Random forest for dissimilarity based multi-view learning : application to radiomics

Hongliu Cao

Normandie Université

THESE

Pour obtenir le diplôme de doctorat

Spécialité INFORMATIQUE

Préparée au sein de l'Université de Rouen Normandie
En partenariat international avec l'École de Technologie Supérieure, Université du Québec, Canada

Random Forest For Dissimilarity Based Multi-view Learning : Application To Radiomics

Présentée et soutenue par
Hongliu CAO

Thèse soutenue publiquement le 02 décembre 2019 devant le jury composé de		
Mme Bernadette DORIZZI	Professeur des Universités Télécom SudParis	Rapporteuse
M. Pierre GEURTS	Professeur d'Université Université de Liège, Belgique	Rapporteur
M. Frédéric PRECIOSO	Professeur des Universités Université Côte d'Azur	Examinateur
M. Alessandro KOERICH	Professeur École de Technologie Supérieure, Montréal	Examinateur
M. Lionel PREVOST	Professeur des Universités ESIEA Paris	Examinateur
M. Laurent HEUTTE	Professeur des Universités Université de Rouen Normandie	Directeur de thèse
M. Robert SABOURIN	Professeur École de Technologie Supérieure, Montréal	Codirecteur de thèse
M. Simon BERNARD	Maître de Conférence Université de Rouen Normandie	Co-encadrant de thèse

Thèse dirigée par Prof. Laurent HEUTTE, Laboratoire LITIS, Université de Rouen Normandie et Prof. Robert SABOURIN, Laboratoire LIVIA, École de Technologie Supérieure

UNIVERSITÉ DE ROUEN NORMANDIE ÉTS ed MIIS litis LIVIA

Résumé

Les travaux de cette thèse ont été initiés par des problèmes d'apprentissage de données radiomiques. La Radiomique est une discipline médicale qui vise l'analyse à grande échelle de données issues d'imageries médicales traditionnelles, pour aider au diagnostique et au traitement des cancers. L'hypothèse principale de cette discipline est qu'en extrayant une grande quantité d'informations des images, on peut caractériser de bien meilleure façon que l'oeil humain les spécificités de cette pathologie. Pour y parvenir, les données radiomiques sont généralement constituées de plusieurs types d'images et/ou de plusieurs types de caractéristiques (images, cliniques, génomiques).

Cette thèse aborde ce problème sous l'angle de l'apprentissage automatique et a pour objectif de proposer une solution générique, adaptée à tous problèmes d'apprentissage du même type. Nous identifions ainsi en Radiomique deux problématiques d'apprentissage: (i) l'apprentissage de données en grande dimension et avec peu d'instances (high dimension, low sample size, a.k.a. HDLSS) et (ii) l'apprentissage multi-vues. Les solutions proposées dans ce manuscrit exploitent des représentations de dissimilarités obtenues à l'aide des Forêts Aléatoires. L'utilisation d'une représentation par dissimilarité permet de contourner les difficultés inhérentes à l'apprentissage en grande dimension et facilite l'analyse conjointe des descriptions multiples (les vues).

Les contributions de cette thèse portent sur l'utilisation de la mesure de dissimilarité embarquée dans les méthodes de Forêts Aléatoires pour l'apprentissage multi-vue de données HDLSS. En particulier, nous présentons trois résultats: (i) la démonstration et l'analyse de l'efficacité de cette mesure pour l'apprentissage multi-vue de données HDLSS; (ii) une nouvelle méthode pour mesurer les dissimilarités à partir de Forêts Aléatoires, plus adaptée à ce type de problème d'apprentissage; et (iii) une nouvelle façon d'exploiter l'hétérogénèité des vues, à l'aide d'un mécanisme de combinaison dynamique. Ces résultats ont été obtenus sur des données radiomiques mais aussi sur des problèmes multi-vue classiques.

Mots-clés: Espace de dissimilarité, forêt aléatoire, apprentissage multi-vue, dimension élevée, taille réduite de l'échantillon, apprentissage de dissimilarité, sélection dynamique

Abstract

The work of this thesis was initiated by a Radiomic learning problem. Radiomics is a medical discipline that aims at the large-scale analysis of data from traditional medical imaging to assist in the diagnosis and treatment of cancer. The main hypothesis of this discipline is that by extracting a large amount of information from the images, we can characterize the specificities of this pathology in a much better way than the human eye. To achieve this, Radiomics data are generally based on several types of images and/or several types of features (from images, clinical, genomic).

This thesis approaches this problem from the perspective of Machine Learning (ML) and aims to propose a generic solution, adapted to any similar learning problem. To do this, we identify two types of ML problems behind Radiomics: (i) learning from high dimension, low sample size (HDLSS) and (ii) multi-view learning. The solutions proposed in this manuscript exploit dissimilarity representations obtained using the Random Forest method. The use of dissimilarity representations makes it possible to overcome the well-known difficulties of learning high dimensional data, and to facilitate the joint analysis of the multiple descriptions, i.e. the views.

The contributions of this thesis focus on the use of the dissimilarity measurement embedded in the Random Forest method for HDLSS multi-view learning. In particular, we present three main results: (i) the demonstration and analysis of the effectiveness of this measure for HDLSS multi-view learning; (ii) a new method for measuring dissimilarities from Random Forests, better adapted to this type of learning problem; and (iii) a new way to exploit the heterogeneity of views, using a dynamic combination mechanism. These results have been obtained on radiomic data but also on classical multi-view learning problems.

Keywords: Dissimilarity space, random forest, multi-view learning, high dimension, low sample size, dissimilarity learning, dynamic selection

Acknowledgements

I would like to firstly thank my advisors Laurent Heutte and Simon Bernard from Laboratoire d'Informatique, de Traitement de l'Information et des Systems (LITIS) in France. They have given me a great deal of independence pursuing my ideas as well as lots of encouragement. At the meantime, they have given me a lot of great advices on doing rigorous science research and writing rigorous research articles. They have been good role models for me in the field of scientific research. I am sincerely thankful for all their efforts during my thesis.

I also would like to thank my supervisor Robert Sabourin from Laboratoire d'imagerie, de vision et d'intelligence artificielle (LIVIA) in Canada. Working with Robert makes me a more efficient person. His way of thinking and analyzing inspires me a lot during my thesis. He also helped me a lot on understanding quebecois french, which made my life easier in Montreal.

My colleagues in LITIS France and LIVIA Canada have helped a lot too. I learned a lot from Group Reading in LIVIA, where Phd students can share their understanding of different research fields. I would like to especially mention Rafael Menelau, Fabien Orlando, Lucas Shorten, Jonathan De Matos, Alexandre Reeberg and Fabio Alexandre Spanhol for being nice colleagues and also good friends.

Last but not least I would like to thank my parents and my sister for their caring and supporting. Nothing could express how grateful I am to them for everything. This thesis would not exist without their support.

Contents

General Introduction

In many real-world Pattern Recognition problems, the data available are complex, in the sense that they cannot be described by a single numerical representation. For example, they can come from multiple sources as in video surveillance problems ([119]), where multiple cameras are used to captured the same scene, from different angles. This is necessary to avoid blind spot or occlusions of object or persons. The images captured by these multiple cameras are expected to complement each other to have a more accurate and complete representation of the scene.

Another widespread situation is problems for which the raw data are described via multiple feature extractors, with the goal to better capture their complexity. In certain image recognition task for example, an image is described by multiple feature representations, e.g. colour descriptors, shape descriptors, texture descriptors, etc. Each of these descriptor families is used to capture a particular characteristic of the images, and using them all together is expected to help to better address the complexity of the recognition task.

The starting point of this work is a medical imaging problem of this type, i.e. for which the data are derived from several image modalities and/or several feature extractors. This problem is computer-aided cancer diagnosis/treatment based on Radiomic features, a key application of the DAISI project, co-financed by the European Union with the European Regional Development Fund (ERDF) and by the Normandy Region. As part of the DAISI project, this thesis has been initiated around the Radiomics application in collaboration with experts and doctors from the Henri Becquerel center, one of the French Comprehensive Cancer Centers (FCCCs).

The objective of Radiomics is to describe radio-graphic medical images with a large number of heterogeneous image features, in the hope of discovering characteristics of the disease that cannot be discerned with the naked eye. The main hypothesis is that exploiting numerous heterogeneous features may be useful for predicting prognosis and therapeutic response for various conditions, thus providing valuable information for personalized therapy.

This thesis work addresses the Radiomics problem from a Machine Learning (ML) point of view. In doing so, Radiomics can be considered a Multi-View Learning (MVL) problem, in which a 'view' is the set of features obtained

from one modality and/or one feature extractor. Each instance of the problem is therefore described by multiple views, and the goal is to learn a predictive model by taking into account the complementarities of all views. MVL tasks usually needs dedicated methods because it exhibits specific ML difficulties. The first difficulty often relates to the overall dimension of the problem. Considering all the features extracted in all views, most MVL problems are (very) high dimensional learning problems. This will largely narrow down the choice of machine learning methods. For example, the popular machine learning method Support Vector Machine (SVM) can suffer from data piling problem on very high dimensional features ([163]). The high feature dimension usually requires a large number of labelled training instances. However, it is impossible to collect a lot of training instances in various fields, especially in the medical field. Due to the problem of data collection and sharing, the number of patients is always very small compared to the high feature dimension, which usually leads to the High Dimension Low Sample Size (HDLSS) problem. Due to the small sample size, the instances are very sparse in the feature space, which also makes it harder to deal with noise or outliers. When multi-view problems are coupled with HDLSS problems, it becomes more difficult because many multi-view techniques are unable to handle the HDLSS problems while HDLSS solutions do not take multi-view information into consideration.

Radiomics is a typical example of HDLSS multi-view problem. Each view of Radiomic problem can easily have hundreds or thousands of features while the sample size is usually smaller than one hundred. The state-of-the-art works in Radiomics, however, concatenate the multi-view information into one view and use feature selection techniques to decrease the feature dimension. These feature selection methods may lose some important information, especially when only few features are selected. Furthermore, these methods also overlook the multi-view challenge, which deviates from the original intention of extracting heterogeneous information. Features from different views generally have distinct descriptions of the same instance and can provide additional information for the learning task, making it necessary to use multi-view alternatives to take benefit of the complementary information.

In the first part of the thesis, we aim at finding a solution to deal with both HDLSS and multi-view challenges. A literature review on multi-view learning methods with a focus on the HDLSS problem is given in Chapter 1. One straightforward solution is to find a low dimensional intermediate representation that can be comparable among all the views to tackle the HDLSS challenge. Then, these intermediate representations can be merged into a joint representation to tackle the multi-view challenge. In this work, we propose to use dissimilarity representations as the solution because it can naturally reduce the feature dimension to a lower space. The dissimilarity values are also directly comparable from one view to another and easy to integrate.

As the only information shared by the different views is the class information, we propose to use a dissimilarity measure that can take this information into account. Among different supervised dissimilarity learning methods, we propose to use the Random Forest Dissimilarity (RFD). Random Forest (RF) is a successful machine learning method due to its interpretability and its accuracy in classification and regression tasks. RF can also naturally handle HDLSS problems thanks to the implicit feature selection mechanism as well as the bootstrap procedure ([103]). RF also embed a dissimilarity measure, the RFD measure, which reflects both feature and class dissimilarities. A first solution of RFD based intermediate integration is proposed in Chapter 2 as the solution for HDLSS multi-view problem: the features from each view are firstly projected into a dissimilarity space built with RFD, then these dissimilarity spaces are merged to form a joint dissimilarity space, used as the new representation for learning. The proposed solution is compared to multiple state-of-the-art Radiomic and multi-view solutions during the experiments on real world datasets. We show that this proposed approach is accurate and competitive on several real-world HDLSS multi-view problems.

In the second part of this thesis, we mainly focus on deepening this first solution and make it more efficient for the HDLSS multi-view problem. We identify two ways to improve the proposed method in Chapter 2: one in the way the dissimilarity is computed from the RF and the other in the way the view-specific dissimilarity representations are merged. For the first one, we propose a finer and a more accurate dissimilarity measure based on Random Forests, by better exploiting the tree structures. Each tree in a Random Forest

estimates the dissimilarity between two instances with a binary value, which may not be precise enough. Furthermore, the RFD estimate is a simple average over these binary values, while different trees may contribute differently to the final dissimilarity value. To overcome these limitations, our proposal is based on the evaluation of each terminal node confidence so that we can tell which tree to trust for each test instance. Based on this proposal, we also propose another more refined confidence measure using instance hardness to make the dissimilarity measure more adaptive to the HDLSS problem.

The second improvement is to propose a more adaptive dissimilarity matrix combination method to take better advantage of the multi-view information. For the experiments in Chapter 2 and Chapter 3, the joint dissimilarity representation is calculated by averaging over all the dissimilarity representations from each view. With averaging, different views are assumed to have the same importance for the classification task. However, for multi-view problem, different views usually offer very heterogeneous information and may have different importance for the given task. To estimate this importance, two methods based on static weighting and dynamic view selection respectively are proposed in Chapter 4. We firstly propose to use the Out Of Bag (OOB) accuracy of the Random Forest classifier used to built the dissimilarity representation to estimate the importance of each view. The intuition is that if a Random Forest has better generalization performance, the corresponding dissimilarity measure also has better quality. Secondly, a dynamic view selection method is proposed to select different view combinations for different test instances with the intuition that the information provided from one view for one test instance may not be as useful for another test instance.

Finally, we summarize our contributions in the last chapter, along with the future works we are interested in for both short and long terms.

Chapter 1

HDLSS Multi-view learning

Contents

1.1 Introduction

During the last decades, the field of computer vision and machine learning has made a lot of progress in both hardware and software. To tackle more complex problems, many studies use a large number of features as well as training instances. Nowadays, it is common to have several heterogeneous descriptions of the same set of instances. These multiple descriptions can come from "hardware" such as several complementary modalities or from "software" such as heterogeneous feature extractors for the same raw data. In machine learning, these datasets are called multi-view problem with "view" being the name of each type of information.

(a) Optical data (b) Range data

FIGURE 1.1: In the scenario of pedestrian detection, (a) optical data can provide useful RGB information while (b) range data can provide useful distance information. Data from different sensors can provide complementary information to improve the detection performance. This figure is extracted from [186].

From the perspective of hardware, plenty of new sensors or devices are invented to collect more complementary data. For example, the Internet of Things (IoT) has attracted a lot of attention in many countries by collecting

data using numerous sensors. It is also very common to have data from different sensors to describe the same object in self driving car scenarios. In [186], the authors use data from camera (Figure 1.1a) and lidar (Figure1.1b) to have a better pedestrian detection system. Data from different sensors usually can provide complementary information (color and depth information in the example of [186]) to improve the accuracy of machine learning models.

From the perspective of software, a large amount of new algorithms have been proposed for both information extraction and data analysis. To make the process of data analysis more efficient, many different feature extractors have been proposed to better represent the characteristics of instances. Multiple feature extractors are usually used together to represent the data heterogeneity for complex problems. For example in Figure 1.2, the AWA dataset used in [143] for natural scene image classification, is build with six different groups of traditional features including RGB color histograms ([143]), Scale-Invariant Feature Transform (SIFT [158]), rgSIFT ([229]), Pyramid Histogram of Oriented Gradients (PHOG [31]), Speeded Up Robust Features (SURF [22]) and local self-similarity histograms ([207]), which results in a total of 10940 features.

There are also a lot of "genuine" multi-view problems. For example, social media data from Facebook or Instagram contain image data along with textual data such as hashtags; videos are made up of visual data such as images and texts (subtitles) as well as audio data; news data can always be found in multiple different languages, etc. These multiple views generally convey additional information, and successfully combining them often makes it possible to obtain better overall performance than treating them individually.

