Learning Concepts through Multi-Class Diverse Density

Chalita Hiransoog

Master of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2007

Abstract

This research investigates the possibility of creating an intelligent system based on the philosophy that the world is ambiguous and a system gains knowledge by learning from these ambiguous examples where the learning can especially be improved when a system is allowed to play an active role in requesting these ambiguous examples. The above philosophy will bridge the gap between the traditional Artificial Intelligence (knowledge-based AI) and the behaviour-oriented Artificial Intelligence (intelligence emerging from behaviour). Concept learning, due to its simplicity and features needed to prove this philosophy, is chosen as the studied platform. Based on the aforementioned philosophy, the task of concept learning is comparable to the multiple-instance learning framework where the learning framework will be modified to tackle more classes compared the the original two-class problem, named here as the multi-class problem. The multi-class multiple-instance learning problem is thus defined. One of the methods used to solve the original multiple-instance learning framework, the Diverse Density method, is selected due to its simplicity, robustness, and incremental property. The method is then modified to solve the newly defined multi-class multiple-instance learning problem. To explore the functionality and the efficiency, the modified method, multi-class Diverse Density, was tested on both artificial data and real-world applications: stock prediction task, assembly task, and document search. It was found that redefining the two-class problem as multi-class problems allows a wider range of ambiguous concepts to be better captured than is possible with the original multiple-instance learning framework. Moreover interactivity, the ability to play an active role in requesting or suggesting examples to learn, was proven to enhance the learning process when integrated into the multi-class Diverse Density method. In summary this research proves that the task of concept learning of ambiguous objects can be solved using the proposed multi-class Diverse Density method where the added interactivity feature improves the learning further

Acknowledgements

This work has been a long and enduring process. I would like to thank Chris Malcolm, my supervisor for his patience and valuable comments. I am very grateful to my family especially my parents and my aunt, for their unconditional support throughout. I would like to dedicate this thesis in my late aunt's, Na-Toom's, loving memory. Thank you to all my friends who have always been there for me. I am indebted to Lucien, who has put in a lot of patience, time, and effort and into helping me correct my confusing writing style. Without him, this thesis would not have been completed in this condition.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Parts of this work have been published as:

- Hiransoog, C. and Murray-Pitts, L. A. (2006). “Introduction to interactive document search with Multi-Class Diverse Density”. *To appear in Proceedings of Advances in Intelligent IT: Active Media Technology 2006 (AMT06) in ‘Frontiers in Artificial Intelligence and Applications’*, IOS Press.

(*Chalita Hiransoog*)

In loving memory of Na-Toom, my beloved aunt.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Cognitive scientists have stated the assumption that the human mind has mental representations [Stanford University, 2004]. These representations are broken down into five categories; concepts, logical representations, rules, analogies, and images. Each of these mental representations corresponds to different types of mental procedures. For example, deductive and inductive procedures are applied to logical representations to produce inferences; retrieval, mapping, and adaptation are applied to analogies to produce behaviours; and images are constructed and manipulated to produce behaviours. In the case of 'concepts', they are viewed as sets of typical features that match the real world. A procedure like matching is applied to these concepts to produce behaviours. Concepts are not only considered as simple ideas (i.e. each concept is only a set of features) but also a powerful platform for any system to create an understanding of the world around it. Comparing the human mind to a computer, the mental representations are analogous to computer data structures and Artificial Intelligence has attempted to create methods to reflect the human behaviour / mental procedure that operates on these data.

In traditional Artificial Intelligence, the designer of a system has tried to extract human concepts of the world and represented them in a form appropriate for such a system to reason with. However, this extraction is a model of either the system user's or system designer's interpretation of the world that is then pre-built into the system. Because of this pre-built approach, a system suffers from "brittleness" problems [Holland, 1986] where it fails when confronted with problems even slightly outside its domain. This suggests that all the possible concepts must be pre-built into the system, which is clearly impossible for real world applications. On the contrary, behaviour-oriented Artificial Intelligence proposes building up or emerging concepts of the world by letting a system interact with the environment alone. Systems based on the behaviour-oriented approach exist but their functions are still limited. It is the belief of this research that it is impossible for a system with more complex functions to be created without receiving guidance from an expert on what concepts are to be built. The expert could be anything from other similar systems to the designer of the system that it is interacting with. Having guidance can be seen as an influence from traditional AI, however the learning should not be carried out passively by the system; the system needs to be interactive with the expert. This requirement for interaction can be considered as an influence from behaviour-oriented AI.

This thesis proposes to take the middle ground between the two approaches (i.e. traditional AI vs behaviour-oriented AI) in order to create a learning method that is appropriate for concept learning in which the interaction between a system and the expert is also allowed. In other words, the developed system creates a concept about something by learning from examples given by an expert but at the same time the system also actively requests for examples that would aid its learning process. The latter can be seen as the interaction between the system and the expert. This thesis has chosen to view the concept learning problem as multiple-instance learning problem [Dietterich et al., 1997] by creating a new model that will extend it to multi-class classification. It will also be shown that the learning method called ‘Diverse Density’ [Maron and Lozano-Perez, 1998] is the best fit for modification into a multi-class learner with the above requirements.

This chapter is the basis for setting up the claim of this research. In the first section of this chapter, a summary of concept learning and why the Diverse Density method is a suitable starting point for the development of the proposed concept learning algorithm are reviewed. Second section deals with the overview of the modified Diverse Density method. The last section looks at a few applications which are proposed as the test platform for the modified Diverse Density method.

1.1 Concept Learning

The interest of this thesis is to learn a ‘concept’ with a lesser degree of restriction on concept definition but still with some guidance from an expert. By way of examining the feature of a ’learned concept’ in section 1.1.1, it is possible to provide the requirement for such an algorithm. This can then be followed by the comparison to general machine learning algorithm in section 1.1.2 and an analysis in section 1.1.3 of one particular learning algorithm (the multiple-instance learning framework [Dietterich et al., 1997]) that matches the requirement of the proposed concept learning algorithm very closely. Section 1.1.4 introduces Diverse Density method proposed by [Maron and Lozano-Perez, 1998] as the method to be further developed for the purpose of this thesis in comparison to other methods employed to solve the multiple-instance learning problems.

1.1.1 General Summary

It is commonly understood that 'concepts' summarise entities (things, objects, situations etc.) into groups. Each group is categorised using a number of features that are shared by each member of the same group. For example, plants share a set of features among themselves while animals share another set of features that distinguish them from being plants or other living organisms. Once a concept is formed by a system, it will then be used to classify future experiences so that a system responds better to the situation around it. However, as mentioned previously, it is not yet conclusive as to how concept formation should be organised. On a high level of organisation, neither having concepts pre-specified for a system like that of the traditional AI approach, nor ignoring the idea of concept but letting the interaction with the environment dictate its behaviour like that of the behaviour-oriented AI approach, are sufficient enough. By proposing that concepts of the world should be formed by letting a system interact with and receive guidance from an expert, then the system can build up its own knowledge while more guidance is supplied.

Moreover, in any real life situation, the target concept and any guidance given could be ambiguous. Ambiguous guidance given to a system can be thought of as missing or incomplete information. In the case of ambiguous concepts, there is not yet a clear idea of how a concept should be described. For instance, features describing good stocks registered in a stock market do not guarantee that those stocks will perform well at all times. Therefore any learning algorithm developed for the purpose of concept formation should be able to handle some degree of ambiguity.

On a detailed level, there have already been many attempts to create a perfect concept formation algorithm. A review on different approaches to concept formation and their examples can be found in Chapter 2: section 2.1. Although these approaches tackle many aspects of concept formation such as supervised vs unsupervised, logical vs probabilistic representation, and overlapping etc. (as discussed in Chapter 2: section 2.1.3), they all assume that objects under observation are single entities. In other words, one type of object represents one type of entity. This is an assumption that will fail the objective of this research's concept learning system in the two following aspects:

- A single-entity assumption will not cover enough ground for all types of objects to be learned by the system, as there are two cases where objects have to be described as multiple entities. In the first case, each of the entities could be describing the object at one particular time but not at another time. In the second case, all of these entities could be describing the object but only a few of these entities can be used to distinguish this object from others. One example of the former case is drug molecules where each molecule could take on many different atom configurations. These drug molecules take on one configuration during a chemical reaction with a disease protein on one occasion, and take on another configuration on different occasions. However, only one of these different configurations can bind well with the disease protein. An example of the latter case is an image data retrieval task. A few features presented in an image (e.g. mountain scenery) are desirable by the users, all other features are not but this information is not specified to the system. It is possible to expand on the previous stock example in section 1.1.1. For each stock, some but not all of its properties can be seen as influencing its price at a particular time. Thus it is important that the proposed concept learning algorithm be as general as possible, so it should be able to deal with this multiple-entity assumption as well.

- The concept formed based on a single-entity assumption can be too biased; there is little room allowed for the system to explore the concept it has created. This is a restriction rather than an enhancement on the second part of the objective to make the system learn through interaction back and forth with the expert or with the outside world.

Therefore the proposed system should be able to handle objects that cover both single-entity and multiple-entity descriptions and it should also allow interactivity with the expert. In the next topic, the proposed system will be compared to the general machine learning algorithm, in order to judge whether there is an existing system that could suit this need.

1.1.2 Comparison to A General Machine Learning Algorithm

In the field of machine learning, Mitchell [Mitchell, 1997] defined the learning process in term of a computer program as follows; "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.". These variables E, T, and P will be assigned differently according to the learning algorithm chosen.

If concept formation is looked at from a point of view of the learning process defined by Mitchell above, experience E would be assigned to the observation of objects, task T would be assigned to the classification of objects, and performance measure P would be assigned to the accuracy of the classification. However the proposed system is supposed to receive some guidance from the outside world, therefore experience E, which in this case is assigned to the observation of objects, should integrate guidance from the outside. With the above aspect, the proposed system is comparable to the traditional supervised learning framework. In supervised learning, examples given by the teacher to the learner, would be equivalent to experience E, the classification of unseen examples by the learner would be equivalent to task T, and the accuracy of the classification would be equivalent to performance measure P.

Despite satisfying the guidance integration requirement, supervised learning is unable to support learning of objects with multiple-entity definitions. However a novel system introduced by [Dietterich et al., 1997] called the multiple-instance learning framework can replace supervised learning as a potential candidate. Like the supervised learning framework, the multiple-instance learning framework adopts similar assignments for variables E, T, and P above. The difference between these two learning frameworks is that both make different assumption about the nature of examples given by the teacher. The supervised learning framework assumes that one example consists of one entity (or instance), which can be described by a number of features. The class label given by the teacher for this one example is a sole response to this one entity. However the multiple-instance learning framework, which was motivated by the problem in drug discovery [Dietterich et al., 1997], considers that in many circumstances such one-example-to-one-entity (or instance) assumptions cannot be held. As a result, the multiple-instance framework takes on a one-example-to-many-instances assumption. In other words, an example consists of many or multiple instances. However, we do not know which one or more of these instances are responsible for the class label of the example.

Figure 1.1 illustrates the different nature of the examples given to the supervised learning and to the multiple-instance learning algorithms. The above suggests that the multiple-instance learning framework is a suitable starting point for the development of concept learning of ambiguous objects. The next section deals with the extension necessary to create such a framework.

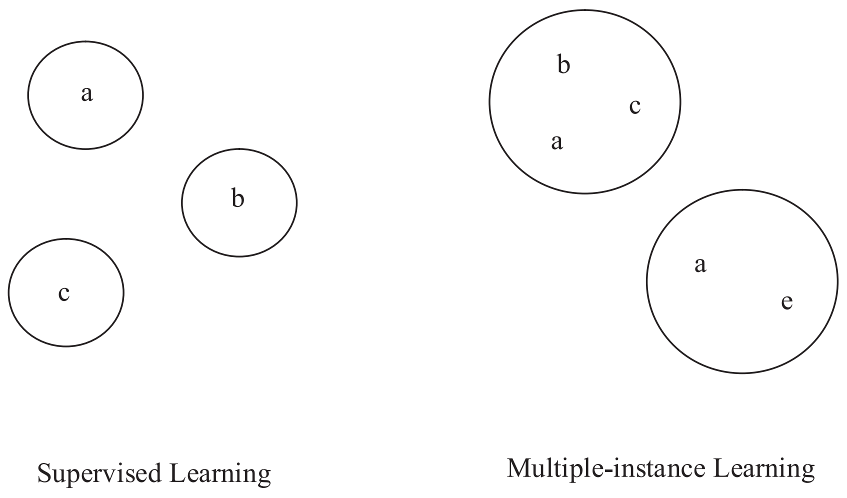


Figure 1.1: The different nature of the examples given to the supervised learning and the multiple-instance learning algorithms. A circle represents an example and a lower case letter represents an instance.

1.1.3 Concept Learning and The Modified Multi-Class Multiple-Instance Learning Framework

The multiple-instance learning framework was formalised by [Dietterich et al., 1997] as follows. Each example presented to a learning algorithm or a learner is a set (or bag) of instances. A bag will be labelled positive if at least one instance in the bag is positive. On the other hand, a bag will be labelled negative if all the instances in the bag are negative. The goal for the learner is to accurately predict the labels of unseen bags by way of induction from a collection of labelled bags (examples) presented.

However, there would still be some discrepancies if multiple-instance learning framework were to be applied to the process of concept formation. This is due to the fact that the original multiple-instance learning framework only deals with examples given to the learner being of only two classes i.e. positive or negative. In concept formation, examples given could be of multiple classes (e.g. living organisms could be classed as either plants or animals or bacteria or virus etc.) with each class being described by a set of unique features. If objects are labelled negative, it is not necessarily only because they are not positive like is assumed by the multiple-instance learning model but also because they have a unique set of features describing their negativity. Hence to apply multiple-instance learning to the concept formation process of ambiguous objects, the multiple-instance learning problem should be modified to accommodate the possibility of having multi-class examples. In order to solve the multi-class problem, the order of influence of the class label must be used. This leads to the question of which learning algorithm targetted for the multiple-instance framework would be suitable for the multi-class extension and this will be discussed in the next section.

1.1.4 Concept Learning and The Diverse Density Method

As was mentioned in the last section, in order to extend the multiple-instance learning problem to deal with the multi-class problem, the order of influence of the class label must be introduced otherwise a bag of instances could be given conflicting labels. Other methods such as Citation KNN (Chapter 2: section 2.2.3.1), Multi Instance Neural Network (Chapter 2: section 2.2.3.2) and BP-MIP (Chapter 2: section 2.2.3.2) solve the issue in different ways but none of them support all of the requirements of this research, thus the Diverse Density method is chosen as the starting point for modification. A full review of this is made in Chapters 2 and 7.

Prior to discussing why the Diverse Density method was chosen as a starting point for modification to create the proposed concept learning method, the original Diverse Density method should first be reviewed. In multiple-instance learning, each instance in a set (or bag) of instances is a point in feature space. A multiple-instance learning algorithm needs to find a point/points in feature space that always exists/exist in all bags that are labelled positive and does not/do not exist in all bags that are labelled negative. In other words, an unseen bag of instances that contains the true positive instance/instances, will be labelled positive; otherwise it will be labelled negative. Although the simplest learning problem only involves one dimensional feature space (i.e. each instance is described by one feature), a problem described by two-dimensional feature space will give clearer illustration of where a solution to the multiple-instance learning problem can be found in the feature space diagram. Figure 1.2 shows an example of the multiple-instance learning problem involving two-dimensional feature space. In this figure, each bag only gives some arbitrary samples of instances. Hence each bag does not need to have a continuous manifold in a feature space, which is appropriate to most of the multiple-instance learning problems.

Maron and Lozano-Perez [Maron, 1998], [Maron and Lozano-Perez, 1998] suggested that if what is in common among the positive bags and does not appear in the negative bags is found, then a concept that agrees with the training examples can be identified. In other words, in an ideal situation where each bag is treated as a set, we should be looking for ‘the intersection of the positive bags minus the union of the negative bags’ as a solution for

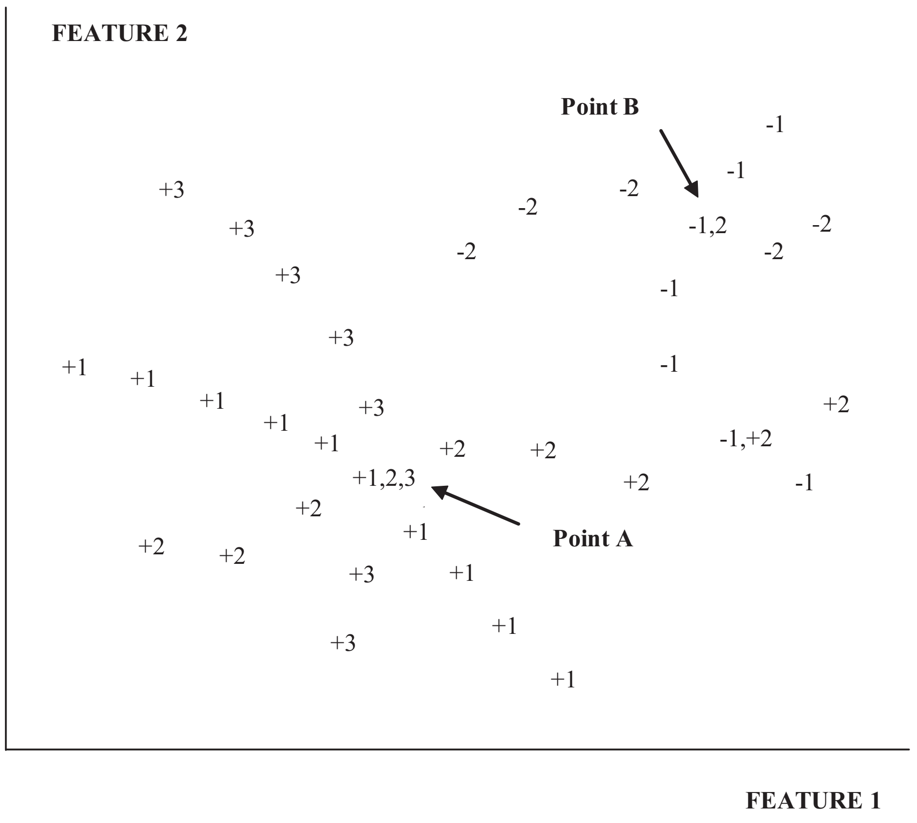


Figure 1.2: A two-dimensional feature space diagram describing the multiple-instance learning problem. '+1' represents an instance in positive bag number one, while '-1' represents an instance in negative bag number one and so on.

a multiple-instance learning task. The solution can be represented as point A in Figure 1.2. However, since the real world data given to the system could be corrupted (i.e. data with noise), they later proposed to use a softer version of the set operations instead of the stricter ones mentioned previously at the beginning of this paragraph. The softer version is based around the idea that some probability distribution can describe an instance being at a certain location in a feature space resulting in it being treated as evidence for or against the location of the true concept.

They then defined ‘Diverse Density’ at a point in feature space to be a measure of (or evidence of) how many different positive bags have instances near that point and how far the negative instances are from that point. Note that the term ‘Diverse’ in ‘Diverse Density’ was chosen because this method not only looks for the area of high density of positive instances but also looks for the area that diverges as much as possible away from the negative instances. They also show that Diverse Density can be computed as a combination of the probability densities from each bag. The point in feature space that maximises Diverse Density will be the true underlying concept we are looking for (e.g. the highest Diverse Density should be at point A in Figure 1.2).

It is therefore the task of this research to adjust the definition of the original Diverse Density method to suit the multi-class problem such that the revised Diverse Density method can still be computed in order to learn the target concept. Moreover the Diverse Density method was designed to be very robust against very noisy data by adopting the use of probability. Hence, the more evidence (or examples) supplied to the learner, the lesser the effect of noisy data on the learning process. It will be shown that the new improved method for multi-class problems will also inherit the same robustness.

As an enhancement to the second requirement of having the interaction with an expert, a method of incremental calculation is desired in order to further increase efficiency. In fact a subsequent benefit of adopting the Diverse Density method is its incremental property. This is because Diverse Density at a point is a combination of Diverse Density at that point calculated for each bag. Therefore, when the system is given more bags of instances, Diverse Density already calculated for the old bags will not need to be altered, only the multiplication of densities from the old and the new bags is required.

1.2 The Modified Diverse Density Method for Multi-Class Problems

One example where the Diverse Density could be applied is image data retrieval tasks. The tasks were originally described as two-class multiple-instance learning problems. Maron and Lozano-Perez [Maron and Lozano-Perez, 1998] introduced the task where a simple description of a person from a series of images must be learnt. The images were labelled positive if they contained the specific person and negative otherwise. Another similar task using natural scene features like fields, mountains, waterfalls etc. as investigated by [Maron and LakshmiRatan, 1998], the users labelled images positive if they had the desirable features otherwise they were negative.

Describing the image data retrieval tasks as a two-class problem is rather restricted in terms of concept description. Positive features only describe target images, where in fact, negative features could also define those images. In other words, a user should be allowed to impose undesirable features in the images. For example, a user could be looking for images that do not have certain features as well as having other desirable features, hence the need to approach the task as a multi-class problem. The users will label images negative only in the case that the undesirable features present in the image are more influential than the desirable ones. The calculated multi-class Diverse Density should tell us about the features that are desirable, the ones that are not and which of the two types is more influential according to the users' decision.

To summarise, in the original multiple-instance framework, the label of each bag of instances is from the sets of positive and negative labels and only the features underlying the positive class are searched for. In other words, the positive class can be seen as having more influence over the negative class (i.e. instances that underlie the positive class will always exist in all the positive bags but never in any of the negative bags). This is a limitation of this framework because the labelling of objects should come from the set of multiple classes for an accurate identification of objects. Furthermore, the framework should allow the possibility that each class could exert its influence on the label of the object. For example, an image is selected on the basis that it does not have a certain feature. Therefore all negative images will have this feature as an underlying concept. Hence we should be looking for this unique set of features underlying every class, not just for the positive class alone.

As a result, we have included multi-class labelling of bags of instances as part of the multiple-instance framework. Adding more classes to the label means that the influence of one class label over others must be allowed (i.e. a bag of instances will be labelled according to its instances that underlies the highest influential class label). However the basic idea behind the learning algorithm for multi-class problem is still similar to the original two-class problem. The difference is that the multi-class learning algorithm not only has to find an underlying concept for each individual class but also has to find the order of influence of each class label.

Let us consider the multi-class problem where examples are classified into two classes only. The positive and negative label system is chosen in this case. We are looking for at least two points in a feature space; one is the point underlying the positive class (which can be illustrated as point A in Figure 1.2) the other is the point underlying the negative class (which can be illustrated as point B in the same figure). However, we cannot think of the first point as the soft intersection of the positive bags minus the union of the negative bags, like that described in the original Diverse Density method [Maron, 1998] and [Maron and Lozano-Perez, 1998] because not all of the negative instances within the multi-class problem are true negatives as is assumed in the two-class problem. Nonetheless, the assumption that a false instance is likely to be the instance that appears in at least one of the positive bags and in at least one of the negative bags is drawn instead. For example, if instances a, b, and c are in a positive bag while instances c, d, and e are in a negative bag, we can assume that instance c is likely to be a false instance because it appears in both positive and negative bags. From now on, we will call instances such as instance c in the example a joint instance (i.e. an instance that appears in at least two bags of different classes).

Although joint instances cannot be used as evidence against a specific concept being the target concept in the same degree as negative instances have provided in the two-class problem, evidence from the joint instance can be used to degrade the evidence from the soft intersection (as detailed in [Maron, 1998]) of all bags from each individual class. If the joint instance is not the same as the instance being the soft intersection of all the bags from one individual class, then this joint instance is very unlikely to be the underlying instance in the same way that we consider negative instances in the two class problem. On the other hand, if the joint instance is the same as the instance being the soft intersection of all the bags from one individual class, the instance could still be the underlying instance for this specific class but it is not likely to have a strong bag-labelling influence.

Adopting the above knowledge about the joint instances is insufficient thus further investigation given here shall conclude that “joint instances” function in the same manner as negative instances in the two-class problem for the calculation of Diverse Density. In other words, joint instances will be used to provide the evidence against a specific concept being the target concept for a specific class within the multi-class problem. Hence finding the true positive instance, point A in Figure 1.2, should become the search for the area of high density of positive instances where it also diverges as much as possible away from the joint instances. This is equivalent to the soft intersection of the positive bags minus the union of the joint instances instead of negative instances. The same applies to the finding of the true negative instance, point B, that it is the soft intersection of the negative bags minus the union of the joint instances.

Furthermore, we also need to find out, which of the two true instances (one from the positive and the other from the negative class) is more influential in labelling a bag of instances. Referring back to the image data retrieval tasks example, does the user choose the image because it has got positive features or because it has not got the negative features? Which one has more influence on the user decision? Ideally, the new version of Diverse Density should have the highest value at the concept underlying the most influential class, and the less influential the class the lower the value at the concept underlying it. Hence the problem becomes one of finding the suitable probability density describing a bag of instances and that for a set of joint instances to give rise to the required effect when combined. It is proposed that the addition of all the modified Diverse Density for a given concept being a target concept underlying each individual class, can be used as an indicator for the order of influence. Those two issue above will be investigated further in Chapter 4.

As an extension to the two class problem as investigated by [Maron, 1998], given the number of possible classes is N, the true concept underlying each individual class N becomes the soft version of [the intersection of the bags of Class_N minus the joint instances from bags of different classes available]. Furthermore, with the appropriate probability density model, it is possible to add up the modified Diverse Density of every class to yield the order of influence of each individual concept.

From now on, the modified Diverse Density for the multi-class problem will be called multi-class Diverse Density. Proving the feasibility of the multi-class Diverse Density method will be carried out through tests with artificial data described in Chapter 4: section 4.3, and this is then followed by tests with real-life data, such as stock data, in Chapter 5: section 5.4.2. Besides proving the feasibility of the multi-class Diverse Density method for the concept learning task, instance volume sensitivity will also be explored using stock data. Furthermore to satisfy the original objective of creating a system that is interactive with the outside environment, the multi-class Diverse Density method will be further extended to include the interactivity feature. This feature will be detailed and tested with the applications described in Chapter 6: section 6.4 and section 6.5. It is hoped to show that the proposed approach will succeed in the test platforms discussed in section 1.3 and preform better with the enhanced features mentioned above added.

1.3 Test Platforms

Three scenarios are laid out. They can be described as a problem of the multi-class multiple-instance learning task. The first scenario, the stock prediction task, is chosen for this study because instance volume is likely to have an effect on the learning hence its sensitivity can be investigated. When considering self-recovering tasks and interactive document search tasks, the interactivity feature can be studied as it is an active part in solving these two problems.

1.3.1 Stock Prediction Task

Like the image data retrieval task, discussed in section 1.2, the original stock prediction task explored by [Maron, 1998] can be modified to allow a better description of the target concept, in this case stocks, by using the multi-class multiple-instance learning approach. In the original the learner was looking for stocks that performed well for fundamental reasons, rather than on chance. Stocks that performed well, whether because of fundamental reasons or because of luck, were put in positive bags. However in our adaptation we are also looking for stocks that perform badly because of fundamental reasons and not because of the fear of speculation by investors etc. In other words, we also pay attention to negative bags in this case and look for the true negative (i.e. fundamentally bad stocks) as well as the true positive.

It is hoped that the multi-class approach will prove to be more robust that the original two-class approach. For example, certain assumptions such as every stock in the negative bags has to decrease in value for fundamental reasons (i.e. every stock must be a true negative instance) do not need to hold. So far we have considered one instance as one unit, and with this task, the question whether in certain circumstances it is better to consider instance as having a volume will be answered. If this is the case, how can instance volume be best represented and how can the multi-class Diverse Density method be modified to fit the new description? Chapter 5 deals with answering this question and showing that the stock prediction task can be better solved with the multi-class Diverse Density with instance volume feature included in the calculations.

1.3.2 Self-recovering System

The self-recovering system is capable of discovering its own faults or malfunctions and repairing the damage itself. The learning section of such a system is responsible for uncovering the root of the fault through learning from the operation record of its success and failure. An example of such a learning problem can be found in problem determination in large, dynamic internet services. One solution is to use the record of both the success and failure of client requests through e-commerce systems to determine which software components are likely to be at fault. The Pinpoint system developed by Chen and his colleagues [Chen et al., 2002] employed a data clustering algorithm to identify such software components. Components were grouped together according to their similarity on how often components were or were not used together in client requests. The failure points were marked at the location where the requests failed. The components that clustered around the failure point are likely to be the root cause of the failure. In our case, the multi-class Diverse Density method could replace the clustering algorithm with an increase in dynamic response of the system. This is because the multi-class Diverse Density method allows examples (i.e. record of client requests in this case) to be learned incrementally.

Another example of the self-recovering system is the assembly task that will be used as a platform to investigate the possibility of adding interactive feature to the multi-class Diverse Density method. The investigated assembly task is a novel problem, which actually provided the insight into both the multi-class problem within the multiple-instance learning framework and the use of the multi-class Diverse Density method to find the solution. The assembly task is described as a certain number of plans generated by the planner failing unexpectedly. This failure is believed to be a result of the lack of knowledge of the behaviour of certain assembly parts when putting them into a plan due to either location or proximity. For example, there might be two parts that cannot be put adjacent to one another (e.g. magnetically repellent) and this will cause the assembly to fail, or when some parts cannot be put at certain places in an assembly, and alternatively when some parts must be put at certain places for the assembly to be successful. Unfortunately, the assembly system does not have such built-in knowledge in the first place. Therefore, it is the task of the learning system to figure out, from the results (i.e. success or failure) of testing plans, which of the relationships mentioned above cause the plans to give unexpected results.

All the assembly plans generated for a specific task can be viewed as either of the two classes: plans that succeed belong to a positive class; and plans that fail belong to a negative class. Let's assume that there are two magnetic parts and if these two parts are adjacent in any plan, the assembly will fail. Therefore the plan that fails will always have the two magnetic parts adjacent to one another among other relationships which can vary from one failed plan to another. The adjacency of these two specific parts can be called a true negative relationship, while other relationships belonging to the same plan are called false negatives. For a successful plan the two parts will be separated from one another. Hence all the relationships that signify the separation of the two parts are true positive relationships. It follows that, other relationships within the same plan are false positives. In this case if a sufficient number of plans are tested, the influence of true negatives should be higher than the true positives. This is because there will be multiple numbers of true positives (i.e. not every plan has the same true positive relationship) compared to only one true negative.

One way of describing the above scenario is by classifying it as a multi-class multiple-instance learning problem. Each assembly plan generated by an assembly planner could be treated as a bag of instances, where each instance represents possible individual relationships between assembly parts and the relationship between each part and its location in an assembly. When a plan is assembled successfully it is considered a positive bag, and if a plan causes a failure it is considered a negative bag. This scenario is a multi-class problem because not only do we have to find both the true positive instance (i.e. the relationship that has to be present for the assembly to succeed) and the true negative instance (i.e. the relationship that if present will fail) but also we need to compare their influence on the bag label. Calculating the multi-class Diverse Density will give the answer.

This novel assembly task can be used as a platform to explore how best to add the interactivity feature to the multi-class Diverse Density method. This interactivity feature should allow the learning system (a.k.a. learner) to interactively request, based on its current knowledge, new examples that would lead to much faster learning of the target concept. This issue will be further discussed in Chapter 6.

1.3.3 Interactive Document Search Task

Searching for a document using search engines has always been a passive procedure for the user. The user supplies a search engine with a few keywords and then the engine comes up with a list of the documents associated with these keywords. Some of the documents listed are what the user requires but some are not. The user then has the task of sifting through the list manually for what they are looking for. One way to reduce the amount of sifting is for the user to provide more keywords to the search engine. However, searching can sometimes be a progressive procedure. Sometimes the user is not sure what they are looking for exactly, and only after seeing a few documents from the search results can the user gain the concept of what they are after. Therefore, an alternative is to have the search engine interact with the user. The user may randomly pick documents and report back to the search engine on how closely the documents match their requirements. The search engine then uses this information to incrementally build up the user's search concept so that the list becomes smaller.

The above task can be viewed as in conjunction with the multi-class multiple-instance learning tasks because of the following. Firstly, each document consists of words but only a few words are the main idea identified with the user's concept, therefore each document can be considered as a bag of instances and words as instances in a bag. Secondly, allowing users to classify documents in multiple groups depending on how closely they match with the user's concept leads us to consider this task as a multi-class problem. Furthermore this task is a good platform to investigate the effect of interaction between a system and an environment, to build a concept within the system. More on this in Chapter 6: section 6.5

1.4 Summary

In short both traditional AI and behaviour-oriented AI fail to create concept learning system that can overcome "brittleness" problem or can deal with complex functions. This thesis introduces a new way of building a concept learning system that integrates both the above approaches into one. The system will receive guidance from an expert similar to learning from examples while the system will also interact with the expert by actively request for examples similar to interacting with an environment. It is proposed to view such a concept learning task as the multiple-instance learning task [Dietterich et al., 1997] but with the extension to multi-class problem. The Diverse Density method [Maron, 1998] is chosen to be modified to solve the proposed task. The validation of the modified method is then explored in Chapters 3 and 4.

Traditional concept formation systems and the development of the multiple-instance learning framework before and after the proposal of the Diverse Density method are reviewed in Chapter 2. Chapter 3 details the alteration of the Diverse Density method for solving multi-class problems. Chapter 4 investigates different issues concerning the feasibility of using multi-class Diverse Density. The exploration of instance volume sensitivity and interactivity features are detailed in Chapters 5 and 6 respectively. Chapter 7 provides a detailed comparison with other methods followed by the conclusion in Chapter 8.

Chapter 2

Literature Review

This research focuses on creating a concept learning system that can build concepts from ambiguous objects. The interest is in concepts which can be described by a number of features. Objects that are described by the same concept will share the same set of features. This is not to say that concepts could not be viewed in a different manner. One example is concept learning as pointed out by Wittgenstein in his language-games investigation [Wittgenstein, 1965]. “When one shows someone the king in chess and says: “This is the king”, this does not tell him the use of this piece–unless he already knows the rules of the game up to this last point...” This statement was to point out that there is more to the concept of “the king” in chess game rather than it being a piece in a chess game. Wittgenstein suggested that use of a word should be guided through a complicated network of similarities, overlapping, and criss-crossing and family resemblance also serves to exhibit the lack of boundaries and the distance from exactness that characterize different uses of the same concept. In other words, concepts should not be exact.

In comparison, Wittgenstein looks at concepts as not having an exact description while this research tries to create concepts from ambiguous objects. Ambiguity is dealt with at a different states in the flow of concept learning for these two approaches. Wittgenstein’s approach creates ambiguous concepts as the end product while this research proposes removal of ambiguity from objects at the beginning. Since it is proposed to use concept learning as a test platform to remove ambiguity at the beginning of the concept learning process, the first main section of this chapter is then a review of concept formation systems developed so far. However object ambiguity is not a main focus in the systems in this section. Since this research proposes to eliminate the ambiguity of objects by considering a concept learning task as the multiple-instance learning problem, the second section is then a review on different methods developed for the multiple-instance learning framework.

2.1 Existing Conception Formation Systems Review

This section reviews the development of concept formation systems to date. Most of the systems discussed here were not developed with the object ambiguity issue in mind. They were created for a general purpose of concept formation. They are reviewed because concept learning is proposed as a study platform for this research and there is also a possibility that the proposed system (i.e. concept learning through multi-class Diverse Density) can be added onto these existing systems. The proposed system should be able to eliminate ambiguity from the objects of interest and create a concept of these objects. This concept then can be refined by passing it onto one of these general concept formation systems reviewed here. This way, the same concept formation systems can deal with both normal objects and ambiguous ones without any adjustment to the original algorithm of the systems. The next subsection deals with a general discussion of concept formation models. A comparison of these models is set forth. Their implementations in real systems are described followed by the description of real systems. The last subsection discusses various features, suggested by other researchers, to be accommodated in concept formation systems apart from object ambiguity which is the main concern of this research.

2.1.1 General Models of Concept Formation

Up to now, many concept formation systems have been invented and are employed in different areas such as expert systems, data mining, robotics, and so on. Researchers have been known to use many different models to develop their concept formation systems. The differences between early models are usually based on the differences in the algorithm used. As a result, the early improvement of such systems relies heavily on making the algorithms more efficient in speed and computer usage. However this direction still produced quite a limited success in capturing human concept formation feature. Researchers then moved on to integrating more human-like features (e.g. unsupervised feature, incremental feature) to their model as was also suggested by research in cognitive science. As a result, it is more common to investigate the classification of concept formation system according to these cognitive models.

Most of the concept formation systems reviewed here could be seen to be rooted from, or be connected to, the two main cognitive models even though some details of features in each system varied in detail. The techniques used vary (e.g. Neural Network, Fuzzy logic, Genetic Algorithm or a combination etc.), the system capability and application domain (e.g. medicine, robotics) also all varied between systems. The two models mentioned above are as follows:

- Hunt, Martin, and Stone model of concept formation [Hunt, 1962].

  This model was based on the idea that human mind breaks down items into subsets with common characteristics, to make discrete concepts from new cluttered or unstructured data. They proposed the concept formation strategy as outlined below and called this strategy ‘Concept Learning System’.

  1. Pick a criterion that appears to be the most useful for separating a set into subsets according to some heuristic such as human vs animal.
  2. Separate the set according to the selected criterion.
  3. Return to the first step, choosing new and increasingly precise criteria until a unique definition of the new data is reached.

  Examples of some of the systems that used such model are as follows: (i) AQ [Michalski, 1969], (ii) Version Space Search [Mitchell, 1978], (iii) CLUSTER/2 [Michalski and Stepp, 1983], (iv) ID3 [Quinlan, 1986], (v) COBWEB [Fisher, 1987], (vi) AUTOCLASS [Cheeseman et al., 1988], and their descendants. With items or objects being ambiguous, criterion that succeeds dividing normal objects into subsets cannot achieve the same objective since ambiguous objects that should belong to different subsets could have satisfied the same criteria.

- Smith and Medin model [Smith and Medin, 1981]

  Smith and Medin purposed that human concepts and concept formation are viewed in three different approaches: the Classical, Probabilistic and Exemplar approaches. Here argues that the former two approaches are essentially an alternative description of Hunt model with additional details. The Classical approach states that all instances of a concept need to share common properties and sufficient conditions for defining the concept. Whereas the Probabilistic approach describes concepts in terms of properties that are only characteristic or probable of class members. Thus membership in a category can be graded rather than all-or-none. However the third approach, the Exemplar approach, is the centre of interest here because it is another

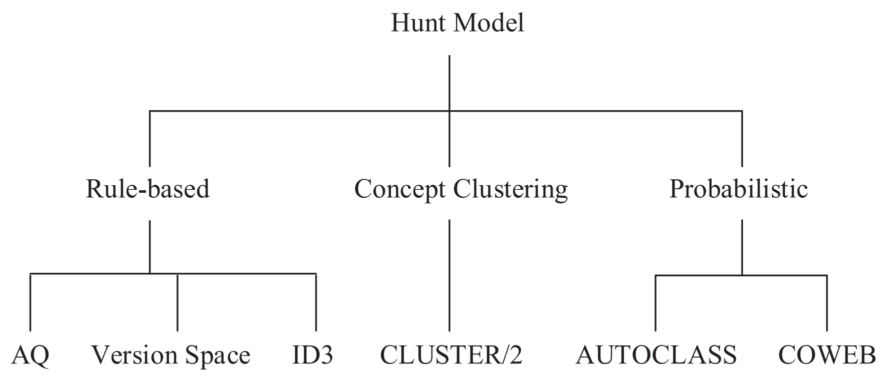


Figure 2.1: Hunt, Martin, and Stone model.

model of concept formation that varies from Hunt model and was used by many researchers such as [Kolodner, 1983], [Lebowitz, 1987] and [Kibler and Aha, 1987] etc. Exemplar based systems store the description of training examples ('exemplars') and then a matching algorithm is used to find the best match between exemplars and the new example presented to the systems. The new example is then assigned to the class of the best match exemplar. Nonetheless there exist other concept formation systems that are the combination of the two cognitive models to some degree.

