the matrix representation of the metadata file. These sets of processes continue until all the required data sources (files) have been processed. Once a matrix for each data source has been generated they are merged into a single matrix.

At this point, the matrix is then ready to be loaded into any machine-learning tool, e.g. Matlab, Octave, R or even sklearn (a python machine learning API). The details of this process will be discussed in more detail in the next section.

I. Framework Implementation

A prototype system has been developed to implement the design stages discussed in the previous section. More specifically, we have developed a Java application that utilises the Jena Semantic Web API for processing RDF. In order to test our system, RDF documents, produced by the BBC Nature website, that describe animals have been used. In the next section, the algorithm is presented, which moves towards the goal of performing probabilistic searches on semantic content.

A. Technical Details

The algorithm utilises RDF as a semantic notation for capturing structured information. Twenty RDF documents are used from the BBC Nature website, which describe different kinds of mammals. Figure 2 below provides an excerpt from an RDF document that describes a Jaguar.

```
<owl:sameAs rdf:resource="http://dbpedia.org/resource/Jaguar"/>
<wo:adaptation rdf:resource="/nature/adaptations/Altricial#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Ambush_predator#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Camouflage#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Carnivore#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Polymorphism_(biology)#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Predation#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Territory_(animal)#adaptation"/>
<wo:adaptation rdf:resource="/nature/adaptations/Vivipary#adaptation"/>
<wo:livesIn rdf:resource="/nature/habitats/Flooded_grasslands_and_savannas#habitat"/>
<wo:livesIn rdf:resource="/nature/habitats/Mangrove#habitat"/>
```

Fig 2. RDF Excerpt for Jaguar

Each RDF file used is loaded using the Jena API, and the syntax is validated and loaded into a model using the ModelFactory.createDefaultModel() method. This model is then used to convert the RDF serialisation into a RDF N-Triples format. Again, an excerpt can be seen in Figure 3.

```
<file:/nature/class/Mammal#class> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://purl.org/ontology/wo/Class> .
<file:/nature/class/Mammal#class> <http://www.w3.org/2000/01/rdf-schema#label> "Mammalia" .
<file:/nature/family/Felidae#family> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://purl.org/ontology/wo/Family> .
<file:/nature/family/Felidae#family> <http://www.w3.org/2000/01/rdf-schema#label> "Felidae" .
<file:/nature/genus/Panthera#genus> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
<http://purl.org/ontology/wo/Genus> .
<file:/nature/genus/Panthera#genus> <http://purl.org/ontology/wo/species>
<file:/nature/life/Jaguar#species> .
```

Fig 3. RDF N-Triples

The model is then converted into a three dimensional binary matrix where the first dimension is the predicate, the second is the subject, and the third is the object. For each predicate (Feature), there is a two dimensional binary matrix [subject, object] that represents all the content in the RDF file. The binary matrix is converted into a two dimensional matrix [N x 3] containing string elements where N is the number of

statements in the RDF file and the number 3 represents the subject, predicate, object values. For example, matrix [][0] denotes the subject and matrix [], [1] denotes the predicate and finally matrix [][2] denotes the object.

Once each of the required RDF files, i.e. all the mammal RDF documents, have been converted into a corresponding matrix representation, a list of unique predicate values are extracted, which are found in all the RDF files. For instance, if 20 mammal RDF files are processed then this would yield 21 distinctive predicates. However, this does result in predicates that are not of any use in the classification stage. For example, predicates such as type, title, subject and label are of little use in this stage. Therefore, they are removed. The remaining features form the basis for creating blank object instances. For each of the object instances, values are assigned to each of the features in the object vector. Note that a particular mammal may have numerous object instances because specific features may have multiple values. An excerpt of the final matrix containing several object instances is illustrated in Figure 4. This matrix provides a one-to-one mapping with the information contained in the RDF document(s). This means that the algorithm can also return the matrix back to its original RDF representation.

```
[Jaguar,979,17,10,171,977,974,30,975,224,961,,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,17,10,23,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,17,10,24,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,17,10,169,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,17,10,13,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,17,10,168,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,683,10,171,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,683,10,23,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,683,10,24,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,683,10,169,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
[Jaguar,979,683,10,13,977,974,30,975,224,961,972,973,213,61,970,82,971,-1,978,-1,-1]
```

Fig 4. Matrix for RDF information on a Jaguar

Note that the value-1 in the matrix means that the particular object instance does not contain information for that specific feature. Once each of the RDF files have been processed, according to the above processes, they are all merged into a single file, i.e. a file that contains a matrix representation for information about all the different mammals that were processed. This file is used to form a single dataset for use in the classification, which will be discussed in more detail in the following section.