In real world applications, the feature dimension is becoming higher and higher in each view, which makes the feature dimension extremely high when all multi-view features are concatenated. For example in [145], a total of 1403 handcrafted features and 98304 deep features from magnetic resonance imaging (MRI) are extracted. Because of the curse of dimensionality, a lot of machine learning methods will suffer from over-fitting problem ([163, 256]). To avoid this problem, a large amount of training instances are required, especially in multi-view learning when extra validation instances are often needed

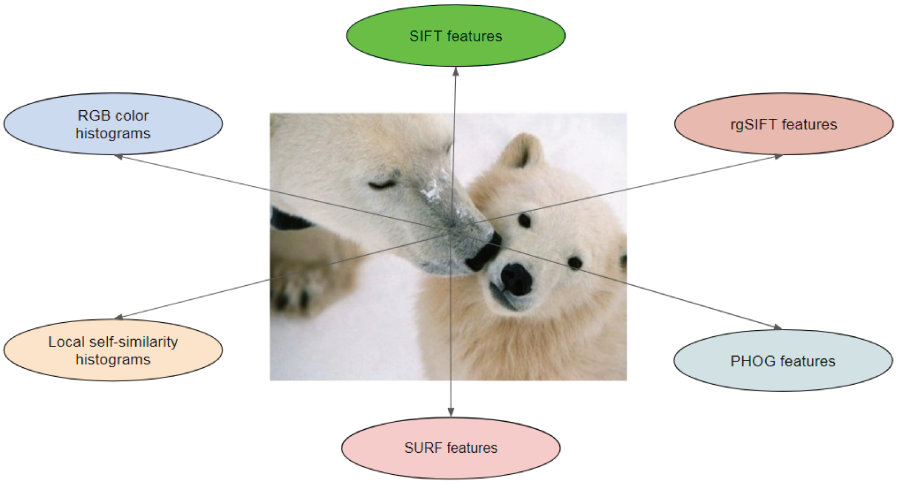


FIGURE 1.2: Six different groups of features including RGB color histograms, SIFT, rgSIFT, PHOG, SURF and local self-similarity histograms are extracted from each individual image to constitute the AWA dataset ([143]).

for the view combination. However, it is very expensive to collect a lot of training instances in many fields, which can easily make the sample size much smaller than the feature size. When the high feature dimension problem is combined with low sample size, it will lead to the High Dimension Low Sample Size (HDLSS) problem.

For multi-view problems, HDLSS may happen in the concatenated feature space as well as in each individual view. It becomes more challenging when multi-view problem meets HDLSS problem, because a lot of multi-view learning methods can not deal with HDLSS problem while HDLSS solutions do not take multi-view into consideration. These two difficulties, when combined, are referred to as HDLSS multi-view learning problems. The work of this thesis was motivated by a real problem named Radiomics with these characteristics, on which we will give more details in section 1.2. Then in section 1.3, a literature review of HDLSS multi-view learning is given. In section 1.4, the most appropriate solution for HDLSS multi-view problems is concluded and the contributions of this thesis are listed.

1.2 The example of Radiomics

In this section, the HDLSS multi-view problem is introduced and analyzed in detail through the medical example of Radiomics. Although this work is not restricted to this particular application, it is important here to present in detail what Radiomics is and why it is a typical HDLSS multi-view learning problem. The main mission of the thesis is to propose efficient new solutions for the HDLSS multi-view problems such as Radiomics.

1.2.1 Background and workflow

Background of Radiomics

One of the main difficulties in the diagnosis and treatment of cancer rises from the fact that tumors can show very heterogeneous profiles. This phenomenon is called the tumour heterogeneity and may occur for both between tumours (inter-tumour heterogeneity) and within tumours (intra-tumour heterogeneity). The tumor heterogeneity makes the diagnosis and treatment of cancer more difficult.

The usual process of cancer detection is from certain signs and symptoms to the further investigation by medical imaging and at last confirmed by biopsy ([3]). However, with the improvement of medical imaging technology, more and more attention has been paid on the data collected from medical images such as computed tomography (CT), positron emission tomography (PET) or magnetic resonance imaging (MRI) in medical research during the last two decades. Compared to the traditional procedure, medical imaging has the advantage of being easy to perform, low cost, and non-invasive ([64]). One of the most important reason which makes medical imaging popular in diagnosis and treatment of cancer is that tumor phenotype characteristics can be visualized. One typical example can be found in Figure 1.3: representative CT images of lung cancer are shown on the left and the corresponding 3D visualizations are shown on the right. It can be seen that strong phenotypic heterogeneity can be visualized ([3]): some tumors are smaller, some are bigger; some are round shaped while others are more spiky.

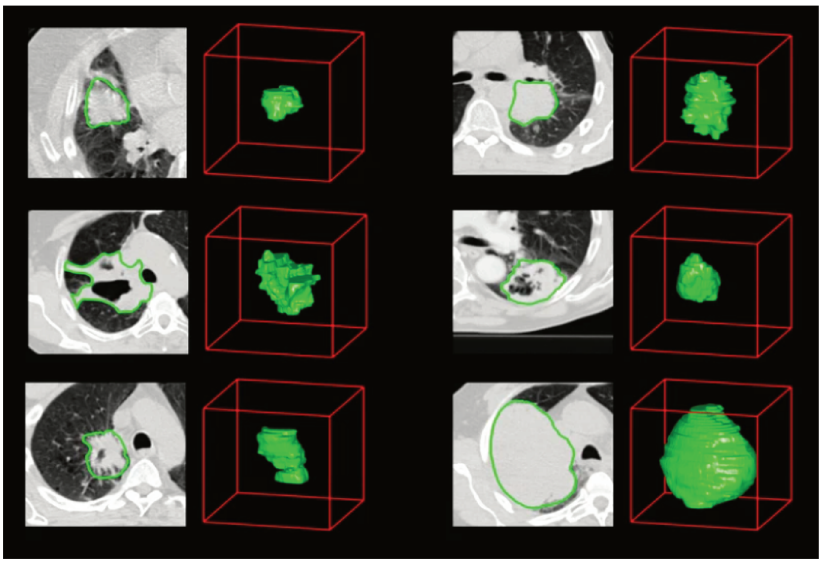


FIGURE 1.3: Visualization of tumor heterogeneity from medical imaging technology: example of CT images of lung cancer, CT scans are shown on the left and the corresponding 3D visualization of tumor is shown to the right. This figure is extracted from [3].

Radiomics was originally introduced in 2012 mainly for the diagnosis, prognosis and prediction of cancer. It is easy to perform, low cost and non-invasive as the information is collected from medical images such as CT or MRI other than biopsy ([142]). The core principle of Radiomics is to extract a very large number of features from multiple medical imaging modalities, in order to better capture the tumor heterogeneity and improve the diagnosis and treatment for cancer, but it has also been extended to many other medical studies where a disease can be tomographically imaged ([170]).

Radiomics has aroused great interest over the past few years ([92]). Compared to the current qualitative analysis in radiology, Radiomics can provide a quantitative analysis including much more useful information to make optimal treatment decisions and make cancer treatment more effective and less expensive ([136]). Many studies focus on the prediction for survival or the prediction for the response of patients to the treatment. A lot of classification tasks like classifying between patients with cancer and without cancer

have also been done. Furthermore, the information provided by Radiomics is thought nowadays to be complementary to clinical, pathological, and genomic information ([146, 234]).

Workflow of Radiomics

The process of Radiomics mainly contains three steps: data acquisition and segmentation , feature extraction and data analysis. One typical example is shown in Figure 1.4:

- From step 1) to 3) the data are collected from different medical imaging devices and the regions of interest (ROI) are segmented. Collecting data is quite hard in medical fields due to different data acquisition protocols, data sharing policies, data privacy, etc. In many Radiomics studies, researchers use no more than 50 patients in their dataset ([18, 43, 48, 61, 87]). In [43], they have only 13 prostate cancer patients for example.
- Then from step 4) to 6), different feature extraction techniques are used to extract multiple heterogeneous feature groups that can be complementary to each other. Due to the limited number of cancer patients, it is common to have Radiomic datasets with many features but very small number of training instances in comparison. Before the data analysis, Radiomic features can also be combined with other features such as genomics, clinics or protein features in practice, which makes the problem "more multi-view" with both multiple data sources and feature extractors, leading to a even higher feature dimension at the meantime.
- Finally, machine learning methods are used to build predictive, diagnostic or prognostic models to help realize the personalized treatment. Due to the large amount of features, it is common for traditional machine learning methods to suffer from the curse of dimensionality. Hence, feature selection methods are usually used before machine learning model. The most used machine learning methods in Radiomics include Random Forest (RF), support vector machine (SVM) and neural networks ([84, 125, 211, 242]).

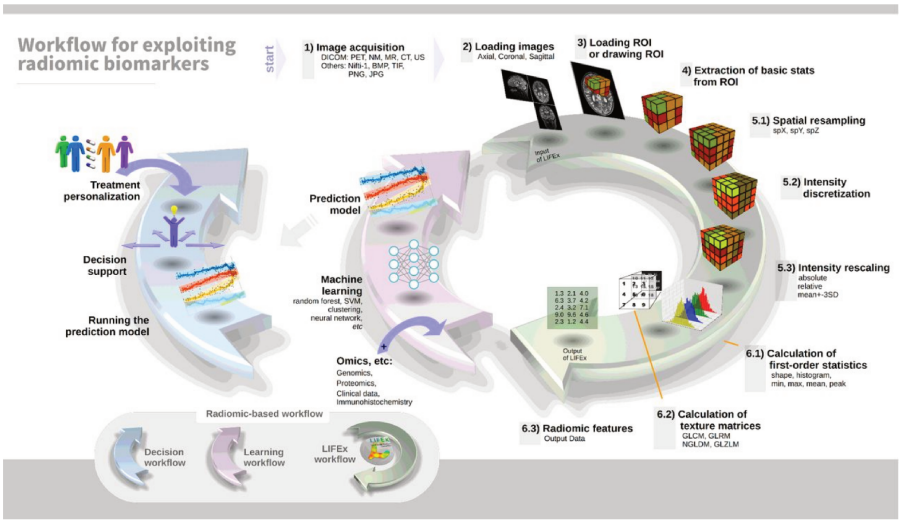


FIGURE 1.4: Example of workflow in Radiomics: it can be divided into three parts: 1. Data collection and segmentation from step 1) to 3); 2. Feature extraction from step 4) to 6); 3. Data analysis. This figure is extracted from [172]

In the following sections, we will review different features used in Radiomic works as well as the machine learning methods used in the Radiomic literature.

1.2.2 Radiomic features

Handcrafted feature extraction

Most of the publications in Radiomics report the use of handcrafted image features because of their interpretability even though they may cause an inevitable bias. There is no standard way to extract features from medical images. For different imaging modalities (CT, PET, MRI, etc.), features may be different due to different modality characteristics. Different researchers also have their own preference for extracting features too.

Even though the choices of feature extractors may be different across Radiomics studies, there are two aspects in common. The first one is that multiple heterogeneous feature extractors are often used to better represent the tumor heterogeneity. Aerts et al have used four feature groups for lung and head-and-neck cancer captured by CT images in [3] (shown in Figure 1.5), including features calculated with first-order statistics from the histogram, shape features, texture features and wavelet features. Similar feature extraction method can also be found in [173, 174] (CT scans for lung and head-and-neck cancer), [65, 248] (CT scans for non-small-cell lung carcinoma), [87] (MRI scans for head-and-neck cancer) and [237] (DCE-MRI for breast cancer). However, for Radiomics in brain tumor, three different feature extractors are used including local binary patterns (LBP), histogram of oriented gradients (HOG) and SIFT features ([259]). Despite of the different feature extractors for different cancers, there are at least two or three feature groups extracted in Radiomic works. These features are also often combined with additional information, like clinical and genomic information for example ([3]).

The second aspect in common for handcrafted Radiomic features is the "large quantity of features". However, "large quantity" is a qualitative expression and there is no study that defines the minimum number of features to be large. In the workflow of Radiomics, we have pointed out that the sample size of Radiomic problem is usually very small (fewer than one hundred patients), while the number of features is usually at least 4 or 5 times above the number of learning instances ([3, 64, 173]). Hence, the large number of features generally means that the feature size is much bigger than the sample size in Radiomics. For example, the Radiomic dataset used in [258] is made up of 84 patients and 6746 features.

Automatic feature learning

As manually extracted features would contain inevitable bias, researchers try to develop some automatic feature extraction methods. Deep learning is the most used, especially convolutional neural network (CNN). The advantage of using CNN is that given the region of interest as input, and an objective as output (e.g. classification), the features can be learned automatically.

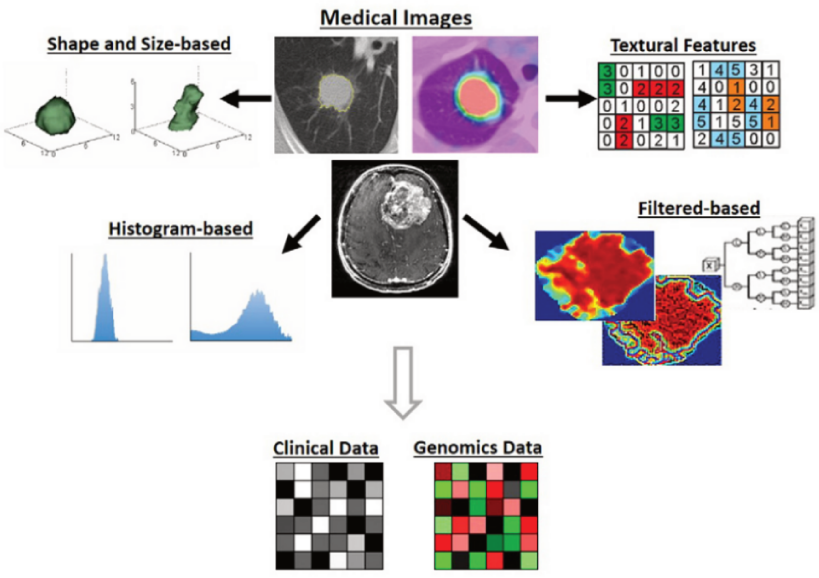
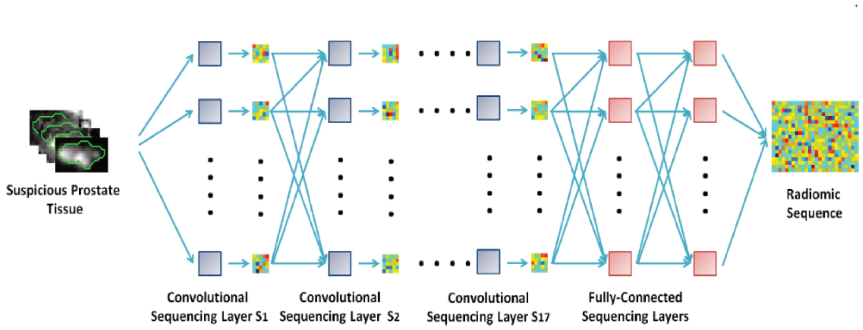


FIGURE 1.5: Example of popular feature extractors used in Radiomics. Heterogeneous feature groups make Radiomic problem multi-view. This figure is extracted from [3].