With ambiguous objects, there will be extra features of the objects that will not fit into an exemplar even if this exemplar might actually be the best match exemplar when the ambiguity is removed. The next subsection compares the similarities and differences of these two models through selected existing systems.

2.1.2 Concept Formation Models Comparison

In this section the comparison of concept formation model will be made through chosen systems. These concept formation systems are selected because they are very well known as well as being a good representation of each model in various aspects. First, let's look at how each of the two cognitive models could be classified further. For Hunt's model (Figure 2.1), the second level on the diagram represents various adopted methodologies, while the second level in the Smith's model (Figure 2.2) represents different number of exemplars required to define the target concept. At the bottom level for both diagrams are well known examples of systems belonging to each model.

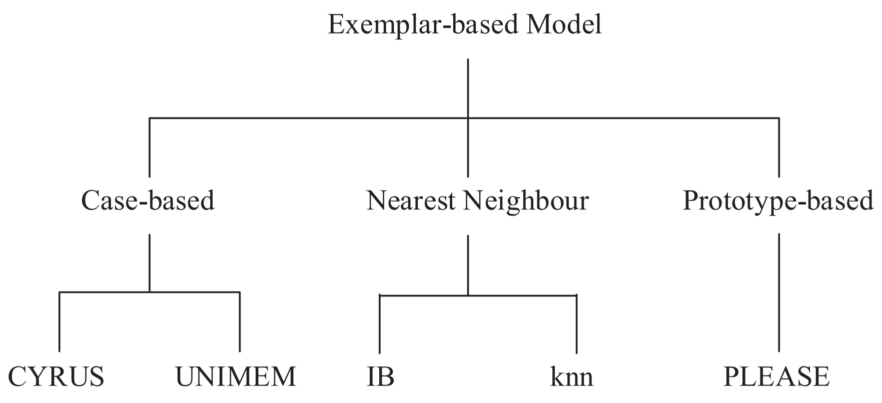


Figure 2.2: Smith and Medin model.

2.1.2.1 Hunt Model

The following deals with each node in the second level of Figure 2.1 , and the systems implemented on these methodologies.

Rule-Based: A concept is usually represented by a general statement in the form of a set of conjuncts or disjuncts. A new example is considered to be a member of a concept if its description holds true compared to the concept description.

- ***AQ*** - The AQ algorithm generates a set of if-then classification rules. To find classification rules for each concept class, it is to generate a rule first and then remove the examples covered by that rule. This step is repeated until enough rules are found to cover all positive examples of that class. Then the whole process is repeated for each class. Starting from the most general rule "if true then predict class C" (i.e. "all examples are of class C"), a rule is repeatedly specialised by appending further conditions until it covers only examples of class C and no other examples. The AQ algorithm was suggested by [Michalski, 1969] and the latest version is AQ18 [Michalski, 1998].

- ***Version Space Search*** - The set of candidate concept definitions is called a version space which is constructed from positive and negative examples of the concept, and representational bias (also called Concept Description Language (CDL)). The conceptual bias is the set of terms used to describe the concept, while the logical bias is a restriction on the logical combination of those terms. The examples are used to eliminate all inconsistent hypotheses. The conceptual bias determines the maximum size of the version space. And the logical bias decreases the size of the version space by disallowing certain hypotheses. The version space is represented using two boundary sets, which contain G - the most general hypotheses and S - the most specific hypotheses. Those boundary sets are updated as training examples are presented to the system, using the Candidate Elimination Algorithm [Mitchell, 1978]. It was later suggested that is also possible to create version space from single examples and intersect them for use in explanation-based learning [Mitchell et al., 1986].

- ***ID3*** - Concepts are represented as decision trees, in which a node of the tree represents a test on an attribute and each branch corresponds to a possible results of the test. A test on an attribute is chosen on the basis that the test will give rise to the greatest gain in information content (i.e. the amount of data held in each unit of representation), while decreasing system entropy (i.e. the least units of representation necessary to communicate a given data set). During the years ID3 [Quinlan, 1986] has been developed to a better version C4.5 [Quinlan, 1993] and the commercial version C5.0 [Quinlan, 1999].

Concept Clustering: Early approaches to concept analysis (numerical taxanomy) represent the objects as points clustered in a multi dimensional metric space. Distance metrics, such as Euclidean and Mahalanobis etc., are adopted to define dissimilarity between objects. Michalski and Stepp [Michalski and Stepp, 1983] later introduced Conceptual Clustering paradigm which includes not only clustering, but also characterisation (formation of concepts from defined cluster). Their first system is CLUSTER/2 [Michalski and Stepp, 1983].

- ***CLUSTER/2*** - The system uses a divisive technique to generate a disjoint hierarchy of concepts. The divisive technique is based on measures of common attribute values within a cluster, non-intersecting attribute values between clusters, and simplicity of the conjunctive expression for describing a cluster. The system starts with a root node consisting of all objects in the data set. It then splits a root node into a set of mutually exclusive clusters using the divisive technique mentioned and recurses to construct sub-hierarchies below each node. CLUSTER/2 was extended to CLUSTER/S [Stepp, 1984] to cope with learning in the structured domain.

Probabilistic Model: Like concept clustering paradigm, it also follows concept analysis (numerical taxonomy) but instead of using non parametric methods which describe class structures as mean vector in multi-dimensional space, it assumes that the probability distribution of each is known.

- ***COBWEB*** - Categories in COBWEB are placed in a concept hierarchy. Each node in a hierarchy, like a tree, specifies a set of children and the conditional probability given its parent category. Furthermore, the relationship between a parent and its children must obey the rule that the probability distribution for each attribute of a parent is a weighted mixture of those of its children. The leaves or terminal node in COBWEB hierarchy correspond to specific examples, normally training cases observed during training. Hence, the root node corresponds to the most general category and categories become more specific as it moves down the tree. After COBWEB [Fisher, 1987], many systems were developed to cover other aspects in concept formation that had not been covered in COBWEB. LABYRINTH [Thompson and Langley, 1991] was developed to induce concepts from structured domains. OXBOW [Iba, 1991] tackled concept formation on the temporal domains such as in the learning movement concepts, which requires a sequence to describe concepts. ARACHNE by [McKusick and Langley, 1991] tried to minimise effects on the learning induced by noise and training order. Learning overlapping concepts was included in TWILIX [Martin and Billman, 1994].

- ***AUTOCLASS*** - Although similar to COBWEB in that it describes category in terms of probability distribution, AUTOCLASS [Cheeseman et al., 1988] does not store training cases as terminal nodes, but instead assigns training cases to every category with some probability. Moreover AUTOCLASS also stores the conditional covariance matrices while COBWEB only stores the conditional probability distribution for each attribute given the category. Since AUTOCLASS takes a very strong Bayesian belief on classification of new instances, a test case to each category at each level is assigned with a certain probability, rather than the most probable one like in COBWEB and its relatives. A later version of AUTOCLASS, [Cheeseman and Stutz, 1996] expanded one level clustering of concept to multi-level descriptions.

2.1.2.2 Exemplar Model

Referring to Figure 2.2 of the exemplar model, the following deals with each node in the second level, and the systems implemented on the usage of the different number of exemplars.

Case-based or Instance-based: The system was inspired by psychological findings on memory usage showing that humans often recall past experiences to guide themselves to the solution to new problems. Case-based learning discovers concepts by determining which case in memory is the most similar to the new situation or examples. The system forms a model in memory of the relationships between examples. These relationships could either be induced or supplied by an expert user.

- ***CYRUS*** - It was developed by [Kolodner, 1983] to help the understanding of human memory. Whenever a new event or case is introduced that shares features with an existing case, CYRUS will create a generalisation node containing the common features where the events are indexed below it by their differences. Like other case-based system CYRUS stores all examples (in other words cases or events) as exemplars.
- ***UNIMEM*** - Like CYRUS, UNIMEM [Lebowitz, 1987] organises the examples into a generalisation based memory, a hierarchy where each node represents a collection of common features shared by the examples, but with the difference that UNIMEM builds a classification tree based on a Hamming distance.

Nearest Neighbour: Unlike Case-based, Nearest Neighbour algorithm does not store all of the examples and tends to measure similarity between cases numerically. A nearest neighbour system uses a metric that measures the distance between a new example and a set of exemplars in memory. The new example is then classified to be in the same class as its nearest neighbour. In order to measure the distance between symbolic features, the distance is determined by counting the matching features.

- ***IB version 1-5*** - After PROXIMITY, GROWTH, and SHRINK had been developed by [Kibler and Aha, 1987], the IB family was developed to overcome the inadequacy of the standard Euclidean distance in measuring the similarity of examples. PROXIMITY is a pure nearest neighbour algorithm, by which retaining all examples using an unweighted Euclidean distance function to perform classification.

While GROWTH receives examples incrementally, and only stores those examples that the current exemplar database misclassifies. SHRINK accepts all exemplars at first, and drops out the rest of database that are classified correctly. IB families [Aha et al., 1991] were designed to overcome:

  - the expensiveness due to large storage requirements in nearest neighbour systems
  - the sensitiveness to the choice of similarity function
  - the difficulty working with missing attribute values
  - the difficulty working with nominal attributes
  - the conciseness of concept summaries

- ***knn*** - It is another variation of nearest neighbour algorithm where the most popular class of k nearest examples is used for prediction. This method deals very well with noisy examples because it prevents a single noisy example from incorrectly classifying the new one. However the early version of knn algorithm [Kibler and Aha, 1987] experienced difficulty determining the right value of k: the noisier the input set, the larger k required. Because the system does not have this prior information, a cross validation is used by having user training and testing a variety of k values and then letting the system adopt the value of k that produces the best result. knn algorithm was investigated further and later known as the lazy learning model [Aha, 1997].

Prototype - based: This is the other extreme case of exemplar-based system where only a single exemplar represents a concept. As also suggested earlier by [Reed, 1972], a prototype is constructed out of salient or noticeable features of a target concept. Hence a concept description produced by a prototype-based system is supposed to be more concise than the description of concept from other exemplar-based system described earlier. In general prototypes are constructed from descriptions of individual instances of a given class by abstracting the more frequent properties of these instances. The similarity of an input instance to a prototype is calculated by using the difference of the weighted sum of attributes having the same and different value.

- ***PLEASE*** - This system is one example of prototype-based learning system, which also allows multiple prototypes per class in order to overcome linearly non-separable category learning. PLEASE [Knight and Sen, 1995] also employ a genetic algorithm in the learning process.

Above is a summary of the different approaches to concept formation tasks. These approaches do not take into account the possibility that objects or examples could be ambiguous or at least not in the same sense as it is considered in this research as introduced at the end of section 1.1.1 of Chapter 1. However, some of these approaches or methods have been extended to cope with object ambiguity, and these extensions will be discussed in the section 2.2. First it is necessary to discuss features, other than object ambiguity, that should also be accommodated in concept formation systems.

2.1.3 Other Features Accommodated in Concept Formation

It can be seen from the previous section that researchers have chosen to build their concept formation systems to accommodate various cognitive models and have used a variety of techniques (e.g. rule-based, probabilistic, nearest neighbour algorithms etc.). Outside of the main algorithm, researchers have considered other features and have tried to integrate such features into their systems in the hope to make their systems more closely resemble human concept formation. The features under consideration are as follows: Supervised vs Unsupervised; Logical vs Probabilistic Representation; Overlapping; Incremental vs Non-Incremental; and Noisy Data. These are discussed in more detail below.

2.1.3.1 Supervised vs Unsupervised

In supervised concept formation, training examples or observations are labelled by an external teacher indicating the class membership. For example, strawberries could be described by their size (large - medium - small), their shape (heart - round - irregular), or their colour (dark red - red - pink). An expert could label large - round - red strawberries and likewise medium - heart - dark red strawberry as tasty, while medium - irregular - pink strawberries are labelled as not tasty. The concept formation system has to find how to describe tasty and not tasty strawberries by their size, shape, and colour. This supervised concept can be found, for example in Version Space Search [Mitchell, 1978] and ID3 [Quinlan, 1986]. Whereas unsupervised concept formation assumes that no previous information about the class membership of training examples exists. In this research, the ambiguity problem of training examples is in between the supervised and the unsupervised problem. This is because training examples (bags of instances) are partially labelled. In other words, a group of instances is labelled but not the individual and this makes the label ambiguous.

2.1.3.2 Logical vs Probabilistic Representation

Concepts are represented as conjunctive of attributes in the logical representation, which implies that each attribute is of equal weight. While with probabilistic representation, a weight or probability is associated to each attribute, this allows attributes with different strengths to describe concepts as in COBWEB [Fisher, 1987]. In the original multiple-instance problem, each instance represents a point in a feature space and only the features of the true positive instance will be used to describe the concept. This is equivalent to a logical representation unless a different weight is allowed for each feature like as is used within the original Diverse Density. Further if each instance represents multiple points in a feature space and has some assigned probability, the probabilistic representation will more appropriate.

2.1.3.3 Overlapping and Multiple Concepts

If logical representation is used, it can be said that an object is a member of more than one concept, ‘overlapping’, if all its attributes can not be accounted for by any single concept. However if an object to be clustered in multi-dimensional space together with other objects, the object that is located in between two clustering groups could be considered to belong to either of the two groups. This as well could signify an overlapping of concept. An example of existing systems that consider this idea is TWILIX [Martin and Billman, 1994]. From the above two points of view of overlapping concept, it has not yet been suggested whether a new concept would be allowed to be formed from the overlapping region (i.e. whether the overlapping object could as well be a new concept totally different from the nearby objects). It is favourable that the idea concerning overlapping concept should exist, but at the same time a concept formation system should allow a new concept to be formed if further evidences suggest so.

Another issue to be considered is the issue of ‘multiple concept’. ‘Multiple concept’ exists when one object could be a member of more than one concept. All of its attributes belong to one concept as well as when some of these attributes match with other concepts. For instance, a father could be a doctor and vice versa. This is different from overlapping concept because the object satisfies all of the conditions to be a member of both concepts at the same time. Overlapping and ‘multiple concept’ in the above sense were not emphasised in the original multiple-instance learning problem. The problem assumed that each instance only belongs to one concept therefore no further investigation was made to find out whether an instance could be a member of more than one concept.

2.1.3.4 Incremental vs Non-Incremental

Incremental feature means that the system allows concepts to be formed incrementally. In other words, the description of the concepts can be refined at the latter time. For example, in the early version of ID3 [Quinlan, 1986], the process is non-incremental because once the concept tree is constructed, it cannot be altered unless the system is retrained with all the examples. The incremental process is favourable, as more and more evidence from cognitive science is suggesting the resemblance to human learning of new concepts where the learning process develops as more information is fed through our senses. In the original Diverse Density method for solving multiple-instance problem, the target concept could be changed from one concept to another as more and more training examples were introduced (i.e. combined new evidence could result in a different direction from the previous one).

2.1.3.5 Noisy Data

Noisy data normally causes the system to be mislead when generating a description of the concept. The valid concept formation system should try to eliminate or limit the effect of noisy data to the minimum. Early development of concept formation system tends to single out the most unexpected data and disregards the information such data would provide. Later development acknowledges that by eliminating or ignoring this unexpected data could possibly mislead the process as well. This generally happens when the data is considered noisy, but in fact it is a result of incomplete information supplied (e.g. the data is biased only in a certain direction). The main issue here is to find a good balance between what the system needs to ignore and consider in the data. Again since the original Diverse Density method is incremental in nature, the importance of the unexpected data will be either kept or driven away as more and more evidence suggest one way or another.

Summarising all the above research it should be clear that the main concern is to try to eliminate the ambiguity of the observed objects before any concept about these objects is formed. Thus any feature that concerns the manipulation of the object data will be considered.

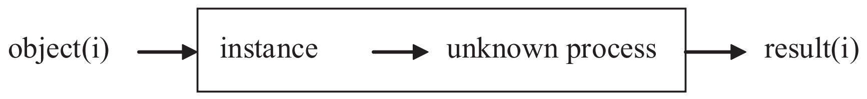
2.2 Multiple-Instance Learning Framework Review

It was already established in the first chapter that concept formation task can be viewed as the multiple-instance learning problem with multi-class classification. Even though multiple-instance learning framework was first formalised by [Dietterich et al., 1997], in such a short span of time, there has already been a number of different researches into this framework. They have come up with different approaches for solving the problem and tested their methods on various problem scenarios (e.g. drug discovery, image data retrieval etc.) with impressive results. It is the interest of this research to modify the Diverse Density method to solve the multi-class problem of the multiple-instance learning task. In order to better grasp the multi-class Diverse Density method an understanding of its creation is necessary, therefore this review will be divided into three sections. Section 2.2.1 is a review of the methods proposed from the time of [Dietterich et al., 1997] to the time just before Diverse Density [Maron and Lozano-Perez, 1998] was proposed. Section 2.2.2 will report on Diverse Density and its descendants. Finally the various approaches developed at the same time and after Diverse Density will be discussed in section 2.2.3.

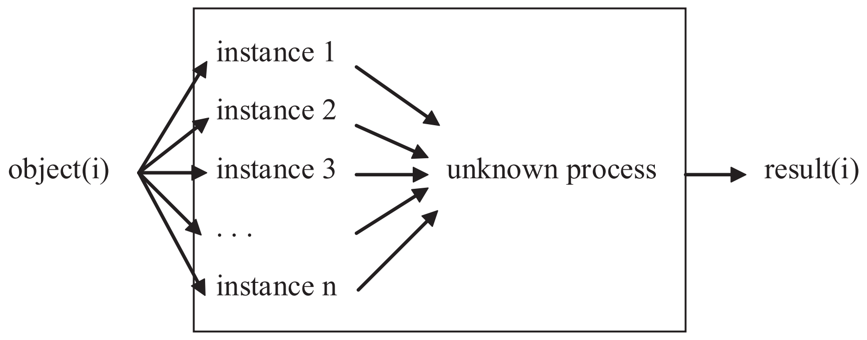
2.2.1 Research Done Before Diverse Density

2.2.1.1 Research Leading to the 'iterated-discrim APR' Method

This research was carried out by Dietterich and his colleagues [Dietterich et al., 1997]. Their first concern was to develop an algorithm to tackle the drug discovery problem, which led them to formalise the multiple-instance learning problem. Unlike the supervised learning framework, the learning system in multiple-instance framework is given partial or incomplete knowledge about each training example. The knowledge given to the supervised learning is of the form (object(i), result(i)) as depicted in Figure 2.3. Only one instance at a time is used to described object(i), whereas in multiple-instance framework, there are multiple number of instances given to the learning system but only one of these instances might truly describe the object-result relationship. Keeping the drug discovery problem in mind, Dietterich and his colleagues suggested the use of axis-parallel hyper-rectangles (APRs) to bound the positive drug molecules given as examples to the learner (i.e. molecules that have at least one of their configurations specified as potential binder to the disease protein). The aim is to find the smallest rectangle that covers at least one instance of each positive example and none of any negative example. In the drug discovery problem, each molecule is an example and each instance in one example is one of the



Supervised Learning Framework



Multiple-instance Learning Framework

Figure 2.3: The knowledge given to supervised learner compared to that given to multiple-instance learner.

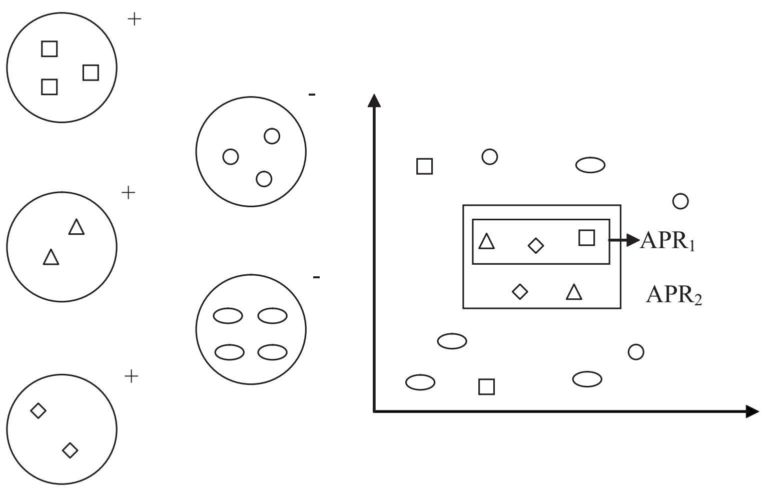


Figure 2.4: The rectangle solution of the multiple-instance learning problem, taken from [Zucker and Chevaleyre, 2000]. A big circle with attached + sign represents a positive bag of instances, where a big circle with attached - sign represents a negative bag.

configurations of that molecule.

The best algorithm, using axis-parallel hyper-rectangles (APRs) and tested by Dietterich and his colleagues, is called 'iterated-discrim APR'. This algorithm consists of three procedures. The first procedure, *grow*, grows a seed APR that covers just only one instance from a positive bag until the smallest APR that covers at least one instance from every positive bags is found. The second procedure, *discrim*, selects the features that discriminate negative from positive instances. The grow procedure is then repeated using features selected by the discrim procedure. These two procedures will need to be iterated several times until the smallest APR is discovered. Then the third procedure, *expand*, expands the bound of APR in order to increase the APR power to generalise. This is done by estimating the density of instances along each feature of the APR, for which the probability of excluding a positive instance is kept at the minimum. Figure 2.4 illustrates the expand procedure of APR in feature space, from the smallest APR (APR1) to more generalised APR (APR2).

[Dietterich et al., 1997] chose to test their algorithm on the task of classifying aromatic molecules into whether they are musky or not. They considered two data sets. The first set contained 47 musk molecules and 45 similar non-musk molecules. The second sets contained 39 musks and 63 non-musks. The two sets also shared a common 72 molecules. This scenario is the multiple-instance learning problem because each of these molecules (musks or non-musks) can have many configurations in the nature, and each with different energy properties. The data set_1 was used as a training set, while the data set_2 was used as a test set. The performance of 'iterated-discrim APR' on the Musk data sets can be found in Table 2.1 at the end of the review section together with the performance of other algorithms. Although it is possible to extend axis-parallel hyper-rectangles (APRs) to solve multi-class problem, it is not going to be straight forward or convenient to extend because the method does not possess the incremental property therefore each time a new example is presented to the method, it has to at least restart the whole procedure to obtain the starting APR of every class.

2.2.1.2 Researches Leading to the 'Multinst' Algorithm

Theoreticians, Long and Tan [Long and Tan, 1996] investigated the learnability, based on the theory of the learnable by [Valiant, 1984], of an axis-parallel concept from multiple-instance examples. They proved that it is possible to PAC-learn this concept but it is possible only by allowing extra assumptions as follows. Firstly, an algorithm must be allowed to have a very high bound running time. Secondly, the dimension of the instance vectors must be independent. Lastly, each instance must be generated independently of its bag. However, the last assumption does not generally hold for applications of multiple-instance problem (e.g. the configuration of a molecule does depend on that molecule). Auer proposed a new algorithm, 'Multinst' [Auer, 1997], which improved the running time given by [Long and Tan, 1996] and imposed less strict assumptions about independency of the features. By remaining firm on the idea that the concept underlying training examples is within APR, Multinst approximates the probability that an instance is within such APR from the training data. Hence if the boundary that makes such a probability equal to zero is identified, we find the required APR. Multinst performance on the Musk data sets was also compared to other algorithms in Table 2.1. Based on Auer's works, [Blum and Kalai, 1998] showed that multiple-instance problem can be reduced to PAC-learning with one-sided random classification noise and slightly improved sample bounds given earlier by [Auer, 1997]. Similar to 'iterated-discrim APR', the Multinst algorithm makes use of APR which introduces the same set of obstacles like those experienced by 'iterated-discrim APR' when the method is extended to solve multi-class problem.

2.2.2 Diverse Density and Its Descendants

2.2.2.1 Diverse Density

The purpose of this research is to modify Diverse Density algorithm, which was developed by [Maron and Lozano-Perez, 1998]. Therefore instead of discussing about this algorithm here in this section, the detailed discussion will be deferred to the next chapter (Chapter 3). However the Diverse Density algorithm can be described in general term as a measure of evidence in term of probability to support whether a concept is an underlying concept given training data. The point in a feature space, which is describing such a concept, should have the highest Diverse Density value (i.e. positive instances describe this concept and none of negative instances describe this concept). [Maron and Lozano-Perez, 1998] tested the Diverse Density algorithm on the Musk data sets with the resultant performance in Table 2.1. [Maron and Lozano-Perez, 1998] also applied the algorithm to the stock prediction application and the image data retrieval task since these two applications can be considered as the multiple-instance learning tasks.

[Maron and LakshmiRatan, 1998] investigated the possibility of using Diverse Density in the natural scene image classification task. They converted each colour images of nature scenes into a set of matrix of colour blob. Simple features such as a row's mean colour, colour differences, and colour distributions among neighbours etc. were used.

[LakshmiRatan et al., 1999] made an improvement by using the segmented region of high resolution images to generate complex instances, using colour, texture and simple geometric properties as features. Segmentation such as that introduced in a research by [Felzenszwalb and Huttenlocher, 1998] was employed to help extend the retrieval of natural scenes images to the retrieval of images based on specific parts (e.g. an images that contains a certain object class like car).

[Yang and Lozano-Perez, 2000] proposed another approach where templates for object shapes are not required. A feature vector of images uses weighted Euclidean distance to reflect a weighted similarity measure which is defined as the distance. A measure of image similarity starts with the correlation coefficient of corresponding regions after smoothing and sampling. By further allowing different weight factor for different locations, weight similarity is compared. This approach, like described in [LakshmiRatan et al., 1999], succeeded in retrieving natural scene images as well as object images.

Another application suggested by [McGovern and Barto, 2001] is finding effective subgoals of reinforcement learning. Effective subgoals of reinforcement learning can be discovered by searching for ‘bottlenecks’. A Bottleneck is a region in the agent’s observation space that the agent visits frequently on successful paths to goal but not on unsuccessful paths. The agent, that is good at discovering and learning to reach the bottlenecks fast, could become more effective at exploring the environment. This is because it can facilitate this learning in similar task. [McGovern and Barto, 2001] treated the task of finding bottleneck regions as a multiple-instance learning problem where the instances are the agent’s observation along the trajectory. A successful trajectory corresponds to a positive bag and negative otherwise. The Diverse Density method was then employed to find the solution to this problem.

2.2.2.2 Expectation Maximisation Based (EM Based)

EM-based multiple-instance learning algorithm - One modification of Diverse Density algorithm based on Expectation Maximisation (EM) is used in a World Wide Web based video query system called the Intelligence Multimedia Processing System (IMIPS) [Xu et al., 2000]. The image classifier section of this system made use of EM-based multiple-instance learning algorithm [Xu and Fu, 2000]. Based on Diverse Density ($DD(t)$), the density function of the class t is a D-dimensional Gaussian mixture with uncorrelated features. Using the model of the mean, variance and the cluster prior probability of each cluster in class t, the probability of t being a concept given subimages can be found. Hence $DD(t)$ can be found. To find t with max $DD(t)$ is performed by comparing the difference between $DD(t)$ of two consecutive t until the difference is smaller than a predefined threshold. Each subimage is features or instance’s features are similar to the ones used in [Maron and LakshmiRatan, 1998]. This system was tested on the natural scene images in five classes: human; star; sky; animal; and fire.

EM-DD - Another modification of the Diverse Density algorithm leading to EM-DD (Expectation Maximisation - Diverse Density) was proposed by [Amar et al., 2001]. Given a hypothesised concept of a bag, instead of having the probability of the label of a bag to be 1 if the label is the same as the one given by a bag, and 0 otherwise, the linear formula was applied to the value of this probability. Their aim was to enable the Diverse Density algorithm to accept real-valued labels. Alternatively, [Zhang and Goldman, 2001] used the Gaussian formulation to calculate the value of the above probability, which was proved to obtain better results than that of the linear formula. The tests were on the Affinity data set, which gives the label of each bag in real-value.

[Zhang and Goldman, 2001] also investigated the statistical algorithm EM, first proposed by [Dempster et al., 1997], to find a concept that maximises the Diverse Density value instead of just using a gradient search (softmax) as in the original. In this integrated method, a search to find the underlying concept is simplified because the search is carried out on one selected instance from each bag instead of a search on every instances like that of the original Diverse Density method. Each selected instance is likely to be the instance responsible for the label given to each of those bags. To compare the EM-DD algorithms with other established algorithms, the tests on the MUSK data sets were carried out with the result in Table 2.1. [Zhang et al., 2002] later tested EM-DD algorithms on a content-based image retrieval task similar to that proposed by [Maron and LakshmiRatan, 1998]. Other researchers, [Ray and Page, 2001], also used EM technique in their multiple-instance regression problem with real-valued data.

2.2.3 Other Approaches

2.2.3.1 A Lazy Learning Approach

[Wang and Zucker, 2000] investigated the use of lazy learning and Hausdoff distance to solve the multiple-instance learning problem. As a result of their research, two variants of K-nearest neighbour algorithm (K-NN) were created: Bayesian-KNN and Citation KNN. The original K-NN assumes that all examples (instances in the supervised learning framework) correspond to points in n-dimensional space and the standard Euclidean distances are computed to describe the nearest neighbour of an example. However, each example (or bag) in multiple-instance problem consists of one or more instances and each instance could correspond to different points in n-dimensional space. As a result, the Hausdoff distance should replace the standard Euclidean distance. However, by adopting the Hausdoff distance alone was proven not to be sufficient for using K-NN to solve the multiple-instance problem because of the two following reasons. Firstly, the K-NN algorithm is very sensitive to which nearest neighbours are chosen in the majority vote method. And secondly, it is very likely that the false positive neighbours are selected resulting in the instance being wrongly classified as positive instead of negative.

To overcome the above problem, [Wang and Zucker, 2000] introduced two possible extensions. The first approach was to introduce the Bayesian probabilistic approach to the majority vote method, hence the name Bayesian-KNN. The second approach was based on the citation of documents. This idea of citation suggested that the algorithm should not only take into account the neighbours of a particular bag but also the bags that count this particular bag as a neighbour, hence the name Citation-KNN. The tests on the Musk data sets showed that Citation-KNN was doing slightly better than Bayesian-KNN (as illustrated in Table 2.1). This approach will be further discussed in comparison to the proposed multi-class Diverse Density in Chapter 7: section 7.3.1. The main reason for not adopting the lazy learning approach to solve multi-class problem is because interactivity feature cannot be introduced with such an approach, see section 7.3.1 for details.

2.2.3.2 Multiple-Instance Neural Network

Multi instance neural network - One of the algorithms that adopted neural network approach is the multi instance neural network developed by [Ramon and Raedt, 2000]. It was originated from the idea that attribute-value learning can be solved by the use of neural networks. In attribute-value learning, the target function that maps each example onto its class, can be represented by a neural network with n input nodes and one output node for the two-class problem (the number of output nodes increase as the number of classes increase). However this cannot be applied to the multiple-instance problem because there is not any function that can map each instance into its class. The class of each instance in the multiple-instance problem is ambiguous (i.e. at least one instance in a positive bag is positive but not all of them are). For each bag of instances, [Ramon and Raedt, 2000] proposed to connect a number of neural networks equal to the number of instances in a bag, to the next level node that decides on the class of all the lower level nodes combined. Each of the lower level nodes represents a target function of the normal attribute-value learning. In other words, each lower level node represents each instance of a bag, while the upper level node represents the class of that bag. Using the backpropagation technique, [Ramon and Raedt, 2000] extended a standard neural network to handle the standard multiple-instance data. Even though adjusting the network representation might help solve multi-class problem but error from the error function cannot be used as an index, similar to the use of the Diverse Density value, in order to establish the interactivity feature. Further detailed discussion on this will be held in Chapter 7: section 7.3.2.

BP-MIP - A second example of multiple-instance neural network is based on the neural network algorithm BP [Rumelhart et al., 1986], hence the name BP-MIP or in other words BP for multiple-instance problems [Zhou and Zhang, 2002]. The network proposed is a feedforward network with p input units where p is equal to the number of the dimension of feature vector representing each instance. These *p* input unit are connected to one hidden layer, where the outputs of this layer are connected to one output unit (the instance label unit). The activation function is the Sigmoid function. If the actual output is more than 0.5 then the instance is regarded as a positive instance, it is negative otherwise. The error function is defined as follows: error is equal to 0 if a bag containing this instance is positive and the actual output is more than 0.5, or if a bag containing this instance is negative and the actual output is less than 0.5. Otherwise the error is equal to half of the square of the difference between 0.5 and the actual output. The weights in the network are modified according to the BP weight-updated rule. The algorithm was only tested on the Musk1 data set achieving the predictive accuracy of 83.8% +/- 9.2. Similar to 'Multi instance neural network', the neural network representation of BP-MIP can be adjusted to cope with multi-class problem but this representation also suffers from not having the right kind of index to help interactivity feature be integrated into the system.

2.2.3.3 Multiple-Instance Learning Problems with Decision Trees and Decision Rules

Multiple-part learning problems with decision trees and decision rules - Zucker and Chevaleyre [Zucker and Chevaleyre, 2000] argued that multiple-instance learning framework is only covered when each instance (like that depicted in Figure 2.3) is a possible description of an object but not when each instance is part of a description of an object. Thus, they proposed the multiple-part problem framework, which covers just this latter situation. Let us consider how a learning problem can be addressed differently within these two frameworks. Consider an application in Chemistry, in case of the multiple-instance learning framework a bag of instances would contain various configurations of the molecule, while in the case of the multiple-part problem a molecule could be cut into small parts such as into a single atom or a pair of bonded atoms to represent each instance.

[Zucker and Chevaleyre, 2000] suggested the extension of decision tree systems such as ID3 [Quinlan, 1986] and C4.5 [Quinlan, 1993] and decision rule systems such as AQ [Michalski, 1969] to cope with the problem of multiple-part learning problem. For a decision tree system, the algorithm should generate two trees instead of one tree like it does in the original. One tree is used for separating true positive instances from those of false positive and negative instances. The other tree is for separating true negative instances from those of false negative and positive instances. They further modified the Entropy and the information gain in the original ID3 system to suit their problem of multiple-part learning. For a decision rule system, each instance of a new object is tested on the rules. The object should be classified as positive if at least one of its instances satisfies the rules describing positive features and that none of its instance satisfies the rules describing negative features.

[Zucker and Chevaleyre, 2000] proposed the problem of predicting the mutagenic activity of a given molecule as a benchmark for their learning framework. Their extended decision trees and decision rules were tested on the available database consists of 230 nitro-aromatic compounds, which is divided into two sets of 188 and 42 molecules. In addition, the idea of this decision tree extension has helped improve the search bias used in propositional rule learning algorithm that was extended to handle multiple-instance data [Chevaleyre and Zucker, 2001]. Unfortunately in the case of the multi-class problem, this system makes it very difficult to identify the order of influence of the underlying instances. This is further discussed in Chapter 7: Section 7.3.3.

RELIC system - RELIC system was developed by [Ruffo, 2000]. RELIC classified an unseen bag of instances in the context of multiple-instance problem by letting each individual instance of a bag vote for the class of that bag with the decision tree approach. The method was applied to the data mining application of computer security problem.

2.2.3.4 Support Vector Machines

Support Vector Machines (SVMs) [Vapnik, 1998] is generalised for multiple-instance learning by Andrews and his colleagues [Andrews et al., 2002]. SVMs are based on the idea of the maximum margin hyperplane, which is defined by some parameters, patterns and margins. Support vector is the pattern with margin. Generalising SVMs means that label of patterns that only appear in a positive bag are treated as an unknown integer variable. In negative bags, the patterns labels are negative. The goal is to soft-margin generalise of SVMs over the continuous parameter and over the integer variables (i.e. labels of patterns in positive bags). The method was tested on a set of images from the Corel image data base. Each image consists of a set of segments, characterised by color, texture and shape discriptors. This method performed at an average precision of 30.3% for the ‘tiger’ category and 46% on the ‘elephant’ category.

2.2.3.5 Ensembles of Multi-Instance Learners

Zhou and Zhang [Zhou and Zhang, 2003] proposed to solve the multiple-instance learning problem by building an ensemble learners [Dietterich, 1997] from four of the multi instance base learners: Iterated-discrim APR (section 2.2.1.1), Diverse Density (section 2.2.2), Citation-KNN (section 2.2.3.1), and EM-DD (section 2.2.2.2). One of the diverse ensemble learning algorithms, Bagging, was used. Bagging uses bootstrap sampling to generate several training sets from the original training set and trains multiple versions of the base learners on each of these training sets. The predictions of each of the learners are combined via majority voting. It is believed that ensembles are more accurate than single learners and as was proven by their work that multi-instance learners can be enhanced by a simple ensemble learning algorithm. EM-DD ensemble was found to be the best method so far for solving the problem of the musk data set. This work can be seen as an enhancement to the methods proposed so far for solving the multiple-instance learning problem but it does not tackle the multi-class problem. Nonetheless ensemble of the proposed multi-class Diverse Density might still be possible.

2.2.3.6 An Open MIL

Similar to EM-DD algorithm discussed in section 2.2.2.2, an open MIL (i.e. an open multiple-instance learning framework) proposed by [Huang et al., 2003] transforms the bag label space from discrete space (1-positve and 0-negative) to a continuous space [0, 1]. In other words, the label of the bag is now indicating the degree to which the bag is positive. It is also assumed that each instance in a particular bag has a label in the interval [0, 1]. Therefore the label of the bag becomes the label of the instance with maximum label. As a result, MIL can be transformed into the unconstrained optimisation problem using the Minimum Square Error (MSE) between the label of the bag and the label of each instance in the bag. Moreover, since the traditional supervised learning problem can also be converted to an unconstrained optimisation using MSE with the substitution of objects instead of instances with maximum label, the gradient-based search methods used in the standard supervised learning can be applied to MIL directly. Such a gradient-based search method includes the steepest descent, Newton method etc. These search methods are open for selection to suit a particular application, hence an open framework. The experiment was conducted on the Musk-1 data set using the Quasi-Newton method. The open MIL method achieved 92.4% accuracy on this data set as shown by Table 2.1.

2.2.3.7 Two-Level-Classification

Weidmann and his colleagues attempted to generalise the multiple-instance learning framework by requiring that instead of one instance, a certain number of instances need to be in

Table 2.1: Performance comparison of various algorithms on the Musk data sets based on the average accuracy across 10 runs using 10-fold cross validation.