V. Evaluation

This section presents the results from experiments that were performed on a set of matrix data. This set represented data from ten RDF files, from 10 mammals. Each RDF file contained 100 vectors, which contained different feature combinations that best described a particular mammal. A sample of such a matrix containing several vectors and associated features is illustrated in figure 4 above. Each of the matrix files were merged to contain a single matrix that contained all ten mammal. This was achieved using the Linux command cat * > mammal_vector.csv. This resulted in a matrix that contains 1000 vector objects.

The combined matrix of data is imported into Matlab and random permutations of the vector objects are performed, i.e. the 1000 vector objects were shuffled. A single matrix, containing the features, was generated from the original matrix. A single corresponding column vector, containing the class names, was also produced. Using the feature and class label matrix, a dataset was created using the dataset function provided by PRTools. To validate the performance, of the classification algorithms used, this dataset was split into a training set, and a test set. The training set contained 200 samples, from the dataset, and the test set contained the remaining 800. Using the training set, the Polynomial, Logistic, kNN, Decision Tree, Parzen, Support Vector and Naive Bayes classifiers were trained. Using the true class labels, obtained from the test set, the estimated class labels were obtained, and plotted in a confusion matrix. All the classifiers provided 100% classification, except for the Decision Tree classifier which provided 98% classification. The results from the confusion matrix can be seen in Figure 5.

True Labels \ Estimated Labels	1	2	3	4	5	6	7	8	9	10	Totals
1	80	0	0	0	0	0	0	0	0	0	80
2	0	80	0	0	0	0	0	0	0	0	80
3	0	0	80	0	0	0	0	0	0	0	80
4	0	0	0	80	0	0	0	0	0	0	80
5	0	0	0	0	80	0	0	0	0	0	80
6	0	0	0	0	0	80	0	0	0	0	80
7	0	0	0	0	0	0	80	0	0	0	80
8	0	0	0	0	0	0	0	80	0	0	80
9	0	0	0	0	0	0	0	0	80	0	80
10	0	0	0	0	0	0	0	0	0	80	80
Totals	80	80	80	80	80	80	80	80	80	80	800

a) Results for the Polynomial, Logistic, kNN, Parzen, Support Vector, and Naïve Bayes Classifiers

True Labels \ Estimated Labels	1	2	3	4	5	6	7	8	9	10	Totals
1	76	0	4	0	0	0	0	0	0	0	80
2	0	80	0	0	0	0	0	0	0	0	80
3	3	0	77	0	0	0	0	0	0	0	80
4	0	0	0	80	0	0	0	0	0	0	80
5	2	0	0	0	78	0	0	0	0	0	80
6	1	0	0	0	1	78	0	0	0	0	80
7	0	0	0	0	0	0	80	0	0	0	80
8	0	0	0	0	0	0	0	80	0	0	80
9	0	0	0	0	0	0	0	0	80	0	80
10	0	0	0	0	0	0	0	0	0	80	80
Totals	82	80	81	80	79	78	80	80	80	80	800

b) Results for the Decision Tree Classifier

Fig 5. Confusion Matrix

In order to visually determine the performance of a classifier, the receiver operating characteristic (ROC) was used. Using PRTools, the false negatives and false positives are plotted, as illustrated in Figure 6. As the results show, all of the classifiers performed very well and only the Decision Tree classifier making several misclassifications. The following section provides a detailed discussion on the reasons why the results were so high.