In [61], the authors propose a CNN architecture with 17 layers for multi-parametric MRI data. The structure of the designed CNN is shown in Figure 1.6. The difficulty here is that their dataset only contains 20 patients which is obviously too few for learning such a deep architecture. To overcome this difficulty, the authors firstly reduce the numbers of parameters by treating the receptive field as a random field with the connection weights in the receptive field being spatially correlated subject to a spatial correlation parameter. Then they augment the sample size by rotating the original images, resulting in 640 cancerous regions and 5712 healthy regions. However, compared to the number of parameters in CNN, the sample size is still not sufficient. There are also some other works trying to use shallower networks to reduce the number of parameters. In [254], they build a 4-layer CNN with 96 training instances (PET images). Similarly in [48], they also propose a 4-layer CNN for the prediction of 5 year mortality with 48 CT scans of chest.

There are mainly two disadvantages of CNN based feature extractors. Firstly,



Convolutional Sequencing Layer	Number of Receptive Fields	Receptive Field Size
1, 2	64	5×5
3, 4	128	5×5
5, 6, 7, 8	256	5×5
9, 10, 11, 12, 13, 14, 15, 16	512	5×5
17	2000	5×5

FIGURE 1.6: Example of feature extractors based on CNN. This figure is extracted from [61].

they lack of interpretability. CNN is very often treated as black-box function approximator that learns the mapping between a given input to a classification output ([50]). This makes it less suitable for medical case because it is hard to explain the utility of each feature and to provide human understandable justifications for its output. Secondly, CNN can be very accurate when there are a lot of training instances, but in the case of Radiomics, the sample size is usually very small. Some papers use only 13-25 patients in total for training and tests. Due to the lack of instances in Radiomics, more and more works have tried to use transfer learning instead of training a CNN on very small sample size ([120, 145, 154, 176]). To improve the performance of transfer learning based features, handcrafted features are used in addition. For example, in [145], they extract a total of 1403 handcrafted features and 98304 deep features from MRI of 75 patients. We have also combined transfer learning features with handcrafted features for breast cancer histology data in [46]. Deep learning features from ResNet-18 from [115] (512 features), ResNet-152 from [115] (2048 features), ResNeXt from [250] (2048 features), NASNet-A from [262] (4032 features) and VGG16 from [212] (25088 features) along with

175 handcrafted features are used which result in a total of 33903 features (details about this work can be found in Appendix A).

1.2.3 Machine Learning for Radiomics

The previous section has detailed the first steps of the workflow of Radiomics presented in Figure 1.4, until the feature extraction. This section focus on how the literature addresses the next step of this workflow: the learning phase.

In the previous section, it has been shown that Radiomic problems are typical HDLSS multi-view problems. In a large majority of Radiomic works, the multiple views are concatenated to form a single-view feature vector. Among the multiple views, it is very often to have some high dimensional views. When these multiple feature groups are concatenated, it usually worsens the HDLSS problem. This constitutes the major obstacle for the traditional machine learning methods. For example, linear discriminant analysis (LDA) is not suitable for HDLSS problem as the pooled covariance matrix is not invertible ([256]). Support vector machine (SVM) is also proved to have data piling problem on very high dimensional data ([163]), which may adversely affect the generalization performance of SVM in some HDLSS situations.

As a consequence, feature selection methods are systematically used to overcome the difficulty of learning in high dimension. The goal is to reduce the redundancy, noise, or irrelevant features while at the same time keep good performance. There are mainly two feature selection strategies used in Radiomic works. The first one is to select features by some predefined criteria and then build machine learning classifiers on the selected features. The second one combines the feature selection procedure with classification by using the performance of a predefined classifier as the feature selection criteria.

Most of the studies of Radiomics belong to the first category, where feature selection is used independently from the machine learning methods. Usually a feature score showing the relevance or reliability (such as in [65], [174] and [3]) is calculated to rank all the features. Then, the features with the lowest ranks are removed. The number of features or the threshold should be predefined. In Radiomics, most studies choose no more than 15 features among

hundreds of features ([3, 64, 87, 114, 248]). Afterwards, statistical analysis and machine learning methods are applied to the selected feature subsets. The advantages of the first methodology is obvious: it is computationally simple and fast, hence it can scale to very high dimensional feature space easily. However, these methods do not take into account the interaction with classifiers and the search in the feature subset space is separated from the search in the hypothesis space ([201]), which may hurt the performance of classifiers in some cases.

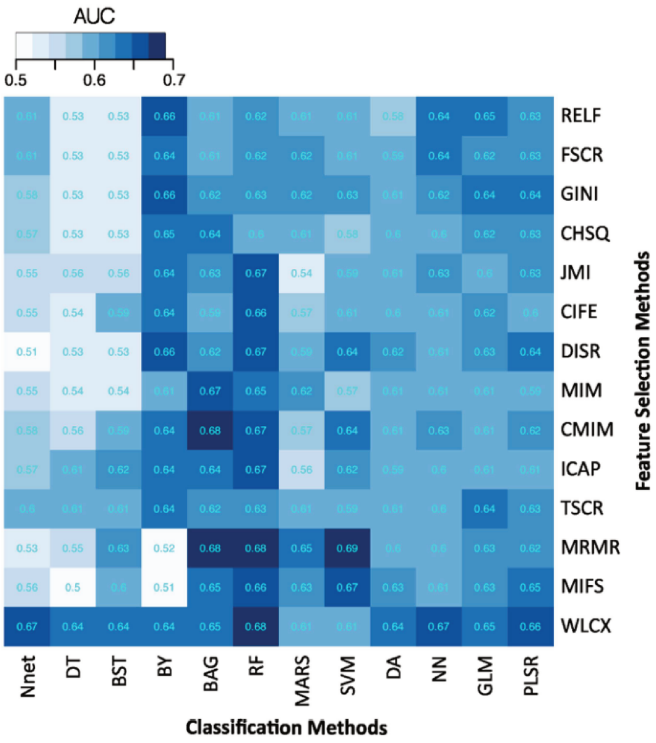


FIGURE 1.7: Heatmap depicting the predictive performance (AUC) of feature selection (in rows) and classification (in columns) methods. It can be observed that RF, BAG and BY classification methods and feature selection methods WLCX, MRMR show relatively high predictive performance in many cases. This figure is extracted from [173].

Some studies also realize the important role of the classifier in Radiomics and propose to combine feature selection methods with reliable machine learning

methods to find the most accurate combination. For example, in [173], the authors use 14 feature selection methods along with 12 classifier families on CT images of lung cancer patients. Two similar studies has been done in [174] and in [248]. The experimental result of [173] is shown in Figure 1.7: it can be seen that feature selection criteria Minimum Redundancy Maximum Relevance (MRMR) and Wilcoxon (WLCX) have the best general performance; machine learning methods Bagging (BAG), RF and SVM have good overall performance with selected features. But choosing the proper classifier on selected features is very important too. For example, the features selected by MRMR have the best performance if fed into BAG, RF or SVM. But when Bayesian (BY) is used as the classifier on the same selected features, the performance is the worst, which motivates the second category of feature selection in Radiomics by taking the interaction of classifiers into consideration.

The second category of feature selection in Radiomics integrate the classifier into the selection procedure to improve the classification performance. Generally speaking, a classifier needs to be chosen in advance and some feature weighting or ranking approach is usually embedded in the training process. The feature elimination is then realized according to the weight/rank of features. Few Radiomic works have successfully used Support Vector Machine Recursive Feature Elimination (SVM-RFE [107]) and obtained very good performance. This approach differs from the most popular approaches used in Radiomics by embedding the feature selection into the learning procedure of the SVM, so that it can take the resulting classifier performance into account. In [238, 257], they showed that SVM-RFE had very good performance on Radiomic problem. The workflow of [257] is shown in Figure 1.8: SVM-RFE is used to rank all the features, then SVM classifier is trained to select the best size of feature subset. For other HDLSS problems, SVM-RFE also had very good performance ([30]).

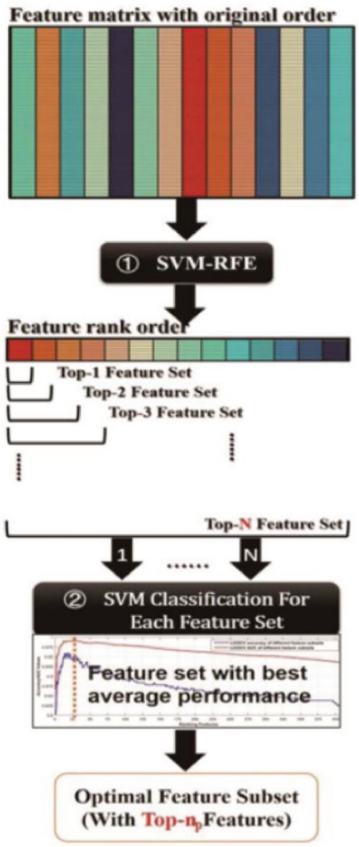


FIGURE 1.8: The process for finding the optimal Radiomics feature subset by SVM-RFE. This figure is extracted from [257].

1.3 HDLSS multi-view learning: a literature review

In the previous section, we have introduced the HDLSS multi-view machine learning problem through the example of Radiomics. The state-of-the-art Radiomics solutions always ignore the multi-view machine learning challenge behind the problem and use feature selection methods. However, multi-view learning solutions that can take advantage of complementary information from

different views should be considered to improve the performance. This section presents a taxonomy of multi-view learning methods with an emphasis on HDLSS problems to provide better solutions to the HDLSS multi-view learning problems. Our goal here is not to give an exhaustive survey on this machine learning field but to present a panorama of the different multi-view approaches, especially for the methods that can deal with the HDLSS problem. According to [204], there are three main kinds of multi-view approaches: early integration, late integration and intermediate integration. Each of these approaches is detailed in the following sections.

1.3.1 Problem statement

Before introducing the different multi-view learning approaches, this section gives a formal definition of this type of problems, and details all the notations that will be used in the rest of this manuscript.

Supervised learning

Supervised learning tasks strive to infer a function h, often called a model, that maps an input domain $\mathcal{X}$ to an output domain $\mathcal{Y}$:

$$h : \mathcal{X} \to \mathcal{Y}$$

For cases where $\mathcal{Y} = \mathbb{R}$, the problem is called a regression problem. For cases where $\mathcal{Y}$ is a finite set of classes, the problems is called a classification problem. In the latter case, the $\mathcal{C}$ classes are denoted $\{\omega_1, \omega_2, \ldots, \omega_{\mathcal{C}}\}$.

For simplicity, and because it concerns most of the Radiomics tasks found in the literature, this manuscript mainly focus on classification. However, note that most of the methods described in this section also suit to regression tasks.

As for the input domain $\mathcal{X}$, it is typically a m-dimensional space, i.e. $\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times \cdots \times \mathcal{X}_m$, where $\mathcal{X}_i$ is the domain of the ith feature of the problem. Consequently, an instance $\mathbf{x} \in \mathcal{X}$ is a m-dimensional vector noted:

$$\mathbf{x} = (x_1, x_2, \ldots, x_{m-1}, x_m)$$

where x_j is the value of the jth feature of $\mathbf{x}$.

In supervised learning, h is said to be learnt from a set $\mathcal{T}$ of labeled instances. This set is usually called a training set, and is composed of n input-output pairs:

$$\mathcal{T} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$$

Such a training set is often written as a $n \times m$ matrix:

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \ldots & x_{1,m} \\ x_{2,1} & x_{2,2} & x_{2,3} & \ldots & x_{2,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & x_{n,3} & \ldots & x_{n,m} \end{bmatrix} \tag{1.1}$$

where $x_{i,j}$ is the value for the jth feature of the ith instance in $\mathcal{T}$. In the same way, the n output values are gathered in a vector $\mathbf{y} = (y_1, y_2, \ldots, y_n)$, where y_i is the class label of the ith instance in $\mathcal{T}$.

Multi-view learning

Multi-view learning is a learning task where each instance is described by Q different input vectors instead of only one. Formally, the task is to infer a model h:

$$h : \mathcal{X}^{(1)} \times \mathcal{X}^{(2)} \times \cdots \times \mathcal{X}^{(Q)} \to \mathcal{Y}$$

where the $\mathcal{X}^{(q)}$ are called the views. These views constitute different description spaces of different dimensions, noted m_1 to m_Q. In such learning framework, the training set $\mathcal{T}$ is actually decomposed in Q training set:

$$\mathcal{T}^{(q)} = \left\{(\mathbf{x}_1^{(q)}, y_1), (\mathbf{x}_2^{(q)}, y_2), \ldots, (\mathbf{x}_n^{(q)}, y_n)\right\}, \forall q = 1..Q$$

Similarly to $\mathcal{T}$, a multi-view dataset can be written as Q matrices:

$$\mathbf{X}^{(q)} = \begin{bmatrix} x_{1,1}^{(q)} & x_{1,2}^{(q)} & x_{1,3}^{(q)} & \dots & x_{1,m_q}^{(q)} \\ x_{2,1}^{(q)} & x_{2,2}^{(q)} & x_{2,3}^{(q)} & \dots & x_{2,m_q}^{(q)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n,1}^{(q)} & x_{n,2}^{(q)} & x_{n,3}^{(q)} & \dots & x_{n,m_q}^{(q)} \end{bmatrix} \forall q = 1..Q \tag{1.2}$$

where $x_{i,j}^{(q)}$ is the value of the jth feature of the ith instance in the qth view.

High dimension, low sample size (HDLSS)

HDLSS problems are learning problems for which the dimension m of the input space is very high in regards to n, the number of instances available in the training set $\mathcal{T}$. Machine learning problems described in high-dimensional spaces are known in the scientific community to be particularly difficult, since an enormous amount of training instances is typically required for learning an accurate classifier. A typical rule of thumb is that there should be at least 5 training examples for each dimension in the representation ([224]). It is well known that such "curse of dimensionality" problem leads to serious breakdown in many algorithms with an under-determined problem. According to [163], in the context of HDLSS, classical multivariate analysis is useless due to the need of the root inverse of the covariance matrix, which does not exist (because the covariance is not of full rank). Many traditional machine learning techniques such as Logistic Regression (LR), discriminant analysis or K-Nearest Neighbors (KNN) are not able to give a solution to HDLSS problem due to the ill-posedness ([103]). For example, Linear Discriminant Analysis (LDA) is not suitable for HDLSS problem as the pooled covariance matrix is not invertible ([256]). Support Vector Machine (SVM) is also known to have data piling problem on very high dimensional data ([163]). When the sample size is small, the distribution of data in the high dimensional space is very sparse, which makes it also harder to deal with outliers ([32]) and may easily cause overfitting problem. Unlike these classifiers discussed above, Random Forest can deal well with high dimensional data due to the implicit feature selection mechanism during the tree construction ([57, 222]).

Nevertheless, as already extensively discussed in the previous section, many machine learning problems are naturally described in high-dimensional spaces but with very few training instances; that is to say with $m >> n$. These problems are called HDLSS learning problems.

When transposed to multi-view learning problems, the HDLSS setting is even more critical. The reason is that the different m_q values are all potentially individually greater than n: $m_q >> n, \forall q = 1..Q$. Besides, if it is not strictly the case for all views, one can reasonably assume that $\left(\sum_{q=1}^{Q} m_q\right) >> n$. As it has been explained previously and as it will be further discussed in the following, concatenating all the views together, to form a new joint description space, is a principle often encountered in the literature when dealing with real-world HDLSS multi-view learning problems. In such a case, the dimension m of this joint description space is $m = \left(\sum_{q=1}^{Q} m_q\right)$, which exacerbates the difficulties that stem from the HDLSS setting. As an example, when compiling works from the Radiomics literature, the values for m_q are often between 400 and 2000 while the values for n are from 50 to 200 ([59, 122, 179, 225]). Furthermore, q is usually bigger than 3 which could lead to a joint description space of dimension m easily bigger than 1000 ([65, 87, 173, 174, 248]).