Algorithm	**Accuracy (%)**	
	Musk1	**Musk2**
EM-DD	96.8	96.0
Open MIL	92.4	-
Iterated-discrim APR	92.4	89.2
Citation-KNN	92.4	86.3
Bayesian-KNN	90.2	82.4
Diverse Density	88.9	82.5
Two-Level-Classification (TLC)	88.7	83.1
Multi instance neural network	88.0	82.0
BP-MIP	83.8	-
RELIC	83.7	87.3
Multinst	76.7	84.0

a bag of instances for the bag to have positive label. They introduced three different generalised multiple-instance (MI) concepts based on the number of instances in each of the underlying concepts: presence-based MI concepts; threshold-based MI concepts; count-based MI concepts. For the presence-based MI concepts, a bag is labelled positive if it contains at least one instance in each of the underlying concepts. The threshold-based MI concepts require a concept-dependent minimum number of instances of each concept while the number of instances per concept is bounded by an upper and a lower limit in the count-based MI concepts. They then proposed the two-level-classification method (TLC) [Weidmann et al., 2003]. The first level is to construct a single instance from a bag of instances using a standard decision tree. The attributes of this single instance represent region of the instance space and the attribute's value is the number of instances in the bag that are in this region. The detail on how the decision tree is constructed can be found in [Weidmann et al., 2003]. Once a bag of instances is transformed into a meta-instance, the method ends up with a single instance appropriate for propositional learning. The method was tested on the musk data sets with the performance as illustrated in Table 2.1 in comparison to other methods. Since the two-level-classification method was designed to tackle the three generalised concept types mentioned at the beginning of this paragraph, the method was tested on an artificial data that represents the three situations. However the generalisation proposed here is different from what is intended in this research where multi-class is of concern. Even though it is possible to create decision tree for multi-class problem, this method will suffer the same drawback as the method described in section 2.2.3.3 where the order of influence of the class label cannot be identified.

2.2.3.8 Generalised Multiple-Instance Learning

Scott and his colleagues [Scott et al., 2003] generalised multiple-instance learning by assigning the target concept to a set of points instead of a single target concept. The bag label is then determined by whether at least a threshold number of those target concepts are near some points from that bag. If a threshold number is equal to 1 then this model becomes the conventional multiple-instance model. The above generalisation allows certain features to be absent for the bag to be positive labelled. In other words, for positive label, not only certain features have to be present as some points of that bag, but also certain feature must be absent like a repulsion point.

Their learning algorithm was based on geometric pattern learning introduced by Goldman and his colleagues [Goldman et al., 2001]. Instances representation uses axis-parallel boxes corresponding to a weighted infinity norm. As a result, axis rescaling is done implicitly unlike EM-DD and DD algorithm. The algorithm was tested on low-dimensional data in robot vision task, content-based image retrieval and protein sequence identification.

Table 2.2, Table 2.3, Table 2.4, and Table 2.5 were created as a summary for the methods reviewed in Section 2.2. The tables summarises the features of each method and the pros and cons of extending those methods into multi-class framework. From this summary, Diverse Density method was chosen to be extended into the method that can handle multi-class label system because of its simplicity, the order of influence identification capability, and the interactivity integration capability.

Table 2.2: The summary of the features of each method reviewed in Section 2.2 and the pros and cons of extending those methods into multi-class framework.

Method	
Iterated-discrim APR	
Features	the solution comes from finding the smallest axis-parallel hyper-rectangle (APR) that covers at least one instance of each of positive examples and none of any negative example
Pros	extendable
Cons	- the method is not incremental therefore the whole process of obtaining the starting APR of every class needs to be restarted every time a new example is presented to the method
	- time consuming when dealing with multi-class extension
Multinst	
Features	using APR and the probability approach
Pros	extendable
Cons	suffers the same drawbacks as Iterated-discrim APR
Diverse Density	
Features	a measurement of evidence in term of probability to support whether a concept is an underlying concept given training data
Pros	- extendable
	- incremental
	- simple
	- possible interactivity integration
Cons	the probability of the label of a bag is either 1 or 0 and nothing in between which can put a limit in some applications
EM-DD	
Features	- modification of Diverse Density
	- the probability of the bag label is described as linear function
	- statistical algorithm (EM - Expectation Maximisation) is used to find the concept that maximises Diverse Density
Pros	- extendable
	- incremental
	- possible interactivity integration
Cons	- complex
	- requires more resource compared Diverse Density

Table 2.3: The summary of the features of each method reviewed in Section 2.2 and the pros and cons of extending those methods into multi-class framework.

Method	
A Lazy Learning Approach (Bayesian-KNN and Citation-KNN)	
Features	- adopting K-Nearest Neighbour algorithm (KNN) with Hausdoff distance instead of standard Euclidean distance in order to handle multiple instance system - Bayesian-KNN introduces Bayesian probabilistic approach to the majority vote methods - Citation-KNN integrates citation method into finding a neighbour
Pros	- extendable if the majority method can be modified to suit a multi-class label system
Cons	interactivity feature cannot be introduced because parameter used in the algorithm such as distance between two instances can not be used as an index required for interactivity feature
Multiple-Instance Neural Network	
Features	adjusting a neural network representation to suit the multiple instance learning problem
Pros	extendable by adjusting the representation of network and the error function
Cons	interactivity feature is not possible because error from the error function cannot be use as an index required for interactivity feature
Decision Trees and Decision Rule Approach	
Features	creating decision trees and decision rules for each individual class of examples
Pros	extendable
Cons	- requires more resources as more trees and rules must be generated for an additional class - the order of influence cannot be identified as trees or rules only separate instances into groups - interactivity is not possible because entropy in decision tree and rules in decision rule cannot distinguish good examples that will speed the learning from the bad ones

Table 2.4: The summary of the features of each method reviewed in Section 2.2 and the pros and cons of extending those methods into multi-class framework.

Method	
Support Vector Machines (SVMs)	
Features	generalises SVMs to label patterns in positive bags and patterns in negative bags
Pros	extendable but with complexity in separating into many classes
Cons	- time consuming - high cost for computation
Ensembles of Multi-Instance Learners	
Features	ensembling of different types of multiple instance learning algorithm such as Iterated-discrim APR, Diverse Density, Citation-KNN, and EM-DD where the predictions of each learner are combined via majority voting
Pros	- the ensemble inherits the feature from each of the combined methods - an enhancement to each of the combined methods
Cons	requires more resources
An Open MIL	
Features	- similar to EM-DD, it transforms the bag label space from discreet space (positive, negative) to continuous space [0,1] - it also transforms multiple instance learning problem into the unconstrained optimisation problem where a gradient-basedsearch can be easily applied
Pros	an open framework
Cons	extendable but with complexity
Two-Level-Classification	
Features	- more generalisation of the multiple instance learning framework - two level classification, first using decision tree to create a single instance from a bag of instances then propositional learning is applied to this meta instance system
Pros	extendable
Cons	suffers the same drawbacks as decision trees approach

Table 2.5: The summary of the features of each method reviewed in Section 2.2 and the pros and cons of extending those methods into multi-class framework.

Method	
	Generalised Multiple-Instance Learning
Features	- assigning the target concept to be a set of points instead of a single target concept - allows the absence of certain features for the bag to be labelled positive - the algorithm is based on geometric pattern learning
Pros	more generalised in term or bag label system
Cons	unnecessarily complicated if to expand to multi-class system because of the complex bag label system

Chapter 3

Diverse Density for Multi-Class Problem

This chapter explores the possibility of extending the original Diverse Density method [Maron, 1998] used for solving multiple-instance learning tasks, from a two-class category to a multi-class category. Emphasis has already been made on the use of the multiple-instance learning scheme and the need to extend this scheme to the multi-class problem for the purpose of concept formation of ambiguous object in Chapter 1. This chapter will start with the discussion about the multiple-instance learning framework. It is then followed by the detailed summary of the original Diverse Density method including its definition, the calculation of the of Diverse Density value, and various algorithms or techniques used to obtain the maximum Diverse Density value from training examples in order to identify the solution to the learning problem. In the last section, we propose the extended definition of the multi-class Diverse Density value, the possible ways to calculate its value, and algorithms to find its maximum value.

3.1 The Multiple-Instance Learning Framework

The multiple-instance learning problem is the problem associated with ambiguity of the training examples or objects. Examples or objects in the multiple-instance learning problem are ambiguous because rather than having only one instance to describe a single object, there will be many instances responsible for the description of this single object. Yet not all of these instances are responsible for the classification of the label of this single object. For example, a drug molecule can be described by various conformations but only one conformation will bind with the disease protein, hence without knowing exactly which conformation the molecule takes during the binding process, the drug molecule will still be labelled positive. As a second example, consider a scenery images, it will contain various features (mountain, forest, waterfall etc.) but a few of these will be desirable by the user. The main goal in the multiple-instance problem is for the learner to accurately predict the label of unseen objects. A further goal is to isolate the true instance responsible for the label of the object from the false instances.

The researches carried out so far in the area of the multiple-instance learning framework has been limited to the two-class problem (i.e. examples or objects are usually labelled either positive or negative). The problem can be stated as follows: “The task is to learn a concept behind the label of positive and negative bags of instances. A bag is labelled positive even if only one of the instances in the bag falls within the target concept to be learned, while a bag is labelled negative if all the instances in this bag are negative (i.e. none of the instances in the negative bag falls within the target concept).” [Dietterich et al., 1997]

Although the two-class problem might be the most common and simplest scenario in many applications investigated so far (e.g. drug discovery), but there exists the need to extend the two-class problem to a multi-class problem for more accurate learning of ambiguous objects. In some situation the concept underlying the positive class is not the only thing we need to look for because the answer to this more general learning problem can be a combination of the concept underlying each individual class, which may be more than two classes. In the multi-class problem, a bag of instances will be labelled into classes (e.g. class_1, class_2, ... , class_N) according to whether there exists the true instance underlying specific class in that bag. Further if there are multiple true instances from different classes in that bag, the most influential instance will dictate the label of the bag. One example of multi-class labelling is illustrated in Figure 3.1. Let us assume that the number 3 is the true instance underlying the number class while the square instance is the true instance underlying the geometric class. Here the order of influence is such that the geometric class dominates or is more influential than the number class. Hence bag_2 is labelled geometric class even though the bag contains the number 3 as well.

In the original two-class problem, the positive class can be seen as having more influence over the negative class (i.e. instances that underlie the positive class will always exist in all the positive bags but never in any of the negative bags). Furthermore, the goal in the multi-class problem is more specific than that of the two-class problem. Thus the Concept underlying each individual class has to be identified, unlike in the two-class problem where only the underlying concept for the positive class is identified. Moreover, the

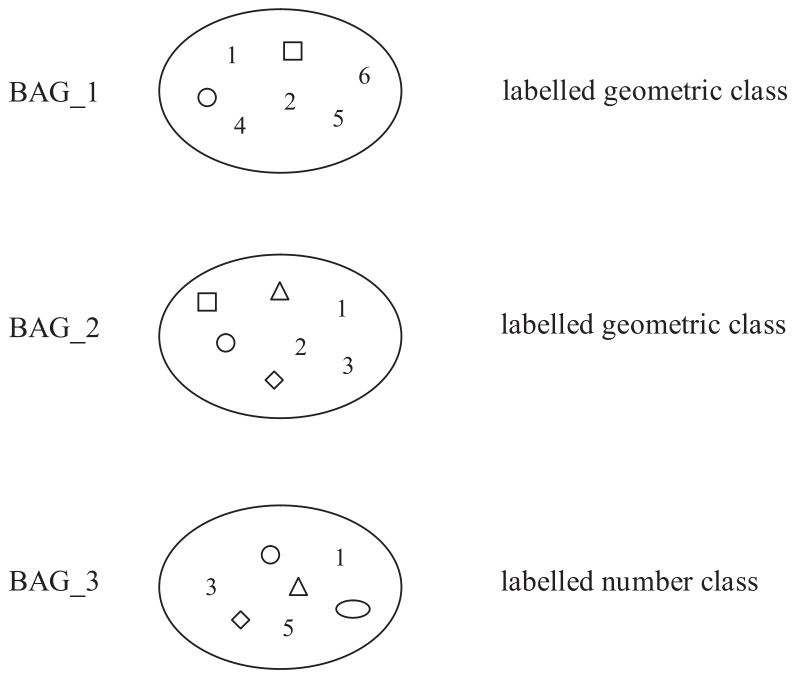


Figure 3.1: Bag of instances labelling system within multi-class problem.

order of influence among the class labels must also be identified. However, it is not possible to use the same learning algorithm used in the original two-class problem with the multi-class problem without any modification.

This research proposes to modify the Diverse Density method, which has already been employed to solve the original two-class problem. Before concentrating on the modified method, the original Diverse Density method will be expanded upon in the next section.

3.2 The Original Diverse Density Method

So far, in Chapter 2: section 2.2, a number of methods for solving the original two-class multiple-instance learning problem have been examined. One of these methods was the original Diverse Density, proposed by [Maron, 1998]. The method proposed that the solution to the original multiple-instance learning problem is a point/points in the area where positive instances have a high density and that this point is as far away as possible from the negative instances. The method identifies these points as the points with the highest Diverse Density values. Diverse Density at a point in feature space is a measure of how many different positive bags have instances near that point and how far the negative instances are from that point. The higher the number of positive bags containing instances near to the point and the lesser the number of negative instances near to the same point, the higher the value of Diverse Density of that point. The Diverse Density method is highly flexible since it allows different probability models describing the Diverse Density to be used as long as the Diverse Density value is the highest at the target concept. Therefore different models can be used to solve different problem scenarios depending on which model is better for describing the scenario. As an example, whilst some scenarios are suitable for a discreet model, some are suitable for a continuous model. Apart from being flexible, the Diverse Density method is incremental because Diverse Density at a given point in a feature space is the multiplication of Diverse Density at that point given each bag of instances, therefore only requiring a multiplication between the previous value of Diverse Density and the new value obtained form a new bag of instances. By adjusting the probability model, the Diverse Density method can be modified to handle the multi-class problem as well as inheriting the flexible and incremental properties of the original method. A more detailed look at the Diverse Density method will now be given, this will be done over the next two subsections. The first subsection deals with the definition of Diverse Density and how its value can be calculated. Then the second subsection discusses how Diverse Density can be employed to learn concepts.

3.2.1 The Computation of Diverse Density

The Diverse Density definition was given by [Maron, 1998] as the probability that a specific concept (i.e. a specific point in a feature space) is the target concept given the training examples. Hence by maximising the Diverse Density over all points in a feature space, the target concept can be found.

Let us denote B_i^+ as the positive bag i, B_{ij}^+ as the j_{th} instance in the positive bag i, and B_{ijk}^+ as the k_{th} feature of the j_{th} instance. Likewise, a j_{th} instance of a negative bag i is represented as B_{ij}^-. The probability that a specific concept t in a feature space is the target concept is written as $Pr(t)$. Hence given n positive bags and m negative bags, the Diverse Density of the t_{th} concept is $DD(t)$ where

$$DD(t) = Pr(t|B_1^+, \dots, B_n^+, B_1^-, \dots, B_m^-) \tag{3.1}$$

By applying Bayes' rule to the above equation 3.1, the right hand side is expanded into three terms. The first term is $Pr(t)$ and is constant because any prior knowledge over the concept location is not assumed. The second term, $Pr(B_1^+, \dots, B_n^+, B_1^-, \dots, B_m^-)$, is also constant with respect to t. Therefore these two terms need not to be computed if it is only required to perform the comparison between different concepts. Hence finding the target concept or maximising $DD(t)$ over all points in a feature space (*w.r.t.* t) has been reduced to maximising just the third term:

$$\max\ DD(t) = Maximising\ the\ likelihood\ Pr(B_1^+,\ldots,B_n^+,B_1^-,\ldots,B_m^-|t) \tag{3.2}$$

With the additional assumption that all bags are conditionally independent given the target concept, the eqation 3.2 becomes:

$$\max\ DD(t) = \arg\max_t \prod_{1\le i\le n} Pr(B_i^+|t) \prod_{1\le i\le m} Pr(B_i^-|t) \tag{3.3}$$

In general applications, it is not known how the bags of instances will be generated. In other words, the bags' generative model is not known, hence the term $Pr(B_i^+|t)$ and $Pr(B_i^-|t)$ cannot exactly be found. However by reapplying Bayes' rule to equation 3.3 and inserting it into the definition of $DD(t)$ (i.e. this new term replaces the third term after equation 3.1 was applied by Bayes' rule), $DD(t)$ can alternatively be expressed in term of $Pr(t|B_i)$. By assuming a uniform prior knowledge over the concept location, the term $Pr(t)^{n+m-1}$ is a constant. While another term in the new expression, $(\prod_{1\le i\le n} Pr(B_i^+) \prod_{1\le i\le m} Pr(B_i^-))/(Pr(B_1^+,\ldots,B_n^+,B_1^-,\ldots,B_m^-))$, is also constant with respect to t, and disappears if it is assumed that the bags are generated independently. As the result, the above likelihood equation 3.3 becomes:

$$\max\ DD(t) = \arg\max_t \prod_{1\le i\le n} Pr(t|B_i^+) \prod_{1\le i\le m} Pr(t|B_i^-) \tag{3.4}$$

where $Pr(t|B_i) = Pr(t|B_{i1},B_{i2},\ldots,B_{ij})$ if bag i has j instances.

As a summary:

$$DD(t) = (constant_{w.r.t.\ t\ and\ uniform\ prior}) \prod_{1\le i\le n} Pr(t|B_i^+) \prod_{1\le i\le m} Pr(t|B_i^-) \tag{3.5}$$

And because a general estimator for $Pr(t|B_i)$ can be found, the maximum Diverse Density can also be found in terms of the products of $Pr(t|B_i)$. This is also easier to compute because the instances are used as pieces of evidence for potential concept locations rather than the generative model of the bags of instances.

Maron investigated three different models of estimators: all-or-nothing; noisy-or; and most-likely-cause. In order for $DD(t)$ to have the highest value when t is the target concept, each estimator should model the probability $Pr(t|B_i)$ so that $Pr(t|B_i^+)$ is high and

$Pr(t|B_i^-)$ is low when t is close to instances in bag i and vice versa when t is far way from instances in bag i. The difference of each model is in the detail such as whether the probability is continuous or discrete etc. In other words, $Pr(t|B_i)$ will be described differently in term of $Pr(B_{ij} \in t)$ for each estimator model. For example, in all-or-nothing estimator model:

$$Pr(t|B_i^+) = 1 \quad at\ B_{ij}^+ \in t\ \ and\ 0\ otherwise \tag{3.6}$$

$$Pr(t|B_i^-) = 0 \quad at\ B_{ij}^- \in t\ \ and\ 1\ otherwise \tag{3.7}$$

As for noisy-or [Pearl, 1988], $Pr(t|B_i)$ is assigned as follows:

$$Pr(t|B_i^+) = 1 - \prod_{1 \le j \le p} (1 - Pr(B_{ij}^+ \in c_t)) \tag{3.8}$$

$$Pr(t|B_i^-) = \prod_{1 \le j \le p} (1 - Pr(B_{ij}^- \in c_t)) \tag{3.9}$$

where p is the number of instances in bag i and $Pr(B_{ij} \in c_t)$ is the probability of a particular instance being the target concept.

If t is a target concept, it is because one of the instances in one of bags suggested it. It is also assumed that the probability of instance j not being the target is independent of any other instance not being the target.

Single point concept class is the simplest concept class, where every concept corresponds to a single point in a feature space. And because a single point in a feature space represents an instance, a single concept corresponds to a single instance. In this case, [Maron, 1998] estimated $Pr(B_{ij} \in c_t)$ using a Gaussian-like distribution of

$$Pr(B_{ij} \in c_t) = \exp(-\sum_{1 \le k \le l} (B_{ijk} - c_{tk})^2) \tag{3.10}$$

Where B_{ijk} is the k feature value of the j_{th} instance of the i_{th} bag and c_{tk} is k feature value of concept c_t. $\sum_{1 \le k \le l} (B_{ijk} - c_{tk})^2$ can also be viewed as the distance between an individual instance and the potential target. The consequences of estimating $Pr(B_{ij} \in c_t)$ in this manner are that :

(i) if one of the instances in a positive bag is close to the concept c, then $Pr(t|B_i^+)$ will be high, on the contrary, if one of the instances in a negative bag is close to c, then $Pr(t|B_i^-)$ will be low;

(ii) if every positive bag has an instance close to *c* and no instance in any negative bag is close to *c*, then *c* will have the highest Diverse Density value;

(iii) as the number of bags increases, Diverse Density at an intersection of those bags grows exponentially.

The proceeding shows an example where the value of Diverse Density for each concept can be illustrated in a two dimensional diagram. Other examples might be hard to illustrate this way because there is not a one-to-one mapping between concepts and instances. The example is a number set where a set (or bag) of numbers is labelled according to the existence of a specific number in that set. Let us say, there are five numbers from 1 to 5 and the existence of number 1 in a set or a bag will make the label of this bag positive and negative otherwise. The first bag given contains numbers 1, 2, and 3. Numbers 3 and 4 are in the second bag and numbers 1, 4, and 5 are in the third bag. Hence we have B_1^+: 1, 2, 3; B_1^-: 3, 4; B_2^+: 1, 4, 5.

The all-or-nothing model is adopted for $Pr(t|B_i)$, $Pr(t|B_i^+) = 1$ if positive bag i has t as one of its instances and 0 otherwise, $Pr(t|B_i^-) = 0$ if negative bag i has t as one of its instance and 1 otherwise. For example, consider the first bag of instances, $DD(t)$ of concept $(t = 1, t = 2, t = 3)$ will all equal to 1 because the numbers 1, 2, and 3 exist in this bag and this bag is a positive bag. $DD(t)$ for each of all the possible concepts (i.e. $t = 1, \dots, 5$ for the above example) is then computed. Having done that, the $DD(t)$ distribution based on the evidence from the first bag of instances (i.e. the top left graph in Figure 3.2) is obtained. The same procedure is carried out for the second and the third bag of instances (i.e. the middle and bottom left graph of Figure 3.2). According to the earlier proof (see equation 3.5), in order to combine all the evidence together, $DD(t)$ from each bag is multiplied in order to derive the final distribution of $DD(t)$ (the right hand column graphs of Figure 3.2). In this example, it was found that concept the $(t = 1)$ had the highest $DD(t)$, hence $t = 1$ is the target concept. In other words, whenever there is number 1 in a bag of instances, the bag will be labelled positive, in any other case it will be negative.

On the other hand, if noisy-or estimator is employed, the development of the value of Diverse Density as new bags of examples are introduced to the system can be illustrated as in Figure 3.3.

This concludes the section on calculation of Diverse Density. In the next section the existing algorithms for concept learning is tackled.

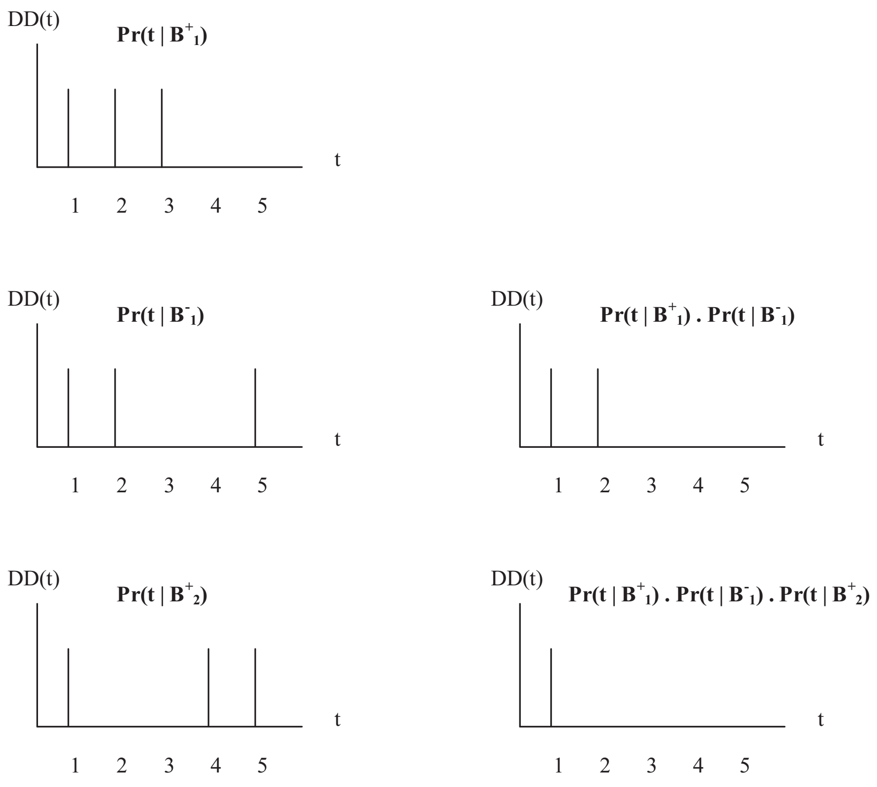


Figure 3.2: Calculation of $DD(t)$ for each concept t, given three bags of instances within the 1-dimensional number domain (B_1^+: 1, 2, 3; B_1^-: 3, 4; B_2^+: 1, 4, 5). $Pr(t|B_i)$ uses the all-or-nothing model where $Pr(t|B_i^+)$ = 1 if positive bag i has t as one of its instances and 0 otherwise, $Pr(t|B_i^-)$ = 0 if negative bag i has t as one of its instance and 1 otherwise.

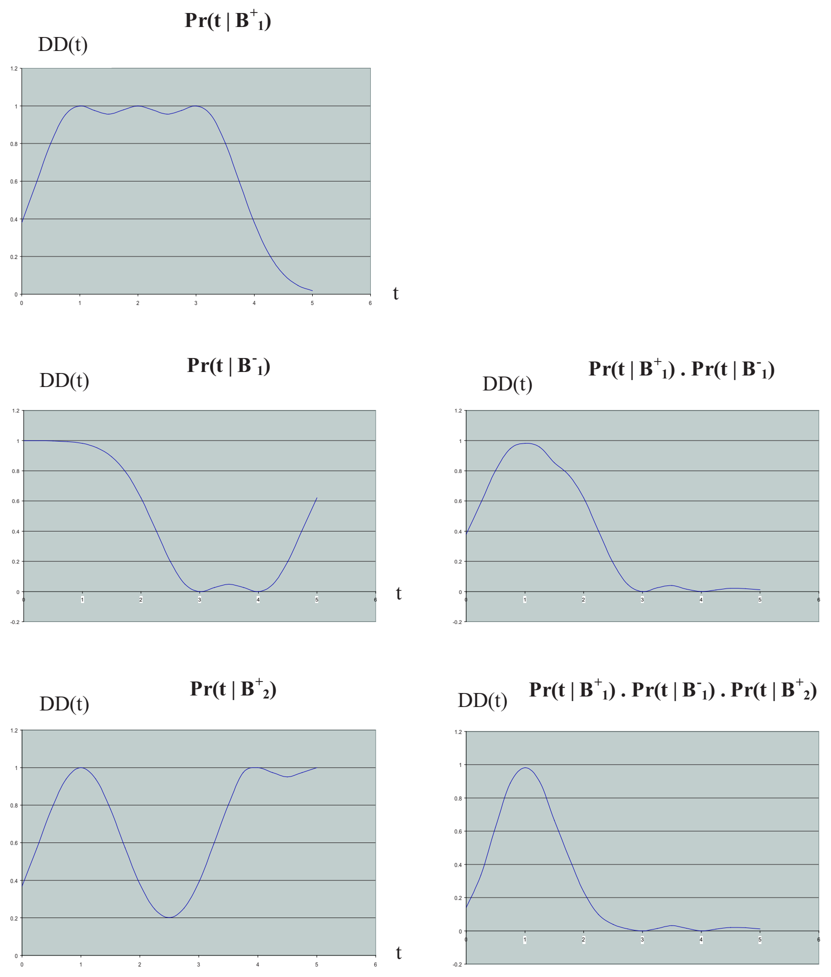


Figure 3.3: Calculation of $DD(t)$ for each concept t, given three bags of instances within the 1-dimensional number domain as before (B_1^+: 1, 2, 3; B_1^-: 3, 4; B_2^+: 1, 4, 5). The calculation is based on the noisy-or model where $Pr(t|B_i)$ is model as follows: $Pr(t|B_i^+) = 1 - \prod_{1\leq j\leq p}(1 - Pr(B_{ij}^+ \in c_t))$, and $Pr(t|B_i^-) = \prod_{1\leq j\leq p}(1 - Pr(B_{ij}^- \in c_t))$ with p equals to the number of instances in bag i. The probability of a particular instance being in the target concept, $Pr(B_{ij} \in c_t)$, is a Gaussian based on the distance from the particular instance to the target concept, $\exp(-\sum_{1\leq k\leq l}(B_{ijk} - c_{tk})^2)$.

3.2.2 Concept Learning Algorithm Using Diverse Density

According to the original definition of Diverse Density, the target concept to be learned is the concept with maximum Diverse Density. It is possible to measure the value of Diverse Density at every concept in order to find the maximum value if the size of the concept class is not too big. However in a real world problem (e.g. drug discovery, stock prediction etc.), the concept class can be as large as the entire feature space, leading to unattractive compute time. Therefore, a further algorithm must be introduced to help the learner find the concept with highest diverse density value.

So far there have been four algorithms developed to learn concepts using the Diverse Density method. The first three were proposed by Maron and his colleagues: maxDD - maximising Diverse Density; PWDD - Pointwise Diverse Density; a parallel implementation of maxDD and PWDD. The fourth algorithm is Expectation Maximisation an (EM) based method, which has already been described in Chapter 2: section 2.2.2.2. The following briefly describes each one.

- **maxDD**

  The maximising Diverse Density (maxDD) is straightforward as it simply calculates the derivatives with respect to a concept class of all the term in equation 3.4 and perform a gradient search in concept space to find the global maxima of Diverse Density. Heuristic can be added to the gradient search. For example, by starting an optimisation search only at every instance in every positive bag (none from the negative bags), at least one starting point will be close to the maximum Diverse Density concept.

- **PWDD**

  Unlike the maxDD algorithm, where its goal is to return the concept that maximise Diverse Density. The goal of PWDD - Pointwise Diverse Density is to return the correct label for every instance in the training bags. In the two-class problem, every instance in the negative bags will be labelled negative because according to multiple-instance learning framework, every instance in negative bags is the true negative. On the other hand, there are both 'true positive instances' and 'false positive instances' in every positive bag. PWDD attempts to separate the two types of positive instances by:

  1. Computing Diverse Density is only at the concept corresponding to instances

in positive bags.

2. Returning the instance with the highest Diverse Density in each of the positive bags.

Because the PWDD algorithm returns the instances closest to the true positive instance or even the true positive itself, PWDD does not need to perform the expensive gradient search, but rather only needs to compute Diverse Density several times. The total number equals to the number of positive instances.

3.3 Multi-Class Diverse Density

The Diverse Density method was designed to be a learning method for the multiple-instance learning framework. However the original method can only deal with the two-class problem. In a two-class problem each training example (or bag of instances) of the multiple-instance learning task will be labelled either one of the two classes. Furthermore examples from the first class is assumed to always be ambiguous, or rather this implies that it is not known within each such example which instance underlies this class label. While examples belonging to the second class are not because they are labelled on the basis that they can never be an underlying instance of the first class. Looking at research in the field of the multiple-instance learning framework, the common scenario is to associate these classes as positive or negative. Only a few instances in each example from the positive class are responsible for examples being labelled positive, while all instances in a negative example are responsible for the example being labelled negative. In other words, positive instances are ambiguous whereas none of the negative ones are.

In the proposed multi-class problem, there are two or more classes of examples, and in particular, instances in each example of every class are ambiguous. Moreover instances underlying one class could influence or over-shadow those of other classes resulting in examples (or bags of instances), where instances underlying different classes are present, labelling the more powerful class. The modification of the Diverse Density method is proposed in order to find the true instances underlying each different class of examples and to identify the order of influence of each class. Again similar to section 3.2, the computation of the multi-class Diverse Density will be discussed first. This is then to be followed by the learning algorithms. However in this section, there is another subsection to compare the proposed multi-class Diverse Density method with another alternative solution to the same problem.

3.3.1 Computation of The Diverse Density for The Multi-Class Problem

The original definition states that Diverse Density is the probability that a specific concept (i.e. a specific point in a feature space) is the target concept given the training examples. The target concept in this case is the concept underlying the positive example, while the target concept within multi-class problem changes from the concept underlying one class to another. However the original definition of Diverse Density should still hold for the multi-class problem as well, as long as we consider the concept underlying each class separately from each other. Furthermore, the equation 3.1 has to be adjusted accordingly to accommodate bags of instances with multiple class labels.

$$DD(t^r) = Pr(t|B_1^1,\ldots,B_n^1,B_1^2,\ldots,B_m^2,\ldots,B_1^p,\ldots,B_o^p) \tag{3.11}$$

where $DDm(t^r)$ denotes multi-class Diverse Density of concept t being a target concept underlying a particular class r and B_o^p is bag number o of a class_p bag.

Like before, Bayes' rule was applied to the right hand side resulting in three new terms. The first term, $Pr(t)$, will be constant due to the assumption that there in no prior knowledge over the concept location. The second term, $Pr(B_1^1,\ldots,B_n^1,B_1^2,\ldots,B_m^2,\ldots,B_1^p,\ldots,B_o^p)$, is also constant with respect to t. Therefore finding the target concept or maximising $DDm(t^r)$ over all points in a feature space $(w.r.t.\ t)$ has reduced to maximising just the third term:

$$\max\ DDm(t^r) = Maximising\ the\ likelihood\ Pr(B_1^1,\ldots,B_n^1,B_1^2,\ldots,B_m^2,\ldots,B_1^p,\ldots,B_o^p|t) \tag{3.12}$$

Again, with the additional assumption that all bags are conditionally independent given the target concept, the above becomes:

$$\max\ DDm(t^r) = \arg\max_t \prod_{1\le i\le n} Pr(B_i^1|t) \prod_{1\le i\le m} Pr(B_i^2|t) \ldots \prod_{1\le i\le o} Pr(B_i^p)|t) \tag{3.13}$$

As before, the term $Pr(B_i|t)$ cannot exactly be found because it is not known how the bags of instances will be generated. However by reapplying Bayes' rule to equation 3.13 and inserting it into the definition of $DDm(t^r)$ (i.e. this term replaces the third term after Bayes' rule was applied to equation 3.11), $DDm(t^r)$ can alternatively be expressed in term of $Pr(t|B_i)$. By assuming a uniform prior knowledge over the concept location, the term $Pr(t)^{n+m+\ldots+o-1}$ in the new expression is a constant. A further term, the division of the term $(\prod_{1\leq i\leq n} Pr(B_i^1)\prod_{1\leq i\leq m} Pr(B_i^2)\ldots\prod_{1\leq i\leq o} Pr(B_i^p))$ by the term $(Pr(B_1^1,\ldots,B_n^1,B_1^2,\ldots,B_m^2,\ldots,B_1^p,\ldots,B_o^p))$, is also constant with respect to t, and disappears if it is assumed that the bags are generated independently. As the result, the above likelihood of equation 3.13 becomes:

$$\max\ DDm(t^r) = \arg\max_t \prod_{1\leq i\leq n} Pr(t|B_i^1)\prod_{1\leq i\leq m} Pr(t|B_i^2)\ldots\prod_{1\leq i\leq o} Pr(t|B_i^p) \tag{3.14}$$

where $Pr(t|B_i) = Pr(t|B_{i1},B_{i2},\ldots,B_{ij})$ if bag i has j instances.

Therefore:

$$DDm(t^r) = (const_{w.r.t.\ t\ and\ uniform\ prior})\prod_{1\leq i\leq n} Pr(t|B_i^1)\prod_{1\leq i\leq m} Pr(t|B_i^2)\ldots\prod_{1\leq i\leq o} Pr(t|B_i^p) \tag{3.15}$$

Notice the obvious difference between the expression for the original Diverse Density $(DD(t))$ and the multi-class Diverse Density $(DDm(t^r))$ is that a greater number of classes of bags of instances are involved in $DDm(t^r)$. However this is only the first obvious difference between the two methods, there are still other things to be considered. First of all, a summary of the commonality between the original Diverse Density and the new method (multi-class Diverse Density) will be given, followed by a look at what requires modification in order to satisfy the multi-class Diverse Density method. For clarification, the last topic in this subsection will give an example showing the computation of multi-class Diverse Density.

3.3.1.1 Commonality to The Original Computation of Diverse Density

To summarise, the definition of Diverse Density given earlier by [Maron, 1998] remains unchanged. Restated here as; the Diverse Density of a specific concept is the probability that a specific concept is the target concept given the training examples. A slight deviation is that in the original the concept underlying only the positive class was of interest, in multi-class it is the concept underlying each specific class that is of interest. In addition, the assumption that no prior knowledge over concept location together with the assumption that all bags of instances (a.k.a. examples) are conditionally independent given the target concept, remain unchanged and are still valid. As a result, the Diverse Density for either the two-class or the multi-class problem is to be viewed as the combined evidence from each individual bag of instances.

3.3.1.2 The Differences Between The Original Diverse Density and The Multi-Class Diverse Density

Although both Diverse Density and multi-class Diverse Density were mathematically proven to be the measure of the combined evidence from each individual bag of instances (i.e. equation 3.5 and equation 3.15), these two methods will need different models for the estimation of the probability of t being target concept given bag i ($Pr(t|B_i)$). However, both methods require that the value of Diverse Density to be the highest at the target concepts and low elsewhere.

In the case of the original two-class problem, the target concept is the concept underlying positive class. As a result, $Pr(t|B_i^+)$ is modelled to be very high if at least one of the instances in a positive bag is close to concept t while $Pr(t|B_i^-)$ is very low for all of the instances in a negative bag so that the Diverse Density of the target concept will have the highest value. This is consistent with the theory of the Diverse Density value; what is in common among the positive bags and does not appear in the negative bags (i.e. the soft version of ‘the intersection of the positive bags minus the union of the negative bags’ [Maron, 1998]).

While dealing with multi-class problem, the model of $Pr(t|B_i)$ should take into account the following:

(i) Not only the concept underlying positive examples is searched but the concepts underlying other classes of examples are also searched. As a result, the value of multi-class Diverse Density should be modelled to be high at each of the concepts underlying each class of examples.

(ii) The order of which class has stronger influence on the label of bags of instances when two or more of the instances underlying different classes are in the same bag should also be identified. Therefore whenever the above circumstances occur, the multi-class Diverse Density value should be modelled to be high at the concept underlying a more powerful class and lower at a less powerful one accordingly.

Expanding on (i), this is achieved by adopting an estimator for $Pr(t|B_i^+)$ such as all-or-nothing or noisy-or as proposed by [Maron, 1998]. However instead of concentrating only on the positive class where $Pr(t|B_i^+)$ will be high if t is close to at least one of the instances in a positive bag, the same model is adopted for every class. Therefore when finding the target concept underlying class p, $Pr(t|B_i^p)$ is also high if t is close to one of the instances in a bag of class p. For all classes other than p, $Pr(t|B_i)$ of bags of instances for those classes still requires modeling. This cannot be modelled in the same way as it is with the negative bags in the two-class problem because not all instances within those bags are true instances underlying those classes. In other words, there are false instances in every class of the multi-class problem unlike the all true-instances assumption for negative bags in the two-class problem.

However it is possible to make other assumption about those instances that will be useful for the modelling of $Pr(t|B_i)$. The instances that exist both in at least one bag of class p and at least in a bag of one of other classes (i.e. joint instances) are unlikely to be the true instances underlying class p. Therefore a concept that is close to such instances should not have a high Diverse Density value. When each new bag of instances is presented to the learning system, more information about joint instances can be found. As a result, it was chosen to model the Diverse Density of concept t being a target concept of particular class to have a high value only where t is close to true instances of that class and low where t is close to joint instances. This idea can also be expressed in the form of the soft set operation, as already proposed in Chapter 1: section 1.2 as follows; "the intersection of the bags of Class_N minus the joint instances from bags of different classes available".

