9 Software Similarity Searching and Classification

The ultimate problem of this book is to search for similar software to our query from a database and to classify a program as belonging to a particular class. This chapter examines how we transform the pair-wise similarity problem into a similarity search problem over a database. Moreover, we examine statistical classification of birthmarks to identify the class of software it belongs to.

Keywords: Software similarity search, software classification, similarity search, instance-based learning, nearest neighbour, metric trees, locality sensitive hashing, kernel methods.

9.1 Instance-based Learning and Nearest Neighbour

Instance-based learning is a form of machine learning used in classification. To classify an object, it is compared to known instances of that object. If the query is similar to a known instance, or alternatively closest to an instance, known as its nearest neighbour, then it is classified as belonging to the same class. Nearest neighbour and range searches are the fundamental basis for software similarity using software features. If a piece of software represented as an object is in very close range or distance to known software instances, then it is declared a variant.

9.1.1 k Nearest Neighbours query

Definition 9.1. Given a set of objects P and a query Q, and an integer k > 0, the k nearest neighbours (kNN) query is to find a result set kNN that consists of k objects such that for any $p \in (P - kNN)$ *and any* $p' \in kNN, dist(p', q) \leq dist(p, q)$

9.1.2 Range query

Definition 9.2. Given a set of objects P and a query Q, and a range r > 0, the range query is to find a result set rNN that consists of objects such that for any $p' \in rNN, dist(p', q) \leq r$

9.1.3 Metric Trees

Metric trees allow similarity searches (nearest neighbour and range searches) for objects that have a metric distance function. A number of algorithms have been proposed such as BK Trees [1], Vantage Point trees [2], M-Trees [3], Slim trees [4], or DBM Trees [5]. Metric access methods can be categorized by different qualities such as whether the data structures allow for efficient insertion and deletion of objects allowing for dynamic access, or whether the data structures are kept in main memory or on disk.

9.1.4 Locality Sensitive Hashing

Locality sensitive hashing [6] is a scheme whereby similar objects are hashed to the same buckets. This allows a similarity search to perform nearest neighbour searches by hashing.

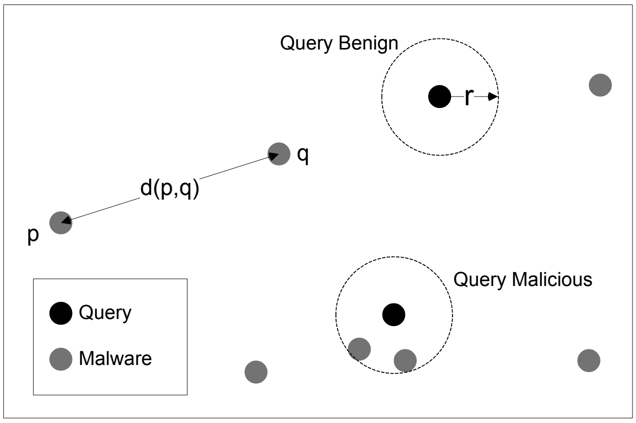

Fig. 9.1 The software similarity search to detect malware.

Definition 9.3. *Let d be a metric distance function. Let* $B(v,r) = \{q \in X \mid (v,q) \leq r\}$. *A family* $H = \{h : S \to U\}$ *is called* $\{r_1,r_2,p_1,p_2\}$ *sensitive for D if for any* $v, q \in S$

- *If* $v \in B(q,r_1)$ *then* $\Pr_H[h(q) = h(v)] \geq p_1$
- *If* $v \notin B(q,r_2)$ *then* $\Pr_H[h(q) = h(v)] \leq p_2$

In order of a locality-sensitive hash (LSH) family to be useful, it has to satisfy inequalities p1 > p2 and r1 < r2.

9.1.5 Distributed Similarity Search

Scalability becomes a problem when database sizes increase. For example, malware databases have been growing exponentially [7] and efficient algorithms are required to handle the problem. Distributed algorithms are one solution to scale similarity searches. Distributed metric space similarity search algorithms include M-Chord [8] and GHT* [9,10]. An approach based on Locality Sensitive Hashing is proposed in [11]

9.2 Statistical Machine Learning

Statistical classification is the process of assigning objects to classes. A typical example is the malware classification problem which is the process of assigning an unknown executable to the class of malicious or non malicious software.

Machine learning can be supervised or unsupervised. In the unsupervised model, none of the objects are labelled, and their class designation is unknown. The usual approach is to perform clustering to identify separate classes. In the supervised approach, a training set of data is labelled and used to build a model of classes in relation to their characteristics. After training, the system classifies unlabelled data and determines their classes.