The state-of-the-art Radiomic solutions usually ignore the multi-view challenge by concatenating all the views together. To take better advantage of multi-view information, we firstly review the multi-view literature with the focus on the state-of-the-art methods offered for HDLSS problem. In the following sections, three different categories of multi-view solutions (early, late and intermediate integration) are introduced.

1.3.2 Early integration

Early integration methods directly concatenate different views together and treat the multi-view learning as single-view learning ([204]). All the Radiomics works discussed in the previous section belong to this category. The flowchart of early integration is given in Figure 1.9. The dimension of the description space formed by this concatenation is inherently high and a lot of machine learning methods can suffer from it. This would be especially the case, if the

number of learning instances is not sufficient in regards of this high dimension, as for HDLSS settings.

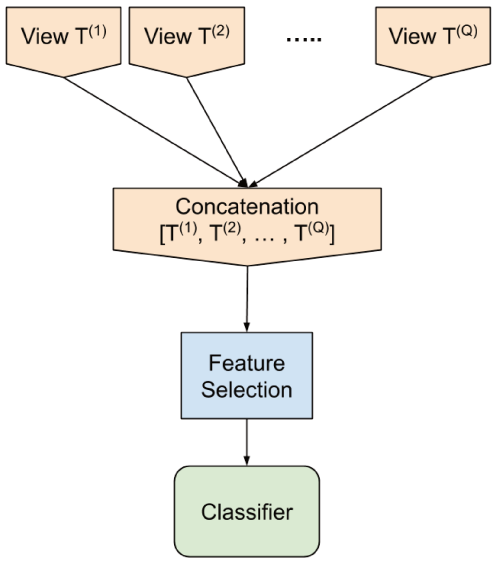


FIGURE 1.9: Flowchart of early integration methods.

To deal with the HDLSS problem, most of the studies use feature selection methods to reduce the dimension. Generally speaking, according to the state-of-the-art works ([30, 52, 175]), feature selection methods can be divided into three groups: filter methods (Figure 1.10a), wrapper methods (Figure 1.10b), and embedded methods (Figure 1.10c). These three groups of feature selection methods mainly differ in the interaction with classifiers, which will be discussed in the following sections.

Filter methods

Filter methods consist in ranking features according to a given criterion measured on each feature separately, and in using only the best ones according to a predefined number or to a threshold ([52]). The general scheme of filter method for feature selection can be found in Figure 1.10a. It is very important to find a suitable ranking criterion so that relevant features for the classification have higher ranks and irrelevant features have lower ranks. The

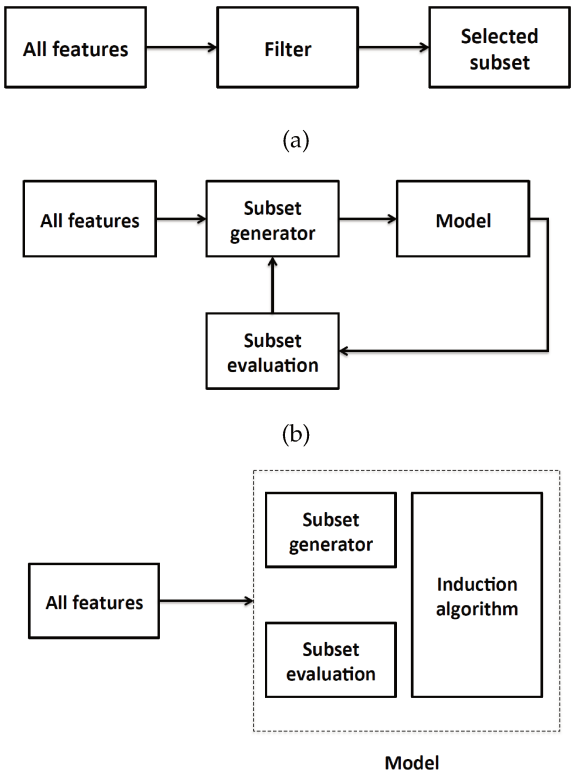


FIGURE 1.10: The general scheme of (a)filter method, (b) wrapper method and (c) embedded method for feature selection. This figure is extracted from [167].

next step is to filter the features by removing features with the rank below the predefined threshold. Finally, the selected feature subset is used to learn the machine learning model.

It can be told that the most important step for filter methods is to measure the relevancy of a feature to the given classification problem. Generally, the relevancy of the feature is related to the dependency of this feature to the class labels: if a feature is totally independent from the class labels, it means that this feature has no influence on the classification task and should be discarded ([52]). However, as filter methods are independent from the classifier and do

not take the correlation among features into consideration, the optimal feature subset may not be unique.

One of the most direct and simplest ranking criteria is the Pearson correlation coefficient ([106]) to detect the dependencies between a feature and the class label. Given the ith feature vector $\mathbf{X}_{.,i}$ and the class label vector $\mathbf{y}$, the Pearson correlation coefficient can be calculated with:

$$P(i) = \frac{cov(\mathbf{X}_{.,i}, \mathbf{y})}{\sqrt{var(\mathbf{X}_{.,i}) * var(\mathbf{y})}} \tag{1.3}$$

where $cov()$ is the covariance and $var()$ is the variance. This correlation based ranking is simple but limited to detect only the linear dependencies between the feature and the class label.

During the last decades, a large number of filter methods have been proposed, especially methods based on Mutual Information (MI), Relief and its variant RELF ([223]). MI is one of the most popular feature selection criteria due to its computational efficiency and simple interpretation ([30, 223]). Relief selects features that help to separate instances from different classes. RELF adds the ability of dealing with multi-class problems and is also more robust and capable of dealing with incomplete and noisy data ([30]). RELF can be interpreted as margin maximization, which explains why it has superior performance in many applications ([64, 223, 248]).

To summarize, filter methods are computationally simple and fast, which makes it very popular when facing HDLSS problems ([223]). However, the major disadvantage is that filter methods ignore the interaction with the classifier, the search in the feature subset space is separated from the search in the hypothesis space ([251]). In contrast to filter methods, we will introduce wrapper methods in the next section which obtain a feature subset relying on the classification.

Wrapper methods

Wrapper methods consist in using a classifier for selecting a subset of features, the objective being to optimize the classifier performance by searching for the

best subset ([30]). The general scheme of wrapper methods for feature selection is shown in Figure 1.10b. Given a predefined classifier, wrapper feature selection methods usually include the following steps: the first step consists in finding a particular subset of features among the 2^m possible subsets. The second step usually consists in training the classifier from a given subset of features and estimating its performance. The estimated performance is usually accessed by a validation dataset or cross-validation. The final subset of features, retained at the end of this procedure, is the one that has allowed to obtain the best performance.

However, exhaustive search systematically enumerates all possible combination of features and find the best feature subset, which is computationally intensive. For dataset with m features, the size of searching space is $O(2^m)$. In the case of HDLSS problem, the feature dimension m is normally very big, an exhaustive search is intractable. This problem is known to be NP-hard ([106]). Wrapper methods usually adopt sub-optimal searches, such as sequential search, or heuristics algorithms ([52]). The sequential selection methods are iterative algorithms that add or remove features at each iteration until the maximum objective function is obtained ([188, 196]). For example, the Sequential Forward Selection (SFS) adds one feature each time so that the maximum classification accuracy is obtained until the required number of features are obtained ([196]). However, SFS suffers from producing nested subsets since the forward inclusion is always unconditional ([52]). The feature selected in the next iteration is highly dependent on the previous selected features. Heuristics algorithms such as Genetic Algorithm (GA) are also very often used for feature subset selection ([9, 63, 97]). The global maximum for the objective function (classification accuracy for example) can be found, which gives the best sub-optimal subset ([52]).

To summarize, the main disadvantage of wrapper methods is the intensive computational cost. For each new subset, the classifier needs to be retrained to evaluate the performance. And overfitting may occur easily when the training instances are not enough. To provide better generalization ability, extra validation datasets are usually needed. For the HDLSS setting, it may be impossible to use an independent validation dataset for the search of the best feature subset. In the next section, the third category of feature selection methods,

namely embedded methods, are presented, which are usually considered to be better alternatives to filter and wrapper methods.

Embedded methods

Embedded methods consist in selecting features during the training process without splitting the instances into training and validation sets ([52]). The general scheme of embedded methods for feature selection can be found in Figure 1.10c. Compared to wrapper methods, embedded methods have the main advantage to be less computationally intensive ([201]); while compared to filter methods, embedded methods take into account the interaction with classifiers.

One of the most used embedded methods are pruning methods. Pruning methods firstly train the classifier with the entire feature set and eliminate features gradually by some ranking criteria while maintaining the performance of the classifier. SVM-RFE ([107]) is the most famous pruning based embedded method. It is a recursive feature elimination method using the learned feature weight as ranking criterion ([107]). As shown in Algorithm 1, firstly, all the features are used to train the SVM classifier, then the weight and the ranking criterion of each feature is calculated. The feature with the smallest ranking is eliminated. This process continues until all the features are eliminated, and at last a ranked list can be given, so that we can choose how many features we want in the ranked list. This approach differs from the filter approaches by embedding the feature selection into the learning procedure, so that it can take the resulting classifier performance into account. SVM-RFE method is known to be efficient and accurate on many kinds of HDLSS applications ([30, 238, 257]).

Apart from using SVM as the predefined classifier, neural networks can also be used for feature selection. A saliency measure calculated from trained multilayer perceptron networks is used to calculate the feature weights ([198, 205]). Network Pruning commonly used to obtain the optimum network architecture for neural networks can be used for feature selection ([52]). In [205], a penalty is applied for features with small magnitude at the node and the nodes connecting to these input features are excluded.

Algorithm 1 SVM-RFE

1: Input: training sample $\mathcal{T}$, labels y, m features
2: Survival subset s=[1,2,...m], rank list r = [ ]
3: Repeat until s = [ ]
4: 1. Train SVM with s
5: 2. Update weight w_i for each feature i.
6: 3. Calculate the ranking criterion: $(w_i)^2$
7: 4. Add the feature with lowest ranking criteria to r, and remove the feature from s.
8: Output: The updated feature ranks.

Discussion

Filter methods have the advantage not to require a validation set to perform feature selection, which is probably the reason why they are mostly used in many real world applications. However, they ignore the resulting classification performance and therefore, are usually less accurate than embedded and wrapper methods. On the other side, these two other families of approaches usually have a higher computational cost and are not very suitable for very low sample size problems.

1.3.3 Late integration

Late integration methods firstly build separate models on each view and combine them afterwards. These methods are named "late" because the data fusion process is done in the late stage of classification after the classifiers are trained. Most late integration methods belong to Multiple Classifier System (MCS) approach, which can be divided into two main categories: co-training and Independent Classifier Combination (ICC). The major difference between these two approaches is the interaction among classifiers: co-training methods re-train classifiers multiple times taking into account the information of classifiers from other views, while most ICC methods train classifiers once for each view independently and then merge the classification outputs. In the following sections, the details of these two approaches are given.

Co-training

Co-training is a semi-supervised principle that strives to maximize the mutual agreement between pairs of views, by training classifiers on labeled instances separately in each view, and by exploiting then their feedback on unlabeled instances. Three assumptions are made for the success of co-training: (1) sufficiency: each view should be informative enough for the classification, (2) compatibility: the target functions in both views predict the same labels for co-occurring features with high probability, (3) conditional independence: the two views of any example are conditionally independent given the class label.

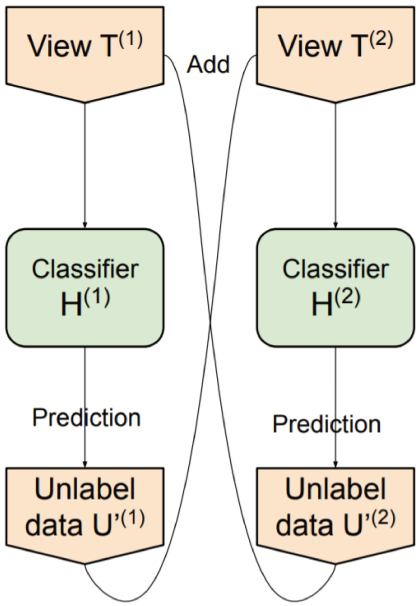


FIGURE 1.11: Flowchart of co-training.

The initial work of co-training was proposed in [29]. The process of co-training style algorithms can be found in Figure 1.11: The training set contains two views with both labelled instances $\mathcal{T} = \{\mathcal{T}^{(1)}, \mathcal{T}^{(2)}\}$ and unlabelled set $\mathcal{U} = \{\mathcal{U}^{(1)}, \mathcal{U}^{(2)}\}$. Firstly, a small unlabelled set $\mathcal{U}'$ is created by random sampling from $\mathcal{U}$. Then, two naive Bayes classifiers $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ are trained on $\mathcal{T}^{(1)}$ and $\mathcal{T}^{(2)}$ respectively. In the second step, $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ are used to label unlabelled set $\mathcal{U}'$. p most confidently labelled instances by $\mathbf{H}^{(1)}$ and p most

confidently labelled instances by $\mathbf{H}^{(2)}$ are chosen (while keeping the class ratio) to be added to the labelled training set $\mathcal{T}$. Finally, $2p$ instances randomly drawn from $\mathcal{U}$ are then added to replenish $\mathcal{U}'$. The updated $\mathcal{T}$ is then used to re-train $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$. This re-training process will stop until a termination condition (e.g. maximum iteration) is satisfied.

Co-training explores the relation between classifiers built on each view by maximizing the mutual agreement on two distinct views of the data in a semi-supervised way. Usually, different views can provide different useful information, which indicates the differences between classifiers $\mathbf{H}^{(1)}$ and $\mathbf{H}^{(2)}$ during the first iterations. As the learning process is going on, the two classifiers are used to predict more and more instances from $\mathcal{U}$, and their disagreement is expected to be smaller and smaller ([240]). Through information exchange between views, the final optimized classifier can be obtained ([29]). Nigam and Ghani [171] showed experimentally that even for single-view data, co-training on multiple views manually generated by random splits of features can still improve performance.

The unlabelled instances play an important role in co-training methods, which enable the information exchange between classifiers. However, as explained in the first chapter, HDLSS multi-view problems are usually composed of very few labeled instances and no additional unlabeled instances are available. Secondly, the co-training method is originally proposed to solve problems with two views. When there are more views, the solution will be much more complex. Thirdly, it is very hard for real world HDLSS multi-view problem applications to fulfill the three assumptions, which may lead to the failure of co-training methods. As a consequence, co-training approaches are not straightforwardly applicable to HDLSS multi-view problems.

Independent Classifier Combination

Co-training makes many assumptions on the given multi-view problem and uses unlabelled instances to exchange information between classifiers, which limits its use in many real world applications. ICC based methods are more flexible by training one classifier for each view independently at first and then combining these classifiers in a proper way. In [111], the authors recall that

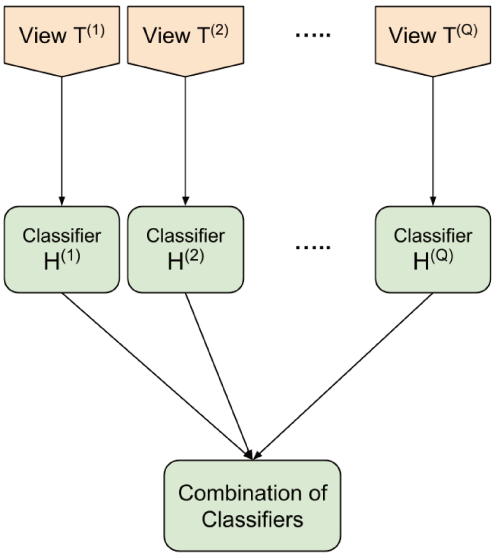


FIGURE 1.12: Flowchart of ICC.

the ICC will have better performance than an individual classifier when there exists disagreement among classifiers. Such independence and disagreement is usually referred to as diversity within the ensemble. For multi-view problem, the information of each view is often very different from other views so that complementary information can be provided, which makes ICC a good solution as the diversity may be guaranteed.