Referring to (ii), the computation of multi-class Diverse Density should help identify the order of influence to the bag labelling of true instances underlying different classes. It is proposed that, with an appropriate probability density model, the addition of multi-class Diverse Density $DDm(t^r)$ for every class r will be used to identify the order of influence between true underlying concepts. The use of different probability models is further discussed in the next chapter (Chapter 4: section 4.1).

Hence, once the addition value has been ordered, the concept with high influence is obtained. Whichever class these concepts underlie is the class where these concepts have highest $DDm(t^r)$.

3.3.1.3 An Example of The Computation of Multi-Class Diverse Density

Below is one of the suggested procedures to compute the value of multi-class Diverse Density. This procedure is aimed at a process where a bag of instances is given one at the time to the learning system. The procedure was also designed on the basis that as much information as possible must be drawn for each bag given to the system. Hence the concept underlying each class of examples does not need to be found one at a time. That is to say when multiple number of bags of instances are given at the same time, it is more cost effective to carry out the intersection operation on every bag of the same class first.

STEP 1: For a given bag of instances of class p, compute $Pr(t|B_i^p)$ according to the model stating $Pr(t|B_i^p)$ is high where t is close to one of the instances in the bag and low otherwise.

STEP 2: As the second bag of instances of class q is given, again compute its $Pr(t|B_i^q)$.

STEP 3: If the second bag is of the same class as the first one $(p = q)$, then obtain the product between the two probabilities of the two previous steps (i.e. this is equivalent to doing the soft intersection of the bags from the same class).

Otherwise, find the instances that appear in at least one bag of one class and in at least one bag of another class (i.e. finding joint instances which will be substituted as J), compute $Pr(t|J)$ according to the model that $Pr(t|J)$ is low where t is close to the joint-instances and high otherwise, and then obtain the product between $Pr(t|J)$ and $Pr(t|B_i^p)$ from each class (i.e. this equivalent to doing the soft minus operation of the joint-instances and the soft intersection of the bags of the same class).

STEP 4: Add all the resultant probabilities together (i.e. the addition of $DDm(t^r)$ for every class r, having r class means having r different $DDm(t^r)$).

STEP 5: As more bags are given, repeat from STEP 2 to 4.

STEP 6: Once all bags are given, starting from the concept with the highest addition of multi-class Diverse Density (i.e. addition of $DDm(t^r)$ of every class r), compare final product between $Pr(t|J)$ and $Pr(t|B)$ of each class for that concept, the concept will underlie the class where this product has the highest value.

Let us now look at an example showing the use of the above procedure for the computation of multi-class Diverse Density. Referring to the number set example in section 3.2.1, number 1 is still the underlying instances for a bag of numbers to be labelled positive. However this time number 4 is also the underlying instance for negative bags. Moreover a bag of numbers will also be labelled positive if number 1 is in the bag regardless of the presence of number 4 in that bag (i.e. number 1 is a more influential instance). This scenario fits a multi-class multiple-instance problem. The learner needs to identify that number 1 is the true instance for positive class bags, number 4 is for that of negative class bags, and number 1 is more influential than number 4. In this example, the all-or-nothing estimator is adopted when calculating $DDm(t^r)$, $Pr(t|B_i^r)$ = 1 at concept t being one of the instances in a bag of class r and 0 otherwise, $Pr(t|J) = 0.5$ at concept t being one of the joint instances and 1 otherwise.

Figure 3.4 shows how the multi-class Diverse Density is computed for the adjusted number set problem. For example, consider the first bag of instances, $DDm(t^+)$ of concept $(t = 1)$ will equal to 1 because the first bag of instances is a positive bag and number 1 exists in this bag. $DDm(t^+)$ is then computed for each of all the possible concept given the first bag. Having done that, the $DDm(t^+)$ distribution based on the evidence from the first bag of instances (i.e. the top left graph of Figure 3.4) is obtained. The same procedure is then carried out for $DDm(t^r)$ where r is the class label of the second and third bag of instances (i.e. the top right and the graph on the second row on the left of Figure 3.4). Then multiply $DDm(t)$ from different bags of the same class *(intersection)*, resulting in the graph on third row on the left of Figure 3.4. Next the joint instances are obtained (instance 3 and 4). $Pr(t|J)$ is then assigned as in the graph on the third row on the right of Figure 3.4. Follow by multiplication of each $DDm(t^r)$ for every class r with $P(t|J)$ *(minus)* as in the two graph on the fourth row of Figure 3.4. Finally the addition is obtained for each of the resultant $DDm(t^r)$ where r is either a positive or negative class. It can then be concluded that number 1 is the underlying concept for a positive class and has more influence than number 3 and 4, which are the concepts underlying the negative class.

The assumption is made that another bag of instance is given $(B_2^- : 4)$. The first step is to update the soft *intersect* operation of the negative bags, followed by the update of the soft *minus* and *addition*. The update values are shown in Figure 3.5. Now, looking at the bottom left graph in Figure 3.5, there is enough evidence to conclude that number 1 is the underlying concept for a positive class and has more influence than number 4, which is the concept underlying the negative class.

3.3.2 Algorithm to Identify True Instances

The original goal was to use the solution from multi-class multiple-instance learning task to solve the problem of object ambiguity in concept formation task. The ambiguity occurs because objects are described as a group of instances and yet only a few instances are responsible for the class-label of objects. This ambiguity should be diminished by isolating the true instances responsible for the class-label of an object from other instances present within the same object. Consequently, it means that an algorithm that can identify true instances within multi-class multiple-instance learning task is required. The first attempt is to adopt and adjust PWDD in the original Diverse Density method (section 3.2.2) in

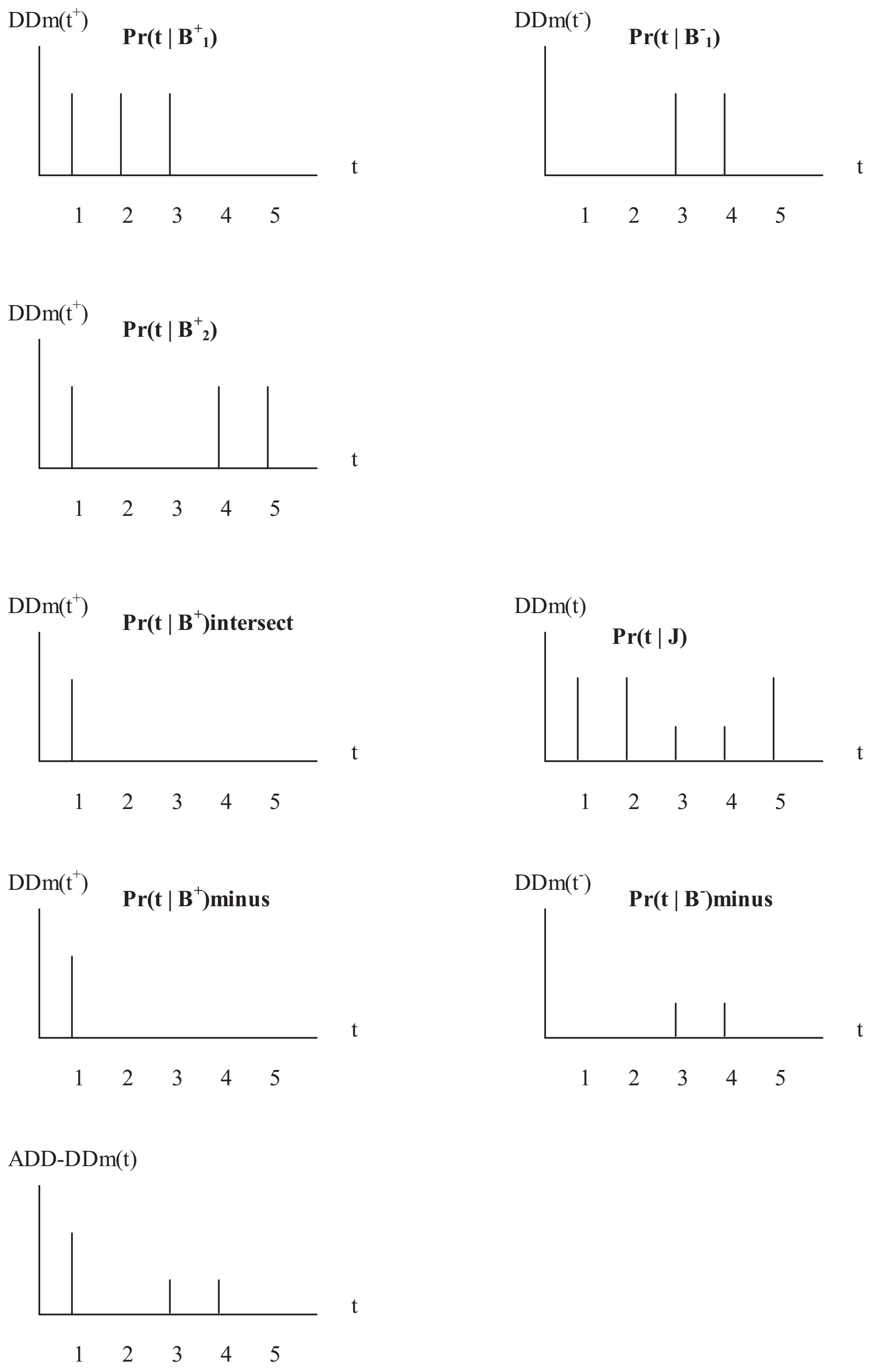


Figure 3.4: Calculation of $DDm(t^r)$ for each concept t and the addition, given three bags of instances within 1-dimensional number domain (B_1^+: 1, 2, 3; B_1^-: 3, 4; B_2^+: 1, 4, 5). $Pr(t|B_i)$ is modelled based on all-or-nothing model where $Pr(t|B_i)$ = 1 if bag i has t as one of its instances and 0 otherwise, $Pr(t|J)$ = 0.5 if t is one of the joint instances and 1 otherwise.

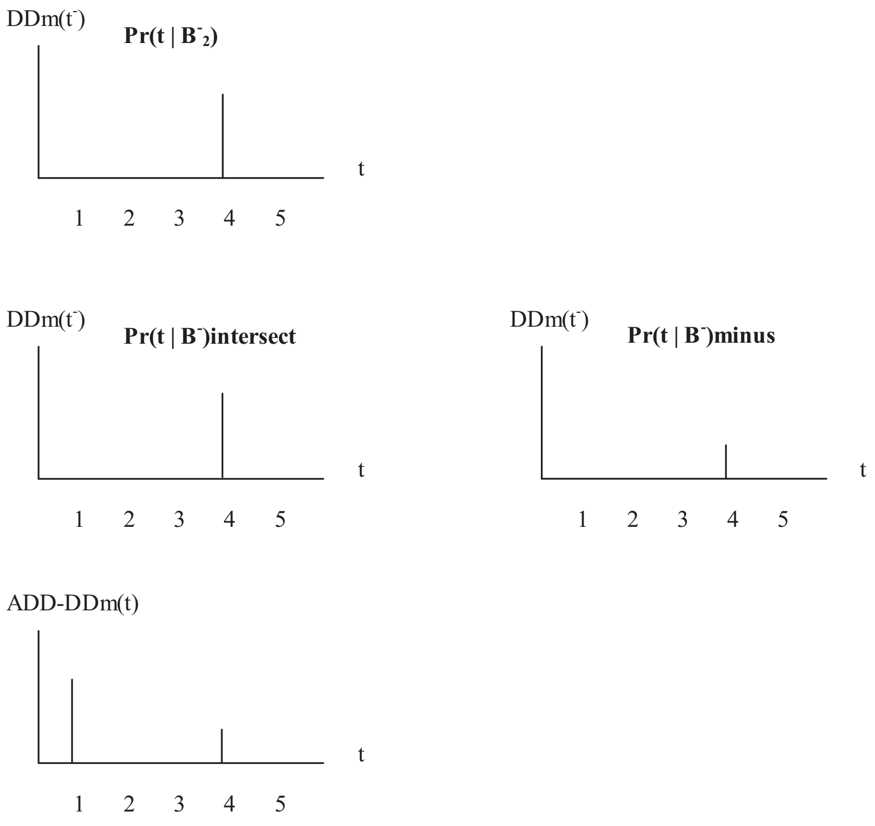


Figure 3.5: Update calculation of $DDm(t^r)$ for each concept t and the addition, given one extra bag of instances (B_2^-: 4).

order to identify the true instances of training examples presented to the learner.

PWDD for multi-class problem: This algorithm has two objectives as follows:

1. Identify true instances underlying each individual class. First of all, identify the instances that are likely to be true instances underlying each class of bags by looking for instances that always appear in every bag of the same class. With the same principle as in the original PWDD, multi-class Diverse Density at the concept corresponding to each individual instance likely to be true instances of each individual class is measured. The highest value of Diverse Density of those instances within the same class is likely to be that of the true instance underlying that class.

2. Obtain the order of influence of the true instances among different classes to the label of a bag.

   By comparing the value of multi-class Diverse Density of all the concepts that are likely to be the underlying concept (i.e. the concepts in objective 1, the order of influence can be found in much the same way as that in STEP 6 of section 3.3.1.3. The difference is that we only concentrate on a few specific concepts and not all the concepts like described in STEP 6.

3.3.3 Multi-Class Diverse Density vs Alternative Solution

Let us consider the number set example again, instead of using multi-class Diverse Density to find the numbers responsible for the label of a bag of instances, the deduction method could be adopted to analytically deduce from one example to the next until the true instances can be identified. With exactly the same three bags presented as examples in section 3.2.1 and section 3.3.1.3 $(B_1^+ : 1,2,3; B_1^- : 3,4; B_2^+ : 1,4,5)$, the analytical deduction method could deduce the following:

- From the first and the third bag, number 1 could be the true instance underlying the positive class is deduced.
- From the second bag, number 3 and 4 could both be the true instances underlying the negative class.
- From the first and second bag, even though number 3 could be the true instance underlying the negative class but it will have less influence compared to the one underlying the positive class as number 3 existed in bags of both classes.

- From the second and the third bag, similar to number 3, number 4 could be the true instance underlying the negative class but it will have less influence compared to the one underlying the positive class as number 4 also existed in bags of both classes.

Given the above initial evidence, it can be concluded that number 1 should be the true instance underlying the label of the positive bags of instances and it is more influential that number 3 and 4 which could both be the underlying instances of the negative class. This conclusion is consistent with the finding when the multi-class Diverse Density method was applied to the same problem (section 3.3.1.3). However, while the simple deduction method only gives logical facts, the multi-class Diverse Density method gives quantified facts in terms of the multi-class Diverse Density values. At least this extra information helps indicate the degree of the influence the specific instance has over the label of a bag of instances. Moreover, there are further benefits in favour of the multi-class Diverse Density over this simple deduction as follows:

1. The multi-class Diverse Density approach is better at managing noisy examples (incorrectly labelled examples). For example, if there is a fourth bag of instances with number 1 in it but was incorrectly labelled as a negative class, adopting the noisy-or model and adding more bags will result in eliminating the evidence that number 1 is not the underlying instance for the positive class. Another example is when a bag might accidentally contain no underlying instances (e.g. a bag containing number 2 and 5). This bag could be labelled to either classes which will affect the simple deduction drastically because the fact deduced will be changed almost completely when the label changes from one class to another, while this does not affect the multi-class Diverse Density due to the same reason as was mentioned earlier.

2. The different sequence of presentation of data to the algorithm does not affect the accuracy of learning by multi-class Diverse Density method. This is because of its incremental property which is also inherited from the original Diverse Density. In other words with this property, each bag of examples contributes to an individual set of evidences directing the algorithm to the target concept and when there are enough of theses sets of evidence, the algorithm will combine them to produce the target concept. Therefore whether the data comes in as a batch of examples or comes in incrementally (i.e. one example at a time), multi-class Diverse Density will combine each individual set of evident from each example together in order to obtain the target concept. Whereas in the case of analytical deduction method, each deduction

made intertwines with previous deductions made before it (i.e. current assumption is developed based on previous assumption acquired so far). Hence the different sequences of the presentation of data could affect the accuracy of learning by analytical deduction process. Note that the accuracy of learning is not to be confused with the speed of learning where selected set of examples or different sequences of presentation of examples could direct the algorithm to learn the target concept faster because these sequences does not produce the duplication set of evidence.

3. In the multi-class Diverse Density method, evidence is collected in the form of the value of multi-class Diverse Density where in analytical method evidence is collected as new facts are deduced. Hence there is a constant number of multi-class Diverse Density kept in the system (maximum space required equals to the number of possible concepts), while the number of facts could keep expending in the analytical deduction method. On the contrary in order to find the maximum Diverse Density, the technique only requires to look at a portion of all the possible instances. Furthermore during the process, the number of the instances considered could be reduced if the PWDDm algorithm was adopted.

4. When tackling the multi-class problem larger than two classes, the complexity of computing multi-class Diverse Density does not change, while the complexity of hypotheses in the analytical deduction method could increase dramatically.

5. When applying to different applications, there is no need to alter the way multi-class Diverse Density is calculated, hence the multi-class Diverse Density method is more generalised.

The above is only a comparison between multi-class Diverse Density and a simple method but further comparison between multi-class Diverse Density and the more sophisticated methods than the one above can be found in Chapter 7. In the next chapter, the multi-class Diverse Density method is tested for its feasibility.

Chapter 4

Feasibility Test

The multi-class Diverse Density method was introduced in previous chapter in comparison with the original Diverse Density method. This highlighted three further issues that have to be addressed before being able to conclude that it is feasible to employ the multi-class Diverse Density method as an indicator of the underlying concept describing ambiguous objects. The first of these issues requires creating a new model for the probability density of a bag of instances by modification of model used in the original Diverse Density method. It follows then that the second issue must be to find the suitable algorithm to identify a target concept using the multi-class Diverse Density method. This raises the possibility of applying large-scale-data to the algorithm using the multi-class Diverse Density method.

4.1 Probability Density Model

Multi-class Diverse Density $(DDm(t^r))$ is defined as the probability that a specific concept t is a target concept underlying a particular class r for a given training examples. It was also proven that $DDm(t^r)$ is the multiplication of the probability; where each probability is the probability that t underlying class r given each individual bag of instances $(Pr(t|B_i))$, as shown by equation 3.15. To be able to use $DDm(t^r)$ as a good measure for finding the target concept underlying each of a particular class r, its value should be very high at the target concepts and low everywhere else for easier identification. There is also further requirement that the addition of $DDm(t^r)$ for every class r should have a highest value at t being the concept underlying the most influential class and lower at t being the concept underlying the less influential class respectively. There is a high dependency on the chosen model for $Pr(t|B_i)$ so that the two requirements above can be satisfied.

The new representation of $Pr(t|B_i)$, based on the modifications of the model used in the original Diverse Density, is proposed as follows:

- Each bag of instances labelled class r gives out two sets of information: $Pr(t|B^r)$ and $Pr(t|J)$. However $Pr(t|J)$ is a global value held for every bag of instances.
- $Pr(t|B^r)$ is modelled in the same fashion as $Pr(t|B^+)$ in the original Diverse Density.
- $Pr(t|J)$ is modelled in the same fashion as $Pr(t|B^-)$ in the original Diverse Density but with an additional offset. As a result, probability will not be equal to 0 but equal to the offset when concept t is close to the joint instance and 1 otherwise. Using the offset value provides the evidence support the possibility that joint instance could still be the concept underlying the less influential class.
- The offset for $Pr(t|J)$ is modelled in such a way that its value has the most effect where concept t is close to any joint instance and exerts a decreasing effect the farther concept t is from any joint instance.
- The value of offset for $Pr(t|J)$ should also be varied, depending on the frequency the concept t appears as an instance in bags of different classes. For example, the offset associated with a particular concept t that appears in bags of two different classes, should be higher than the offset associated with those that appear in three or more classes. This arrangement will allow the offset of the more influential concepts to be higher than those of less influence. Hence the order of influence can be detected.

Revised all-or-nothing density estimator:

$$Pr(t|B_i^r) = 1 \quad if\ \exists j\ such\ that\ B_{ij} \in t,\ \ and\ 0\ otherwise \tag{4.1}$$

$$Pr(t|J) = 0.5^{(nc-2)}F \quad if\ \exists j\ such\ that\ J_j \in t,\ \ and\ 1\ otherwise \tag{4.2}$$

where j is the j_{th} instance of a class_r bag i or j_{th} of the joint instances, nc is the number of classes concept t as a joint instance j exists in, and fraction F is to be $\leq 0.5^{(total\ number\ of\ classes\ -\ 1)}$.

Revised noisy-or density estimator:

$$Pr(t|B_i^r) = 1 - \prod_{1 \leq j \leq p} (1 - Pr(B_{ij}^r \in c_t)) \tag{4.3}$$

$$Pr(t|J) = \prod_{1 \leq j \leq p} (1 - Pr(J \in c_t)) + 0.5^{(nc-2)} F | \prod_{1 \leq j \leq p} (1 - Pr(J \in c_t)) - 1| \tag{4.4}$$

where j is the j_{th} instance of a class_r bag i or j_{th} of the joint instances, p is the total number of instances in bag i or is the number of joint instances, nc is the number of classes where concept t exists in the bags of instances and nc must be ≥ 2 otherwise the equation 4.4 is not applicable because there will not be any joint instance, fraction F is to be $\leq 0.5^{(total\ number\ of\ classes\ -\ 1)}$, $Pr(B^r_{ij} \in c_t)$ is the probability of a particular instance being the target concept, and $Pr(J_j \in c_t)$ is the probability of a particular joint instance being the target concept.

With single point concept class, $Pr(B^r_{ij} \in c_t)$ and $Pr(J_j \in c_t)$ use Gaussian-like distribution of:

$$Pr(B^r_{ij} \in c_t) = \exp(-\sum_{1 \leq k \leq l} (B^r_{ijk} - c_{tk})^2) \tag{4.5}$$

$$Pr(J_j \in c_t) = \exp(-\sum_{1 \leq k \leq l} (J_{jk} - c_{tk})^2) \tag{4.6}$$

Where B^r_{ijk} is the k feature value of the j_{th} instance of the i_{th} r_class bag and c_{tk} is k_{th} feature value of concept c_t. J_{jk} is the k_{th} feature value of the j_{th} joint instances set.

The offset is set to be $0.5^{(nc-2)}F$. In addition the value of fraction F is set to be $\leq 0.5^{(total\ number\ of\ classes-1)}$ so that the addition of $DDm(t)$ of the less influential t would never added up to be equal to those of the more influential t. The offset is also varied according to a $0.5^{(nc-2)}$ factor and the value of nc.

4.1.1 Comparison of $Pr(t|J)$ Between The All-Or-Nothing Model and The Noisy-Or Model

Two separate comparisons are considered here. The first case is when the offset value stays the same as t changes. This situation only occurs in a two-class learning problem (i.e. nc will never have a value of more than 2). The scenario is chosen so that the effect of the offset from different models can be shown more clearly. The second case is to show how the maximum value of the offset is varied according to the value of nc for each concept t.

For the first case, there are only two possible classes of a bag of instances. Hence fraction F is equal to 0.5 according to equations 4.2 and 4.4. The offset, $0.5^{(nc-2)}F$, is

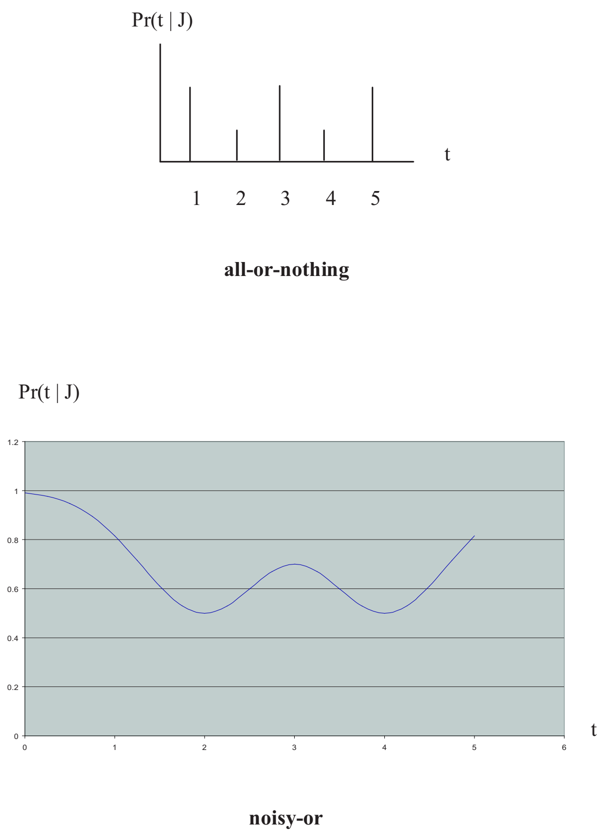


Figure 4.1: The difference to $Pr(t|J)$ given the same concept space and the same set of joint instances between all-or-nothing model and noisy-or model.

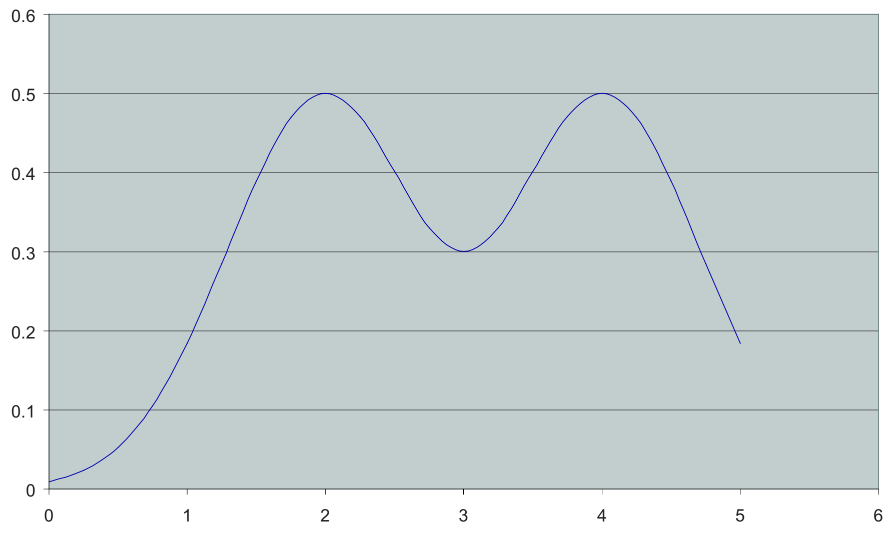


Figure 4.2: The extra term $(0.5^{(nc-2)}F|\prod_{1\leq j\leq p}(1-Pr(J\in c_t))-1|)$ is plotted against concept t.

therefore equal to 0.5 because $nc = 2$ and then the offset just becomes the value of F. As a result there is no variation of the offset values. Figure 4.1 illustrates the difference of $Pr(t|J)$ between the all-or-nothing model and the noisy-or model given the same concept space and the same set of joint instances. The concept space of a number set between 1 and 5 is chosen as an example. The joint instance is chosen to be the numbers 2 and 4. With the all-or-nothing model, $Pr(t|J)$ has a discrete value of 0.5 when t is one of the joint instances and 1 at any other concept (i.e. according to equation 4.2). Contrary to the noisy-or model, $Pr(t|J)$ has a continuous value again with a minimum of 0.5 where t is one of the joint instances (as in equation 4.4) and increases with the distance between t and the joint instances. This increase is calculated differently from that calculated for the negative class in the original Diverse Density because there is an extra term that contributes to the increase (i.e. the term $0.5^{(nc-2)}F|\prod_{1\leq j\leq p}(1-Pr(J\in c_t))|$). This extra term tends from the offset to 0 as the concept gets farther away from the joint instances. Figure 4.2 plots this extra term against concept t.

For the second case, the learning task increases from two classes to four classes. Therefore the fraction F will reduce from 0.5 to 0.125 according to equation 4.2 and 4.4. Let us assume that number 2 appeared in three different classes of bags of instances, while number 4 only appeared in two different classes of bags. Hence different values of nc for $t = 2$ and $t = 4$ results in $Pr(t|J)$ at $t = 2$ being smaller than that at $t = 4$. Figure 4.3 illustrates $Pr(t|J)$ according to noisy-or model given the above situation.

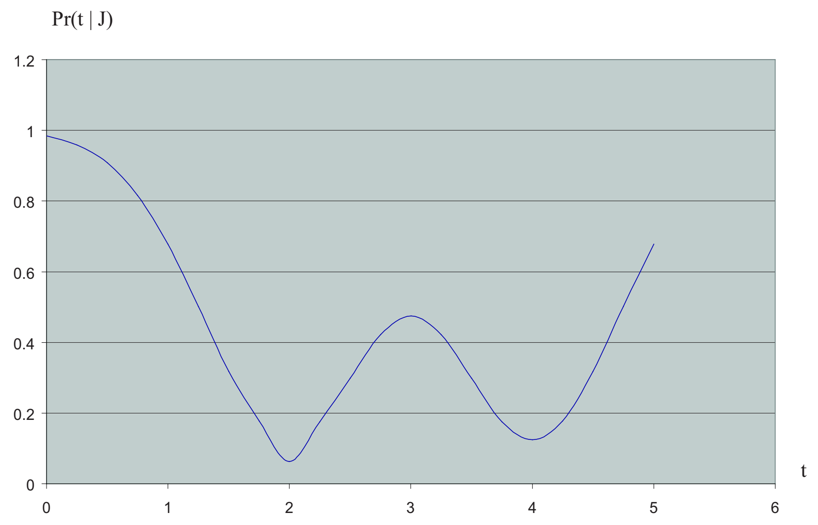


Figure 4.3: $Pr(t|J)$ according to noisy-or model given four-class learning problem with number 2 appeared in three different classes of bags of instances, while number 4 only appeared in two different classes of bags.

4.1.2 An Indication for The Order of Influence by The Addition of Multi-Class Diverse Density from Each Class

If $ADD-DDm(t)$ is the addition of multi-class Diverse Density ($DDm(t^r)$) for every class r, t_{1st} is the concept underlying the most influential class, t_{2nd} is the concept underlying the second most influential class, t_{nth} is the concept underlying the n_{th} influential class and so on, then $ADD-DDm(t_{1st}) > ADD-DDm(t_{2nd}) > \ldots > ADD-DDm(t_{nth}) > ADD-DDm(other\_t)$ needs to be assured so that the order of influence of the underlying concept can be detected. $DDm(t^r)$ for every class r can be seen as the multiplication of two sets of values: $Pr(t|J)$) and the intersects of every bag in class r (i.e. $\prod_{1\leq i\leq n} Pr(t|B_i^r)$). The value of $\prod_{1\leq i\leq n} Pr(t|B_i^r)$ will be very high at t where t is the underlying concept of class r, compared to a very low value at any other t. This concept applies to every class r while the value $Pr(t|J)$ holds constant for every class r. As a result, the multiplication between $Pr(t|J)$ and $\prod_{1\leq i\leq n} Pr(t|B_i^r)$ is the key to the isolation of underlying concept from another concept t. The higher the frequency of concept t appearing in a bag of different classes, the lower $Pr(t|J)$ should be as has been shown in Figure 4.3 from the previous section.

Table 4.1 summarises the maximum bound of $ADD-DDm(t)$. This happens when instances underlying the less influential classes exist in every bag of the more influential class. For example, there are three instances, the numbers 1, 2, and 3, and there are three classes with number 1 underlying the most powerful class, number 2 underlying the second most powerful class respectively and so on. In this case, every bag labelled the most influential class, there always exists number 1, 2, and 3. While in every bag labelled the second most influential class, there exists only number 2 and 3.

Table 4.1: The maximum bound of $ADD - DDm(t)$.

Class Order	**Concept t**					
	t_{1st}	t_{2nd}	t_{3rd}	$\dots$	t_{nth}	t_{others}
$\prod_{1\le i\le n} Pr(t\|B_i^r)$						
1-st	1	1	1		1	≈ 0
2-nd		1	1		1	≈ 0
3-rd			1		1	≈ 0
$\dots$						
n-th					1	≈ 0
$Pr(t\|J)$						
1-st $\dots$ n-th	1	$0.5^{(n-1)}$	$0.5^{(n-1+1)}$		$0.5^{(n-1+n-2)}$	1
$DDm(t)$						
1-st	1	$0.5^{(n-1)}$	0.5^n		$0.5^{(2n-3)}$	≈ 0
2-nd		$0.5^{(n-1)}$	0.5^n		$0.5^{(2n-3)}$	≈ 0
3-rd			0.5^n		$0.5^{(2n-3)}$	≈ 0
$\dots$						
n-th					$0.5^{(2n-3)}$	≈ 0
$AAD - DDm(t)$						
	1	$2 * 0.5^{(n-1)}$	$3 * 0.5^n$		$n * 0.5^{(2n-3)}$	≈ 0

Calculation of $\prod_{1\le i\le n} Pr(t|B_i^r)$, $Pr(t|J)$, $DDm(t)$, and $ADD - DDm(t)$ illustrated in Tables 4.1 emphasises the fact as proposed that $ADD - DDm(t_{1st}) > ADD - DDm(t_{2nd}) > \dots > ADD - DDm(t_{nth}) > ADD - DDm(other\_t)$. Hence the resultant of adding together of $DDm(t)$ for every class signifies the order of influence.

4.2 Algorithms Used to Identify Target Concepts

So far the methods to identify target concepts and order of influence has been discussed. In this section the implementation of these methods as algorithms is examined in order to allow testing on large scale data in the next section. Three algorithms have been developed to identify target concepts in order to explore 1) basic generation of the distribution of $ADD-DDm(t)$ for every concept t, 2) ability to add further bags of instances to an existing system, and 3) ability to reduce the number of calculation required in order to obtain the highest $ADD-DDm(t)$. Each of these three algorithms is linked to another section of the algorithm called TrueInstances, which is used to identify both the class the target concepts underlie and the order of influence.

all-bags-at-once: The algorithm outputs distribution of $ADD-DDm(t)$ over every concept t.

increment: This algorithm allows new bags of instances to be added to the old set before all-bags-at-once is reapplied. It is used when the comparison between $ADD-DDm(t)$ of the old set and the new set of bags of instances is required.

PWDDm: The general idea behind this PWDDm (Pointwise multi-class Diverse Density) algorithm has already been described in Chapter 3: section 3.3.2. The algorithm all-bags-at-once is used on a few initial training bags to obtain the $ADD-DDm(t)$ distribution. At this point $ADD-DDm(t)$ threshold value has been chosen by the user, only concepts with $ADD-DDm(t)$ above the threshold are selected. Further training bags are then submitted, only $ADD-DDm(t)$ of the selected concepts is recalculated. Hence less computation is required.

TrueInstances: The objective of this section of the algorithm is to identify the true instances underlying each class of bag labels, and the order of influence of these true instances. Therefore, once the distribution of the $ADD-DDm(t)$ has been obtained either from one of the algorithms described above: all-bags-at-once, increment, or PWDDm, the $ADD-DDm(t)$ value of each concept is then ranked against one another. Because of the way the $ADD-DDm(t)$ is calculated, only a top few ranks are of importance (i.e. only true instances will have a steep peak value of $ADD-DDm(t)$), with the highest value being the most influential and so on. The class which the true instance belongs to is found by comparing $< Pr(t|B_i) >_{intersect}$ of different classes at the true instance. The class label with the highest $< Pr(t|B_i) >_{intersect}$ at the true instance will be the underlying class of this true instance.

For example, given the following training bags of instances:

Bag_1: A, C with positive label

Bag_2: A, B with positive label

Bag_3: B with negative label

The program will output instance A as the most influential true instance underlying the positive class. Instance B is output as the second most influential true instance underlying the negative class. This is because instance A is always in the positive bags but instance B is in both types of bags. However according to the evidence provided by the negative bag, B is the only instance in the bag. Therefore, instance B is the true instance underlying the negative class.

However, if the training bags are as follows:

Bag_1: A, B, C with positive label

Bag_2: A, B with positive label

Bag_3: B with negative label

The difference between this and the previous example is that instance B is now in all the positive bags. As a result, there will be two equally influential true instances (i.e. A and B) with instance A still underlying positive class but instance B underlies both the positive and the negative class.

4.3 Artificial Data and Scaling of Test Sets

The calculation of both the multi-class Diverse Density method and the original Diverse Density method are very similar to the computation needed for a Nearest Neighbour algorithm where the distance from every instance to the hypothesized concept must be calculated. While the original Diverse Density calculation requires $O(N)$ space and $O(N)$ time, where N is the total number of instances in all bags. The multi-class Diverse Density method, on the contrary, requires more: $O(N)$ space and $O(2N)$ time. The extra N time represents the extra calculation for joint instances. However, a technique such as PWDDm or the interactivity feature, which is explored in next chapters, can be employed to reduce space and time of the calculation of multi-class Diverse Density.

Two sets of experiments were carried out with the artificial data set in order to verify the multi-class Diverse Density functionality and its applicability to large scale data, such as might be found in the real world. The first experiment is the investigation of the effect of the size of feature space, the number of bags of instances given to the learner, and the maximum number of instances given per bag. The second experiment tests multi-class Diverse Density on a problem with a large feature space. Before looking into the two sets of experiments, the generation of the artificial data set will be described in the next subsection.

4.3.1 The Generation of The Artificial Data

The aim of the generator is to automatically generate artificial data as bags of instances input to the algorithms in section 4.2. The generator generates concepts or instances which are co-ordinates of points in N-dimensional space. Two different points are chosen to be the underlying true instance for each of positive and negative bag labels.

Bag Generator: The function of the bag generator is to generate a set of training bags of instances with appropriate bag label. In the real world data, the parameters are set while in the case of artificial data, the user is allowed to input the following parameters in order to generate a random set of training bags of instances:

range indicates the number of dimensions and the maximum instance value allowed for each dimension (e.g. {5, 6, 8} means three dimensions where the first dimension has the range of 0-5, the second dimension has the range of 0-6 and the third dimension has the range of 0-8)

numberOfBags indicates the number of bags of instances to be randomly generated

maxInstPerBag indicates the maximum possible number of instances to be generated for each bag

influentLabelOrder indicates the order of influence of the true instances (e.g. {‘N’, ‘P’} means that the true instance underlying the negative bag label is more influential than that of the positive bag label)

underlyingPositive indicates the true instance underlying the positive bag label, and must fall within the range for its associated dimension (e.g. {1, 6, 7} could be chosen for the above example range of {5, 6, 8})

underlyingNegative indicates the true instance underlying the negative bag label, and must fall within the range for its associated dimension (e.g. $\{3, 3, 1\}$ could be chosen for range of $\{5, 6, 8\}$)

Instances in a bag are randomly generated and labelled according to the selected true instances and its order of influence. However, if any of the selected true instances does not exist in a randomly generated bag, the last instance generated for that bag will be removed and replaced with one of the true instances, which is also randomly selected.

Feature Space Generator: The function of feature space generator is to generate the list of all the concepts given the chosen range parameter. As was mentioned earlier in this subsection, the generated artificial data will therefore be a list of co-ordinates.

4.3.2 Experiments on Data with Small - Medium Feature Space

There are two objectives in carrying out these experiments.