Statistical classifiers include the popular and efficient Bayesian classifiers. Artificial Neural Networks (ANN) are another popular choice. The classifiers can also

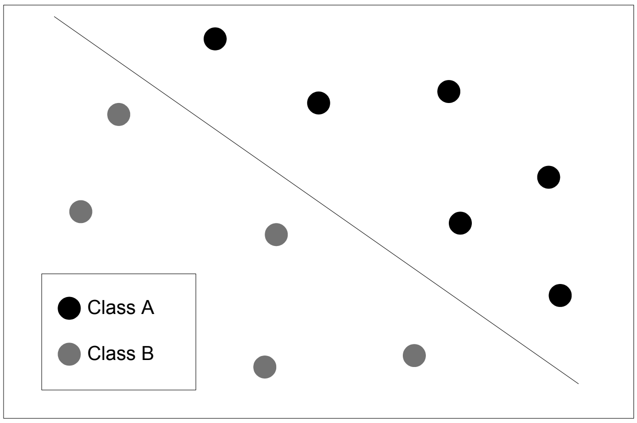

Fig. 9.2 A linear classifier separating two classes.

be grouped into linear and non linear systems. In a linear classifier, the input space can divide the classes using hyperplanes.

Vectors are used in many machine learning algorithms so often it is most useful to represent software as feature vectors. Features that are extracted from software can be used to construct feature vectors. Kernel machines provide an alternative approach to using feature effects and the most popular kernel method based classifier is the Support Vector Machine [12]. In this approach, a kernel for a particular object must be constructed. For classification of objects such as graphs, a variety of graph kernels can be used.

9.2.1 Vector Space Models

In the vector space model, a feature vector is constructed in $\mathbb{R}^n$ and classes are separated by partitioning over that space. The original feature vectors may have a high dimensionality, but in reality many of these features may be of low importance or redundant. Dimensionality reduction reduces the size of the feature vector.

9.2.2 Kernel Methods

The most well known kernel based classifier is the support vector machine (SVM) [12]. It is a linear classifier and works by constructing a hyperplane that maximally separates the margins between each class.

9.3 Research Opportunities

Nearest neighbour searches using metric distance functions to perform similarity searches has been employed in some malware detection literature. Much existing literature on software similarity has only focused on pairwise similarity and ignored the indexing and searching problem. Opportunities exist to transfer existing techniques into metric indexing methods.

Locality sensitive hashing also represents an opportunity as this indexing and searching technique has not been employed in all areas such as malware detection. Likewise, distributed similarity search algorithms are still to be exploited in the domain of software similarity.

The use of kernel methods for graph and tree based features is an area which is unexplored. The use of graph kernels to enable graph based classification presents much opportunity for researchers in future work.

References

1. Baeza-Yates R, Navarro G Fast approximate string matching in a dictionary. In: South American Symposium on String Processing and Information Retrieval (SPIR'98), 1998. pp 14-22
2. Peter NY Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms, Austin, Texas, United States, 1993. Society for Industrial and Applied Mathematics, pp 311-321
3. Paolo C, Marco P, Pavel Z (1997) M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. Paper presented at the Proceedings of the 23rd International Conference on Very Large Data Bases,
4. Caetano Traina, Jr., Agma JMT, Bernhard S, Christos F (2000) Slim-Trees: High Performance Metric Trees Minimizing Overlap Between Nodes. Paper presented at the Proceedings of the 7th International Conference on Extending Database Technology: Advances in Database Technology,
5. Vieira MR, Chino FJT, Traina C, Jr., Traina AJM DBM-Tree: A Dynamic Metric Access Method Sensitive to Local Density Data. In: Brazilian Symposium on Databases, Brazil, 2004. pp 163-177

6. Indyk P, Motwani R Approximate nearest neighbors: towards removing the curse of dimensionality. In, 1998. ACM, pp 604-613

7. F-Secure (2007) F-Secure Reports Amount of Malware Grew by 100% during 2007.

8. Novak D, Zezula P (2006) M-Chord: a scalable distributed similarity search structure. Paper presented at the Proceedings of the 1st international conference on Scalable information systems, Hong Kong,

9. Batko M, Gennaro C, Savino P, Zezula P Scalable similarity search in metric spaces. In, 2004. pp 213-224

10. Batko M, Gennaro C, Zezula P (2005) A scalable nearest neighbor search in p2p systems. Databases, Information Systems, and Peer-to-Peer Computing:79-92

11. Haghani P, Michel S, Aberer K (2009) Distributed similarity search in high dimensions using locality sensitive hashing. Paper presented at the Proceedings of the 12th International Conference on Extending Database Technology: Advances in Database Technology, Saint Petersburg, Russia,

12. Cortes C, Vapnik V (1995) Support-vector networks. Machine learning 20 (3):273-297