Voting based methods are widely used in the field of ICC ([166]), among which the most popular one is Majority voting. It generally takes the output of each classifier as a vote and gives the final decision by finding the most voted decision. Majority voting assigns equal weights for every view during the combination. But in real world applications, different views may have different importance for the problem at hand. In this case, weighted voting can be used. Each classifier is given a weight according to some confidence evaluation measure, where the sum of weights from all classifiers is one. Apart from these fusers, support function fusion is also widely used, which uses the likelihood of a class such as a posteriori probability provided by each classifier, the output of neuron network or fuzzy membership function ([8, 27, 132]). A

posteriori probability produced by the probabilistic models embodied by the classifiers is the most popular support function fusion method ([247]). Dynamic weighting ([45]) can also be used to combine the decisions, but in the cost of using some validation dataset for most cases.

Aside from the rule based combination, trainable combination is also used in ICC. The new feature vector is constructed by combining the outputs of all the base classifiers, then a 'secondary' classifier can be built on these new features to give the final decision. For example, neural networks are used in [39] to learn the combined matching score based on the classifiers built from face data and voice data. Other classifiers such as SVM, decision trees, Multilayer Perceptrons (MLP) and Random Forest are also used in [53, 91, 105, 169, 197, 232]. Combinations with higher complexity can potentially provide better classification results. But validation datasets to train the secondary classifier are always required to guarantee the improvement. The small number of training instances will limit the choice of combination methods. Hence, choosing different combination approaches is a trade-off between the classifying capabilities of combination functions and the training sample size ([228]).

Discussion

Compared to co-training, ICC is more suitable for HDLSS multi-view problems because fewer assumptions are made beforehand and no extra unlabelled are needed to re-train the classifiers. ICC is usually faster compared to Co-training as all the classifiers are trained only once. However, in some cases, validation datasets are needed for a better combination of classifiers. For both late integration methods, attention should be paid for the choice of classifiers due to the fact that a lot of classifiers can not deal well with high dimensional problems. For low sample size problem, Co-training method is not a good choice due to the need of many unlabelled instances while ICC may not be a good choice neither when validation datasets are required for the combination of classifiers.

1.3.4 Intermediate integration

The two previous sections explained that early integration methods do not take the multi-view learning specificities into account and that late integration methods are not well suited for HDLSS learning settings. This section presents the intermediate integration approach that is – as suggested by its name – an in-between approach. Intermediate integration methods combine the information from different views at the feature level and perform learning in a joint feature space ([149]). The most used intermediate integration methods include shared representation learning and multi-representation fusion methods, which will be discussed in the following sections in detail.

Shared representation learning

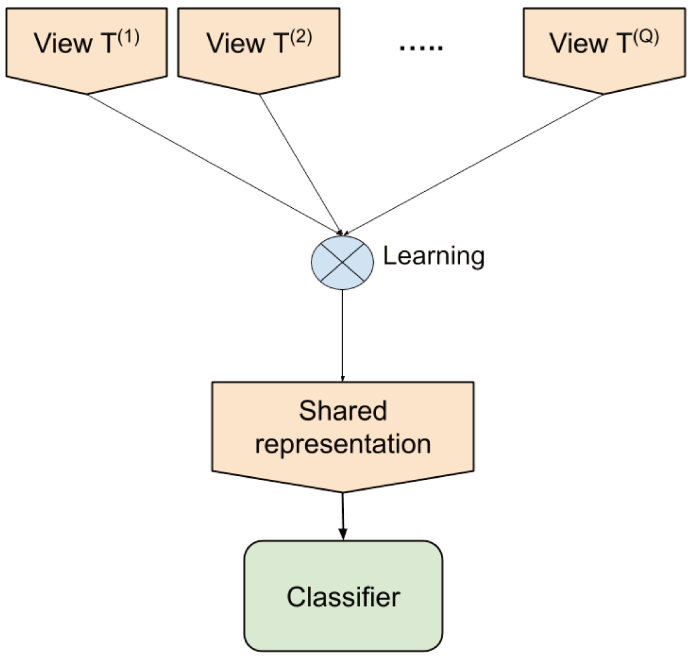


FIGURE 1.13: Flowchart of shared representation learning.

Shared representation learning aims at finding a latent low dimensional space shared by all the views (see Figure 1.13): feature transformation functions for each view are learned with the assumption that all the views are generated from the latent space. Shared representation learning is an efficient multi-view dimensionality reduction technique as the dimension of the latent space is lower than any view, but it is mostly unsupervised and ignores the supervised information, which may lead to a subspace with weak predictive ability ([56]).

Canonical correlation analysis (CCA) is the most well-known shared representation learning method ([113, 131]). CCA works by seeking for a projection for each view so that the correlation among the projected views is maximized. For multi-view problems with two views $\{\mathbf{X}^{(1)}, \mathbf{X}^{(2)}\}$, CCA learns two projections $\mathbf{w}_1 \in \mathbb{R}^{m_1}$ and $\mathbf{w}_2 \in \mathbb{R}^{m_2}$ for view 1 and view 2 respectively so that the following correlation between two projections is maximized:

$$\rho = \frac{\mathbf{w}_1^T \mathbf{X}^{(1)} \mathbf{X}^{(2)T} \mathbf{w}_2}{\sqrt{(\mathbf{w}_1^T \mathbf{X}^{(1)} \mathbf{X}^{(1)T} \mathbf{w}_1)(\mathbf{w}_2^T \mathbf{X}^{(2)} \mathbf{X}^{(2)T} \mathbf{w}_2)}} \tag{1.4}$$

As ρ is invariant to the scaling of $\mathbf{w}_1$ and $\mathbf{w}_2$, Equation 1.4 can be formulated as:

$$\begin{aligned} &\underset{\mathbf{w}_1, \mathbf{w}_2}{max}\, \mathbf{w}_1^T \mathbf{X}^{(1)} \mathbf{X}^{(2)T} \mathbf{w}_2 \\ s.t.\ &\mathbf{w}_1^T \mathbf{X}^{(1)} \mathbf{X}^{(1)T} \mathbf{w}_1 = 1, \mathbf{w}_2^T \mathbf{X}^{(2)} \mathbf{X}^{(2)T} \mathbf{w}_2 = 1 \end{aligned} \tag{1.5}$$

However, CCA can not be applied directly to many real world datasets which exhibit non-linear characteristics, hence the kernel variant of CCA, namely KCCA ([112, 140]), was proposed to firstly map each instance to a higher space in which linear CCA can be applied. With the replacement of kernel matrices $\mathbf{K}_1 = \mathbf{X}^{(1)T} \mathbf{X}^{(1)}$ and $\mathbf{K}_2 = \mathbf{X}^{(2)T} \mathbf{X}^{(2)}$, the optimization in Equation 1.5 can be rewritten as:

$$\begin{aligned} &\underset{\mathbf{w}_1, \mathbf{w}_2}{max}\, \mathbf{w}_1{}^T \mathbf{K}_1 \mathbf{K}_2^T \mathbf{w}_2 \\ s.t.\ &\mathbf{w}_1{}^T \mathbf{K}_1 \mathbf{K}_2^T \mathbf{w}_1 = 1, \mathbf{w}_2^T \mathbf{K}_2 \mathbf{K}_2^T \mathbf{w}_2 = 1 \end{aligned} \tag{1.6}$$

Apart from the basic CCA and KCCA, there are also a lot of other studies

related to CCA. In [260], the authors propose MKCCA for dimensionality reduction by performing PCA followed by CCA to better remove noises and handle the issue of trivial learning. More recently, more and more deep learning based CCA such as Deep CCA [12] and its variant [241] have also been proposed.

Multi-representation fusion

In contrast to shared representation learning methods, multi-representation fusion provides a more transparent way to take advantage of the complementary information among different views (see Figure 1.14). Multi-representation fusion projects each view in a space in which every instance is described by its (dis)similarities to all the training instances. In that way, each view is separately projected in the same description space so that linear or non-linear combinations can be applied directly. Multi-representation fusion is very flexible and efficient, and can be applied to many different types of data.

In [177], the authors proposed an SVM with heterogeneous kernel function, which firstly computes separate kernels for each view and then sums the results. Their proposed kernel is an attempt to incorporate prior knowledge into the task at hand. The kernel function is shown in Equation 1.7, where g and p stand for two different views (gene expression and phylogenetic profiles), and K is a local kernel. In the experiments, they compared the performance of single view data as well as three multi-view integration methods (early, intermediate and late integration), and showed that multi-representation fusion based method is the best performing among all the tested methods.

$$K_{combined}(\mathbf{x}_1, \mathbf{x}_2) = K_g(\mathbf{x}_1^{(g)}, \mathbf{x}_2^{(g)}) + K_p(\mathbf{x}_1^{(p)}, \mathbf{x}_2^{(p)}) \tag{1.7}$$

From the heterogeneous kernel $K_{combined}$, it is easy to tell the difference between shared representation learning and multi-representation fusion. Shared representation learning methods find the data projection to maximize the view correlation and the dimension of projected space needs to be predefined. The

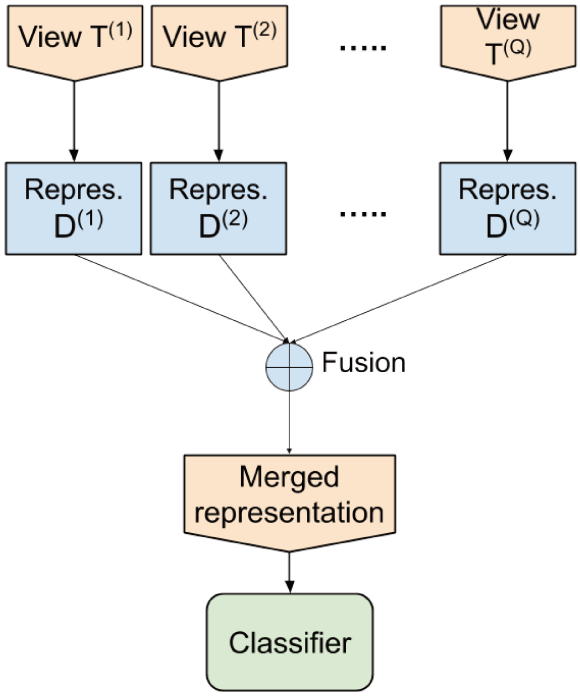


FIGURE 1.14: Flowchart of multi-representation fusion methods.

heterogeneous kernel proposed in [177] takes advantage of the complementary information by the sum of two defined kernels. They firstly define a kernel for each view using prior domain knowledge. The kernel matrix can be seen as a similarity matrix, which projects data into a lower dimension and the dimension is the same as the training sample size (no need to be predefined). However, the label information is not used for the construction of the joint kernel in [177].

To take the class information into account, many studies have been done on combining multiple kernels together linearly or non-linearly to improve the performance, named Multiple Kernel Learning (MKL) . MKL ([40]) is a well exploited field and lot of researches have been done, which studies the generation and combination of different kernels for an optimized performance.

For multi-view problems, MKL uses one kernel for each view and the final kernel K_η is used for SVM:

$$K_\eta(\mathbf{x}_h, \mathbf{x}_k) = f_\eta(\{K^{(q)}(\mathbf{x}_h^{(q)}, \mathbf{x}_k^{(q)})\}_{q=1}^{Q}) \tag{1.8}$$

where f_η is a linear or non-linear combination function. The most successful kernels in the literature include linear kernel, polynomial kernel and gaussian kernel. Linear combination is the most popular approach in MKL ([100]), which contains simple sum or average of kernels as well as weighted combination (Equation (1.9)).

$$K_\eta(\mathbf{x}_h, \mathbf{x}_k) = f_\eta(\{K^{(q)}(\mathbf{x}_h^{(q)}, \mathbf{x}_k^{(q)})\}_{q=1}^{Q}) = \sum_{q=1}^{Q} \eta_q K^{(q)}(\mathbf{x}_h^{(q)}, \mathbf{x}_k^{(q)}) \tag{1.9}$$

Lanckriet et al. [144] show how the kernel matrix can be learned from data via semidefinite programming (SDP) techniques. Better results are obtained than SVMs trained with each single kernel in 9 out of 13 experiments. SimpleMKL use the optimization of linear combination with kernel weights on a simplex ([195]). According to their comparison results, using multiple kernels is better than using a single one in terms of accuracy. However, trained linear combination is not always better than averaging for simple linear kernels. For the combination of complex gaussian kernels, linear combination is better than nonlinear combinations, but still not better than unweighted combination. Their results show that simple average of kernels is a strong baseline. Similar to the conclusion of [195], in [6], they also find out that the mean of kernels can obtain very good results. Hence they proposed a time and space efficient MKL method named EasyMKL by maximizing the distance between positive and negative examples ([255]). Their experimental results are shown to perform significantly better than the simple kernel averaging.

Apart from the combination of kernels, dissimilarity measures are also used to merge the information from different views. There are a large quantity of dissimilarity measures in the literature for all different feature types such as binary, categorical, ordinal, symbolic or quantitative features. There are no

restrictions like being symmetric or positive semi-definite (PSD) for dissimilarity measures neither. Hence, a lot of studies have also tried to use multiple dissimilarity fusion.

In [153] for example, the authors propose a method to combine multiple dissimilarity measures together. They firstly define a distance measure D_a based on Kullback-Leibler (KL) divergence, which measures directly the difference between a query and a prototype. Then, they define another distance measure D_d which only considers inter-relations between different training instances. Similarly, Heterogeneous Auto-Similarities of Characteristics (HASC) is proposed to deal with heterogeneous data with a combination of covariance matrix of features (COV), and Entropy and Mutual Information (EMI) matrix ([202]). Many other similar works of combining multiple dissimilarity matrices with heuristic rules can be also found in [11, 121, 150, 185]. An adaptive bilinear mixing of dissimilarities is also proposed in [128]. They claim that if the data are heterogeneous, a single dissimilarity measure might not be sufficient to describe the relations between the data. They focus on the prototype based learning like learning vector quantization dataset. Similar to the idea of MKL, the combination of dissimilarities most adequate for the classification task is also learned. Generalized Learning Vector Quantization (GLVQ) is used to integrate bilinear mixing and weighting of dissimilarities in prototype-based classification learning.

Discussion

In summary, shared representation learning is an efficient multi-view dimensionality reduction technique. However CCA based methods are unsupervised and ignore the supervised information, which may lead to a joint space with weak predictive ability. Multi-representation fusion projects each view in a space in which every instance is described by its similarities (kernel functions) or dissimilarities to all the training instances. In that way, each view is separately projected in comparable lower description spaces, which provides a good solution for HDLSS problem ([67]). Then the joint data representation can be obtained by searching for the best combination of the new data representations, which provides a good solution for multi-view problems.

1.4 Conclusion and contributions

In this chapter, we have introduced the HDLSS multi-view problem through the example of Radiomics. From the perspective of multi-view learning, Radiomic features are always from multiple distinct feature groups or different imaging modalities to better represent the tumor heterogeneity. From the perspective of HDLSS, Radiomic features are easily to be high dimensional (over 1000 features), especially when combined with transfer learning features. But the sample size of Radiomics is usually very small, normally fewer than 100 instances. Most of the state-of-the-art works in Radiomics concatenate all the feature groups together as a single feature vector, which often results in a very high dimension. Hence feature selection is the most used method to reduce the dimension. However, if only a small subset of the features are chosen, certainly a lot of useful information is lost and the heterogeneity can not be well represented. By concatenating all feature groups together, the complementary information from different feature groups is often ignored.