- To test the functionality of the multi-class Diverse Density algorithm.
- To investigate on how the previously given three aspects (i.e. the size of the feature space, the number of training bags of instances given, and the number of instances in a bag) could affect the performance of the algorithm and the value of $ADD-DDm(t)$ at each of the concepts.

Experiment is carried out on 10 different sets of training bags for each learning problem to ensure large coverage of concept space. $ADD-DDm(t)$ is calculated based on the noisy-or estimator model. The first experiment is run on a small feature space with randomly selected concepts underlying positive and negative classes, and with the random order of influence. The second experiment maintains the original set up of the first experiment but increases the maximum number of instances per bag and then increases the number of bag of instances. The third experiment focuses on increasing the size of feature space from small to medium sized.

4.3.2.1 The First Experiment

The setup is as follows.

- Range $\{5, 5\}$

- influentLabelOrder {‘P’, ‘N’}
- underlyingPositive {1, 2}
- underlyingNegative {3, 4}
- numberOfBags 5
- maxInstPerBag at 5 instances

The algorithm correctly learned the two target concepts (i.e. {1,2} and {3,4}) and correctly indicated the order of influence in 6/10 training sets. The reasons for the algorithm not to correctly learn in 10/10 training sets are as follows. In the failed training sets, either only one type of bags (or examples) is presented to the algorithm hence the algorithm will not be able to find the counter evidence it needs to conclude the right target concept or the bags presented to the algorithm produce duplicated evidence which does not cover enough ground to conclude the right target concept. Table 4.2 summarises the profiles of $ADD-DDm(t)$ for the cases where the target concepts were correctly learned. From Table 4.2, among the successful training sets, there are only 2 concepts that have $ADD-DDm(t)$ over 0.5, which are the correct target concepts. Furthermore according to the same table the values of $ADD-DDm(t)$ of other concepts are insignificant compared to those of the target concepts which are close to 1 and are ordered according to the order of influential labels.

Table 4.2: The $ADD-DDm(t)$ profile for the first experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t) between**			
{1,2}	*{3,4}*	*0.0-0.1*	*0.1-0.3*	*0.3-0.5*	*0.5-1.0*
0.93-1.00	0.82-1.00	24-29	5-8	0-2	2

4.3.2.2 The Second Experiment

The first setup: The first setup of the second experiment is to increase the maximum number of instances per bag from 5 to 10 instances.

- Range {5, 5}

- influentLabelOrder {‘P’, ‘N’}
- underlyingPositive {1, 2}
- underlyingNegative {3, 4}
- numberOfBags 5
- maxInstPerBag 10

The algorithm correctly learned the two target concepts and correctly indicated the order of influence in 8/10 training sets. Table 4.3 summarises the profiles of $ADD - DDm(t)$ for the cases where the target concepts were correctly learned.

Table 4.3: The $ADD - DDm(t)$ profile for the first setup of the second experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t) between**			
{1,2}	*{3,4}*	*0.0-0.1*	*0.1-0.3*	*0.3-0.5*	*0.5-1.0*
0.81-1.12	0.50-0.99	14-29	4-10	1-7	2-5

The second setup: The second setup of the second experiment is to increase the number of bags of instances from 5 to 15 bags while other parameters were kept as in the original setup of the first experiment.

- Range {5, 5}
- influentLabelOrder {‘P’, ‘N’}
- underlyingPositive {1, 2}
- underlyingNegative {3, 4}
- numberOfBags 15
- maxInstPerBag 10

The algorithm correctly learned the two target concepts and correctly indicated the order of influence in 8/10 training sets. Table 4.4 summarises the profiles of $ADD - DDm(t)$ for the cases where the target concepts were correctly learned.

Table 4.4: The $ADD - DDm(t)$ profile for the second setup of the second experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t) between**			
{1,2}	*{3,4}*	*0.0-0.1*	*0.1-0.3*	*0.3-0.5*	*0.5-1.0*
0.77-0.99	0.50-0.91	34	0	0	2

4.3.2.3 The Third Experiment

This experiment is designed to investigate the effect on the distribution of $ADD - DDm(t)$ across the feature space when the size of the feature space is enlarged resulting in the true instances being farther apart.

The first setup:

- Range {10, 10}
- influentLabelOrder {‘N’, ‘P’}
- underlyingPositive {2, 5}
- underlyingNegative {7, 8}
- numberOfBags at 20 instances
- maxInstPerBag 20

The algorithm correctly learned the two target concepts and correctly indicated the order of influence in 8/10 training sets. Table 4.5 summarises the profiles of $ADD - DDm(t)$ for the cases where the target concepts were correctly learned.

Table 4.5: The $ADD - DDm(t)$ profile for the first setup of the third experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t)**	
{7,8}	*{2,5}*	< 0.5	≥ 0.5
0.65-0.73	0.50-0.70	119	2

The second setup:

- Range $\{20, 20\}$
- influentLabelOrder $\{$‘P’, ‘N’$\}$
- underlyingPositive $\{5, 7\}$
- underlyingNegative $\{3, 15\}$
- numberOfBags at 20 instances
- maxInstPerBag 40

The algorithm correctly learned the two target concepts and correctly indicated the order of influence in 7/10 training sets. Table 4.6 summarises the profiles of $ADD-DDm(t)$ for the cases where the target concepts were correctly learned.

Table 4.6: The $ADD-DDm(t)$ profile for the second setup of the third experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t)**	
$\{5,7\}$	$\{3,15\}$	< 0.5	≥ 0.5
0.93-0.99	0.77-0.93	439	2

The third setup:

- Range $\{5, 5, 10\}$
- influentLabelOrder $\{$‘P’, ‘N’$\}$
- underlyingPositive $\{1, 3, 7\}$
- underlyingNegative $\{4, 2, 5\}$
- numberOfBags at 20 instances
- maxInstPerBag 30

The algorithm correctly learned the two target concepts and correctly indicated the order of influence in 7/10 training sets. Table 4.7 summarises the profiles of $ADD-DDm(t)$ for the cases where the target concepts were correctly learned.

Table 4.7: The $ADD-DDm(t)$ profile for the third setup of the third experiment for the case where the multi-class Diverse Density method learned the correct concepts.

ADD-DDm(t) at concept		**Number of concepts that has ADD-DDm(t)**	
$\{1,3,7\}$	$\{4,2,5\}$	< 0.5	≥ 0.5
0.81-0.91	0.58-0.70	394	2

4.3.2.4 Discussion of Results

The three experiments provide enough results to discuss the functionality of the multi-class Diverse Density method and the effect some parameters such as the number of bags of instances have in aiding the method to produce clear-cut learning results.

1. The functionality of the multi-class Diverse Density method:

    - The first experiment suggested that even given a small feature space (i.e. 36 concepts in total), the $ADD-DDm(t)$ values can have high peaks at both of the true instances and very small values elsewhere (i.e. only 2 concepts have $DDm(t^r)$ more than 0.5). This behaviour also held true for experiments with larger feature spaces. It can generally be said that the multi-class Diverse Density method also inherits the exponential nature of the original Diverse Density method.
    - The success of the multi-class Diverse Density method is very much based on the evidence provided by the training data. In other words, given certain training data, the algorithm could learn the wrong concept. For example, when only one class of training bags is presented, only the underlying true instance of that class can be learned. Hence the failure to learn the target concepts or to learn the correct order of influence of the target concepts. Restated thus, not all the randomly generated training sets can provide all the evidence required by the multi-class Diverse Density method.
    - As can be seen from the experiment, the more the evidence that is supplied the better the learning becomes.

2. Parameters affecting the performance of multi-class Diverse Density:

    - The first and second experiments suggested that the larger the number of instances per bag, the more difficult it became for the multi-class Diverse Density method to learn the correct concept. This is because true instances will have lower than normal peaks (i.e. $ADD-DDm(t)$) and all other instances have values close to that of the peak. Therefore it is difficult to differentiate true instances from other instances.

- The more training bags of instances that are supplied, the better the evidence the multi-class Diverse Density method has. This will result in less instances with high $ADD-DDm(t)$ value compared to that of the true instances.
- The third experiment suggests that the larger feature space requires more training bags to order to help the true instances get higher peak values.

4.3.3 Experiment on Data with Large Feature Space

In this experiment, one target concept underlying the positive class and another concept underlying the negative class are randomly selected from artificial data set of range $\{100,100\}$. The order of influence of the underlying concept is also randomly selected. Based on similar experiment in [Maron, 1998], the learning algorithm will learn from 10 bags of instances, 5 for each type of bag label, and 50 instances in each bag. Further, in order to compare with the original method, the noisy-or estimator model was used. Unlike the original two-class problem, the learning algorithm has to look for the concept underlying the label of each individual class of bags of instances and also the order of influence of the class labels.

In the experiment by [Maron, 1998], instances in all of the 10 bags were forced to uniformly cover the feature space. This arrangement allowed all the possible instances to be seen by his method, which would benefit the learning enormously (i.e. all the possible evidence were given to his method). However to investigate the robustness of the proposed method (the multi-class Diverse Density method) further, instances were randomly generated and were not forced to uniformly cover the feature space, hence making the learning problem more difficult to learn. The multi-class Diverse Density method was then re-applied on multiple sets of 10 bags instead in order to measure its accuracy. Another difference is that in this experiment, one bag of instances was forced to have both underlying positive and negative instances so that the evidence supporting the order of influence was ensured in every set of examples. It had to be a forced mechanism because the number of bags of instances given to the learner is small, only 10 bags for each experiment. Moreover, because of the large feature space, there is low possibility of both

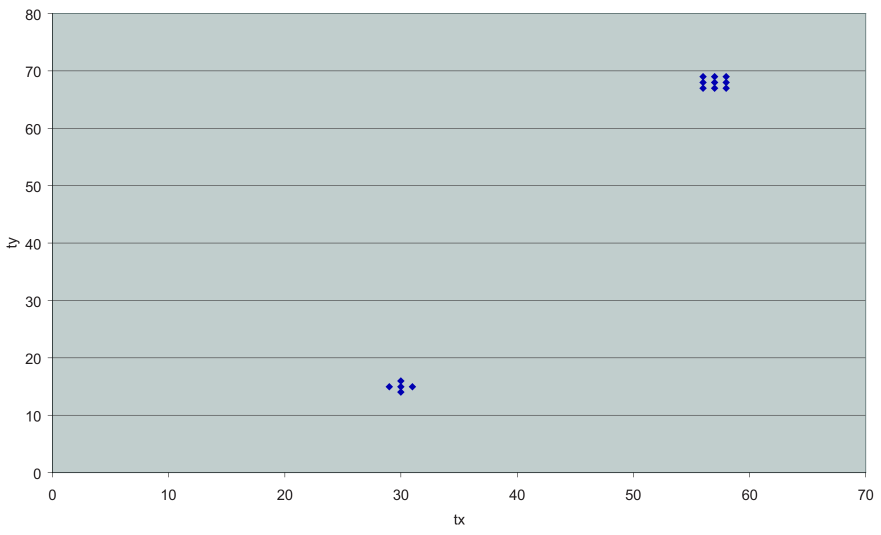


Figure 4.4: The concentration of concepts with high $ADD - DDm(t)$ value after all ten bags were presented to the multi-class Diverse Density learner.

underlying instances being randomly generated for the same bag.

For every set of bags of instances tested in this experiment, the multi-class Diverse Density learner succeeded in identifying the correct underlying concepts and the order of influence of such concepts. One example of the learning tasks was to learn that concept $\{30, 15\}$ was underlying positive-class bags and had more influence than concept $\{57, 68\}$, which was underlying negative-class bags. For this learning problem with a particular set of examples, Figure 4.4 shows the concentration of concepts that had $ADD - DDm(t)$ over 0.0001, whereas $ADD - DDm(t)$ of the rest of the possible concepts were less than 0.0001 or very close to 0. Table 4.8 illustrates the values of $ADD - DDm(t)$ obtained for those concepts in the two concentration areas in Figure 4.4. As can be seen from the figure and the table, the multi-class Diverse Density method was able to distinguish the underlying concepts from the rest of the concepts accurately. The very high peak of $ADD - DDm(t)$ values only occurred at the underlying concepts and the height of the peaks was based on the order of influence. In other words, the higher the peak, the more influence the concept has. The maximum height for each peak also followed the equations 4.1 - 4.4. When there are only two different classes of bags of instances, the experiments confirmed that the most influential concept has a value of 1 and the second most influential concept has a value of 0.5. These results signify the capability of the multi-class Diverse Density method for finding the solution of the multi-class problem proposed within this thesis.

The original Diverse Density method passes on many features to the multi-class Di-

Table 4.8: $ADD - DDm(t)$ obtained for concepts in the concentration area.

tx	ty	ADD-DDm(t)
30	15	1
57	68	0.5
56	68	0.0149
57	67	0.0149
57	69	0.0149
58	68	0.0149
29	15	0.0025
30	14	0.0025
30	16	0.0025
31	15	0.0025
56	67	0.0003
56	69	0.0003
58	67	0.0003
58	69	0.0003

verse Density. One of these features is the exponential nature, which is an advantage over the averaging nature of supervised learning algorithm. It helps the true concepts stand out while the averaging technique does not. Despite being good at solving the two-class problem, Diverse Density is not capable of solving multi-class problem. It is possible to apply Diverse Density twice. The first application finds the concept underlying the positive class. The second application can be done by reversing the label of bags of instances so that the concept underlying negative class can be found. Nonetheless, adopting the original Diverse Density method will not be able to indicate the order of influence of the concepts found, which can be very significant in many learning problems, while the multi-class Diverse Density method will achieve this.

Chapter 5

Instance Volume

So far each instance in a bag of instances is considered as having one unit. In other words, each instance represents equal amount of evidence regardless of its volume. However, it cannot be certain that the above assumption would apply to every circumstance. In some cases, taking into account the volume of each instance present in a bag might be more appropriate. For example, in a bag of instances, there might be 10 units of instance A, and 4 units of instance B. In another bag, there might be 5 units of instance A, and 20 units of instance B. The effect of instance volume on the target concept is investigated in this chapter resulting in a proposal for the representation of instance volume within the multi-class Diverse Density method.

Previous work on the original Diverse Density method by [Maron, 1998] has shown that it is possible to learn to identify or predict stocks that will increase in value for fundamental economic reasons using the multiple-instance learning approach. It is common that the fluctuations in the price of a stock occur for reasons that are not economically fundamental, resulting in mislabelling of training examples used in supervised learning. This would not pose any difficulty if the stock prediction task was to be viewed as a multiple-instance learning problem as have been proved by [Maron, 1998]. In this thesis, the stock prediction task was also one of the applications chosen for the investigation of the use of multi-class Diverse Density as well as for the investigation of instance volume.

This stock prediction task is chosen because it was one of the application domains investigated in previous work on the Diverse Density method [Maron, 1998] and was proven successful. In order to treat this stock prediction task as the multiple-instance learning problem, Maron made two crucial assumptions. The details of these two assumptions can be found in the section 5.2. Although these assumptions are difficult to guarantee in practice, treating the stock prediction task as the multi-class problem however does not require such assumptions to be made. In other words, the proposed multi-class Diverse Density method should be able to handle problems with a higher degree of ambiguity compared to the original Diverse Density method. The task is also chosen because the effect of instance volume on the learning can also be explored with this task. This research hopes to prove that the stock prediction task could be even better viewed as a multi-class multiple-instance learning problem with added instance volume feature.

In the first section, instance volume and its effects on the learning process are investigated as well as the representation of the volume within the multi-class Diverse Density calculation. For a comparison purpose, an initial work on stock prediction task by [Maron, 1998] is discussed in the section 5.2. This then is followed by the detail of the improved algorithm for the same task. The last section in this chapter will be the experimental set-up and results to prove the claim.

5.1 Instance Volume and Its Effects

5.1.1 Instance Volume Definition

It is already known that instances are considered to be of the same kind if each instance is described by the same features. Here, the amount of instances of the same kind in the same bag at any one time is defined as an instance volume. In case of the multiple-instance learning, the idea of instance volume has never been investigated in detail. Currently, in other methods, if the volume is presented, it will usually be considered as one of the features of the instance. The question is whether it is appropriate to do so - considering instance volume as one of the features or attributes of the instance.

If the issue associated with instance volume was not considered, each instance in a bag of instances would provide an equal amount of evidence for the target concept to be identified. By introducing instance volume as one of the features, instances of the same kind need to have the same amount of volume in each bag that they are in, otherwise they would be considered as instances of different kinds. This will result in incorrect identification of the location in the feature space of the target concept.

The nature of evidence instance volume provides will need to be considered. The first question is whether more instance volume (i.e. more instances of the same kind present in a bag) implies greater evidence supporting the instance being target concept. To answer this, the representation of an instance for the learning task of interest need to be considered. For example, a bag of instances could represent a group of software components for performing an operation where they may fail or succeed. In many cases, some software components will be used several times in one operation or in one user request. If the number of the calls from the same component affects the success or failure of the user request, then each type of the component call can be considered as an instance with its volume having an effect on the likelihood of an instance being a target concept. Another example is the stock prediction task. If the quantity of the same stock traded affects the price trend for that stock, then again instance volume (i.e. stock volume) will have an effect on the likelihood of an instance (i.e. stock in this case) being a target concept.

As a summary, the following questions should be answered before the effect of instance volume is decided.

(i) Does each instance in the same bag represent the same amount of quantity?

(ii) Does the quantity affect to the chance of an instance being a target concept?

The answer to the second question is then a guideline on how the instance volume should be represented. The simplest assumption is that the probability of an instance being target concept compared to other instances in the same bag is also proportional to its instance volume. With this assumption in mind, the proportional representation of instance volume is proposed, which will be further discussed in the next section.

5.1.2 Proportional Representation of Instance Volume for Multi-Class Diverse Density

Once the effect of instance volume has been established for a particular learning task, instance volume will be assigned to the calculation of multi-class Diverse Density. It is proposed that the probability of an instance being the target concept compared to other instances in the same bag is proportional to its instance volume. As a result, proportional representation is chosen where the proportion of instance volume to the combined volume of every instance in the same bag is used as a scaling factor applied to $Pr(t|B_i)$ estimator model.

In the original Diverse Density method, [Maron, 1998] has already introduced a scaling factor to single point-and-scaling concept class. This type of concept class projects onto the data having many irrelevant or redundant attributes. Therefore weighting of attributes (i.e. scaling factor) must be assigned so that irrelevant attributes can be ignored (i.e. weight goes to 0). However Maron treated this scaling factor as an unknown to be found for each individual feature in a feature space used to describe instances. Making the scaling factor another unknown is the opposite to the philosophy of making the calculation of multi-class Diverse Density as simple as possible while still preserving sufficient information about instances. Hence only the single point concept class is considered within multi-class Diverse Density. However, the idea of a scaling factor is not rejected totally. The scaling factor should be used only when its value can be obtained from readily available information such as instance volume. Furthermore the scaling factor associated with the instance volume will be applied to each individual instance, unlike the scaling factor proposed by [Maron, 1998] which has a unique value for each individual feature.

Referring to the first paragraph of this section, the instance volume scaling factor for concept t being instance j of bag i ($S_i^v(t=j)$) is described as follows:

$$S_i^v(t=j) = (instance\ j\ volume\ of\ bag\ i)/(total\ instance\ volume\ of\ bag\ i) \tag{5.1}$$

As a result, $Pr(t|B_i)$ for the all-or-nothing estimator could be remodelled as follows:

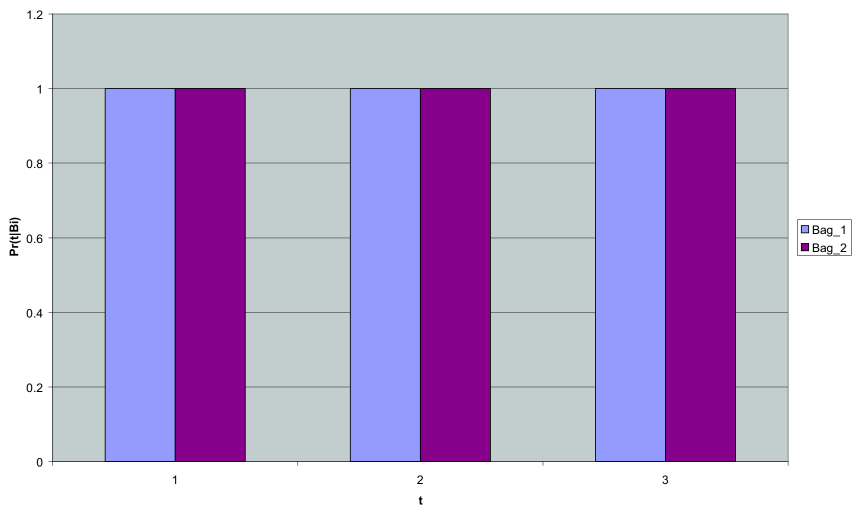
$$Pr(t|B_i) = (1-P_v) + P_v(S_i^v(t=j)) \quad if\ \exists j\ such\ that\ B_{ij} \in t,\ \ and\ 0\ otherwise \tag{5.2}$$

Where P_v is the number between 0 and 1, the larger the P_v, the greater the effect of instance volume has on the value of $Pr(t|B_i)$.

The instance volume scaling factor is used to scale down the probability for concept t being a target concept underlying a particular class given bag i. For example, there are 2 bags of instances, let us say both bags are labelled the same class - class one. The first bag contains the numbers 1, 2, and 3 with each number having a volume of 1 unit. The second bag similarly contains the numbers 1, 2, and 3 but each has different volume of 10, 20, 30 units respectively. Figure 5.1 illustrates the comparison for the value of $Pr(t|B_i^{one})$ obtained for the two bags. Figure 5.1(a) adopts the normal all-or-nothing model for both bags. With both bags having the same set of instances, their distributions will be exactly the same. Figure 5.1(b) adopts the all-or-nothing model with instance volume and P_v = 0.5 for both bags. Figure 5.2 illustrates the effect of varying P_v on the value of $Pr(t|B_i)$.

From Figure 5.1(a), the value of $Pr(t|B_{1and2}^{one})$ for $t=1$, $t=2$, and $t=3$ are the same and have the value of 1. The values for $Pr(t|B_i^{one})$ of the first bag in Figure 5.1(b) are similarly all constant having the value of 0.6667 (0.5 + (1/3)*0.5), while those of the second bag have various values. When P_v = 0, the distribution is effectively that of the

(a) all-or-nothing model



(b) all-or-nothing model with instance volume feature

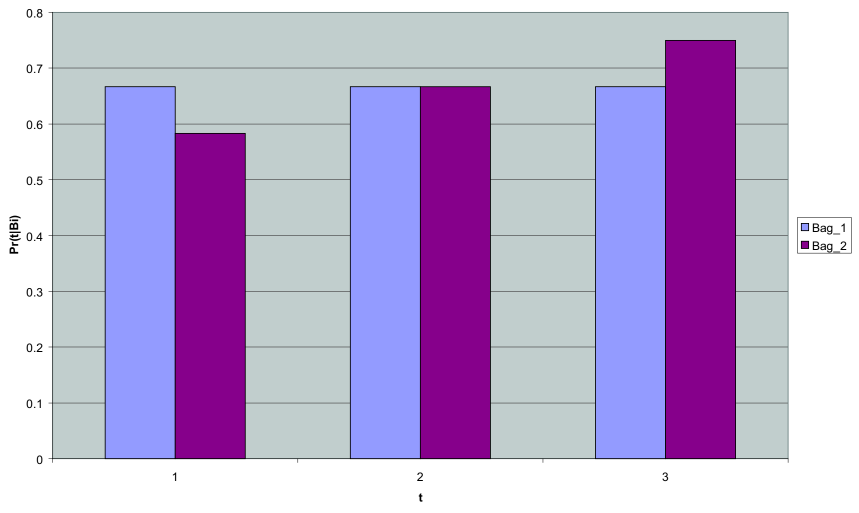


Figure 5.1: The distribution of $Pr(t|B_i^{one})$ of the two bags when (a) adopting the normal all-or-nothing model and (b) adopting the normal all-or-nothing model with instance volume feature.

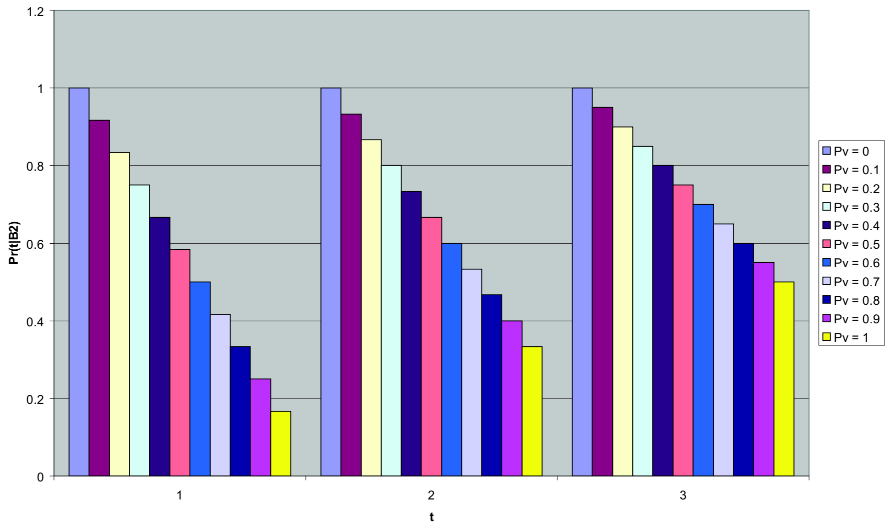


Figure 5.2: The effect of the value of $Pr(t|B_i)$ when P_v varies. The second bag contains number 1, 2 and 3 having the unit of 10, 20, and 30 respectively is used in the calculation.

normal all-or-nothing model. As P_v increases, $Pr(t|B_i)$ reduces further according to the increased effect of instance volume as shown by Figure 5.2.

Having remodelled $Pr(t|B_i)$ to take into account the instance volume as described above, concept t with high $DDm(t)$ implies that i) concept t is close to instances that appear very often in bags of the same class and ii) these instances must have a high volume compared to other instances in the same bag. The next section is a look at the area where Diverse Density has already applied but the problem might be better solved by multi-class Diverse Density with the instance volume feature.

5.2 Initial Work on Stock Prediction Task

This section looks at an the initial work of stock prediction task done by [Maron, 1998] where the task was viewed as a multiple-instance learning problem. Unlike supervised learning, multiple-instance learning views stocks as a collection and not individually. Consider a positive example for multiple-instance learning will be a collection of stocks whose value increases from one period to another. Treating stocks in this manner, it is not necessary to concern whether each stock is a true positive example or not (i.e. stock that increases in value for fundamental economic reasons). This information is not easily provided in general. However, by providing more evidence (i.e. more examples), the true positive stock can eventually be identified using the Diverse Density method developed by

[Maron, 1998].

Maron treated each stock, at a particular time, as an instance in a bag and each bag of instances as an example to the learning system. The stock is described as a point in feature space that could be used to describe the unknown fundamental economic factors. A positive bag is a collection of n different stocks from time T, which increase in value at time $T+1$. While a negative bag is a collection of m stocks from time T which decrease in value at time $T+1$.

It is possible to label a bag of instances in this fashion because Maron has made two very important assumptions. The first assumption is that at least one of the n instances (i.e. stocks) in a positive bag increases in value for fundamental reasons. The second assumption is that all of the m instances (i.e. stocks) in a negative bag decrease in value for fundamental economic reasons. In practice, both of these assumptions are difficult to guarantee. One example is that it is possible that none of the n instances in a positive bag are true positives. Another example is the true positive instance could be put in a negative bag as some of the fundamental good stock could decrease in value for some spurious reasons other than fundamental economic reasons (i.e. not all instances in a negative bag are true negative as assumed earlier).

Maron argued that the above two assumptions are held if the following is guaranteed or ensured.

1. The first assumption is ensured by finding a number of n instances such that at least one of the stocks is fundamentally good with high probability. As n approaches infinity, this probability will approach 1. Nonetheless, it will be more difficult to learn the required concept as n approaches infinity. Furthermore, the probability that each stock that goes up in value and is also a fundamentally good stock, is not known and is not independent across stocks or time. Hence difficulty in calculating the probability that at least one of the stocks is fundamentally good.

2. For the second assumption to be true, Maron proposed two solutions on how to generate a negative bag. The first solution is not to generate any negative bags at all. The second solution is to have an expert select the stocks that are fairly certain not be fundamentally good stocks.

3. Spurious reasons such as world events, public whims, etc., that cause the stock price to go up and down, change from one time period to another, while fundamental economic factors are constant. As a result, the distribution of fundamentally good

stocks in feature space will remain constant while the distribution of other stocks will change from one time period to another as spurious factors change.

Maron tested his algorithm (the Diverse Density method) on real financial data supplied by Grantham, Mayo, Van Otterloo & Co. (GMO). The data consists of monthly information for about 600 companies in the US stock market between 1979 and 1995. Each company's stock was described using 17 attributes such as the stock's size, cyclicality, momentum, and trailing returns etc. The learning algorithm was trained on data from a 10 year period and then tested on the data from the following year. Note that in Maron's work, he has shifted the goal of classifying fundamentally good stocks to one of ranking stocks according to the future performance. The closer the stocks in the feature space are to the learned concept of an ideal stock, the higher the rank.

These ranks will be used to decide on a policy to buy and sell stocks. For example, one policy is to buy stocks in the top few ranks (i.e. guarantee high return) and sell ones at the bottom ranks (i.e. eliminate losses), or just buy stocks in the top ranks etc. The ranking of the stocks was compared within three different predictors: the predictor based on multiple-instance learning of ideal stocks, the predictor based on MULTINST and the one developed by GMO. It was found that depending on the buying and selling policy, the multiple-instance learning predictor could outperform both GMO and MULTINST predictors.

In summary, assumptions made by Maron about bags of instances in stock prediction task are difficult to be guaranteed in practice. Hence, bags of instances do not allow the real representation of the stocks such as the possibility of having the false negative stocks in a negative bag (i.e. the stocks that decrease in value for spurious reasons and could be a true positive stock as well). On the contrary, viewing the same stock prediction task as the proposed multi-class multiple-instance learning problem will allow false negative stocks and true positive stock to exist in a negative bag. Furthermore allowing stocks to be classified into many more classes other than a positive (price increase) and a negative (price decrease) class such as classifying them according to their price in comparison with the market index, should help represent the stock problem closer to reality and hence ambiguity can be removed much more easily. It is hoped that the improved multi-class Diverse Density method proposed in the next section will solve this problem and will result in better stock predictor.

5.3 An Improved Method

The general idea behind the multi-class multiple-instance learning approach, that is an extension from the original multiple-instance learning approach, is that it allows multiple classes of examples. Each class has its own true instances, and true instances from other classes are allowed to be present in the bag of the labelled class as long as they are less influential to the label of that bag. Hence, no assumption has to be made about the reasons that made the stocks go up or down in price like those made by Maron. This is ideal for the stock prediction task. Maron's work can still be followed by treating each stock at particular time as an instance in a bag and each stock is described as a point in feature space of the unknown fundamental economic factors. A positive bag means all stocks in the bag go up in price from one particular time period to the next regardless of the reasons behind the increase in price. Similarly, a negative bag means all stocks in the bag decrease in price regardless of the causes behind the decrease.

Unfortunately at the time when this research was carried out, the US Securities and Exchange Commission (SEC) had already imposed a ruling on Regulation Fair Disclosure [U.S. Securities and Exchange Commission (SEC), 2000] since October 2000. According to this regulation, financial firms are not allowed to give out financial data to third parties unless this data is also made publicly available. This means the same raw data used by [Maron, 1998] cannot be obtained and used in this research. As a result, there is a change in the way the problem is approached. This is done by representing stock using only publicly available data such as the movement of stock price during a certain period of time. Unlike financial data from financial firms, publicly available data can only describe each stock very roughly because such data contains fewer features describing stocks. Consequently, stock must be viewed differently so that missing information is no longer important or is substituted. One approach is to look at each stock as an individual and ignore other features used to describe it. This is similar to the way we look at the number set example, one of the stocks can be seen as number 1, another as number 2 and so on. Multi-class classification also helps refining stock description when dealing with smaller set of features. Moreover since volume of stock exchanged affects the price trend, extra features added to the learning method such as the instance volume feature will be proven here to be very useful. Most importantly, the learning problem must be redefined to suit the nature of stock data available.

The learning objective should be changed to suit the new learning task as follows.

- The task is to find a specific stock underlying a specific class, instead of looking for features used to identify such a stock. This is to overcome the problem of describing stocks with fewer features while still keeping the learning task meaningful.
- A class used to classify a collection of stocks will be more specific because it will be classified as either good or bad stock given particular conditions.
- The possibility that a stock belongs to different category at different time series is also allowed.

In the next section, experiments on artificial data are carried out in order to prove that introducing instance volume feature to the calculation of multi-class Diverse Density will improve inefficiency of the learning method. This then follows by the experiment on real stock data where the problem can be described much more precisely using instance volume feature.

5.4 Experimental Set-up and Results

As explained above the experiment is divided into two parts. The first part deals with tests carried out with artificial data. The task is to prove that even when instances are described by very few features, learning the target concept is possible with the multi-class Diverse Density method. Further it will allowed investigation into the way in which instance volume can aid the learning process. The second part is experimenting on real stock data where the problem can be described much more precisely using instance volume feature.

5.4.1 Artificial Data

The artificial data, used to prove the efficiency of the multi-class Diverse Density method, is generated based on the following assumptions.

- The instance underlying each individual class is pre-selected.
- For instances other than the pre-selected ones, they will be randomly assigned to bags and likewise given a random volume each.

The artificial data will be divided into four classes (classes A-D). Let us assume that instances 1-4 are selected as the instances underlying the classes A-D respectively.

It is designed for every training example generated the instance must be assigned in multiple of four bags, one bag for each class. The first bag of each of the four bags generated must at least contain instance 1, the second bag with instance 2 and so on. Each bag of the training examples will have different number of instances depending on the maximum number of possible instances and the randomness assigned to each bag.

The experiment with artificial data is dealt with in the following two sections. The first section investigates the effect of instance volume purely without considering the underlying instances' order of influence. The second section introduces bags of instances that contain more than one underlying instance of different classes. Hence the effect of the order of influence can be investigated.

5.4.1.1 Instance Volume Effect

In this section, apart from investigating the effect of instance volume, four additional parameters will be assigned to each experiment: the number of training sets; the maximum number of possible instances; the maximum instance volume possible; and P_v. As explained in section 5.4.1, each training set is composed of four bags of instances. The four pre-selected underlying instances are each put in the four respective bags, and will be labelled accordingly. Other instances that are not the pre-selected underlying instances are randomly allocated to one or more of these four bags. Each set of four training bags is then presented to the increment algorithm of the multi-class Diverse Density method.

As far as learning the four pre-selected underlying instances correctly is concerned, the learner should always succeed as long as sufficient evidence supporting the target concepts and evidence counter-supporting other concepts are supplied, regardless of the model used or the variation of the parameter P_v. To investigate the effect of instance volume, the three following models are used in this experiment: (i) all-or-nothing; (ii) noisy-or; and (iii) all-or-nothing with instance volume feature. The $ADD-DDm(t)$ value of the target concepts obtained by (i) should be all equal to 1, while those obtained by (ii) should have a value close to 1, and with (iii) should have the values which vary according to the effect of the instance volume. In other words, for the (iii) model the larger the product of the $S_i^v(t=j)$ ratio, as used in equation 5.1, for every training bag seen by the learner, the higher the value of $ADD-DDm(t)$. With this profound effect, not only the target concepts are identified, but its $ADD-DDm(t)$ value also represents its volume.

The bags of instances in this experiment were generated with the random number of instances at the maximum value of 100, and each instance within a bag has a random instance volume at the maximum value of 100. Ten experiments were carried out for each of the three models. Table 5.1 shows an example of the results output by the learner when it learned the target concept successfully. Table 5.2 compares the average number of training sets required between the three models before the learning was successful. As expected, the average values between the three models are very close together at around 7-9 sets or equivalent of 28-36 bags of instances. Note that with the noisy-or model, the learning was considered successful when at least four concepts achieved its $ADD-DDm(t)$ value over the threshold value of 0.01, hence each of these concepts can be considered as a target concept. If the threshold had been set higher, the number of training sets required would have been reduced as a result.

Table 5.1: An example of an output result produced by the learner.

all-or-nothing model with instance volume feature, P_v = 0.5, and succeeded in 8 training sets		
Concept t	Product of $S_i^v(t=j)$ for every bag i supplied to the learner	$ADD-DDm(t)$
1(A class)	7.90e-12	0.00695
2(B class)	1.02e-14	0.00491
3(C class)	6.64e-15	0.00472
4(D class)	1.74e-15	0.00448

Table 5.2: The average number of training sets required for each of the three models before the learning was successful.

Model	**The average number of training sets**
all-or-nothing	7.1
noisy-or	9.2
all-or-nothing - instance volume (P_v = 0.5)	6.9
all-or-nothing - instance volume (P_v = 1.0)	6.6

The second experiment follows the same set-up as the first experiment above but with a fix number of training sets, in this case 100. This experiment is designed to investigate

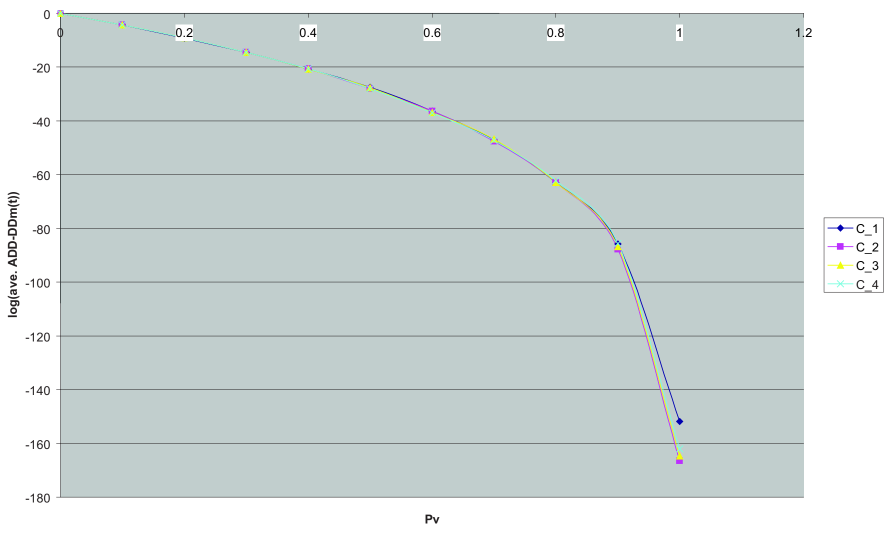


Figure 5.3: The effect of P_v on the average value of $ADD - DDm(t)$ of the 4 target concepts.

how the variation of P_v could have an effect on $ADD - DDm(t)$. For each change in P_v, the 10 experiments were repeated and $ADD - DDm(t)$ was averaged across 10 trials. Figure 5.3 illustrates the effect of P_v on the average value of $ADD - DDm(t)$ of the target concepts. When P_v is small, instance volume has little effect on $ADD - DDm(t)$. Hence $ADD - DDm(t)$ for each target concept is almost the same. Whereas in the case of high P_v, the difference between $ADD - DDm(t)$ of each target concept can be seen more clearly. The higher the P_v, the larger the difference unless instance volume is the same for all four target concepts.