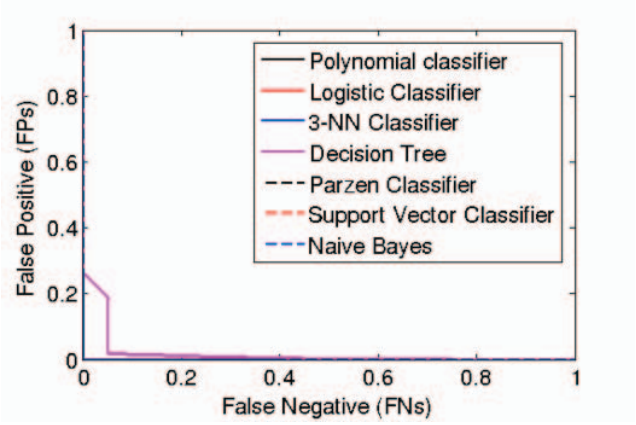

Fig 6. Confusion Matrix

A. Discussion of Results

In this paper, RDF data from the BBC Nature website was converted to a matrix to allow probabilistic searches based on machine learning algorithms. In other words, searching RDF data is treated as a classification problem. The observed data was then run through several well-known classification algorithms. The results show an accurate classification of mammals, described using 22 features that are common to all mammals. The Polynomial, Logistic, kNN, Decision Tree, Parzen, Support Vector and Naive Bayes classifiers provided 100% classification, and the Decision Tree provided 98% classification. Several other classifiers were tested that included the normal densities based linear classifier; normal densities based quadratic classifier, and the normal densities based classifier (independent features). However, all three of these classifiers produced poor results.

The reason the linear based classifiers performed worse was because it is not possible to separate the ten classes, i.e. the problem is clearly a non-linear problem. Using the non-linear classifiers, separating the ten classes is possible by weaving a separation divide between the different classes, which is not possible using linear classifiers due to the linear divide. Better results would have been obtained if only two classes were used.

Figure 7, below, demonstrates the separation between the different classes. Using the nonlinear classifiers, it also illustrates why such a high positive result was possible. The figure clearly shows that there is not an overlap between the ten distinct animals that were used in the evaluation.

This paper provides a novel method for searching RDF data and provides a generic solution that takes full advantage of different knowledge domains. Nonetheless, further research is required. The exceptionally high classification results were possible because there was a clear separation between the various animals in the study. Therefore, it would be interesting to run a related experiment using animals that are very similar, for example, between all of the big cats, to see if this separation is as distinct. Furthermore, a much bigger dataset is required to fully understand the approach. This would also be helpful in evaluating its usefulness on big datasets that are comprised of hundreds of thousands of vectors, within the matrix space. In addition, it would be very useful to use other domain knowledge, such as DBPedia and evaluate how well different feature sets can be found.

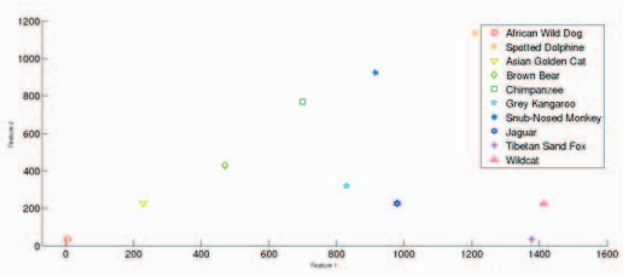

Fig 7. Separation of Classes

Another issue, which needs to be explored further, is the definition of the feature space itself. For the mammal experiment, 22 features, specific to mammals, were used. However, it was not clear at this time, whether all of these features are required. The initial thought was that only a subset of these would be required to sufficiently separate classes. These concerns will be the focus of further future research.

Another direction of future work will focus on how to best describe and combine features, at the application level, to collect search criteria from the user. This will involve an investigation into how they can be applied over different classifications of information. For example, as in the case of mammals and reptiles, which have a diverse feature set length.

VI. Conclusions

The World Wide Web has become a huge information space, with many of today's devices contributing to the amount of data and information it contains. While this was performed in a crude unconstrained way, current approaches are opting to structure data in a more meaningful way to support machine-processible semantics. Using RDF, data is described and linked in a much more informative way. This allows it to be processed in a more formal way. Languages, such as SPARQL, now provide similar capabilities to SQL and database management systems. This has made it easier to store and find the information that we require. However, the process is formal and requires a clear description of queries that precisely match the structures of the RDF – the query is either matched or not matched. A better alternative would be to provide an indication about what information is close to the query, if not exactly matching.

This paper explores this idea and proposes a probabilistic approach that treats the querying of RDF data as a classification problem. RDF data is flattened into a matrix format, which describes classes of mammals with an associated set of features. Rather than building complex SQL-type query, a set of features are defined and used to classify the feature space. Then, using a training set, their features and corresponding classes are described. A successful working prototype algorithm was developed, and evaluated, using several classification algorithms. The results are positive and illustrate how a nonlinear classifier performs. The results provided a 100% classification, except for the Decision Tree classifier that provided a 98% classification. The same dataset was also applied to classifiers; however, the results were poor.