To make better use of multi-view information, we have reviewed the state-of-the-art multi-view solutions including early integration, intermediate integration and late integration with an emphasis on HDLSS problems. We have shown that early integration ignores the potential complementary information that different views may offer, while late integration methods are not very suitable for low sample size problem since they may require additional training instances to optimize the combination of classifiers. In our opinion, intermediate integration methods offer a better way to deal with the HDLSS multi-view problem by studying the relations between views and combining the views together so that traditional machine learning methods can be applied in a joint low dimensional space. Among the different intermediate integration approaches introduced above, we have shown that multi-representation fusion is the most appropriate solution for the HDLSS multi-view problem for the following reasons: to deal with the HDLSS problem, features from each view can be represented by (dis)similarity matrix, which leads to a low-dimensional description space for HDLSS problems; it also makes the fusion of each view very straightforward since dissimilarities are always comparable from one view to another; then, the multi-view problem can be solved in a

more transparent way by searching for the best combination.

Multi-representation fusion can be divided as a two stage learning process: the first stage is to learn the new data representation to project data from each view into a lower common space; the second stage is to learn the combination of views to better exploit the complementary information among views. Most multi-representation fusion methods focus on the second step only. For example, multiple kernel learning, the richest literature in multi-representation fusion, focus more on the second step about how to combine different kernels while ignoring the first step about how to learn the appropriate kernel for each view (most MKL methods just choose the well-known predefined kernels such as linear polynomial or gaussian kernels without identifying the nature of the data). In this work, we believe that more attention needs to be paid on the first step: firstly, data from different views normally have different nature, type or complexity. An appropriate (dis)similarity representation should be learned accordingly. Secondly, learning the proper (dis)similarity representation for each view makes the data combination more meaningful because the only information shared by all the views (the class information) will be included in each (dis)similarity representation.

The main contributions of this thesis are:

- Random Forest Dissimilarity (RFD) is chosen as an intermediate representation for multi-representation fusion. To deal better with HDLSS problem, the parameterization of RFD is also studied. In the experiments in Chapter 2, by comparing with different early, late and intermediate integration methods on real world datasets, RFD based intermediate integration method show the potential of being a good solution for HDLSS multi-view problem.
- Some limitations and possible modifications of the classic RFD measure are studied. Two more accurate dissimilarity measures are proposed based on Random Forest to improve the classification performance of multi-representation fusion. Instance hardness measure in the subspace defined by each leaf node is used to weight the dissimilarity values so that they are no more binary and more accurate. The experimental results in Chapter 3 show significant improvement over classic RFD based

multi-representation fusion.

- Instead of using simple averaging to form a joint dissimilarity matrix, static weighting and dynamic weighting methods are explored to take better advantage of the complementary information. A static weighting based on OOB accuracy and a dynamic view selection method are proposed. From the experimental results in chapter 4, both methods can improve the classification performance while the dynamic view selection is significantly better than averaging.

Chapter 2

Random Forest Dissimilarity for intermediate integration

Contents

2.1 Introduction

The previous chapter introduced the HDLSS multi-view problem. In particular, we have shown that a key aspect of multi-view learning is to efficiently exploit the complementarity of the views. To do so, multi-view methods based on the fusion of multiple intermediate representations are the most relevant from our point of view, especially for HDLSS problems. Multi-representations fusion methods deal with HDLSS multi-view problems through two steps : i) building new low dimensional representations of the data from each view separately, and ii) merging these view-specific representations into a joint representation.

Most multi-representation fusion methods focus on learning a good (parametric) combination operator rather than learning good view-specific intermediate representations. Dissimilarity representations are usually chosen as the intermediate representation because they offer the pairwise information between instances indicating if these two instances are similar or dissimilar. As the pairwise information is comparable across the views, it makes the representation merging task easier. Another advantage of using dissimilarity representation is that the original high dimensional $n \times m$ data (n is sample size, m is feature size) will be presented as a $n \times n$ matrix, which offers a natural solution to HDLSS problem.

There are a lot of different dissimilarity measures in the literature that can be used for multi-representation fusion. In general, they can be divided into two groups: learning free measure and learning based measure. Learning free measures are general purpose measures (e.g., the Euclidean distance and the cosine similarity for feature vectors). They are mostly defined without specific context and are problem independent (not learned from data). Learning based measures learn a specific measure for the data by taking the class information into account. For example, in the case of classification tasks, the goal would be to make the distance reflect the best possible the class membership: instances from the same class should be close to each other while instances from different classes should be far from each other [23, 206]. For multi-view problems, the features from different views may be very diverse, and the only information shared by all the views is the output information. Using learning based

dissimilarity measure as the intermediate representation can help to filter the "noisy" information from each view that is not related to the classification task and make the fusion task more efficient and transparent. Hence, in this work we firstly focus on generating the proper intermediate representation with class information and secondly focus on the combination. The experimental results of this chapter have been published in [47].

The remainder of this chapter is organized as follows. In section 2.2, different dissimilarity learning methods are introduced and compared. Random Forest dissimilarity, the most appropriate method for the HDLSS problems, is introduced in details in section 2.3. The parameterization of RFD are studied in section 2.4. The proposed RFD based intermediate integration methods are compared to the state-of-the-art Radiomic and multi-view methods on 15 datasets in section 2.5. The conclusion and future works are introduced in section 2.6.

2.2 Related works

In the literature, methods that learn a similarity or dissimilarity measure from a dataset can be generally called dissimilarity learning. Dissimilarity is a very general term which relates to many notions such as distance, kernel, similarity etc. With different problem formulations or constraints, there exists a considerable number of approaches that aim at learning the dissimilarity, among which the most useful ones include metric learning, kernel learning and random partitions. The details of each of these methods are given in the following sections.

We first give the definitions of distance, kernel, similarity and dissimilarity. In terms of classification, instances from the same class should be similar in some way and instances from different classes should be dissimilar. The notion of "(dis)similarity" plays a pivotal role in pattern recognition and machine learning. Similarity measure is a numerical measure of how close two instances are. The value is bigger when two objects are closer. On the contrary, dissimilarity measure is a numerical measure of how different two instances are. The value is smaller when two instances are closer. Normally, it's possible to transfer a

similarity value into a dissimilarity value and vice versa. The notion of proximity usually refers to similarity or dissimilarity. A specific dissimilarity form (distance) and a specific similarity form (kernel) are defined in the following.

Distance function

A distance function over the domain $\mathcal{X}$ is a pairwise function $d(\cdot,\cdot)$: $\mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ which satisfies the following properties:

- Reflexivity: $d(\mathbf{x}_i, \mathbf{x}_i) = 0$
- Definiteness: $d(\mathbf{x}_i, \mathbf{x}_j) = 0 \Rightarrow \mathbf{x}_i = \mathbf{x}_j$
- Nonnegativity: $d(\mathbf{x}_i, \mathbf{x}_j) \geq 0$
- Symmetry: $d(\mathbf{x}_i, \mathbf{x}_j) = d(\mathbf{x}_j, \mathbf{x}_i)$
- Triangle inequality: $d(\mathbf{x}_i, \mathbf{x}_k) \leq d(\mathbf{x}_i, \mathbf{x}_j) + d(\mathbf{x}_j, \mathbf{x}_k)$

When all the previous properties are respected except for the definiteness, the function is called a pseudo distance.

Kernel function

A symmetric similarity function $K(\cdot,\cdot)$ is a kernel if $K(\cdot,\cdot)$ can be written as an inner product in Hilbert space $\mathcal{H}$ with the mapping function $\Phi : \mathcal{X} \rightarrow \mathcal{H}$:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \tag{2.1}$$

Equivalently, $K(\cdot,\cdot)$ is a kernel function if it is positive semi-definite (p.s.d):

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \tag{2.2}$$

for all finite sequences $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathcal{X}$ and $a_1, \ldots, a_n \in \mathbb{R}$ ([23]).

The most popular kernels in the literature are the Linear kernel, the Polynomial kernel and the Gaussian Radial Basis Function (RBF) kernel. :

$$\begin{aligned} \textit{Linear kernel} : K_{lin}(\mathbf{x}_i, \mathbf{x}_j) &= \mathbf{x}_i^T \mathbf{x}_j \\ \textit{Polynomial kernel} : K_{poly}(\mathbf{x}_i, \mathbf{x}_j) &= (\mathbf{x}_i^T \mathbf{x}_j + 1)^p \\ \textit{Gaussian kernel} : K_{gau}(\mathbf{x}_i, \mathbf{x}_j) &= exp(-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma^2}) \end{aligned} \tag{2.3}$$

where p is the hyperparameter (degree) of polynomial kernels; σ is the hyperparameter (bandwidth parameter) for gaussian kernel.

Compared to the term of distance or kernel, dissimilarity and similarity are more general terms, which do not have the constraints to be a metric or positive semi-definite. A dissimilarity can be asymmetric, non-PSD or can violate the triangle inequality ([182]).

2.2.1 Metric learning

The goal of metric learning is to learn a distance metric function $d(\mathbf{x}_i, \mathbf{x}_j)$, for all $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{T} \times \mathcal{T}$, which mostly consists in estimating some parameters from the data, in order to make a generic distance function suit the best possible to some constraints defined by the ground truth. The most popular form of metric learning methods is based on the Mahalanobis distance due to its simplicity and nice interpretation in terms of a linear projection ([24]).

Mahalanobis distance

$$d_M(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T \mathbf{M} (\mathbf{x}_i - \mathbf{x}_j)} \tag{2.4}$$

Originally from [161], the Mahalanobis distance is a measure that takes into account the correlation between features. The Generalized Mahalanobis Distance (GMD) formula is shown in Equation (2.4): when $\mathbf{M} = \mathbf{\Omega}^{-1}$ (where $\mathbf{\Omega}$ is the covariance matrix), Equation (2.4) represents the original Mahalanobis distance. Most metric learning methods strive to estimate the parameter $\mathbf{M}$

from a training set, and with the goal to make the resulting distance measure reflect the output, e.g. the class membership for classification tasks. According to the survey of [24], the positive semi-definite matrix $\mathbf{M}$ is mostly learned from pair constraints or triplet constraints, which usually have the following form:

- Must-link / cannot-link constraints (sometimes called similar set / dissimilar set):

  $S_p = \{(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ should be similar}\}$

  $D_p = \{(\mathbf{x}_i, \mathbf{x}_j), \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ should be dissimilar}\}$

- Relative constraints (sometimes called training triplets):

  $R_t = \{(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k), \mathbf{x}_i \text{ should be more similar to } \mathbf{x}_j \text{ than to } \mathbf{x}_k\}$

All the methods in metric learning try to optimize at least one of the following objectives ([148]):

$$\min_{\mathbf{M}} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in S_p} d_M(\mathbf{x}_i, \mathbf{x}_j) \tag{2.5}$$

$$\max_{\mathbf{M}} \sum_{(\mathbf{x}_i, \mathbf{x}_j) \in D_p} d_M(\mathbf{x}_i, \mathbf{x}_j) \tag{2.6}$$

The optimization can be written in a more general way:

$$\min_{\mathbf{M}} l(\mathbf{M}, S_p, D_p, R_t) + \lambda R(\mathbf{M}) \tag{2.7}$$

where $l()$ is a loss function to measure the loss when specified constraints are violated; λ is the parameter used to control the regularizer $R(\mathbf{M})$. Generally speaking, the Mahalanobis metric learning formulations differ by the choice of constraints, loss function and regularizer ([24]).

One simple example for the illustration of metric learning is shown in Figure 2.1: the original instances in the Euclidean space are shown on the left pane. With constraints of making instances from the same class closer and instances from different class farther, the metric learning objective is to get instances on the right pane with fewer violations of constraints.

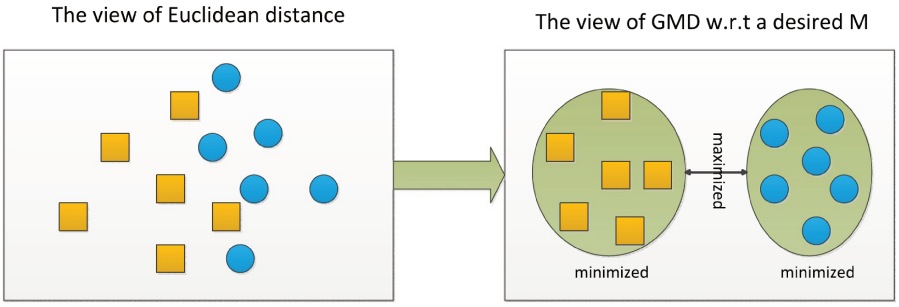


FIGURE 2.1: Illustration of metric learning from a simple toy example. For simplicity, there are only two classes, orange squares and blue circles. Pairwise constraints (left pane) are then to group the orange squares and group the blue circles while making these two groups far away. We wish to adapt the metric so that there are fewer constraint violations on the right pane. This figure is extracted from [148].

There are several surveys about metric learning in the literature [24, 135, 168, 236, 252]. Among various distance metric learning methods, the most used include Relevant Component Analysis (RCA [19, 208]), Information-Theoretic Metric Learning (ITML [77]), Large Margin Nearest Neighbors (LMNN [80]), Multiple Metrics LMNN [243, 244]. RCA uses S_p constraints only to minimize the sum of distances between each point and its corresponding class center. LMNN uses both S_p and R_t constraints to optimize the intra-class distance. In addition to that, Multiple Metrics LMNN learns multiple metrics for each class ([148]). ITML defines both S_p and D_p with the use of LogDet divergence regularization, which is used in several other Mahalanobis distance learning methods too ([124, 190]).

The Mahalanobis distance based metric learning requires at least $O(m^2)$ parameters, which is not very suitable for instances in high dimensional space. Data preprocessing techniques such as PCA or random projections are used when feature dimension approaches several hundreds ([102, 148, 152, 253]). However, the use of these preprocessing methods will cause an important information loss and decrease the interpretability of the model and affects the accuracy.

2.2.2 Kernel learning

Kernel functions normally have the form of Equation 2.1, representing the inner product of two vectors (see in the beginning of 2.2). If these two vectors are unit vectors, the inner product also represents the angle between these two vectors. Therefore, a kernel is also often interpreted as the similarity between instances and kernel learning is often seen as similarity learning ([1]). Unlike distance metric learning methods, the biggest constraint of learning a kernel is the positive semi-definiteness ([129]). According to [1], there are generally three families of kernel learning: data-dependent, nonparametric and parametric.

Data-dependent kernel learning

For using the polynomial and the RBF kernel, both hyperparameters p and σ need to be set a priori and are sometimes quite complicated to be tuned ([239]). In contrast, data-dependent kernels such as Fisher kernel ([123, 230]) or marginalization kernel ([116]) strive to learn the parameters of the kernel from training instances with generative models. However, the positive semi-definiteness still needs to be proven mathematically for any data-dependent kernel, which sometimes constitutes an obstacle for the development of new kernels of the kind, and which also makes this approach less popular than using pre-defined kernels.

Nonparametric kernel learning

In contrast to metric learning methods, nonparametric kernel learning does not use any prior model. The kernel matrix is learned without a pre-defined kernel form that implicitly generates it ([117, 118, 194]). The objective function of the nonparametric kernel learning is usually expressed as a set of user-defined criteria that aim at finding the best kernel. The downside of these methods is that during testing, the best kernel must be built from the training and test examples, which makes them impossible to be applied to new instances.