In summary, in case where the order of influence is not considered, adding instance volume to the calculation of the multi-class Diverse Density not only preserves the target concepts' information but also manage to adjust the $ADD - DDm(t)$ values according to the instance volume information. The higher the P_v value the more the effect of instance volume on the $ADD - DDm(t)$.

5.4.1.2 Instance Volume and Order of Influence

In the previous section instance volume was investigated, by extending this test in this section to allow for the possibility that any two of the four underlying instances could exist in the same bag of instances. Hence the order of influence of the underlying instances must be considered so that a bag of instances can be labelled accordingly. The learner was tested on two different bag labelling systems: (i) the pre-selected order system; and (ii) the high-volume system. For example, if the pre-selected order system assigns the underlying instance 1 to be the most influential, followed by 2, 3, and 4 respectively, a bag of instances contains instance 2 and 4 will be labelled class B because of instance 2. Whereas the high-volume system will compare the volume of instance 2 and 4 and assign the class according to the instance with the higher volume.

(i) The pre-selected order system: This experiment was carried out on 100 possible instances in a feature space, and each instance could have a maximum volume of 100. In addition, it is set such that there is a 50% chance that a bag of instances could contain two underlying instances. It was found that the learner failed to learn the correct order of the target concept only when P_v is equal to 0.1 or above. This is because instance volume has more and more effect on the value of $ADD-DDm(t)$ as P_v increases, and this interferes with the evidence supplied by the pre-selected order of influence. Table 5.3 compares the value of $ADD-DDm(t)$ calculated after 100 sets of training examples (each set contains four bags of different classes) were presented to the learners of different models.

Table 5.3: The comparison of $ADD-DDm(t)$ obtained between different models.

Concept	**Model**					
	all-or-nothing	noisy-or	All-or-nothing with instance volume			
			P_v = 0.0001	P_v = 0.001	P_v = 0.01	P_v = 0.1
1	1	0.668	0.99	0.908	0.385	1.00e-5
2	0.125	0.125	0.123	0.108	0.0309	1.00e-8
3	0.0625	0.0625	0.0619	0.057	0.0225	1.00e-6
4	0.03125	0.03125	0.0311	0.0298	0.0193	1.00e-4

The average number of the training sets required before the learner learned the correct order successfully is obtained from averaging the value from 10 successful experiments carried out for each of the models. Table 5.4 illustrates the number of the training sets required for different models before the successful learning.

As expected, it was found that with higher P_v, it is harder to learn and the more likely it is that the learner will learn the wrong order of influence. This is because the values of $ADD-DDm(t)$ reflects the proportion of its instance volume to the combined volume. The lower the $ADD-DDm(t)$ value is compared to that of the normal all-or-nothing

Table 5.4: The number of the training sets required for different models before the successful learning.

Model	**The average number of training sets required before the successful learning for 10 experiments**
all-or-nothing	10.7
noisy-or	12.1
all-or-nothing with instance volume feature	
P_v = 0.0001	11.0
P_v = 0.001	12.3
P_v = 0.01	13.0

model, the smaller the portion of its volume to the volume of every instance combined and vice versa. In other words, the learning of pre-selected order of influence of the underlying instance will only be successful when the value of P_v is lower than 0.1, otherwise the effect of the instance volume is so high that it counteracts the effect of the pre-selected order system.

(ii) The high-volume system: The experimental conditions were exactly the same as those for the pre-selected order system. However, the learning is only considered successful when both of the following two criteria are satisfied. The first criteria is that the four pre-selected underlying instances, instances 1, 2, 3, and 4, should still underlie class A, B, C, and D respectively. The second criteria is that the $ADD-DDm(t)$ of the four target concepts must be ranked according to the product of the $S_i^v(t=j)$ ratio. Note that this ratio will be obtained in a slightly different manner for that used in the previous section (section 5.4.1.1) where there is only one underlying instance in each bag. In this section it is possible that there are two underlying instances of two different classes in the same bag, therefore $S_i^v(t=j)$ ratio will be discarded if instance j is the underlying instance with smaller volume. It has to be discarded because $Pr(t|B_i)$ obtained will belong to a class other than its actual underlying class. The learner was again tested on three models: all-or-nothing; noisy-or; and all-or-nothing with instance volume feature and again with various P_v settings. Table 5.5 shows, for the three different models, the number of successes where the learner learned the correct target concepts and was able to put them in the order of their $S_i^v(t = j)$ ratio. Each model was tested on 10 different random selections from 100 sets of training examples.

Table 5.5: The number of successes out of the 10 runs that the learner of each model learned the correct target concepts and can put them in the correct order according to their $S_i^v(t = j)$ ratio ranking.

Model	**Number of successes out of 10 runs**	**Highest ADD-DDm(t) recorded**
all-or-nothing	0	-
noisy-or	0	-
all-or-nothing with instance volume feature		
P_v = 0.0001	5	3.09e-2
P_v = 0.001	7	2.92e-2
P_v = 0.01	7	1.57e-2
P_v = 0.1	8	3.59e-5
P_v = 0.2	8	6.22e-9
P_v = 0.3	8	7.91e-12
P_v = 0.4	8	2.13e-16
P_v = 0.5	8	3.83e-20
P_v = 0.6	8	1.21e-24
P_v = 0.7	8	2.54e-34
P_v = 0.8	9	1.14e-45
P_v = 0.9	9	2.75e-60
P_v = 1.0	10	6.58e-108

It can be concluded that the learner has to include the instance volume feature into its calculation to be able to learn the target concept according to the high-volume labelling system successfully and its ability to learn improves as the value of P_v used increases. This result is also in coherence with results from previous experiments in section 5.4.1.1.

5.4.1.3 Benefits of Introducing Instance Volume

Without instance volume feature, it will be impossible to learn the correct order of influence of the target concepts if this order is based on the high-volume bag-labelling system. On the other hand, with the other type of labelling system (i.e. the pre-selected order system), it is possible to learn the correct order even when instance volume feature is added to the algorithm as has been shown by Table 5.3 and Table 5.4. Not only is the learner able to learn the correct concepts and their correct order of influence, but the learner is also able to identify the relative volume of the instances close to each identified concept by using the value of $ADD - DDm(t)$ obtained compared to the one obtained from the normal model (e.g. all-or-nothing) as an indicator. However there is still a drawback associated with the multi-class Multiple-Instance learning with instance volume feature. The drawback is that for a labelling system other than the high-volume system, the learning is successful within the P_v range used. For example, P_v above 0.1 cannot be used with the pre-selected order system.

Nonetheless there is a lot of potential here where the multi-class Multiple-Instance learner with instance volume feature can be applied. The task for the designer of the system is to figure out what the system needs to learn. For instance, if the designer knows for sure that instance volume plays an important role in the bag labelling then the learner should have a high P_v value. There is no hard rule on which P_v value the learner should take, through experimentation a good value can be found. Experiment on a few of those values should be a good indicator. Another advantage of this characteristic is that the learner is given the flexibility to adjust itself to various types of learning and different conditions.

5.4.2 Stock Data

In this section, the test on the multi-class Diverse Density method with instance volume feature will be carried out on a large scale, real world, stock data. This data was collected from the trading records of the London stock exchange. The learning system was to monitor the price change of selected stocks over a certain period of time. These stocks would then be classified into groups (classes). Their trade volume was also recorded. For stock data, instance volume is defined here as the multiplication of its price at the time of trading and the stock volume traded. Instance volume has to be defined in this way because of the price differences for each stock.

5.4.2.1 The Task

The task of the learner is to classify stocks into different groups. Because of the limited information on each stock (i.e. only stock price, trading volume, and time of trade are available), it was decided to divide stocks into four groups: good stocks regardless of the trading trend; bad stocks regardless of the trading trend; stocks that rise with the trend; and stocks that fall with the trend.

5.4.2.2 The Bag Generation

The four groups of stocks mentioned in the section 5.4.2.1 can be defined in term of the stock parameters as follows:

"good stocks regardless of the trading trend" is equivalent to stocks that rise in price when the index is down (i.e. class A).

"bad stocks regardless of the trading trend" is equivalent to stocks that fall in price when the index is up (i.e. class B).

"stocks that rise with the trend" is equivalent to stocks that rise in price while the index is also up (i.e. class C).

"stocks that fall with the trend" is equivalent to stocks that fall in price while the index is also down. (i.e. class D).

Note that good stocks and bad stocks could also behave according to the trading trend. However for concept t (in this case stock t) being a good stock, the value of $DDm(t^A)$ should stand out compared to the value of $DDm(t^C)$. Similarly, for t being a bad stock the value of $DD(t^B)$ should stand out compared to the value of $DDm(t^D)$.

For a specific time period (e.g. in every hour that stocks were traded), two bags of instances, each belonging to two different classes are generated depending on the direction of the index. In other words, when the index is up, class B and C bags are generated, while when the index is down, class A and D bags are generated. In our test, we recorded the movement of all of FTSE-techmark stocks (total 204 stocks) on the 19th February 2003. Bags of instances were generated every hour from the opening of the London stock market until it closed on that day. Table 5.6 shows an example of how a stock is assigned to a bag of a certain class. Note that 'TIDM' in table 5.6 stands for Tradable Instrument Display Mnemonic.

Table 5.6: An example on how a stock is assigned to a bag of instances of a certain class.

TIDM	Time	Comparison to previous hour		Volume traded x Price	Class assigned
		price	index		
AIH	9am	down	down	25200 x 27	D
	10am	up	down	35200 x 28	A
AU	11am	down	up	12471 x 138.25	B
	12pm	up	down	18279 x 141.50	A
	1pm	down	up	86352 x 140.50	B
	2pm	down	up	203922 x 140	B
	3pm	down	up	427520 x 139.25	B
	4pm	down	down	549828 x 138	D

5.4.2.3 The Results

- As expected, the learner identified fewer underlying stocks belonging to those four various classes as time progressed (i.e. more bags were generated from the buying-selling record). It went from 23 underlying stocks at 1pm (i.e. 10 training bags so far) down to 17 stocks at 2 pm, then 14 stocks at 3pm, and 10 stocks at 4pm.

- With the all-or-nothing model, $ADD-DDm(t)$ of the underlying stocks were grouped into four levels (1, 0.25, 0.125, 0.0625) according to their order of influence on the label of a bag of instances.

- With the all-or-nothing model with the instance volume feature added, underlying stocks in each of the four levels, as mentioned earlier, were sorted according to the magnitude of their instance volume, but they still stayed in the same groups. Not only was this new ordering based on the stock's instance volume but also on the value of P_v employed. It was found that the order did not change until P_v was set to a high value. For example, at 2 pm the ordering changed when P_v was set at 0.8 and above.

- One way to test the accuracy of the stock class prediction is to compare the price trend at the time when the prediction was made to its price at the time of closing. For example, ACAMBIS stock was identified as class D at 2 pm with the price of 214 pounds, therefore it is expected that its price and index at closing would be lower than that at the identification point if the learner's prediction was to be accurate. Table 5.7 illustrates such results.

Table 5.7: The number of underlying stocks, P_v when the order of $ADD - DDm(t)$ changed, and the accuracy of class prediction by the learner over time.

Time	No. of underlying stocks	P_V when ADD-DDm(t) order changed	Successful class prediction	Accuracy(%)
1pm	23	0.6	13/23	56.52
2pm	17	0.8	10/17	58.82
3pm	14	0.6	11/14	78.57
4pm	10	0.8	9/10	90.00

5.4.2.4 The Discussion

The accuracy of the stock class prediction is very impressive given that there are only a few features describing each stock. The accuracy went up to 90% at the end of the session as described in Table 5.7. Moreover the results have proven that the multi-class Diverse Density method with integrated instance volume feature can be used in a very dynamic domain such as this stock prediction task where the stock data was collected hourly.

In order to assess how powerful the multi-class Diverse Density method is compared to the original Diverse Density method, the same raw data should be tested on the original Diverse Density method. Using the original Diverse Density method, bags of instances for this stock data will have to be arranged as follows:

- There will be only two classes: a positive bag is where the price of stocks in this bag increases from the price at the time they were last sampled (i.e. one positive bag of instances will be generated at every hour); a negative bag is where the price of stocks in this bag decreases at all times. This is due to the assumption imposed by [Maron, 1998] that negative instances have to be true negative instances.

- Due to the same assumption regarding the negative instances as mentioned above, there will be only one negative bag of instances.

With this set-up, it will be impossible to get any meaningful results from the same data using the original Diverse Density because of the following:

- Lack of features describing stocks means that the stock underlying the positive class will only be a stock that rises in prices for the whole session.
- As more data are collected, such a positive stock may never exist.
- Another situation is that as more data are collected it becomes unlikely to guarantee the true negative instances. In other words, a negative bag of instances can not be generated.

Therefore the multi-class Diverse Density method was proven to be better than the original Diverse Density method when dealing with data described by fewer features and when used in a dynamic situation such as the stock prediction task over a short period of time. It was also proven that the instance volume feature improves the learning in the situation where the instance volume has a effect on the bag labelling system.

Chapter 6

Interactivity

The general objective of this research is to show that it is possible to create a concept learning system that can handle ambiguity. It is hoped that this would lead to the creation of an intelligent system based on the philosophy that lies in the middle ground between the traditional and the behaviour-oriented Artificial Intelligence. Not only is learning from ambiguity important but also active learning is the second feature in this proposed philosophy. Active learning happens when the learning system actively decides what questions to ask in order to best refine the concepts it is developing. In other words, a learner is actively asking a teacher to show more examples that the learner thinks would help it learn the concept much more quickly. It is proposed to call this feature ‘interactivity’. This research is also aimed to prove that integrating ‘interactivity’ in the proposed concept learning system would always improve the ability to learn by the system. Since the multi-class Diverse Density method is the proposed concept learning method developed in this research, it is first to show that the interactivity feature can be integrated into the multi-class Diverse Density method. Using the multi-class Diverse Density value as an indicator, the learner should be able to selectively ask for examples that provide better information about the target concept. For instance, a ‘better example’ might be considered as an example that has an instance suspected to be the target concept, hence reinforcing evidence is provided.

A self-recovering system is a good starting platform for this investigation. This is because of the two characteristics of this type of system. Firstly, ambiguity is commonly associated with the causes of the system’s failure. For instance, there are many sources that could cause the same failure but the system will not be able to identify the source without further investigation into the problem. Secondly, the interactivity feature can naturally be integrated into such a self-recovering system. This is because it is natural that such a system might request more tests to be carried out in order to identify the cause of the failure. During this research a novel assembly task that has these two characteristics is formalised. The purpose is to show that the proposed block-world assembly task can be viewed as the multiple-instance learning problem with multi-class labelling system. The nature of the assembly task should also allow the interactivity feature to be embedded with ease, hence the possibility to investigate its effect on learning. The first section describes the proposed block-world assembly task. This then followed by the section explaining how the assembly task can be viewed as the multiple-instance learning problem. The third section reviews the use of the multi-class Diverse Density method to solve the assembly task. The fourth section investigates the function of interactivity feature.

Another good platform for the investigation of the interactivity feature is an interactive document search task. This is a task where the search engine and the users exchange their search information interactively. The search engine has to uncover the list of succinct keywords to describe the user's search while the user identifies the relevancy of documents listed by the engine as a result of the search. Interactive exchange of information between the engine and the search should improve the search dramatically as the keywords uncovered by the engine are refined instead of having the user sift through the document results themselves. The final section investigates an interactive document search task as the multi-class multiple-instance learning problem and the use of the multi-class Diverse Density to solve such a problem.

6.1 The Block-World Assembly Task

An assembly task is a task of putting parts of an assembly together according to a specific plan. The problem scenario is that of an assembly failing unexpectedly even when it was supposed to be a successful operation and the assembly plan generated by an assembly planner was also strictly followed. The assembly planner is believed to generate only successful plans, therefore, it is possible to assume that the failure is caused by unexpected circumstances. One of these circumstances could be uncertainty of equipment, and another could be due to the assembly parts themselves. The latter case is of particular interest to this research because the multiple-instance learning framework could be applied to solve such a problem. As mentioned previously, there should also be an opportunity to explore the interactivity feature within this application.

The nature of the problem is that it is impossible to fit some of the assembly parts

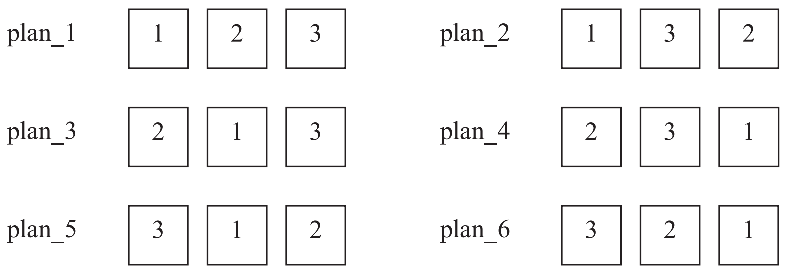


Figure 6.1: Six possible plans generated by assembly planner to arrange three cubes in a row.

together. This is because the planner plans the assembly without having the knowledge of such parts in mind. For example, particular parts cannot be fitted together because they are magnetically repellent to one another. One way to solve such a problem is to identify those parts using an assembly record of failed and successful plans. Having these parts close together should come up repeatedly in the record of failed plans while these parts are not put next to one another in the record of successful plans. If however each plan is viewed as one object, this object is actually ambiguous since it is not known which sections or partitions of the plan are causing the failure. Having viewed the assembly plans in this manner allows the problem to be treated as the multiple-instance learning problem.

It is chosen to represent the above problem scenario as a block-world assembly task because of its simplicity, while preserving the complexity of the actual problem. The assembly task is to put a number of blocks, all in a cube shape, into required patterns such as in a row or in a square of different sizes. For instance, the assembly system is required to put three blocks into a row. The planner should generate six (the factorial of the number of plans, 3!) possible plans to arrange these three blocks into a row. Figure 6.1 illustrates all the six possible plans.

Let us say that, as an example, parts no.1 and no.2 have the magnetic property as described earlier. Four out of six plans generated by the planner (i.e. plan_1, plan_3, plan_5, and plan_6 as illustrated in Figure 6.1) will fail when an actual assembly is carried out. Hence the learning section of the system has the task to identify that assembly parts no.1 and no.2 cannot be placed adjacently if the assembly plan is to succeed. The next section discussed how to view this assembly problem as the multi-class multiple-instance learning problem.

6.2 Block-World Assembly Task as Multi-Class Multiple-Instance Learning Problem

In previous section, it has already been decided to view each assembly plan as an ambiguous object. The next problem must be to characterise such ambiguity so that instances in a bag of instances within multiple-instance learning framework can be defined. As an example used in the previous section, some of the assembly parts having the magnetic property cause the assembly to fail. An assembly plan should be described so that the magnetic property description is also embedded in the plan as ambiguity to be found. With this requirement in mind, relationships between assembly parts can be used as one of the alternative descriptions. In other words, parts that are magnetically repellent cannot be put next to one another or there is to be another normal part put in between any of these parts. This can be described as a relationship between assembly parts and is one of the simplest descriptions possible.

Referring to the assembly task of arranging three blocks in a row in section 6.1, an individual plan could be represented by the relationship between each individual part to one another and to its location in an assembly. For example, plan_1 in Figure 6.1 (i.e. $|1|2|3|$) could be represented by all of the following relationships: primary adjacency - a part is placed next to another; secondary adjacency - there is one other part placed in between this part and another; part and its location - location in an assembly where this part is placed. The list of relationships associated with the primary adjacency relationship group is between parts no.1 and no.2, parts no.2 and no.3. The list of relationships associated with the secondary adjacency is between parts no.1 and no.3. The list of relationships associated with 'part and its location in the plan' is between part no.1 and the first location, part no.2 and the second location, and part no.3 and the third location. In general, other orders of adjacency are also possible: tertiary adjacency - there are two other parts placed in between this part and another; quaternary adjacency - there are three other parts placed in between this part and another and so on.

According to the proposed assembly problem scenario, one or more of these relationships could be responsible for the unexpected failure of the plan. By representing an assembly plan in this way, two possible assumptions could be made about the assembling results.

1. If the plan succeeded, one of the following two assumptions would be correct. The first assumption is that a failure relationship (i.e. relationship that always causes the failure), is not present within the successful plan. An example of such a relationship is primary adjacency between parts no.1 and no.2 used as an example in section 6.1. The second assumption is that a successful relationship (e.g. secondary adjacency between part no.1 and no.2) is always present within the successful plan.

2. If the plan failed, it would imply the opposite to the assumptions described in (1).

Note that there could be multiple failed or successful relationships in one assembly task.

Viewing each individual assembly plan as a collection of relationships between each assembly part and between each part and its location, allows us to fit the assembly task into the multiple-instance learning framework. Each assembly plan is equivalent to an example or a bag of instances. Each relationship within a plan is equivalent to an instance. A successful plan is equivalent to a bag of instances labelled positive, otherwise negative for a failed plan.

Although there are many relationships in one plan, only one or a few could cause either the plan to fail unexpectedly (i.e. the plan is labelled negative) or the plan to always succeed (i.e. the plan is labelled positive). Hence the learning task is to find the underlying relationships (a.k.a. instance that will be used within the multiple-instance learning framework) for each class. Since this problem also allows the possibility that one of the underlying relationships from one class could over-rule other underlying relationships from other classes in labelling the plan and vice versa, the assembly task therefore requires to be viewed as the multi-class problem. For example primary adjacency between part no.1 and no.2 used as an example in section 6.1 will always exists in any failed plan while secondary adjacency between parts no.1 and no.2 does not have to exist in all of the successful plans if there are more than three parts to be assembled.

6.3 The Computation of Multi-Class Diverse Density for The Proposed Assembly Task

It is now decided to look at the proposed block-world assembly task as the multi-class multiple-instance learning task. The learning task treats an individual assembly plan as a bag of instances, and treats an individual relationship between assembly parts or a relationship of the parts to its location in an assembly as an instance. Since these relationships are discreet and not continuous (i.e. there is no variable element to these relationships), the all-or-nothing estimator is adopted.

6.3.1 Simple Assembly Task

Let us first start with simple assembly task such as putting three blocks into a row as mentioned in section 6.1. In this learning task, there are total of six possible bags of instances (i.e. the factorial of number of parts, 3! = six possible plans). Each bag will have exactly six instances (i.e. three relationships of the individual part to its location in an assembly, plus two primary adjacency relationships, and plus one secondary adjacency relationship). However there exist 15 possible relationships (i.e. there are possible nine for part-location and three for each of the two types of adjacency relationship). For simplicity, no.1-15 will be assigned to each of these relationships as shown in Table 6.1.

Table 6.1: Relationships in 3-block assembly task represented as instances for the multiple-instance learning problem.

Instance No.	Relationship Type	Description
1	Part-Location	Part no.1 at the 1st location
2	Part-Location	Part no.1 at the 2nd location
3	Part-Location	Part no.1 at the 3rd location
4	Part-Location	Part no.2 at the 1st location
5	Part-Location	Part no.2 at the 2nd location
6	Part-Location	Part no.2 at the 3rd location
7	Part-Location	Part no.3 at the 1st location
8	Part-Location	Part no.3 at the 2nd location
9	Part-Location	Part no.3 at the 3rd location
10	Primary Adjacency	Part no.1 is adjacent to part no.2
11	Primary Adjacency	Part no.1 is adjacent to part no.3
12	Primary Adjacency	Part no.2 is adjacent to part no.3
13	Secondary Adjacency	Part no.1 is one part away from part no.2
14	Secondary Adjacency	Part no.1 is one part away from part no.3
15	Secondary Adjacency	Part no.2 is one part away from part no.3

Because of the small number of possible bags, instances per bag, and the total number of possible instances, the multi-class Diverse Density value can be computed for each individual concept (i.e. assembly relations in this case). The calculations are carried out by following the procedure described in Chapter3: section 3.3.1.3. A bag number is assigned to be equivalent to the plan number illustrated in Figure 6.1. Each bag or plan will have instances according to the assignment shown in Table 6.1. Table 6.2 summarises the instances to be included in each bag and the label given for each bag in this particular problem.

Table 6.2: All the six possible bags of instances representing a 3-block assembly task with label given to each bag. + represents a successful plan and - is for a failed plan.

Bag No.	Equivalent Plan	Instances	Label
1	$\|1\|2\|3\|$	1,5,9,10,12,14	-
2	$\|1\|3\|2\|$	1,6,8,11,12,13	+
3	$\|2\|1\|3\|$	2,4,9,10,11,15	-
4	$\|2\|3\|1\|$	3,4,8,11,12,13	+
5	$\|3\|1\|2\|$	2,6,7,10,11,15	-
6	$\|3\|2\|1\|$	3,5,7,10,12,14	-

Two cases of the calculations are considered: the first case is where bags of instances are given incrementally and the second case is where all the bags are given all at once. The development of the value of multi-class Diverse Density calculated for the first case is illustrated in Figure 6.2 - 6.9. Figure 6.2 illustrates $Pr(t|B_i)$ for each of the six bags of instances. $Pr(t|B_i)$ will be equal to 1 where concept t is an instance in bag i. Figure 6.3 - 6.8 illustrates for each bag, the computation starting from: updating the soft intersect for bags of the same class; updating the joint instances; updating the calculation of $DDm(t^r)$ for both classes; and computing $ADD-DDm(t)$. Figure 6.9 summarises the history of $ADD-DDm(t)$ as bags of instances were given incrementally to the learner. In the second case, the multiplication of $DDm(t^r)$ of bags of the same label (i.e. the soft intersection) is done only once, as is the search for the joint instances. The end results from both cases should be the same, hence the same final values for multi-class Diverse Density.

The computation of the multi-class Diverse Density suggested that there are three underlying instances (i.e. instances with the highest addition of multi-class Diverse Density value of each class $(ADD-DDm(t))$): instances no.8, no.10, and no.13. Refer to Ta-

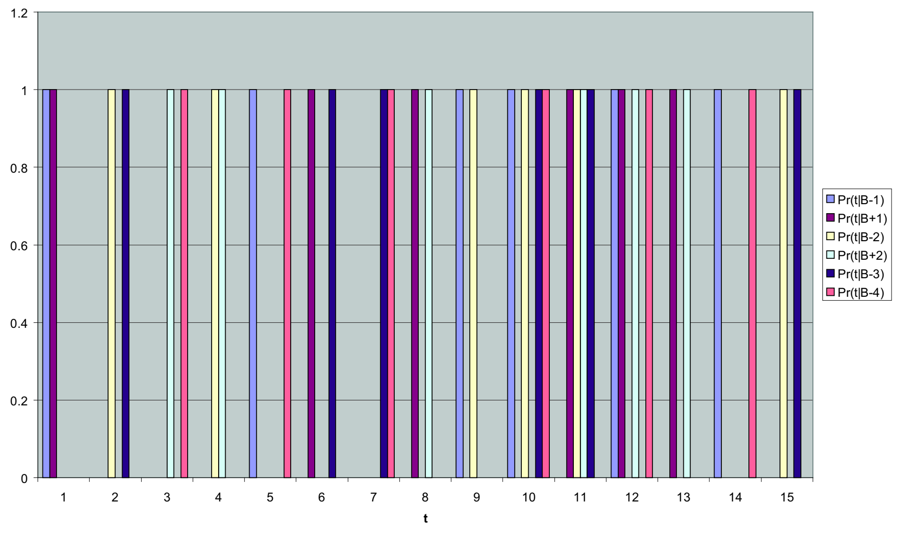


Figure 6.2: The computation of $ADD - DDm(t)$ given all the six bags of instances generated for the 3-block assembly.($Pr(t|B_i)$)

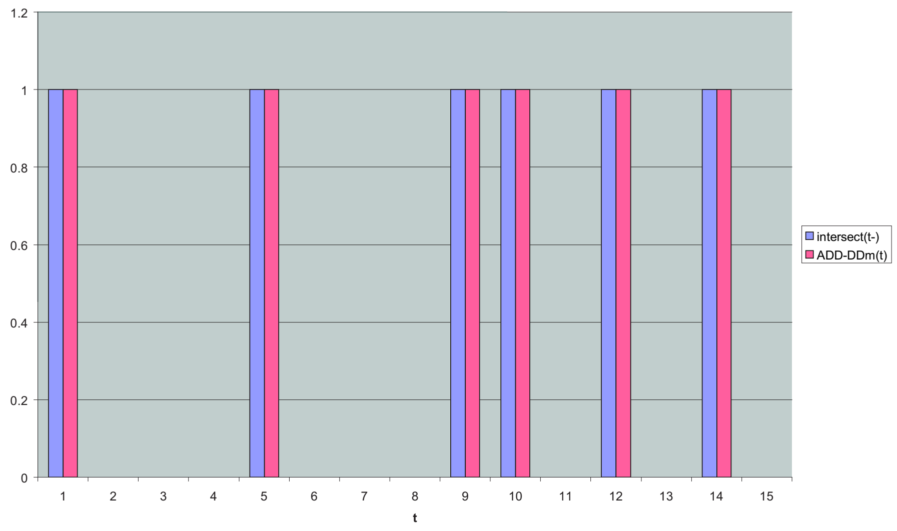


Figure 6.3: The computation of $ADD - DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_1 B_1^-)

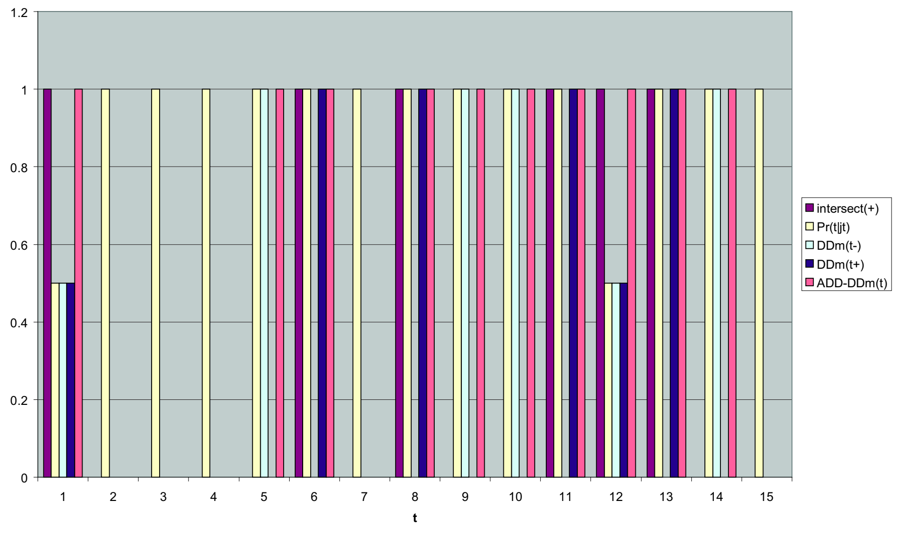


Figure 6.4: The computation of $ADD-DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_2 B_1^+)

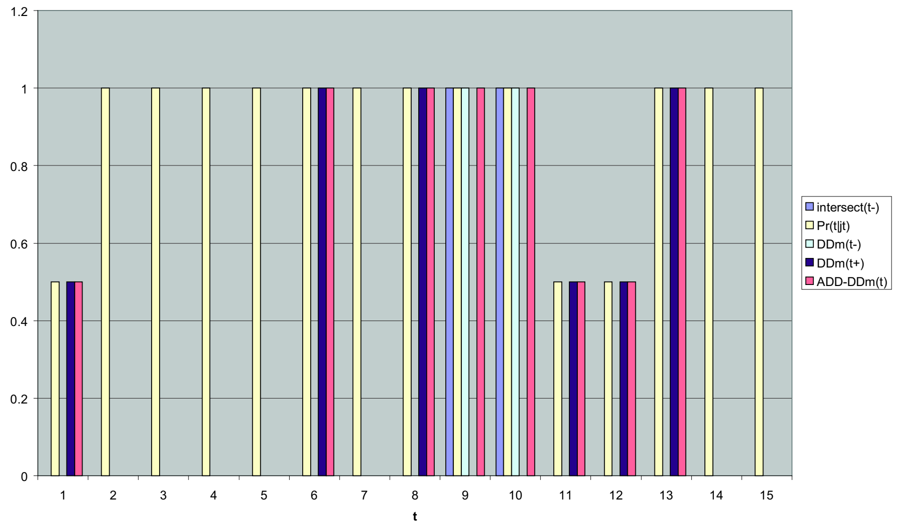


Figure 6.5: The computation of $ADD-DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_3 B_2^-)

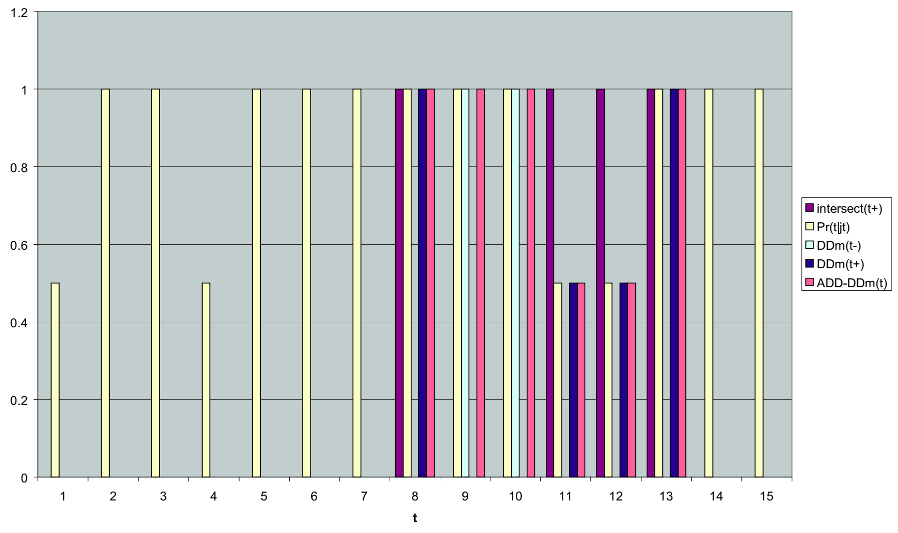


Figure 6.6: The computation of $ADD-DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_4 B_2^+)

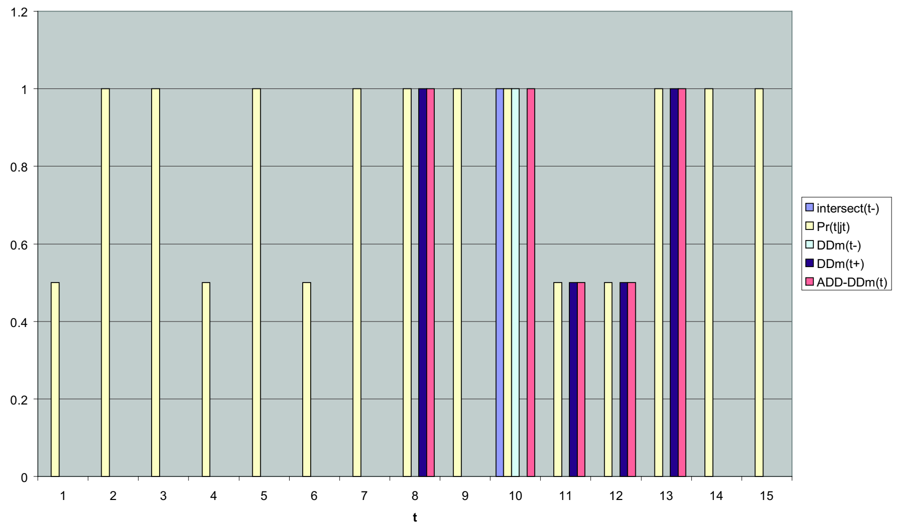


Figure 6.7: The computation of $ADD-DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_5 B_3^-)

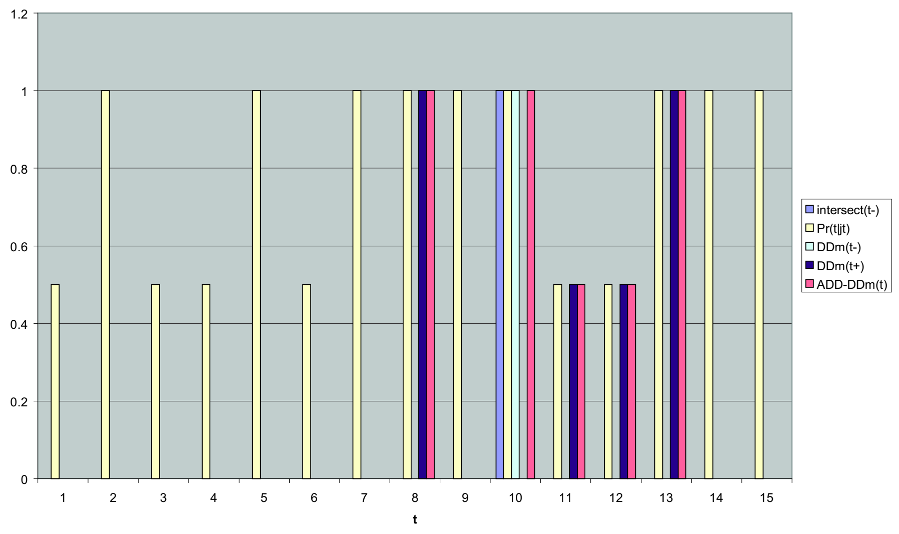


Figure 6.8: The computation of $ADD - DDm(t)$ given all the six bags of instances generated for the 3-block assembly.(Bag_6 B_4^-)

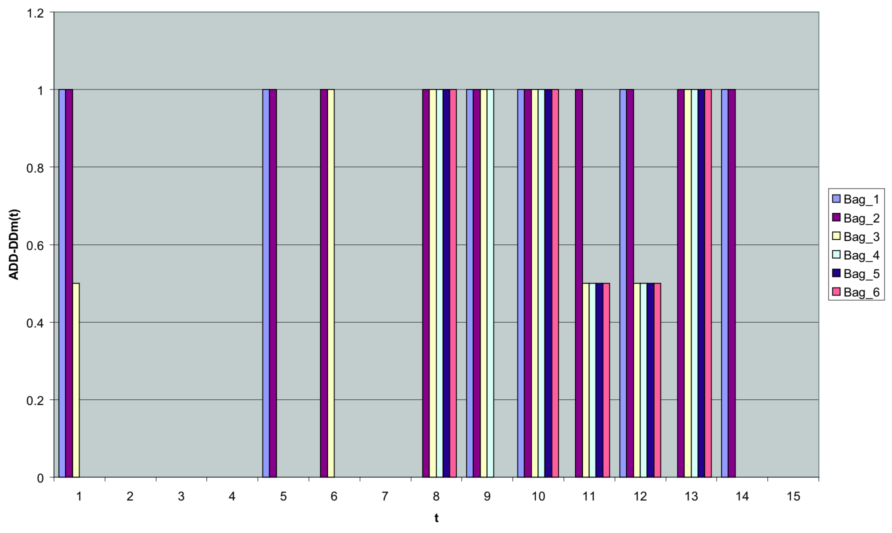


Figure 6.9: The computation of $ADD - DDm(t)$ given all the six bags of instances generated for the 3-block assembly.($ADD - DDm(t)$ history)

ble 6.1, instance no.8 referred to part no.3 being at the second location in an assembly, instance no.10 referred to part no.1 being adjacent to part no.2, and instance no.13 referred to part no.1 being one part away from part no.2. We have already learnt from Chapter 3 that concept t underlies class r where $DDm(t^r)$ contributes the largest portion of $ADD - DDm(t)$. $DDm(t_8^+)$ contributes the most to the value of $ADD - DDm(t_8)$. Similarly $DDm(t_{13}^+)$ contributes the most value of $ADD - DDm(t_{13})$. Furthermore $DDm(t_{10}^-)$ contributes the most value of $ADD - DDm(t_{10})$. It follows that instances no.8 and no.13 underlying the positive class (i.e. successful plans) and instance no.10 underlying the negative class (i.e. failed plans).