The reason for such good results was due to the clear separation of mammals used in the test set. Nonetheless, our initial results are encouraging and provide the base for a much more in-depth investigation.

References

1. Zhou, L., L. Ding, and T. Finin, *How is the Semantic Web Evolving? A Dynamic Social Network Perspective.* Computers in Human Behaviour, 2011. **27**(4): p. 1294-1302.
2. Hendler, J., *Web 3.0 Emerging.* Computer, 2009. **42**(1): p. 111-113.
3. W3C. *Linked Data*. 2010 [cited; Available from: [Online] Available http://www.w3.org/standards/semanticweb/data.
4. Bizer, C., T. Heath, and T. Berners-Lee, *Linked Data - The Story So Far.* International Journal on Semantic Web and Information Systems, 2009. **5**(3): p. 1-22.
5. Bizer, C., *The Emerging Web of Linked Data.* IEEE Intelligent Systems, 2009. **24**(5): p. 87-92.
6. Nowack, s.B. *Easy RDF and SPARQL for LAMP Systems*. 2012 [cited; Available from: [Online] Available: https://github.com/semsol/arc2/wiki.
7. W3C. *SPARQL Query Language for RDF*. 2008 [cited; Available from: [Online] Available: http://www.w3.org/TR/rdf-sparql-query/.
8. Feigenbaum, L. and E. Prud'hommeaux. *SPARQL By Example: A Tutorial*. 2010 [cited; Available from: [Online] Available: http://dig.scail.mit.edu/2010/Courses/6.898/resources/sparql-tutorial.pdf.
9. W3C. *W3C Semantic Web Activity*. 2011 [cited; Available from: [Online] Available: http://www.w3.org/2001/sw/.
10. Fensel, D., et al., *Semantic Web: Why,What, and How*. 2011: Springer, Berlin Heidelberg. 1-653.
11. Berners-Lee, T. *Linked Data*. 2009 [cited; Available from: [Online] Available: http://www.w3.org/DesignIssues/LinkedData.html.
12. Miller, E., *An Introduction to Resource Description Framework.* Bulletin of the American Society for Information Science and Technology, 1998. **25**(1): p. 15-19.
13. W3C. *Linking Open Data W3C SWEO Community Project*. 2011 [cited; Available from: [Online] Available: http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData.
14. Auer, S., et al. *DBpedia: A Nucleus from a Web of Open Data*. in *ISWC*. 2007.
15. Bizer, C., et al., *DBpedia - A crystallization point of the Web of Data.* Web Semantics: Science, Services and Agents on the World Wide Web, 2009. **7**(3): p. 154-165.
16. Pfisterer, D., et al., *SPITFIRE: Toward a Semantic Web of Things.* IEEE Communications Magazine, 2011. **49**(11): p. 40-48.
17. Trelles, O., et al., *Big data, but are we ready?* Nature Reviews Genetics, 2001. **12**(224): p. 647-657.
18. Woon, W.L., et al., *Techniques for early detection of Alzheimer's disease using spontaneous EEG recordings* IOP Publishing 2007. **28**(2007): p. 335-347.
19. Latchoumane, C.F.V., et al., *Multiway array decomposition analysis of EEGs in Alzheimer's disease.* Journal of Neuroscience Methods, 2012. **207**(1): p. 41-50.
20. Ispawi, D.I., N.F. Ibrahim, and N.M. Tahir. *Classification of Parkinson's disease based on Multilayer Perceptrons (MLP) Neural Networks and ANOVA as a feature extraction*. in *8th IEEE Conference on Signal Processing and its Applications*. 2012: IEEE Explore.
21. Whitwell, J.L., et al., *Neroimaging correlates of pathologically defined subtypes of Alzheimer's disease: A case-control study.* The Lancet Neurology, 2012. **11**(10): p. 868-877.
22. Palaniappan, S. and R. Awang. *Intelligent heart disease prediction system using data mining techniques*. in *IEEE Computer Systems and Applications*. 2008: IEEE Explore.
23. Joshi, S., V. Simha, and D. Shenoy, *Classification and treatment of different stages of Alzheimer's disease using various machine learning methods.* International Journal of Bioinformatics Research, 2010. **2**(1): p. 44-52.