Parametric kernel learning

Most of the available approaches in the literature of kernel learning lie in this family. Parametric kernel learning is usually formed as an optimization problem to find the parameters of a predefined model with respect to the user-defined criteria ([1]). Methods in this family can be divided into two categorizes based on the number of predefined kernels used:

- Single kernel: With a single predefined kernel, the objective of parametric learning is to improve the base kernel and make it optimal for the learning task ([2, 10, 55]). There are generally two ways to realize this objective: the first one is to find the appropriate hyperparameters for the kernel and the second one is to find the transformation from the base kernel to the optimal kernel. Kernel alignment ([66, 239]) methods are very often used in this category.
- Multiple kernels: With a set of available kernels, the objective of multiple kernel learning is to find the best combination of these kernels linearly or non-linearly ([15, 17]). Multiple kernel learning methods are the most popular solutions in this category, which have been introduced in the previous chapter.

2.2.3 Random partitions

Most of the methods presented so far in this chapter follow the same core principle: optimizing some parameters of a predefined generic model, with respect to some constraints. Random partitions adopt a different approach in the sense that the method strives to infer the model from the training instances only, without any prior formulation of the measure. The key idea of Random Partitions is to define multiple randomized partitions of the input space that group the instances according to their class membership. It has been proven that such random partitions can be used to define kernels, and as a consequence to define dissimilarity measures.

Given a dataset $\mathcal{T}$, a cluster $\mathbf{C}$ is a non-empty subset of $\mathcal{T}$. The partitions of $\mathcal{T}$ divide $\mathcal{T}$ into multiple non-overlapping clusters $\varrho = \{\mathbf{C}_1, \ldots, \mathbf{C}_N\}$ with respect to:

$$\begin{aligned} \mathbf{C}_i \cap \mathbf{C}_j &= \emptyset \\ \bigcup_i \mathbf{C}_i &= \mathcal{T} \end{aligned} \tag{2.8}$$

A random partition of $\mathcal{T}$ is a sample from the partition distribution $\mathcal{P}$, where $\mathcal{P}$ is a discrete probability density function (pdf) that represents how likely a given clustering is ([76]). For any instance $\mathbf{x}$, $\varrho(\mathbf{x})$ gives the cluster $\mathbf{x}$ belongs to.

A lot of works have been done on random partitions especially in the field of non-parametric Bayesian statistics [7, 141, 184, 199]. In recent years, the relation between random partitions and dissimilarity measures has been highlighted. In [76], the authors find out that kernels can be generated from random partitions. Given a random partition distribution $\mathcal{P}$, a kernel can be defined as:

$$K_{\mathcal{P}} = \mathbb{E}[I[\varrho(\mathbf{x}_i) = \varrho(\mathbf{x}_j)]]_{\varrho \sim \mathcal{P}} \tag{2.9}$$

Random partitions can be easily constructed from existing machine learning methods. For example, any clustering algorithm such as K-means ([160]) or DBSCAN ([86]) can be transformed to stochastic clustering algorithm by adding the randomness (different initializations, number of clusters, feature projections, etc.). The output of stochastic clustering algorithm is a random partition. Another example is decision tree based ensemble methods. The nodes of a decision tree divide instances into different non-overlapping partitions, which makes the ensemble of trees such as Random Forest ([35]), boosted decision trees ([94]) or bayesian additive regression trees ([58]) a natural solution to construct random partitions. A simple kernel generated from the random partitions of Random Forest is given in [76] by generating the random partition from the tree depth sampled randomly of the trained decision tree.

The similarity measure generated from random partitions can be easily transformed to the dissimilarity measure. In [81], the authors proposed a partitioning clustering procedure with bootstrap learning sets to improve the accuracy of a given clustering procedure. M bootstrap sets are used to create a dissimilarity matrix: for each bootstrap dataset, if object i and object j belong to the same cluster, the similarity between them $\mathbf{S}_{ij}$ adds 1. At last the dissimilarity matrix is obtained by:

$$\mathbf{D}_{ij} = 1 - \frac{\mathbf{S}_{ij}}{M} \tag{2.10}$$

2.2.4 Discussion

In this section, different dissimilarity learning methods have been introduced. Generally speaking, most distance metric learning methods are based on Mahalanobis distance learning due to its simplicity and nice interpretation, but these methods require some pre-specified free parameters, and most of them involve some expensive computational procedures such as eigenvalue decomposition or semi-definite programming. Although a lot of metric learning methods have been proposed and shown to perform well in many different applications, few of them try to deal with HDLSS problem ([24, 148, 252]). Since most methods learn $O(m^2)$ parameters , metric learning methods are intractable for real-world high dimensional applications. Kernel learning learns the dissimilarity measure in the form of a kernel which can be seen as a similarity function. However, kernel learning methods usually require a large labelled dataset either for defining an ideal kernel or for cross validation ([1]). Compared to these two methods, dissimilarity learning based on random partitions is more flexible in the way that it can learn both a dissimilarity measure and a similarity (kernel) measure.

To deal with HDLSS problem, Random Forest Dissimilarity (RFD) is introduced in the next section. RFD allows to overcome the aforementioned drawbacks as it is proved to be particularly robust to high dimensions and as it does not require an exponential amount of training instances.

2.3 Random Forest dissimilarity measure

2.3.1 Random Forest

Random Forest (RF) has been a very popular data mining and statistical tool for years due to its transparency and great success in classification and regression tasks as well as in unsupervised learning ([4, 209]) or active learning tasks ([101, 162]). In the past 15 years, it has shown to be among the most accurate general purpose machine learning methods on a wide variety of real-world problems as illustrated by the consequent experimental comparison in [90]. One other important property of RF is that it does not overfit if there are enough trees in the ensemble according to Breiman's work ([35]). We now recall the RF principle.

The name "Random Forest" in this chapter refers to Breiman's work in [35]. The algorithm works by growing M different (randomized) trees with the following rules ([26]). Firstly, a bootstrap sample is generated for each tree by randomly selecting n instances, with replacement, from the initial training set made up of n different instances. Each of these bootstrap samples is then used to build one tree. During this induction phase, at each node of the tree, a splitting rule is designed by selecting a feature over $mtry$ features chosen uniformly at random among the m initial features. This selection can be performed by maximizing the well-known Gini impurity criterion. At last, while the induction of a single tree is usually prematurely stopped by a stopping criterion, e.g. a minimum number of training instances in the node, Random Trees in Random Forest classifier are grown to their maximum depth. As for the final prediction of the RF, it is obtained via majority voting over the component trees ([26]). The resulting Random Forest classifier with M decision trees is typically noted as:

$$\mathbf{H}(\mathbf{x}) = \{h_k(\mathbf{x}), k = 1, \ldots, M\} \tag{2.11}$$

where $h_k(\mathbf{x})$ is a random tree grown using the process discussed above. We refer the reader to [26, 35] for more details about this procedure. Note however that there exist many different RF learning methods that differ from the

one in [35] by the use of different randomization techniques for growing the trees. We choose to use this reference method since it is the most commonly used in the literature and since each RF learned is mainly used to compute the dissimilarities and not only to exhibit the best accuracies.

For predicting the class of a given test instance $\mathbf{x}_i$ with a random tree, $\mathbf{x}_i$ goes down the tree structure, from its root till its terminal node. The descending path is decided by successive tests on the values of the features of $\mathbf{x}_i$, one per node. The prediction is given by the terminal node (or leaf node) in which $\mathbf{x}_i$ has landed. We refer the reader to [26] for more information about this process.

Hence if two test instances land in the same terminal node, they are likely to belong to the same class and they are also likely to share similarities in their feature vectors, since they have followed the same descending path. This is the main motivation behind using Random Forest for measuring dissimilarities between instances, by using the procedure explained in the following.

2.3.2 Random Forest Dissimilarity (RFD)

The RFD measure is inferred from a RF classifier $\mathbf{H}$, learned from $\mathcal{T}$. Let us firstly define a dissimilarity measure inferred by a decision tree $d^{(k)}$: let L_k denote the set of leaves of the k^{th} tree, and let $l_k(\mathbf{x})$ denote a function from $\mathbb{X}$ to L_k that returns the leaf node of the k^{th} tree where a given instance $\mathbf{x}_i$ lands when one wants to predict its class. The dissimilarity measure $d^{(k)}$, inferred by the k^{th} tree in the forest is defined as in Equation (2.12): if two training instances $\mathbf{x}_i$ and $\mathbf{x}_j$ land in the same leaf of the k^{th} tree, then the dissimilarity between both instances is set to 0, else set to 1.

$$d^{(k)}(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 0, & \text{if } l_k(\mathbf{x}_i) = l_k(\mathbf{x}_j) \\ 1, & \text{otherwise} \end{cases} \tag{2.12}$$

The RFD measure $d^{(\mathbf{H})}(\mathbf{x}_i, \mathbf{x}_j)$ between $\mathbf{x}_i$ and $\mathbf{x}_j$ consists in calculating $d^{(k)}$ for each tree in the forest, and in averaging the resulting dissimilarity values over

the M trees, as in Equation (2.13):

$$d^{(\mathbf{H})}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{M} \sum_{k=1}^{M} d^{(k)}(\mathbf{x}_i, \mathbf{x}_j) \tag{2.13}$$

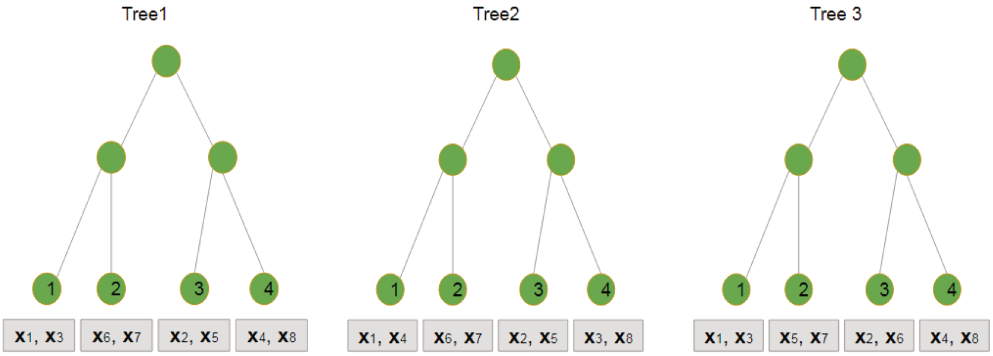


FIGURE 2.2: A simple example to calculate the Random Forest dissimilarity

Let us take a simple example shown in Figure 2.2: Assume we have built a forest with M=3 trees, each tree has 4 terminal nodes; given 8 instances $\{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_8\}$ as input to the forest, they will finally land into the terminal nodes of each tree. Here n is 8, so the dimension of dissimilarity matrix is $8x8$. Take samples $\mathbf{x}_1$ and $\mathbf{x}_3$ as an example, to calculate the dissimilarity, firstly use Equation (2.12) to calculate the similarity between $\mathbf{x}_1$ and $\mathbf{x}_3$ for each tree:

1. Tree1: $\mathbf{x}_1$ and $\mathbf{x}_3$ both land in terminal node 1, so $d^{(1)}(\mathbf{x}_1, \mathbf{x}_3) = 0$
2. Tree2: $\mathbf{x}_1$ lands in terminal node 1, and $\mathbf{x}_3$ lands in terminal node 4, so $d^{(2)}(\mathbf{x}_1, \mathbf{x}_3) = 0$
3. Tree3: $\mathbf{x}_1$ and $\mathbf{x}_3$ both land in terminal node 1, so $d^{(3)}(\mathbf{x}_1, \mathbf{x}_3) = 0$

Then with Equation (2.13), we can calculate the average dissimilarity between $\mathbf{x}_1$ and $\mathbf{x}_3$:

$d^{(\mathbf{H})}(\mathbf{x}_1, \mathbf{x}_3) == \frac{d^{(1)}(\mathbf{x}_1,\mathbf{x}_3)+d^{(2)}(\mathbf{x}_1,\mathbf{x}_3)+d^{(3)}(\mathbf{x}_1,\mathbf{x}_3)}{3} = 0.3333$

2.3.3 RFD matrix

Let $\mathbf{D}$ denote a $n \times n$ matrix, called a dissimilarity matrix, built from a given RFD measure d and from a training set $\mathcal{T}$, and defined as in Equation (2.14):

$$\mathbf{D} = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1n} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{n1} & d_{n2} & d_{n3} & \dots & d_{nn} \end{bmatrix} \tag{2.14}$$

where d_{ij} denotes $d(\mathbf{x}_i, \mathbf{x}_j)$, for all $(\mathbf{x}_i, \mathbf{x}_j) \in \mathcal{T} \times \mathcal{T}$.

$\mathbf{D}$ is non-negative and respects the reflexivity condition. Such a dissimilarity matrix can be viewed as a new training set, where each training instance $\mathbf{x}_i$ is described by a vector $\{d_{i1}, d_{i2}, \dots, d_{in}\}$. In the same way, using the dissimilarity to each of the training instances, any new instance $\mathbf{x}_i$ can be mapped into a n dimensional dissimilarity space DS. For HDLSS problem, the dimension of this dissimilarity space is necessarily smaller than the dimension of the original feature space.

Properties of RFD matrix

In the following, we give the proof that the RF similarity matrix is symmetric and positive semi-definite. These two properties are essential since they ensure that such a matrix can be used as a kernel matrix in kernel methods like non-linear SVM ([181]). Note that RF similarity matrices are easily obtained from the RFD matrices defined in the previous section, by $\mathbf{S_H} = \mathbf{1} - \mathbf{D_H}$. RF similarity is also called the RF proximity in the literature.

Let us firstly recall the two following theorems from [147], that gives the basic conditions for determining whether a matrix is p.s.d. or not

Theorem 2.3.1 *If both $\boldsymbol{A}$ and $\boldsymbol{B}$ are two p.s.d. matrices then so is $\boldsymbol{A} + \boldsymbol{B}$. This follows immediately from the equation $\mathbf{x}^T(\boldsymbol{A} + \boldsymbol{B})\mathbf{x} = \mathbf{x}^T\boldsymbol{A}\mathbf{x} + \mathbf{x}^T\boldsymbol{B}\mathbf{x} \geq 0$. Consequently any sum of p.s.d. matrices is p.s.d.*

Theorem 2.3.2 *A symmetric binary matrix* $\boldsymbol{MA} \in (0,1)^{n\times n}$, *with* $n \geq 3$, *is p.s.d. if and only if it satisfies the following inequalities:*

$$\boldsymbol{MA}_{ij} \leq \boldsymbol{MA}_{ii}, (1 \leq i < j \leq n) \tag{2.15}$$

$$\boldsymbol{MA}_{il} + \boldsymbol{MA}_{jl} \leq \boldsymbol{MA}_{ll} + \boldsymbol{MA}_{ij}, (1 \leq i < j \leq n, l \neq i, j) \tag{2.16}$$

Using these theorems, let us demonstrate that the similarity matrix inferred by a single tree k, noted $\mathbf{S}^{(k)}$ is p.s.d. From Equation (2.12), one can see that $\mathbf{S}^{(k)}$ has the following properties:

- $\mathbf{S}^{(k)}$ is a symmetric matrix with principal diagonal values equal to 1.
- The off diagonal entries in $\mathbf{S}^{(k)}$ are either 0 or 1.