Note that in this example, both instances no.8 and no.13 always exist together in a positive bag because the relationship represented by one of the two instances cannot exist without the other being in place as well. Furthermore, all three of the underlying instances equally exert an influence on the label of a bag since underlying instances of positive class never exist in a bag of negative class and vice versa. For example, in this assembly task, it is impossible to have part no 3 in the second location (instance no.8) and have part no.1 being next to part no.2 (instance no.10) at the same time.

6.3.2 Larger Assembly Task

In section 6.3.1, the computation of multi-class Diverse Density of simple assembly task is considered. This section investigates the feasibility of the computation in regard to two aspects: (i) the calculation for assembly tasks with the number of assembly parts increases, and (ii) the calculation for the assemblies of various patterns.

Aspect (i) stipulates that an increase in the number of assembly parts with or without changing the pattern to be assembled will result in the increase in the number of possible instances and the number of possible bags of instances. In addition, as will be seen in aspect (ii), changing the pattern to be assembled will also increase the number of instances per bag. For example, in the previous assembly task where the system has to put parts in a row, the number of possible plans increases from six (3!) plans for a 3-parts task to 24 (4!) plans for a 4-parts task and to 120 (5!) plans for a 5-parts task and so on. The number of instances per bag and the number of possible instances are calculated in a similar manner as described by Table 6-1. Moreover, there will be an increase in the number of different types of adjacency relationship (e.g. two parts in between the two interested parts), hence more instances. Table 6.3 shows the comparison of the number of possible bags, the number of instances per bag, and the total number of possible instances for the assembly tasks of putting three, four, and five parts in a row respectively.

Table 6.3: Comparison of the number of possible bags, the number of instances per bag, and the total number of possible instances for the tasks of putting 3, 4, 5 blocks in a row respectively.

No. of assembly parts	**No. of possible bags of instances**	**No. of instances per bag**	**No. of possible instances**
3	6	6	15
4	24	10	34
5	120	15	65

For Aspect (ii), the consequence of altering the pattern of an assembly without increasing the number of parts (e.g. from a row to a square pattern) will also change the number of possible bags of instances, the number of instances per bag, and the number of possible instances. The number may increase or decrease depending on the pattern. For example, there will be a total number of 24 possible bags for a task of assembling a square of four blocks or six possible bags if taking the rotation of the parts into account. There will be 10 instances per bag (four of the part-location relationship, plus four of the primary adjacency relationship, and plus two of the diagonal relationship), and 28 possible instances.

6.3.3 Experiment of Multi-Class Diverse Density on Larger Assembly

Having proven the multi-class Diverse Density method for a small scale assembly, the main concern is whether it is still feasible to use the multi-class Diverse Density method for a larger assembly (i.e. the number of possible bags and instances keep increasing). If it is still feasible to employ the multi-class Diverse Density method, then it will be of interest to measure the efficiency. The experiment applies the multi-class Diverse Density method to order to find the causative relationship (i.e. the underlying relationship either for the successful or the failed plans) in assembly tasks. The assembly task by arranging blocks in a row, starting from three blocks. Each specific assembly task is tested several times (i.e. 100 tests in total) so that a large portion of bags of instances is covered. In each test the multi-class Diverse Density method was to learn from *n* randomly selected bags of instances, where *n* is a number of blocks to be assembled, so that a reasonable amount of evidence can be drawn from this initial examples. From this point forwards the

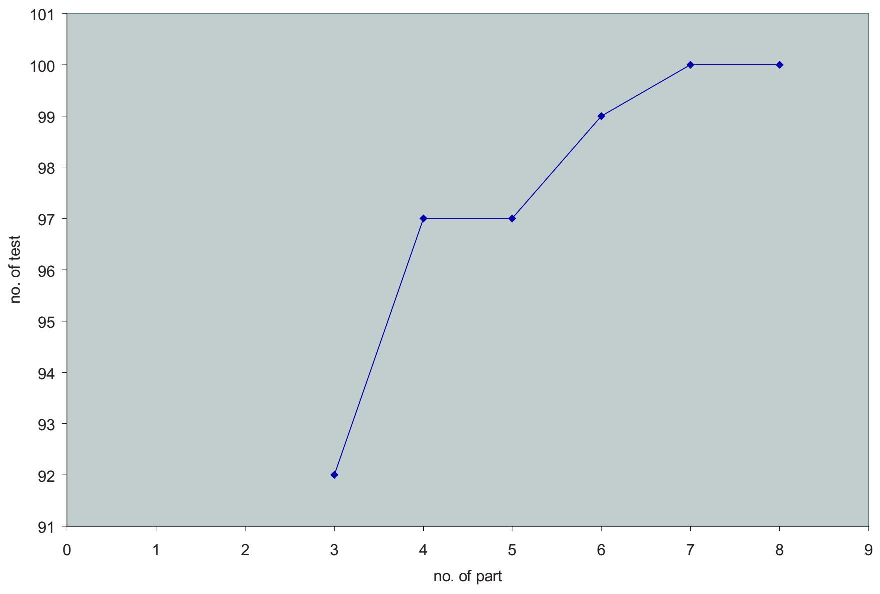


Figure 6.10: The number of tests that the learner learned the correct concepts out of 100 tests VS the number of parts to be assembled.

increment algorithm, as in Chapter 4: section 4.2, is used to aid the learner to learn from a further randomly selected bag of instance, continuing until the causative relationship can be identified. Table 6.4 shows an example of five test results, each for 4-block and 5-block assembly, which were classified into 'learn correct concept' category and 'learn incorrect concept' category. Figure 6.10 illustrates the number of tests that the learner learned the correct concepts out of 100 tests as the number of parts to be assembled increases. Figure 6.11 shows the average number of random bags of instances used before the correct concepts was learned as the number of parts to be assembled increases.

Table 6.4: Examples of test result for 4-block and 5-block assembly. The test results show the number of bags of instances used before the learner learned a concept, five tests for each task.

No. of assembly parts	**No. of bags of instances supplied to the learning system**	
	Learn correct concept	Learn incorrect concept
4	8,8,6,5	4
5	11,6,7,8	5

The learner was less accurate with the assembly of three parts. The learning process

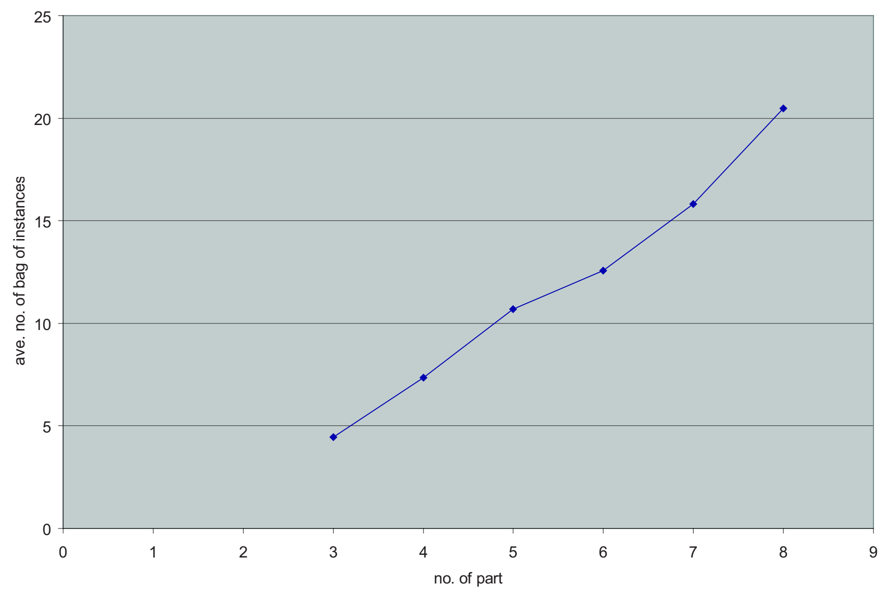


Figure 6.11: The average number of random bags of instances supplied to the learner before the correct concepts was learned vs the number of parts to be assembled.

was stopped in this case because certain initial data contained a combination of three random bags that incorrectly highlighted the faulty concepts due to the high $ADD - DDm(t)$ values. However as the assembly tasks increased in size there was less discrepancy in the evidence supplied to the learner thus the accuracy improved with the size of the assembly. When considering the number of random bags required before it learned the correct concepts, the number did not increase dramatically as the number of parts to be assembled increased. The average number for the 8-block assembly was only 20.48 bags out of the possible 40320 (8!) bags to choose from.

The question now is whether the interactivity feature can be integrated into the learning process so that the process can be achieved with less number of examples supplied to the learner. This is to test whether the learner can be more selective of the examples or bags to look at. This issue will be investigated in the next section.

6.4 Interactivity Feature

It has already been emphasised that the smaller the number of examples (or bags of instances) required to help the learner learn the target concept (in this case is a causative relationship within an assembly task), the faster the learning process and hence the cheaper the computation cost. This section investigates the possibility of integrating the learning algorithm with the ability to select examples to be seen. This ability to select examples should give rise to the smallest number of examples required in order for the concept to be identified. Furthermore, this ability can also be seen as another sign of intelligence - self improved learning. Firstly, the feature that would make the integration of the selection process possible is discussed. This is the feature that the multi-class Diverse Density method also inherits from the original Diverse Density method. This discussion will be in subsection 6.4.1. Secondly, knowledge that could help the example selection process will be discussed. Lastly, the discussion of the integration of multi-class Diverse Density and example selection process will be given.

6.4.1 Diverse Density and Example selection

The original Diverse Density method passively receives training examples (or bags of instances) from the outside world and uses all these examples in the calculation to identify the target concept with the highest Diverse Density value. Each individual bag of instances produces a certain amount of evidence toward the target concept. Depending on how much evidence has already been gathered, supplying one further bag of instances, which provides the right kind of evidence, could give rise to the correct target concept being identified.

For example, let us refer to the number game for the multi-class problem described in Chapter3: section 3.3.1.3, Figure 3.4 and Figure 3.5. If the fourth bag given had contained the numbers 3, 4, and 5 instead of the number 4 as described in Figure 3.5, it would not have been possible to identify that only the number 4 is the instance underlying the negative class. This is due to the fact that there would still be strong evidence suggesting that the number 3 could be the instance underlying the negative class as well. In this example, evidence from the first three bags suggested that both the numbers 3 and 4 have the highest possibility of being the instances underlying the negative class. Therefore in order to provide even stronger evidence to support either numbers 3 or 4, one should try to remove the possibility that one of these two numbers could still be the instances underlying the negative class. The obvious choice is to look for a bag that has one of these two instances present in a bag but does not have the other, and observe the label of this new bag.

It is can be concluded that a new bag of instances, that could provide controversial evidence for or against the evidence gathered from previous bags, is more desirable or recommended than the bag that does not. Therefore, if it is possible to let the learner set a hypothesis from the current evidence and ask for a new bag of instances that could give stronger evidence either to support or to counter-support the hypothesis, then the hypothesis will be proven or ruled out. Allowing the learner to request bag of instances directly related to the target concept identification, this reduces the number of bags and therefore allows the learner to reach the conclusion more quickly. The next section discusses the knowledge that is integrated into the example selection process of the proposed multi-class Diverse Density method.

6.4.2 Knowledge and Example selection Process

The multi-class Diverse Density method could be classified as inductive learning. In inductive learning, the learner looks for hypothesis that is consistent with all training examples. The multi-class Diverse Density also looks for, in this case, the target concept (i.e. a point in a feature space) or the underlying instance of certain classes, that correspond to all the training bags of instances. Generally, one way of helping the inductive learning method to arrive at the target concept much faster is to incorporate background knowledge about the learning task, this background knowledge also describes the training examples. Hence the target hypothesis should be consistent with both the training examples and the background knowledge. Thus, there is more restriction on the search space for the target hypothesis. This idea is not new. This approach is also known as the combining inductive and analytical learning approach. However in the case of the multi-class Diverse Density method, the aim is for the learner to be selective of the training examples considered or used in the calculation of multi-class Diverse Density. The selectivity can be achieved by the learner given some background knowledge about the learning task. Two different types of background knowledge are explored: task specific background knowledge and general background knowledge.

6.4.2.1 Task Specific Background Knowledge

This type of background knowledge is task specific. It is knowledge that might give some clues to how a bag of instances is generated for this specific task. This knowledge also allows the learner to set certain hypotheses about instances. These hypotheses will in turn be proven according to the label given to a test bag of instances. To illustrate this type of knowledge, let us consider the novel assembly task proposed earlier. A planner generates assembly plans, but the planner has certain rules or restrictions on how to generate the plans. For example, in a task of putting three blocks in a row, the planner has the knowledge that in each plan there are to be three part-location relationships, two primary adjacency relationships, and one secondary adjacency relationships. A plan is then transferred to a bag of instances where all relationships between the assembly parts and its location in an assembly and among themselves are described as instances in a bag. Hence instances in one bag are restricted by the assembly plan that the bag describes. Just by altering assembly plans analytically, it is possible to run test to find the trouble relationship by deducting from the success and failure of the plans.

6.4.2.2 General Background Knowledge

General background knowledge is a type of background knowledge that is applied to all learning tasks within the multi-class multiple-instance learning framework. Some examples of this type of background knowledge are as follows:

- A minimum of one instance underlying a certain class must exist in a bag for that bag to be labelled as that class. This knowledge gives the power to the learner to request a bag of instances containing the suspected underlying instance and thus be able to conclude its suspicion from the label of the requested bag.
- A bag of instances will be labelled according to the class of the most influential instance in that bag.

Comparing both types of background knowledge (i.e. task specific vs general), general background knowledge is always true regardless of the application and should constantly be applied while task specific knowledge should only be used as the second resource to enhance the example selection process carried out using general background knowledge. As already mentioned, task specific knowledge might be able to narrow down the search space further but for the case of using the multi-class Diverse Density method, the search is already narrowed down due to the fact that the suspected instances can be eliminated just by testing the label of new bags of instances. This assumption will be proven using experiments in the latter sections.

6.4.3 Integration of Multi-Class Diverse Density and Example Selection Process Using Task Specific Background Knowledge

This section focuses on task specific background knowledge. Firstly, the method by which task specific background knowledge alone can shape the example selection process is described. Following on from this section it will be proven that integrating it with the multi-class Diverse Density method improves the process further.

6.4.3.1 Analytical Deduction Algorithm to The Novel Assembly Task

This section will show that without general background knowledge of the multi-class multiple-instance learning problem, task specific knowledge alone can also be applied to initiate the example selection process and then using analytical deduction algorithm. For comparison purpose the efficiency of this approach will be explored. However it is expected that this approach will not perform so well and there will be some disadvantages, thus the need to integrate the multi-class Diverse Density method. Let us take the same task of putting three blocks into a row as described in section 6.1 as an example, two sets of task specific knowledge can be used (section 6.2). The first set is that the assembly plan for this task can be described as relationships among assembly parts and relationships between the part and its location in the assembly. The second set is that these relationships can be classified into different groups (e.g. part-location relationship, primary adjacency relationship, secondary adjacency relationship etc.). Since relationships in one group could represent the same effect as relationships in another group (e.g. having part_1 and part_2 in secondary adjacency will force part_3 to be in the second location), the causative relationship should be identified within one group before moving on to the next group as it can be seen to be a simple deduction process. By adopting simple algorithm below, it is possible to identify the causative relationship.

REPEAT

- Generate assembly plans, test those plans and keep a record of the results

  (Generate the first plan randomly. Then generate the next plan analytically, which should cover other possible relationships within the selected group and should also allow the possibility to explore the relationships from that of other groups as well)

- Deduct

(Make a deduction to identify the causative relationship from the results of the tests)

UNTIL either the causative relationship from this group is identified or no relationship from the group is proven to be the causative relationship

REPEAT the process for other groups of relationships but also use the record that is created in the previous step to separate relationship to be used in the deduction process from the ones that has already been tested.

For the 3-block assembly task, the following example shows the deduction process being achieved within four steps. Each step is described by the following four aspects: Plan, Reason to choose this plan, Test result, and Deduction result.

STEP1:

- Plan - |1|2|3|
- Reason to choose this plan - randomly selected
- Test result - negaitive (i.e. plan failed)
- Deduction result - the initial choice is to investigate relationship in a group of primary adjacency relationship, hence relationship 1:2 (i.e. part no.1 is adjacent to part no.2) and relationship 2:3 (i.e. part no.2 is adjacent to part no.3) could cause the assembly plan to fail.

STEP2:

- Plan - |2|3|1|
- Reason to choose this plan - it covers the last relationship in the primary adjacency group that has not been investigated (i.e. 1:3) and also explores many different relationships of the part-location group
- Test result - positive (i.e. plan succeeded)
- Deduction result - 2:3 is not the causative relationship that will result in the assembly plan failing because it occurred in both the positive and the negative plans. 1:2 could be the underlying relationship for the negative plans and 1:3 could be that of the positive plans.

STEP 3:

- Plan - |3|1|2|
- Reason to choose this plan - in order to verify which one of 1:2 or 1:3 is more influential
- Test result - negative
- Deduction result - 1:2 could be the underlying relationship for the negative plans.

STEP 4:

- Plan - |1|3|2|
- Reason to choose this plan - From the first three plans, it is gathered that 1:-:3 (i.e. part no.1 is one part away from part no.3) and 2:-:3 (i.e. part no.2 is one part away from part no.3) could not be the underlying relationship for the negative plans. However 1:-:2 could be underlying instance of the positive plans so another plan that contains 1:-:2 can be tested
- Test result - positive
- Deduction result - 1:-:2 could also be the underlying relationship for the positive plans.

Also from these first four plans, it is gathered that 3[2nd] (i.e. part no.3 is in the second position) could also be the underlying relationship for the positive plans because it occurred in all the positive plans.

The above illustrated the simple analytical deduction approach using task specific knowledge of the assembly task. Even though it was successfully applied to the task, the method also has two major disadvantages.

1. The deduction process tries to identify one relationship at a time. Which will lead to difficulty if there is more than one causative relationship in the same group because the test result will not be consistent with the hypothesis about the causative relationship assigned by the deduction process.
2. There is no way of weighting the influence of the identified relationships

It has already been shown in section 6.3 that these two pitfalls could be solved by adopting the multi-class Diverse Density method.

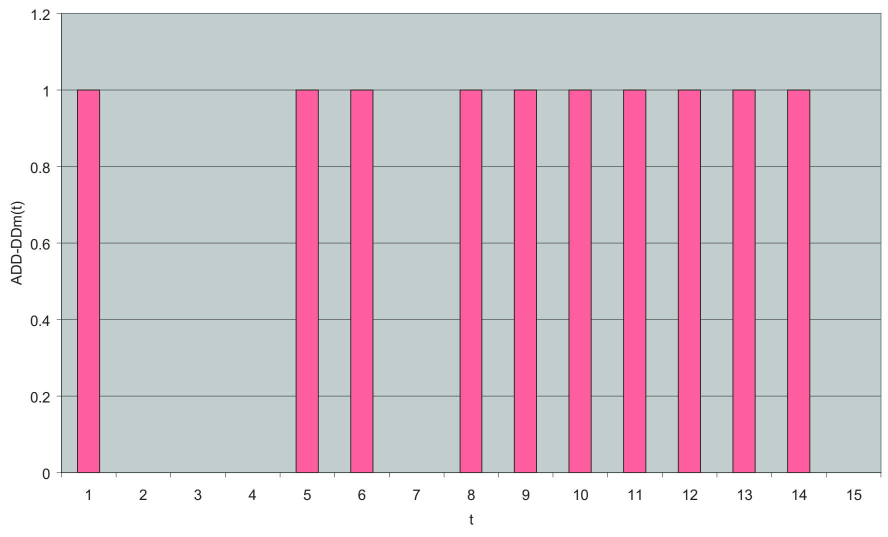


Figure 6.12: $ADD-DDm(t)$ - a copy of Figure 6.4.

6.4.3.2 Multi-Class Diverse Density to Deduction

When comparing the analytical deduction to the algorithm proposed in this section, it is clear that the learning will start by obtaining multi-class Diverse Density profile from a few random training examples before employing the task specific background knowledge to make deductions about the selection of the next example to be tested.

Again the example here will refer to the assembly task considered in the previous approach (section 6.4.3.1). Let us say that the first two assembly plans to be tested are selected at random, and are the same as the first two plans referred to in Figure 6.3 - 6.4 (i.e. bag no.1 and no.2). Hence the $ADD-DDm(t)$ of these two bags combined is the same with that described in Figure 6.4. Figure 6.12 re-illustrates such a result.

This result and the task specific background knowledge about the assembly task suggested the following. The next assembly plan that has not got instances no.1, 5, 6, 8, and 9 should be looked at in order to create diversity in this group of relationships (i.e. part-location relationship). The same assembly plan should have instances no.10 and 11 (i.e. primary adjacency relationship) so that only one type of relationship diversity is investigated. As a result plan $|3|1|2|$ (i.e. bag no.5) was suggested, and the $ADD-DDm(t)$ values turned out to be as shown in Figure 6.13. The evidence suggested that instances no.8, 10 and 13, were underlying instances, where instance no.8 and 13 are of the positive class, and instance no.10 is of the negative class. From this point onward, each further plan tested will only result in stronger evidence supporting this finding. These underlying

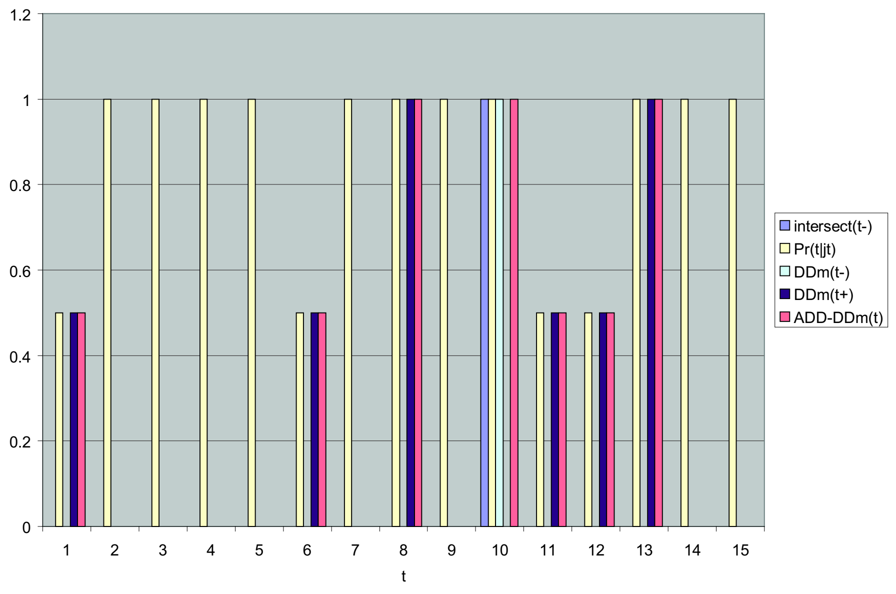


Figure 6.13: The computation of $ADD-DDm(t)$ after bag no.5 (B_3^-) was supplied to the learner.

instances coincide with the result from the previous approach, however a fewer number of bags of instances were required to find the target concepts.

Therefore, it can be concluded that multi-class Diverse Density value can be used in conjunction with task specific knowledge to improve the learning.

6.4.4 Integration of Multi-Class Diverse Density and Example Selection Process Using General Background Knowledge

Applying general background knowledge to create the interactivity feature implies that this feature can be added to the learning process regardless of the learning tasks or applications. The knowledge selected is the knowledge that instances with a high $ADD-DDm(t)$ value of are likely to be class-underlying instances. As the result, the learner can be interactive by requesting for more unseen bags of instances containing the same aforementioned instance found in previous bag. The process is performed in order to gather more evidence to support the case of such instances being target instances.

There is still an additional knowledge that the $ADD-DDm(t)$ value of the target instances will distinguish themselves from other instances if there is strong evidence both supporting these instances being target concepts and counter-supporting other instances being target concepts. The supporting evidence would be that the instance with the highest

$ADD-DDm(t)$ value obtained so far will always be present in bags of the same potential target class but never in bags of other classes, while the counter-supported evidence is the opposite (i.e. either the above instance will not appear in every bag of the potential target class, or the above instance will also appear in bags of other classes). With the above additional knowledge, the learner can therefore be interactive by alternating between asking for another bag of instances containing an instance with the highest $ADD-DDm(t)$ value obtained so far (i.e. finding supporting evidence), and asking for randomly unseen bags (i.e. hoping to find counter-supporting evidence). Randomly unseen bags are favoured because these bags cover a wider spectrum of evidence. In other words, as much as having the suspected instances in bags of different classes acts as the counter evidence, so does not having the suspect instances in other bags of the same class.

Two sets of experiments are proposed. One set of experiments examines the learner being interactive based on high $ADD-DDm(t)$ value alone, and the other set examines the learner being interactive using the alternating cycle. The results of both experiments should point in the direction that the alternating cycle approach is better at learning compared to non-interactivity approach (section 6.3.3) and the interactivity approach based on only high $ADD-DDm(t)$ value. This is to say that it should use the least number of bags of instances before arriving at the target instances.

In this experiment, the multi-class Diverse Density method was applied to find the causative relationship of assembly tasks of arranging blocks in a row starting from three blocks. Each specific assembly task is tested several times (i.e. 100 sets each in total) so that a large portion of bags of instances was covered. Each test will begin by letting the multi-class Diverse Density method learn from n randomly selected bags of instances in order to identify the instance with the highest $ADD-DDm(t)$ among these n bags, where n is the number of parts to be assembled. Further unseen bags are searched for until a bag containing the same high $ADD-DDm(t)$ valued instance is found. This bag is then supplied to the increment algorithm as described in Chapter 4: Section 4.2. This procedure is repeated until until the number of instances that have $ADD-DDm(t)$ values more than a threshold value, is less than or equal to number of the blocks to be assembled. This criterion is only used as a rough maximum bound for the number of causative relationships possible.

The next experiment follows a similar procedure to the previous one. However, bag selection is realised slightly differently. Rather than always searching for instances with high $ADD-DDm(t)$ value and submitting the bags containing these, this experiment in-

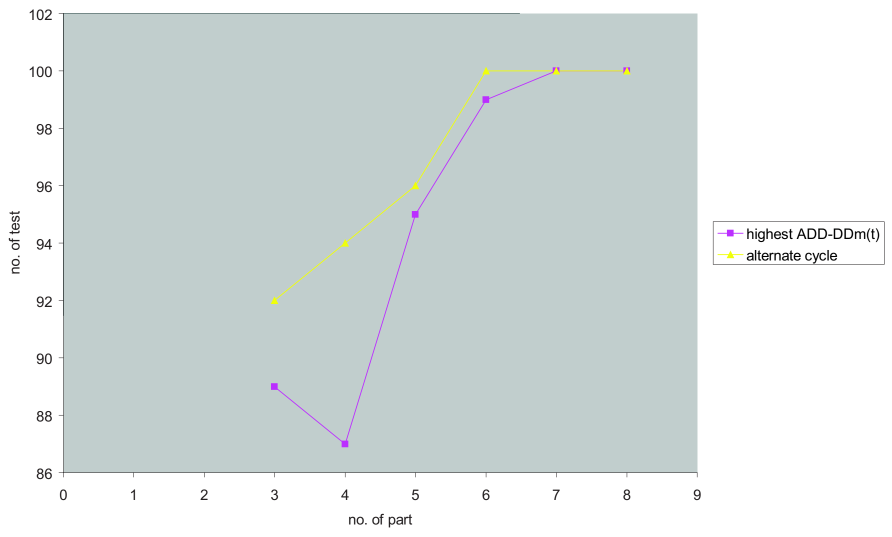


Figure 6.14: The number of tests that the learner learned the correct concepts out of 100 tests for both experiments (i.e. the highest $ADD - DDm(t)$ approach and an alternate cycle approach) vs the number of parts to be assembled.

tersperses each of such unseen bags with randomly selected unseen bags. Figure 6.14 illustrates for both experiments, the number of tests out of the 100 tests where the learner learned the correct concepts against the number of parts to be assembled. Figure 6.15 shows, for both experiments, the average number of bags of instances used before the correct concepts were learned against the number of parts to be assembled. Figure 6.16 compares, between three approaches, the average number of random bags of instances used before the correct concepts were learned against the number of parts to be assembled. The three approaches are as follows: a random approach with results already shown in Figure 6.11; the highest $ADD - DDm(t)$ approach; and an alternating cycle approach.

The results of these experiments showed that the interactivity feature can be integrated into the multi-class Diverse Density method. This is also met with the expectation that learning benefits the most from the alternating cycle approach. With this approach, the learner requests for better-balanced evidence (i.e. supporting and counter-supporting evidence) compared to the other two approaches and hence faster learning is achieved.

6.5 Interactivity Feature and Document Search Task

The majority of research on document search engines concentrates on getting a perfect search. Such search methods list documents in order of relevance using information from

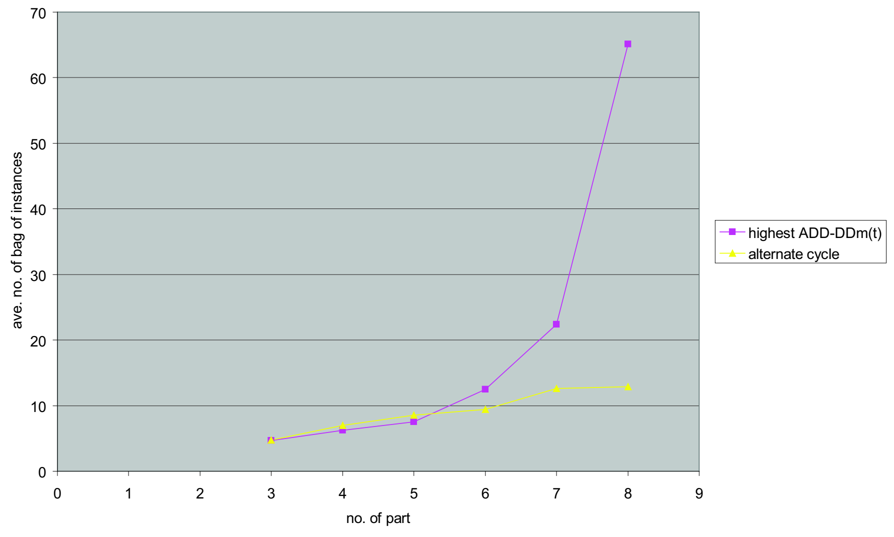


Figure 6.15: The average number of bags of instances used before the correct concepts were learned for both experiments (i.e. the highest $ADD-DDm(t)$ approach and an alternate cycle approach) vs the number of parts to be assembled.

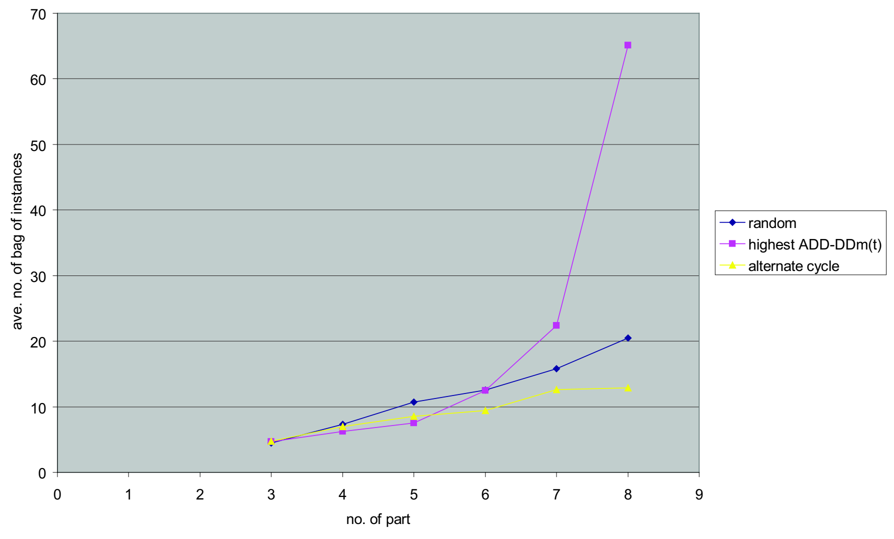


Figure 6.16: The comparison of the average number of bags of instances used before the correct concepts were learned for the three approaches (i.e. a random approach, the highest $ADD-DDm(t)$ approach and an alternate cycle approach) vs the number of parts to be assembled.

the sources, such as citation information, instead of getting further help from the user to direct the search. The article by [Hu et al., 2001] outlined the World Wide Web search technologies and classified them into five major categories: (i) hyperlink exploration, (ii) information retrieval, (iii) metasearches, (iv) SQL approaches, and (v) content-based multimedia searches. None of the five categories of search mentioned above makes use of user feedback to refine the search. This section sets out an interactive document search method that allows a user to report to the search engine their opinion on how relevant each document is. The search then uses this information to narrow down the list of documents given to a user as a result of the search. As more and more opinion is given to the search engine the list should get even smaller and even more relevant to the user's query.

It is proposed to look at a document search task as the multi-class multiple-instance learning problem and to make the search interactive by introducing the interactivity feature into the search similar to as has been done with the assembly task mentioned earlier in this chapter. The next section outlines a document search task as the multi-class multiple-instance learning problem. This is proceeded by a discussion on the actual interactive document search, followed by the experiment and results on the system in the last section.

6.5.1 Interactive Document Search as Multi-Class Multiple-Instance Learning Problem

When searching for documents, a user usually inputs a few keywords into a search engine. The engine then outputs the list of documents that contain those keywords. It is the job of the user, after that, to go through the list and select the documents that are relevant to the search idea. Not all documents in the list are relevant even if they contain all the keywords supplied by the user. This is because of either of these two cases: only a few keywords are not enough to define the idea behind the search; the user develops a refined idea of their search while looking through some of the documents. A novel alternative to this is instead of having a user reading through and selecting the relevant documents manually, an interaction between a user and a search engine is introduced. A user will be asked to classify any document they read into different groups according to the relevancy of the document to their search idea, and to report these results back to a search engine. The search engine must then learn from the user's response in order to identify any common feature underlying the search idea, and from this output a more refined document list. This interaction will be iterated until either the user is happy with the search list or the engine cannot refine the search any further.

Let us consider a general search idea in more detail. It is proposed that there are three types of keywords that constitute any search idea. A search idea can be more refined when one or more types of keywords are discovered. The three types of keyword are as follows:

General keywords: these keywords are normally supplied by a user, the more the general keywords are discovered, the better the search.

Hidden keywords: this type of keywords can be considered as a part of or a subset of another keyword. For example if 'a learning algorithm' is a keyword, any learning algorithm such as Neural Networks, Reinforcement Learning and the like become hidden keywords because the user could be interested in a document that describes at least one of these methods.

Negative keywords: if these keywords appear in a document, they will remove this document from the user's desired list. For example, after reading through some of documents for a while, a user realises that they are only interested in documents about Neural Networks among all other algorithms, any learning algorithm other than Neural Networks should become negative keywords. Most of the search engines do not pay attention to this type of keywords when in fact they could narrow down the search dramatically.

Keywords of different types will appear in different groups of documents. General keywords should appear in every document desired by a user, while hidden keywords may appear in some of the documents desired by a user, and none of negative keywords should appear in any document desired by a user. Therefore asking a user to classify documents into groups could help a search engine uncover more of these keywords, hence refining the search.

The interaction back and forth between a user and a search engine can enhance the multiple-instance learning framework, if each document is considered as a bag of instances and each word within a document is considered as an instance of a bag. Each document will be labelled into different classes by a user depending on how close the content is to the user's search idea. Each document is also a collection of many words in which only a portion of these words constitute its label. In other words, keywords of each of the three types mentioned earlier underlie different classes of documents. Because all these keywords must be discovered and not just one type of keywords for a search engine to be able to refine its search, the problem becomes that of a multi-class problem.

As an example, let us assume that the following four documents are an initial result of a search with ‘cat’ as a keyword and each contains only one sentence. After removing articles, pronouns, prepositions and other grammatical words that are not going to be the main idea of a sentence, and after considering only a root of words such as cat and cats, get and got etc., words that are treated as instances for each document are obtained.

“The cat is in the hat.” $\Rightarrow$ {cat, hat}

“A cat is a fine pet.” $\Rightarrow$ {cat, fine, pet}

“Dogs and cats make good pets.” $\Rightarrow$ {dog, cat, make, good, pet}

“I haven’t got a cat.” $\Rightarrow$ {have, get, cat}

If a user is looking for a document about cats as a pet as their search idea, the first document could be labelled as a ‘least relevance’ class by a user, the second document as a ‘most relevance’ class, the third document as a ‘relevance’ class and the last document as a ‘least relevance’ class. The search engine should then learn that the words ‘pet’ and ‘fine’ are likely to be hidden keywords, while ‘hat’, ‘have’ and ‘get’ are likely to be negative keywords. In a real world situation, the initial list of documents will be large, therefore hidden keywords and negative keywords uncovered by the search engine will only be a small portion compared to all the words listed as instances. Hence the new list of documents output to a user should be a much smaller list. Moreover this list should get smaller and smaller if the search engine keeps interacting with the user until no more new keywords are discovered.

6.5.2 Interactive Document Search System Structure

A process starts with a user inputting a few keywords to the search engine. The search engine then output the result of the search back to the user. The user randomly selects a number of documents and labels them according to relevancy to the user’s search idea and returns the label to the search engine. The search engine then sends these selected documents together with their labels to the bag generator (BG in Figure 6.17). The bag generator then outputs a set of bags of instances to the multi-class Diverse Density method (DDm in Figure 6.17). Instances (words) with high $ADD-DDm(t)$ are obtained. These instances are the words underlying different class of documents and will be output to the search engine so that the search engine can use this information to remove irrelevant documents. The refined list of documents is then presented to the user again and the same process is repeated until the document list cannot be further refined or the user is happy with the search. The diagram in Figure 6.17 summarises this process.

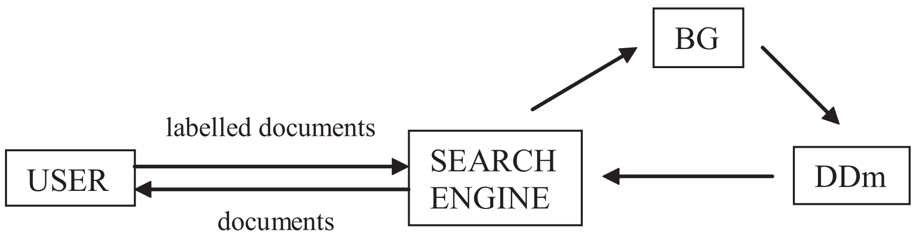


Figure 6.17: The structure of the interactive document search system.

6.5.3 Experiments and Results

6.5.3.1 Bag Generator

Special thank to Lucien Murray-Pitts for the development of Bag Generator. The bag generator generates a list of words and word frequency as they appeared in each document. The generator has a library of articles, pronouns, prepositions and other grammatical words that do not affect the main idea of a sentence. Therefore those words will be removed from the list by the generator. The generator also maintains another library where each new word encountered by the generator is attributed with a unique identification number (UID). This UID will be used as an instance in bag similar to what was carried out on the artificial data in Chapter 3: section 3.3.1.3. Each document is then represented as one bag of instances where each instance is a UID representing words in the document. For example, four documents about cats were given as example documents in section 6.5.1, each word that is treated as an instance (e.g. cat, hat, fine, pet and dogs) will be assigned with the separate UIDs.

6.5.3.2 Experiment

The initial list of documents was a random selection of 50 documents listed during a search using the engine from http://citeseer.ist.psu.edu website with ‘learning’ as the keyword. Only the abstract of each document is considered for the purpose of this experiment. The task is for the multi-class diverse density method to discover that ‘Neural Network’ is a hidden keyword while ‘Reinforcement Learning’ is a negative keyword. From the 50 documents, 10 documents are randomly selected and labelled into one of four classes (i.e. most relevance, 60% relevance, 30% relevance, and irrelevance), based on the two keywords above. The learning algorithm learns from these 10 documents and outputs the more refined list taken from the first 50 documents based on the new keywords learned. In real life, the system initiates the interactivity by outputting the above refined list back

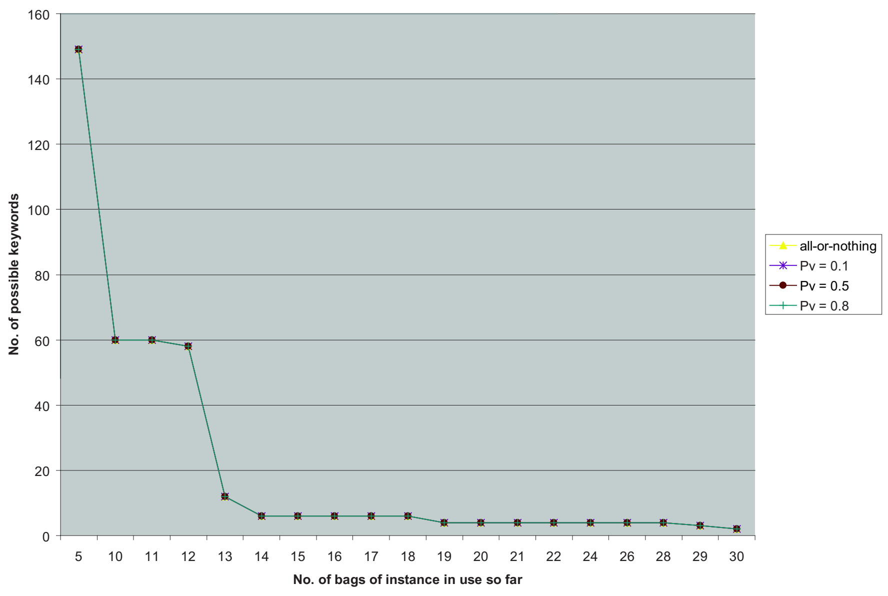


Figure 6.18: The reduction in the number of possible keywords suggested by the multi-class Diverse Density method as the number of bags of instances presented to the method increases.

to the user and the user will then pick some of these documents to be labelled and return the labels back to the system. To duplicate this action for the purpose of this experiment, another 10 documents are randomly selected from the first refined list mentioned earlier instead and the experiment is repeated until the list cannot get any smaller. The experiment is repeated 10 times. It was found that it took an average of three consecutive runs of the multi-class Diverse Density method before the list cannot get any smaller and each time the list was reduced by an average of 50% over the previous list.

Alternatively, instead of considering the reduction in the size of relevant documents, the reduction in the number of keywords suggested by the multi-class Diverse Density method can also indicate the efficiency of the method. Figure 6.18 shows the reduction in the number of possible keywords suggested by the multi-class Diverse Density method as the number of bags of instances presented to the method increases. The same figure also shows the the same reduction when the instance volume feature (as discussed in Chapter 5) was introduced to the calculation model of multi-class Diverse Density value. When introducing the instance volume feature into the calculation of the multi-class Diverse Density value, it does not affect the change in the number of possible keywords obtained from the same set of bags of instances but it affects only the value of multi-class Diverse Density of each keywords. The values decreases as P_v used in the calculation decreases as already detailed in Chapter 5. Introducing the instance volume feature does not affect the change in the number of possible keywords in this interactive document search task because there is not a huge difference in the volume of each word in each document (i.e. only the abstract is considered for the purpose of the experiments). However the instance volume feature will be useful when dealing with normal documents with many repeated words.

To summarise, it was proven that the multi-class Diverse Density method can be employed for the document search task, and the interactivity feature of the multi-class Diverse Density method also improves the efficiency of the search.

Chapter 7

Discussion

This chapter will be divided into three discussion sections. The first section will dedicate to the comparison between the multi-class Diverse Density method and algorithms other than those developed for the multiple-instance learning framework. The comparison between the multi-class Diverse Density method and the Diverse Density method is summarised again in the second section. The final section investigates the differences between the multi-class Diverse Density method and algorithms developed for the multiple-instance learning framework other than the Diverse Density method.

7.1 Comparison to Other Machine Learning Approaches

In order to evaluate the functionality of the multi-class Diverse Density method, a comparison is made between other machine learning algorithms. However, since the multi-class Diverse Density method inherits many features that also belong to the original Diverse Density method, the comparison must refer to the original multiple-instance learning framework and the Diverse Density method as well.

7.1.1 Why Is It Not Possible to Consider A Two-Class or Multi-Class Multiple-Instance Learning Problem as A Regular Supervised Learning Problem?

First let us consider whether it is possible to transform the original multiple-instance learning problem (i.e. two-class case) into a supervised learning problem. The answer is that it is possible to do so but it does not happen without discrepancy.

- If the transformation were to treat every single instance in a positive bag as a true positive label, whereas only one instance in a bag is a true positive in the original multiple-instance learning framework (MIL), and every single instance in a negative bag as negative, the positive labels would be corrupted (i.e. very noisy). This is because of the arbitrary low ratio of true positive to false positive instances. Hence the learning is likely to fail.
- If the transformation to a supervised learning problem was done by making each bag a single example, with all the instances in a bag concatenated together to form a single vector, it would be very hard to pinpoint the feature that represented the true positive instance before it was concatenated. Moreover, every bag can have a different number of instances, hence leading to examples with varying number of features.

As in the case of multi-class problems, not only does the positive label become corrupted but every class label will also become corrupted. Again this is due to the arbitrary low ratio of true class label instances to false instances of the same label. Thus leading to ambiguity of bag labelling. Therefore it is not possible to consider both the two-class and the multi-class multiple-instance learning problem as a regular supervised learning problem.

7.1.2 Are Both The Diverse Density and Multi-Class Diverse Density Methods Employed to Solve The Multiple-Instance Learning Problems Simply ‘Naive Bayes Classifier’?

Instead of looking for the concept underlying the label of the bag of instances, identifying the label of the given bag of instances is another way of looking at the multiple-instance learning problem. Within a regular supervised learning problem, the latter task would be better thought of as identifying the label of an instance given its attributes, and it could easily be accomplished using ‘Naive Bayes Classifier’.

The Naive Bayes Classifier simplifies the following expression $Pr(a_1, a_2, \ldots, a_n|v_j) = Pr(a_1|v_j) \cdot P(a_2|v_j) \cdot \ldots \cdot P(a_n|v_j)$ if each value ‘a’ is conditionally independent given the value ‘v’. The value ‘a’ would be an attribute of an instance and the value ‘v’ would be the possible label of the instance within a regular supervised learning problem.

However this assumption would not be true in the case of MIL if the value ‘a’ is equivalent to each instance in one bag and the value‘v’ is the possible class or label of that bag. It is simply because not all instances are conditionally independent given the class of the bag. For example, false positive instances would not influence the bag to be labelled positive unless it was present together with the true positive. Furthermore if we assumed that the Naive Bayes assumption was held, $Pr(a_1, a_2, \ldots, a_n|+)$ would be low because $Pr(a_{false-instance}|+)$ would be very small and there are so many of false instances in one bag. In fact according to the multiple-instance learning framework, given one of the instances is a true positive, the $Pr(a_1, a_2, \ldots, a_n|+)$ should be high and close to 1.

On the other hand, the Naive Bayes will apply if the value ‘a’ is a single bag of instances because each bag of instances is conditionally independent given the label of the bag. This was employed by the original Diverse Density method for the two-class problem where the Diverse Density value obtained for each concept t, $DD(t)$, is as follows: $DD(t) = the\ maximum\ likelihood\ of\ [Pr(the\ 1^{st} positive\ bag|t) \cdot Pr(the\ 2^{nd} positive\ bag|t) \cdot\ \ldots\ \cdot Pr(the\ n^{th} positive\ bag|t) \cdot Pr(the\ 1^{st} negative\ bag|t) \cdot Pr(the\ 2^{nd} negative\ bag|t) \cdot\ \ldots\ \cdot Pr(the\ m^{th} negative\ bag|t)]$.

The interesting question then, is how each individual probability of a given bag can be modelled to give rise to the highest diverse density value for the correct target concept. This problem has already been investigated by [Maron, 1998] for the two-class problem. The same is also applied for the multi-class case where the problem posed is how each individual probability of a given bag should be modelled to give rise to the highest multi-class Diverse Density value for the correct target concept for each individual class. However, the answer to the question originally posed is that the Diverse Density and multi-class Diverse Density methods are not ‘Naive Bayes Classifier’ unless a whole bag of instances is considered rather than on an individual instance basis.

7.2 Comparison to Diverse Density

Hitherto, the differences between the multi-class Diverse Density and the original two-class Diverse Density methods are only dealt with in regards to the problem description and a mechanism to obtain target concepts. In this section the differences will be discussed based on the gathering of the results from experiments carried out during this research.

7.2.1 Is It Necessary to Replace Diverse Density with Multi-Class Diverse Density in order to Solve The Multi-Class Problems?

It might be possible to consider the multi-class problem as solving multiple sets of two-class problems, if each set was to select bags of one class as positive bags and all others as negative bags. However the problem with this approach is that the multi-class problem was redefined so that a bag of instances labelled as one class could also have true instances of other classes in it depending on the order of influence. Hence it is not possible to treat a bag of instances other than that of the selected class as having all its instances being negative instances of that selected class. Since the model for the probability of concept t suggested by the original Diverse Density method will penalise a great deal if t is close to instances that are considered as a negative instance. Therefore it is not possible to apply such a model to false negative instances presented in multi-class problems. The result was to introduce the notion of joint instances.

In addition to assuming all-true negative instances, the original Diverse Density will not allow the order of influence of the class label to be compared. This is also an important reason why the Diverse Density method cannot be used to solve the multi-class problem.

7.2.2 Instance Volume and Interactivity

7.2.2.1 Instance Volume

The original Diverse Density method not only considered a single point concept class (i.e. a concept represented by a single instance), but also considered other types of concept classes such as a single point-and-scaling concept class and a disjunctive point-and-scaling concept class. A single point concept is seen as a single point in a feature space. However with scaling, each point in a feature space is accompanied by another vector, where each component of the vector is a weight assigned to each feature in a feature space. Each weight is also considered as an unknown to be found by maximising Diverse Density with respect to the weight values.

In the case of the proposed multi-class Diverse Density method, concept classes other than the single point concept class are avoided as much as possible for simplicity. For example, in the assembly task investigated in Chapter 6, each relationship (i.e. relationships between parts and their locations) that could cause an assembly to fail unexpectedly is designed to be an instance within a bag of instances. Since the single point concept class is sufficient to represent such a relationship, scaling is not necessary. In the application where scaling of features could contribute to learning such as in the stock prediction task, scaling was considered in a macroscopic view so that it is no longer an unknown to be discerned like that in the original Diverse Density method. Each stock was to be viewed as an individual. Hence feature values describing each stock were viewed as a whole set instead of individually. Therefore scaling should be applied to the whole set of features. Instance volume was then introduced as a scaling factor applied to the individual stock.

Moreover, although a scaling factor was an unknown to be found in the original Diverse Density method, it can alternatively be described in the multi-class Diverse Density method as instance volume, which is found directly from the data (i.e. stock volume traded). Therefore due to the introduction of the instance volume feature, not only does learning become simple by using the single point concept class but it also becomes possible to do so while keeping the learning task meaningful.

7.2.2.2 Interactivity Feature

One topic within the multiple-instance learning framework that has not been covered before is the area where the learner could become more interactive in selecting learning examples. Instead of passively receiving learning examples, being interactive will allow the learner to adopt a strategy to reduce the numbers of example to be seen. Hence shortening the time and reducing the cost of the learning. A strategy such as 'seeing more examples containing instances close to the concept with the highest $ADD-DDm(t)$' was investigated. Another strategy adopted was based on the former but also introduced diversity by alternating between examples belonging to the former strategy and random examples to be learned. Experiments on a multi-class assembly task showed that the results were in favour of the second strategy over the passive strategy (i.e. random example feeding), and that the passive strategy was in favour over the first strategy.

This finding signifies that the immediate value of $ADD-DDm(t)$ computed should be interactively used by the learner as an indicator for finding supporting evidence for potential target concepts. However, interactive request of examples will not achieve its full potential unless evidence counter-supporting the potential concept is also introduced such as through random example selection.

Since both the multi-class Diverse Density and the original Diverse Density methods rely on the same principle that their value increase exponentially with supporting evidence and decrease exponentially with counter-supporting evidence, the interactive strategy such as was mentioned in the previous paragraph can also be applied to the original Diverse

Density method with similar results. The benefits increase sharply as a problem gets more complicated (i.e. either larger number of instances in bags or more bags of instances are used).

7.3 Comparison to Other Algorithms Proposed for Multiple-Instance Learning

Apart from the original Diverse Density method, there were other algorithms developed for the multiple-instance learning framework, already discussed in Chapter 2: section 2.2. Here the proposed multi-class Diverse Density method is compared to three other main algorithms. Each algorithm was developed for the two-class problem based on the three well-established machine learning algorithms: a lazy learning; neural networks; and decision tree / decision rule. The comparison made for each three methods will start with the suggestion of how to adapt the algorithm to accommodate a multi-class problem. These are then followed by a discussion of their pitfalls or disadvantages compared to the proposed multi-class Diverse Density.

7.3.1 MIL - A Lazy Learning

With reference to Chapter2: section 2.2.3.1, the success of the lazy learning approach applied to multiple-instance learning (MIL) depends on a good selection of appropriate nearest neighbours. For a two-class problem, the only type of nearest neighbour that has to be avoided by the algorithm is false positive neighbours. This is much simpler compared to a multi-class problem. Neighbours to a query instance in this type of problem will belong to one of the class labels of a bag of instances and also could well be a false labelled neighbour of one of these classes. For Bayesian-Knn and Citation-Knn, which are two variants of a lazy learning approach to MIL, to handle multi-class problem, the majority vote method for appropriate neighbours has to be further adjusted to accommodate a multi-class label system. This should be feasible with an appropriate probabilistic model that can assign correct probability to neighbour being labelled one class over others.

Although it might be possible to extend Bayesian-Knn and Citation-Knn to solve multi-class problems, their inferiority over the Diverse Density and the multi-class Diverse Density method is that an interactivity feature cannot be added in much the same manner. The contrast between the two approaches is as follows. The nearest neighbour approach works backwards. Starting from a query bag of instances, the method works out to find the nearest neighbouring bag of instances from a passive set of example bags. Although searching in a backward fashion allows the algorithm not to worry about the order of influence of the target concepts, parameters such as the distance between two instances cannot be used as an index to suggest a specific instance in a new bag for further investigation. On the other hand, the Diverse Density and the multi-class Diverse Density methods work in a forwards fashion. Evidence is gathered incrementally. This allows Diverse Density or multi-class Diverse Density values to be used as a guideline for focusing on a specific instance. This instance is then likely to be the solution for the learning task, and the bag label of a suitable set of new bags of instances will confirm the hypothesis. As a result, the interactivity feature is possible with the multi-class Diverse Density method but is not in the case of the lazy learning approach to the multiple-instance learning problem.

7.3.2 MIL - Neural Network

With reference to Chapter2: section 2.2.3.2, the quest is to find the right representation of the neural network for a multiple-instance learning problem. The same quest is also applied to the problem with a multi-class label system. The first representation proposed by Ramon and De Raedt [Ramon and Raedt, 2000] for the two-class problem should be easily extended to the multi-class problem. Extra nodes, which will represent additional class labels, are added to an appropriate level. Again the disadvantage of this approach compared to the multi-class Diverse Density method is that the system does not have an appropriate index to help selecting an example to be trained interactively. Errors from the error function cannot be easily adapted as an index as well as the Diverse Density value can be.

The second representation, BP-MIP [Zhou and Zhang, 2002], used a different representation from the first system mentioned in the previous paragraph. In order to extend this representation to accommodate a multi-class labelling system, the activation function and error function must be adjusted. In the original, if the actual output is more than 0.5 then the instance is regarded as a positive instance, otherwise it is regarded as negative. With more classes, the threshold for the outputs should be divided and spread between all the classes. How the outputs are spread is also subject to the order of influence of the target concepts, which is considered as another unknown and has to be determined beforehand.

While the error function in the original BP-MIP was defined as follows: error is either equal to 0 if a bag containing this instance is positive and actual output is more than 0.5, or if a bag containing this instance is negative and actual output is less than 0.5. Otherwise the error is equal to half of the square of the difference between 0.5 and the actual output. This too should be adjusted. Again, even though the adjustment can be done to cope with a multi-class problem, this representation also suffers from not having the right kind of index to help integrate interactivity feature into the system.

7.3.3 MIL - Decision Tree and Decision Rule

With reference to Chapter2: section 2.2.3.3, the decision tree and decision rule were adapted by [Zucker and Chevaleyre, 2000] to solve two-class multiple-instance learning problems. For the decision tree approach, the idea was to generate two trees. One tree is for separating true positive instances from false positive instances. The other is for negative instances. Therefore in the case of a multi-class problem, more trees must be generated for additional classes. However the problem with this approach is that the order of influence of the underlying class cannot be decided upon among these classes, as each of the trees will only separate instances into two groups (i.e. true and false instances of a respective class of the tree).

For the decision rule, the object should be classified as positive if at least one of its instances satisfies the rules describing positive features and none of its instance satisfies the rules describing negative features. When applying the decision rule to multi-class problems, additional rules must be generated to describe other classes. Moreover the classification should be similar to the original structure but with an extra condition that none of the instances belongs to the same object can satisfy the rule describing other classes other than the target class. Similar to the decision tree, the order of influence cannot be identified unless the rules are adjusted to include the order of influence effect. Moreover, both the decision tree and the decision rule do not have any direct indicator that can be used to select new training examples so that more precise trees or rules can be generated with less number of possible examples (i.e. interactivity feature). Entropy used in the decision tree can only select a feature that is a better classifier than others but it cannot distinguish better examples (i.e. examples that will speed the learning) from the examples that do not give benefit in learning speed. In other words, both approaches can be viewed as passive learning and therefore not as good as interactive learning proposed here in this thesis.

Chapter 8

Conclusion

It has long been the goal of Artificial Intelligence to create Intelligent Systems. Recently cognitive science has suggested that the human mind has mental representations, which Artificial Intelligence has tried to mimic in order to achieve this goal. Concept learning is one of these such representations. Due to weaknesses in the approaches, both traditional Artificial Intelligence and behaviour-oriented Artificial Intelligence have failed to build an intelligent system based on the idea of concept learning of ambiguous objects. It was the aim of this thesis to create a concept learning system based in-between traditional AI and behaviour-oriented AI together with the introduction of the interactivity feature for enhanced learning. This aim has been fulfilled by viewing the problem as a multi-class multiple instance learning problem, which was first proposed here in this thesis. There already existed the Diverse Density method by [Maron, 1998] for two-class multiple instance learning problem, and it was possible to solve the concept learning problem by modifying this specific method to suit a more general multi-class multiple-instance learning problem. Proofing of the modified Diverse Density was done using artificial data as this allowed controlled rigorous testing, further testing was done using real life data that not only benefitted further investigation of features such as instance volume and interactive learning but also worked to provide examples of the application of the modified Diverse Density in real-world problems. The modified method was proven to be functional and efficient for both artificial and real life data.

The problem and its solution are reviewed in section 8.1 , in section 8.2 the contributions to the work are outlined, and section 8.3 points to future work.

8.1 Thesis Summary

The objective of this research was to create an intelligent system based in-between traditional AI and behaviour-oriented AI philosophies. The in-between philosophy suggests that the world is ambiguous and that a system gains its knowledge by learning from these ambiguous examples. The same philosophy further suggests that learning can be improved if a system is also allowed to play an active role in requesting examples to help it learn. In Chapter 1, it was proposed to call this in-between philosophy an interactive ambiguity-learning approach. This research selected the concept learning algorithm because a task such as learning a simple concept represents all of the important features within the interactive ambiguity-learning philosophy.

As a result, the method of learning ambiguous concepts was then the focus of this research. Concepts are ambiguous in the sense that the objects that represent a concept to be learned are ambiguous. It is assumed that these objects have multiple-entity descriptions. This assumption states that there are two cases where objects can be described as multiple entities. The first case is where each of these entities could be describing the object at one particular time but not at another time. The second case is where all of these entities could describe a specific object but only a few of these entities can be used to distinguish the object from any other objects. The multiple-instance learning framework has already been applied to the learning of such objects.

The multiple-instance learning framework was formalised as follows: each example presented to a learning algorithm or a learner is defined as a set (or bag) of instances. A bag is to be labelled positive if at least one instance in the bag is positive. Conversely, a bag will be labelled negative if all of the instances in the bag are negative. The goal is for the learner to accurately predict a label for any unseen bags by way of induction from a collection of labelled bags (examples) which are presented to it. However in the case of concept learning, there is a need to extend the multiple-instance learning framework to handle multi-class problems. This is because concepts are not simply binary: positive or negative, but of multiple classes. Extending the original two-class classification problem to a multi-class classification implies that the number of possible class labels can be more than two and the instances underlying different classes can co-exist in the same bag of instances, but that the label of the bag will belong to that of the more influential class. This follows the similar premis as that of the original framework in which positive instances can never exist in any of a negative bag of instances while negative instances can exist in a positive bag. More specifically the instances underlying a more influential class will never exist in a bag of a less influential class while instances underlying a less influential class can exist in a bag labelled as a more influential class.

As an example, if a bag of instances can be labelled as one of three classes (class A, class B, and class C) where class A is the most influential class followed by class B and C respectively, then the multi-class problem will be defined as set out below. A particular bag will be labelled class A if at least one instance in the bag underlies class A. Likewise, it will be labelled class B if at least one instance in the bag underlies class B and there is not any instance underlying class A in the bag. Finally, the bag will be labelled class C if at least one instance in the bag underlies class C and there is not any instance underlying class A or B in the bag.

The Diverse Density method, which was one of the methods developed to solve the original multiple-instance learning problem, was chosen to be modified to solve the proposed multi-class multiple-instance learning. The Diverse Density of a given concept is defined as a product of the probability that the given concept is the target concept given each bag of instances. Similar to this original definition, the multi-class Diverse Density of a given concept, being a target concept underlying a particular labelled class, is defined as a product of the probability that this given concept is the target concept for this particular class given each bag of instances. The task is to model the above probability so that multi-class Diverse Density value is at the highest at the target concept in the similar manner as suggested by the original Diverse Density method. While the original Diverse Density method considers only positive and negative instances, joint instances are integrated into the probability model in the multi-class Diverse Density method. Moreover, unlike the original Diverse Density method, the multi-class Diverse Density method has a further task to uncover the influence order of all the instances underlying class labels. This research has proposed to use a summation or an addition of multi-class Diverse Density calculated for each class as an indicator. The higher the addition the more influence the concept has over other concepts, hence the more influence the class underlying that concept has.

Therefore the task of concept learning within the interactive ambiguity-learning environment is reduced to the newly proposed multi-class problem of the multiple-instance learning framework. The thesis also proposed the modification of the Diverse Density method, which will be called the multi-class Diverse Density method, to handle such a problem. The next section reports on the finding of the multi-class Diverse Density method.

8.2 Contribution

The main contribution of this thesis is to create a concept learning system based on in-between philosophy and view the system as the proposed multi-class multiple-instance learning problem and use the modified Diverse Density method (i.e. the multi-class Diverse Density method) to find the solution. Some problem scenarios can be better defined as the multi-class problem whereas others can only be defined as the multi-class problem. It was found that the types of problems that the multi-class Diverse Density method is suitable for are as follows:

- Supervised learning problems with ambiguous examples
- Ambiguous examples that are better classified into multiple classes instead of the two-class classification
- The task of removing instances that make an example ambiguous

However at this stage, the method might be slow at solving learning problems with large feature space. Nonetheless the methods such as point-wise multi-class Diverse Density or interactivity feature will help speed up the learning process. Since multi-class Diverse Density is good at removing false instances, or in other words removing instances with low multi-class Diverse Density value, this method can be used as primary method for removing ambiguity in examples before they are passed on to more appropriate learning algorithms. This does not mean that the multi-class Diverse Density method cannot learn from such examples but that a certain learning algorithm might be more appropriate for such examples with specific nature. For example, once ambiguity has been removed using the multi-class Diverse Density method, a regression learning algorithm can then be applied more successfully.

The proposed multi-class Diverse Density method was tested on both artificial data and real-life problem scenarios: stock prediction tasks; assembly tasks; interactive document search. This is proven efficiency and robustness of the multi-class Diverse Density method. Apart from learning ambiguous concepts, the multi-class Diverse Density was developed to prove that learning can be improved if a system is also allowed to play an active role in requesting examples, therefore the interactivity feature was integrated into this method as a result. The interactivity feature, or the ability to request examples to be presented to the system, that could lead to faster learning, was integrated successfully into the multi-class Diverse Density method. The investigation has proved that as long as a diversity of examples is ensured (i.e. both the evidence supporting the suspecting instances and the counter-supporting evidence are supplied), the multi-class Diverse Density method can arrive at the conclusion much faster. The interactivity feature suggests that the multi-class Diverse Density method will be good at applications involving error recovery task. If what causes the success or the failure of an operation is ambiguous, the system can use an interactive approach to find the underlying causes.

8.3 Future Work

There are two areas that need to be investigated further. The first area is concerning the probability model for a given concept being a target concept for each individual class given bags of instances. The second area is concerning an appropriate introduction of features such as instance volume to the probability model. The model investigated here is a general probability model used to compute the value of multi-class Diverse Density, but a more suitable model could be assigned according to the learning application. For instance, in this research, instance volume is only included into the calculation of multi-class Diverse Density in the stock prediction task but not in the assembly task. Despite any adjustment to the model, it should still follow the same principle that the multi-class Diverse Density value should increase exponentially with supporting evidences and decrease exponentially with counter-supporting evidences so that the underlying concept can have an outstanding value (i.e. very high) compared to the non- target concept.

Bibliography

[Aha et al., 1991] Aha, D., Kibler, D., and Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, 6:37–66.

[Aha, 1997] Aha, D. W. (1997). Lazy learning. *Artificial Intelligence Review*, 11:325–337.

[Amar et al., 2001] Amar, R. A., Dooly, D. R., Goldman, S. A., and Zhang, Q. (2001). Multiple-instance learning of real-valued data. In *Proceedings of the 18th International Conference on Machine learning*.

[Andrews et al., 2002] Andrews, S., Hofmann, T., and Tsochantaridis, I. (2002). Multiple instance learning with generalized support vector machine. In *Proceedings of 18th National Conference on Artificial Intelligence and 14th Conference on Innovative Applications of Artificial Intelligence*, pages 943–944. AAAI Press.

[Auer, 1997] Auer, P. (1997). On learning from multi-instance examples: Empirical evaluation of a theoretical approach. In *Proceedings of the 14th International Conference on Machine learning*.

[Blum and Kalai, 1998] Blum, A. and Kalai, A. (1998). A note on learning from multiple-instance examples. *Machine Learning*, 30:23–29.

[Cheeseman et al., 1988] Cheeseman, P., Freeman, D., Kelly, J., Self, M., Stutz, J., and Taylor, W. (1988). Autoclass: A bayesian classification system. In *Proceedings of the 5th International Conference on Machine Learning*, pages 54–64, Ann Arbor, MI. Morgan Kaufmann.

[Cheeseman and Stutz, 1996] Cheeseman, P. and Stutz, J. (1996). Bayesian classification (AUTOCLASS): Theory and results. In Fayyad, U. M., Piatetsky-Shapiro, G., Smith, P., and Uthurussamy, R., editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180. AAAI/MIT Press.

[Chen et al., 2002] Chen, M. Y., Kiciman, E., Fratkin, E., Fox, A., and Brewer, E. (2002). Pinpoint: Problem determination in large, dynamic internet services. In *Proceedings of 2002 International Performance and Dependability Symposium*, Washington, DC.

[Chevaleyre and Zucker, 2001] Chevaleyre, Y. and Zucker, J. D. (2001). A framework for learning rules from multiple instance data. In *Proceedings of the 12th European Conference on Machine Learning*, pages 49–60.

[Dempster et al., 1997] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1997). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistics Society, Series B*, 39(1):1–38.

[Dietterich, 1997] Dietterich, T. G. (1997). Machine learning research: four current directions. *AI Magazine*, 18:97–136.

[Dietterich et al., 1997] Dietterich, T. G., Lathrop, R. H., and Lozano-Perez, T. (1997). Solving the multiple-instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71.

[Felzenszwalb and Huttenlocher, 1998] Felzenszwalb, P. and Huttenlocher, D. (1998). Image segmentation using local variation. In *Proceedings of Computer Vision and Pattern Recognition*.

[Fisher, 1987] Fisher, D. (1987). COBWEB: Knowledge acquisition via conceptual clustering. *Machine Learning*, 2:139–172.

[Goldman et al., 2001] Goldman, S. A., Kwek, S. K., and Scott, S. D. (2001). Agnostic learning of geometric patterns. *Journal of Computer and System Sciences*, 6(1):123–151.

[Holland, 1986] Holland, J. H. (1986). Escaping brittleness: the possibilities of general purpose learning algorithms applied to parallel rule-based systems. In Michalski, R. S., Carbonell, J. G., and Mitchell, T. M., editors, *Machine learning: An Artificial Intelligence Approach*, volume II, pages 593–623. Morgan Kauffmann, Los Altos, CA.

[Hu et al., 2001] Hu, W.-C., Chen, Y., Schmalz, M. S., and Ritter, G. X. (2001). An overview of the World Wide Web search technologies. In *Proceeding of the 5th World Multi-Conference on Sytem, Cybernetics and informatics, SCI2001*.

[Huang et al., 2003] Huang, X., Chen, S. C., and Shyu, M. L. (2003). An open multiple instance learning framework and its application in drug activity prediction problems. In *Proceedings of the 3rd IEEE Symposium on BioInformatics and BioEngineering*, pages 53–59.

[Hunt, 1962] Hunt, E. B. (1962). *Concept Formation: An Information Processing Problem*. Wiley, New York.

[Iba, 1991] Iba, W. (1991). Learning to classify observed motor behaviour. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 732–738, Sydney. Morgan Kaufmann.

[Kibler and Aha, 1987] Kibler, D. and Aha, D. W. (1987). Learning representative exemplar of concepts: An initial case study. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 24–30, Irvine, CA. Morgan Kaufmann.

[Knight and Sen, 1995] Knight, L. and Sen, S. (1995). PLEASE: A prototype learning system using genetic algorithms. In *Proceeding of the Sixth International Conference on Genetic Algorithms*, pages 429–435. Morgan Kaufmann.

[Kolodner, 1983] Kolodner, J. L. (1983). Reconstructive memory: A computer model. *Cognitive Science*, 7:281–328.

[LakshmiRatan et al., 1999] LakshmiRatan, A., Maron, O., Grimson, W. E. L., and Lozano-Perez, T. (1999). A framework for learning query concepts in image classification. *Computer Vision and Pattern Recognition*, pages 423–429.

[Lebowitz, 1987] Lebowitz, M. (1987). Experiments with incremental concept formation: UNIMEM. *Machine Learning*, 2(2):103–138.

[Long and Tan, 1996] Long, P. M. and Tan, L. (1996). PAC-learning axis alligned rectangles with respect to product distributions from multiple-instance examples. In *Proceedings of the 9th Annual Conference on Computational Learning Theory*, pages 228–234.

[Maron, 1998] Maron, O. (1998). *Learning from ambiguity*. PhD thesis, Department Of Electrical Engineering and Computer Science, MIT.

[Maron and LakshmiRatan, 1998] Maron, O. and LakshmiRatan, A. (1998). Multiple-instance learning for natural scene classification. In *Proceeding of the 15th International Conference on Machine Learning*, pages 341–349, Madison, WI.

[Maron and Lozano-Perez, 1998] Maron, O. and Lozano-Perez, T. (1998). A framework for multiple-instance learning. In Jordan, M. I., Kearns, M. J., and Solla, S. A., editors, *Advances in Neural Information Processing Systems*, number 10, pages 570–576. MIT Press, Cambridge, MA.

[Martin and Billman, 1994] Martin, J. D. and Billman, D. O. (1994). Acquiring and combining overlapping concepts. *Machine Learning*, 16:121–155.

[McGovern and Barto, 2001] McGovern, A. and Barto, A. G. (2001). Automatic discovery of subgoals in reinforcement learning using diverse density. In *Proceedings of 18th International Conference on Machine Learning*, pages 361–368. Morgan Kaufmann.

[McKusick and Langley, 1991] McKusick, K. and Langley, P. (1991). Constrains on tree structure in concept formation. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 810–816, Sydney. Morgan Kaufmann.

[Michalski and Stepp, 1983] Michalski, R. and Stepp, R. E. (1983). Learning from observation: conceptual clustering. In Michalski, R., Carbonell, J., , and Mitchell, T., editors, *Machine Learning: An Artificial Intelligence Approach*, pages 331–364. Tioga Press, Palo Alto, CA.

[Michalski, 1969] Michalski, R. S. (1969). On the quasi-minimal solution of the general covering problem. In *Proceeding of the 5th International Symposium on Information Processing (FCIP 69)*, pages 125–128, Bled, Yugoslavia.

[Michalski, 1998] Michalski, R. S. (1998). The AQ18 symbolic learning and data mining environment: Theory and methodology. Reports of the machine learning and inference laboratory, George Manson University.

[Mitchell, 1978] Mitchell, T. M. (1978). *Version Spaces: An approach to concept learning*. PhD thesis, Stanford University. CS-78-711, HPP-79-2.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

[Mitchell et al., 1986] Mitchell, T. M., Keller, R. M., and Kedar-Cabell, S. T. (1986). Explanation-based generalisation: A unify view. *Machine Learning*, 1:1–33.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

[Quinlan, 1986] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1:81–106.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

[Quinlan, 1999] Quinlan, J. R. (1999). C5.0 decision tree software. http://www.rulequest.com/.

[Ramon and Raedt, 2000] Ramon, J. L. and Raedt, L. D. (2000). Multi instance neural network. In *Proceedings of IMCL-2000 Workshop on Attribute-Value and Relational Learning*.

[Ray and Page, 2001] Ray, S. and Page, D. (2001). Multiple instance regression. In *Proceedings of the 18th International Conference on Machine learning*, pages 425–432.

[Reed, 1972] Reed, S. K. (1972). Pattern recognition and categorisation. *Cognitive Psychology*, 3:383–407.

[Ruffo, 2000] Ruffo, G. (2000). *Learning single and multiple instance decision trees for computer security applications*. PhD thesis, Department of Computer Science, University of Turin, Italy.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E. and McClelland, J. L., editors, *Parallel Distributed Processing: explorations in the microstructure of cognition*, volume 1, pages 318–362. MIT Press, Cambridge, MA.

[Scott et al., 2003] Scott, S. D., Zhang, J., and Brown, J. (2003). On generalized multiple-instance learning. Technical report unl-cse-2003-5, University of Nebraska.

[Smith and Medin, 1981] Smith, E. E. and Medin, D. L. (1981). *Categories and Concepts*. Harvard University Press, Cambridge, MA.

[Stanford University, 2004] Stanford University (2004). Stanford encyclopedia of philosophy. WWW Document url: http://galeb.etf.bg.ac.yu/ igi/.

[Stepp, 1984] Stepp, R. E. (1984). *Conjunctive Conceptual Clustering: A Methodology and Experimentation*. PhD thesis, Department of Computer Science, University of Illinois.

[Thompson and Langley, 1991] Thompson, K. and Langley, P. (1991). Concept formation in structured domains. In Fisher, D., Pazzani, M., and Langley, P., editors, *Concept Formation: Knowledge and Experience in Unsupervised Learning*. Morgan Kaufmann.

[U.S. Securities and Exchange Commission (SEC), 2000] U.S. Securities and Exchange Commission (SEC) (2000). Final rule: Selective disclosure and insider trading. WWW Document url: http://www.sec.gov/rules/final/33-7881.htm.

[Valiant, 1984] Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

[Vapnik, 1998] Vapnik, V. (1998). *Statistical Learning Theory*. Wiley.

[Wang and Zucker, 2000] Wang, J. and Zucker, J. D. (2000). Solving the multiple-instance problem: A lazy learning approach. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1119–1125, San Francisco, CA. Morgan Kaufmann.

[Weidmann et al., 2003] Weidmann, N., Frank, E., and Pfahringer, B. (2003). A two-level learning method for generalized multi-instance problems. In *Proceedings of the 14th European Conference on Machine Learning, Cavtat-Dubrovnik,Croatia*, pages 468–479. Springer.

[Wittgenstein, 1965] Wittgenstein, L. (1965). *Philosophical Investigations*. The Macmillan Company, New York.

[Xu and Fu, 2000] Xu, Y. Y. and Fu, H. C. (2000). Image classification and indexing by EM based multiple-instance learning. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*.

[Xu et al., 2000] Xu, Y. Y., Fu, H. C., and Pao, H. T. A. (2000). WWW-based intelligent multimedia information query and retrieve system. In *Proceeding of IEEE International Conference on Multimedia and Expo, ICME,* volume 2, pages 731–734.

[Yang and Lozano-Perez, 2000] Yang, C. and Lozano-Perez, T. (2000). Image database retrieval with multiple-instance learning techniques. In *Proceedings of International Conference on Data Engineering*, pages 233–243.

[Zhang et al., 2002] Zhang, Q., Goldman, S., Yu, W., and Fritts, J. (2002). Content-based image retrieval using multiple-instance learning. In *The Nineteenth International Conference on Machine Learning*, Sydney, Australia.

[Zhang and Goldman, 2001] Zhang, Q. and Goldman, S. A. (2001). EM-DD: An improved multiple-instance learning technique. Technical report wucs-01-17, Washington University, St. Louis, USA.

[Zhou and Zhang, 2002] Zhou, Z. H. and Zhang, M. L. (2002). Neural networks for multi-instance learning. In *roceedings of the International Conference on Intelligent Information Technology*, pages 455–459, Beijing, China.

[Zhou and Zhang, 2003] Zhou, Z.-H. and Zhang, M.-L. (2003). Ensembles of multi-instance learners. In *Proceedings of the 14th European Conference on Machine Learning, Cavtat-Dubrovnik,Croatia*, pages 492–502. Springer.

[Zucker and Chevaleyre, 2000] Zucker, J. D. and Chevaleyre, Y. (2000). Solving multiple-instance and multiple-part learninglearning problems with decision trees and decision rules. Internal report, University Paris VI.