One can reasonably consider that the number of training instances available is greater than 3, and as a consequence, that $\mathbf{S}^{(k)}$ is a symmetric binary matrix $\in (0,1)^{n\times n}$, with $n \geq 3$. According to Theorem 2.3.2, for this matrix to be p.s.d., it needs to satisfy both Equation (2.15) and Equation (2.16):

1. As $\mathbf{S}^{(k)}$ is a symmetric binary matrix with principal diagonal values $\mathbf{S}^{(k)}_{ii}$ equal to 1, hence $\mathbf{S}^{(k)}_{ij} \leq \mathbf{S}^{(k)}_{ii}$, which satisfies Equation (2.15).
2. To prove $\mathbf{S}^{(k)}$ satisfies Equation (2.16), two situations need to be considered:
   (a) If $\mathbf{S}^{(k)}_{ij} = 1$, then $\mathbf{S}^{(k)}_{ll} + \mathbf{S}^{(k)}_{ij} = 2$. Since $\mathbf{S}^{(k)}_{il} \leq 1$ and $\mathbf{S}^{(k)}_{lj} \leq 1$, then $\mathbf{S}^{(k)}_{il} + \mathbf{S}^{(k)}_{lj} \leq \mathbf{S}^{(k)}_{ll} + \mathbf{S}^{(k)}_{ij}$.
   (b) If $\mathbf{S}^{(k)}_{ij} = 0$, then $\mathbf{S}^{(k)}_{ll} + \mathbf{S}^{(k)}_{ij} = 1$. At the same time, $\mathbf{S}^{(k)}_{ij} = 0$ means that the i^{th} and j^{th} instances fall in different terminal nodes, which implies that $\mathbf{S}^{(k)}_{il}$ and $\mathbf{S}^{(k)}_{lj}$ can not be both equal to 1. Thus $\mathbf{S}^{(k)}_{il} + \mathbf{S}^{(k)}_{lj}$ is necessarily less or equal to 1 and as a consequence, $\mathbf{S}^{(k)}$ also satisfies Equation (2.16).

This proves that $\mathbf{S}^{(k)}$ meets the requirements of Theorem 2.3.2, and is a p.s.d. matrix. It follows from Theorem 2.3.1 that the sum of all $\mathbf{S}^{(k)}, \forall k = 1..M$, is

also p.s.d., meaning the RF similarity matrix $\mathbf{S_H}$ or any linear combination of $\mathbf{S_H}$ is also p.s.d.

Multi-view learning with RFD matrix

For multi-view learning, one needs now to fuse the dissimilarity matrices built on each view and to learn a classifier from the resulting joint dissimilarity matrix.

A natural way to fuse the dissimilarity matrices is to compute the unweighted average matrix. For multi-view learning tasks, the training set $\mathcal{T}$ is composed of Q views: $\mathcal{T}^{(q)} = \{(\mathbf{x}_1^{(q)}, y_1), \ldots, (\mathbf{x}_N^{(q)}, y_N)\}$, $q = 1..Q$. From these views, Q RFD matrices are computed following Equation (2.14) and noted $\{\mathbf{D}_\mathbf{H}^{(q)}, q = 1..Q\}$. For multi-view learning, the joint dissimilarity matrix $\mathbf{D_H}$ can be computed as in Equation (2.17).

$$\mathbf{D_H} = \frac{1}{Q} \sum_{q=1}^{Q} \mathbf{D}_\mathbf{H}^{(q)} \tag{2.17}$$

According to the work in [83], learning from a dissimilarity matrix $\mathbf{D_H}$ can be done in two different ways: (i) by using the corresponding similarity matrix $\mathbf{S_H} = \mathbf{1} - \mathbf{D_H}$ as a kernel matrix in a kernel-based learning method, e.g. a SVM classifier (named RFSVM in the following and illustrated in Figure 2.3a) and (ii) by using the dissimilarity matrix $\mathbf{D_H}$ as a new training set (named RFDis in the following and illustrated in Figure 2.3b).

Multi-view Random Forest kernel SVM (RFSVM): Instead of using traditional kernels, such as the Gaussian Radial Basis Function kernel, SVM classifiers can be efficiently trained on user-defined kernels. For example, in [108], the authors proposed a problem dependent distance measure to construct a substitution Gaussian kernel. Such a user-defined kernel can be supplied to SVM classifiers as a kernel matrix as long as it is positive semi-definite (p.s.d). For RFSVM, the joint similarity matrix $\mathbf{S_H}$ is used as a kernel matrix. The proof that it is p.s.d. has been given in the previous section. Then, given a test

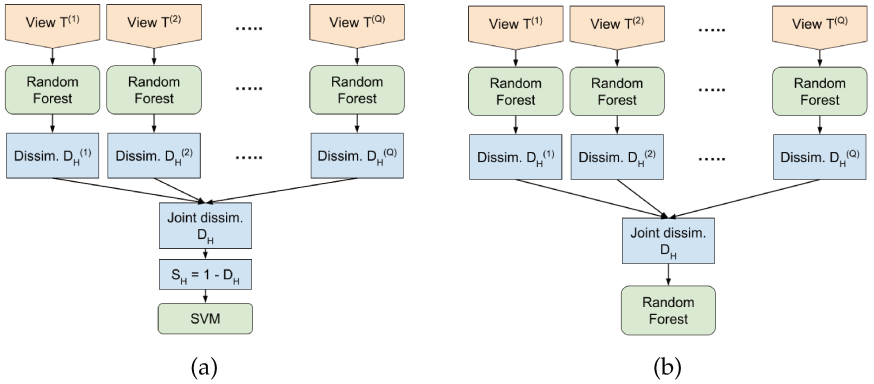


FIGURE 2.3: Flowchart of (a) the RFSVM method and (b) the RFDis method

instance $\mathbf{x}_t$, the joint RF similarity vector $\mathbf{s}_t$, which contains the average similarities between the test instance and each training instance among different views, is given to the trained SVM for prediction.

Multi-view Random Forest dissimilarity (RFDis): RFDis consists in learning an RF classifier $\mathbf{H}$ as if $\mathbf{D_H}$ was a new training set. The joint dissimilarity vector is seen as a feature vector, and an RF classifier is built on these new features. In other words, the original data are projected from the feature space to the dissimilarity space. In this case, the dimension of the dissimilarity space is the number of instances in the training set. For HDLSS problem, it will necessarily reduce the feature dimension without feature decimation. Given a test instance $\mathbf{x}_t$, the joint RF similarity vector $\mathbf{s}_t$, which contains the average similarities between the test instance and each training instance among different views, are given to the RF classifier for prediction.

2.4 The parametrization of RFD

The RF learning algorithm, whether it is used for computing a dissimilarity or not, is controlled by important hyperparameters. While most of these hyperparameters have been extensively studied when RF is used as a classifier, their influence on the quality of the RFD measure is not that clear. In particular, the

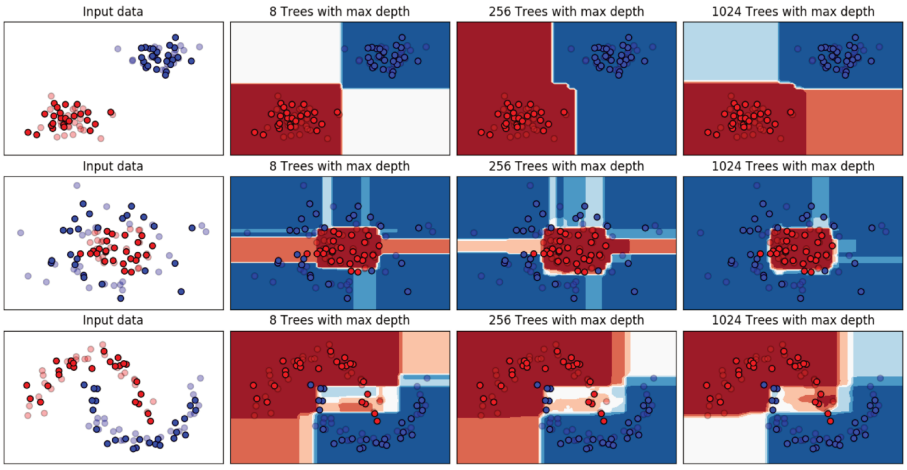
following hyperparameters are assumed to be crucial for having a "good" RFD measure:

- **Forest size** M: As explained in [25, 35], it is now known that the RF accuracy converges for an increasing number of trees in the forest. One can naturally wonder if the same goes for the quality of the corresponding RFD measure. As explained in the previous section, RFD between two instances is computed by averaging over the dissimilarity values inferred by each tree. If the number of trees is very small, say 5 trees for example, the RFD estimate would always be one of the 5 following values: 0, 0.2, 0.4, 0.6, 0.8 or 1. Obviously, this is not accurate enough for describing (dis)similarities between instances. When the number of trees increases, the RFD value is expected to be more accurate and reliable.

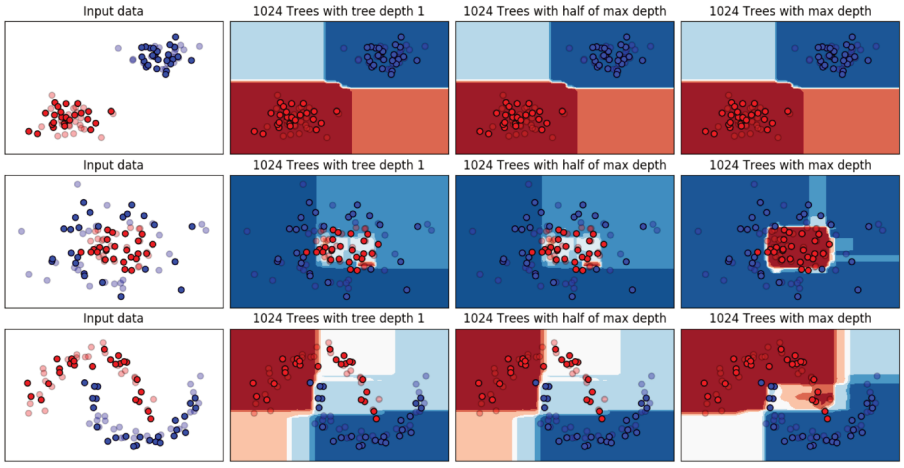
- **Tree depth** δ: The rationale behind studying the influence of the tree depth on the RFD quality is less obvious. When the node is deeper down the tree structure, it is usually more "pure", that is to say it gathers training instances from the same class mostly. This is desirable since it means the RFD values will reflect the class membership: two instances from the same class will be considered quite similar. However, at the same time, the deeper the node, the smaller it will be, that is to say the fewer instances it will gather. As a consequence, the resulting RFD matrix is likely to be sparse, and the dissimilarity measure too loose.

To illustrate the influence of these hyperparameters, an RF classifier is built on the dissimilarity matrix induced from different combinations of numbers of trees and tree depths on three toy datasets. The three toy datasets, composed of 100 instances, two features and two classes, have different shapes and complexities, as shown in the first column of Figures 2.4a and 2.4b: (i) the first row is a dataset with two isotropic Gaussian classes to show how the RFD measure behaves differently from a traditional Euclidean distance measure; (ii) the second row is a donut-shaped dataset, more complex with regards to similarity measures because, contrary to the RFD measure, a distance-based dissimilarity would fail to represent the class membership; and (iii) the third row is a banana-shaped dataset used to confirm that the RFD measure can

properly take into account the class membership to estimate dissimilarities. The decision frontiers given by these RF are shown in the last three columns of Figure 2.4a and Figure 2.4b.



(a) Influence of number of trees on decision boundaries with maximum depth.



(b) Influence of tree depth on decision boundaries with 1024 trees.

FIGURE 2.4: Influence of the hyperparameters on the decision frontiers for 3 toy datasets. The transparent points are the training instances and the opaque points are the test instances. Note that in the sub-figure (b), the decision boundaries do not change in the first row because the maximum depth is 1.

Figure 2.4a shows the influence of the number of trees when their depth is set to the maximum. For the dataset (i) (first row), 8 trees are enough to achieve good performance. But for the two other datasets, the influence of the number of trees can be better highlighted. For an increasing number of trees in the forest, the quality of RFD gets better as shown in Figure 2.4a, and it is also reflected on the decision frontier. It can be seen that for both datasets, the decision frontiers better suit to the classes (i.e. describe more and more correctly the data structure) when the number of trees increases. Therefore if we want the RFD measure to be accurate for each of the cases, it is necessary to have as many trees as possible.

Figure 2.4b shows the influence of the tree depth for a forest of 1024 trees. For the dataset (i), there is no difference in the three scatter plots because the maximum depth is equal to 1. For the two other toy datasets, it can be seen that the decision frontiers are sharper and better fit the training set when the tree is deeper. In particular, when the tree depth is not maximum, the decision boundaries are not sharp enough: this is because, in this case, the trees fail to capture the class membership of similar instances.

In summary, these results show that if we want the RFD measure to be accurate, it is necessary to have the maximum tree depth, with a large number of trees in the forest.

2.4.1 Experiments on real-world datasets

To confirm the trends observed on the toy datasets, the hyperparameters are further studied on real-world multi-view datasets. A general description of these datasets can be found in Table 2.1. The first four datasets are Radiomics problems. The others 11 datasets are non-Radiomic datasets but relate to similar HDLSS multi-view applications. For Radiomic datasets, there are 5 views for each of these 4 datasets: 4 texture feature groups from axial T1-weighted MR images before and after gadolinium-based CE material administration as well as axial T2-weighted and axial T2-weighted fluid attenuated inversion recovery (FLAIR) images. The fifth view are Visually AcceSAble Rembrandt Images (VASARI) features ([104]). More details about the Radiomic

datasets can be found in [258]. As for the non-Radiomic datasets: LSVT is a dataset on vocal performance degradation of Parkinson's disease subjects ([226]); Metabolomic contains biomarkers (CEA and TIMP), fluorescence concentration (PF) and NMR profiles for early detection of colorectal cancer ([37]); BBC and BBCSport are text classification problems constructed from the news article corpora by splitting articles into related segments of text ([249]); the remaining datasets (Cal7 and 20, Mfeat, NUS-WIDE2 and 3, and AWA8 and 15) are classical image classification datasets obtained using different feature extractors. Similar to [151], these latter datasets have been randomly downsampled to simulate the HDLSS setting.

All these datasets are multi-view datasets, that is to say they are supplied with several views of the same instances. However, as the goal of this first experiment is to study the effect of hyperparameters on the quality of the RFD measure, we considered the 71 views (coming from the 15 datasets) separately, as independent datasets. The reason we decided to use multi-view dataset for this experiment is to be able to re-use the same datasets in our next experiments.

Both hyperparameters M and δ have been tested with the following values:

- Forest size $M \in \{8, 16, 32, 64, 128, 256, 512, 1024\}$: first, an RF with 1024 trees is built; the performance is then monitored with the first 8 trees, the first 16 trees, and so on, until all the 1024 trees are used in the RF. Recall that for training a Random Forest, trees are grown independently from each other. Therefore, retaining a subset of trees in a forest already built is just a mean to save computation time.
- Tree depth δ : an RF is firstly built with fully grown trees. For each RF, the maximum tree depth δ_{max} is computed. Then, the quality of the RFD is measured by only considering nodes above depth $i \in \{1, 2, 3, \ldots, \delta_{max}\}$, that is to say by considering that each branch of each tree has not been grown beyond depth i.

Following the conclusion of [83], the quality of the RFD measure obtained with different combinations of these hyperparameters is now assessed with a 1-Nearest Neighbor classifier (1NN): 1NN selects the nearest neighbor for a test instance according to the dissimilarity values and assign the label of

TABLE 2.1: Overview of the real-world datasets used in our experiments. IR (imbalanced ratio) is the number of instances of the majority class over the number of instances of the minority class.

	#features	**#instances**	**#views**	**#classes**	**IR**
nonIDH1[258]	6746	84	5	2	3
IDHcodel[258]	6746	67	5	2	2.94
lowGrade[258]	6746	75	5	2	1.4
progression[258]	6746	84	5	2	1.68
LSVT[226]	309	126	4	2	2
Metabolomic[37]	476	94	3	2	1
Cal7[151]	3766	1474	6	7	25.74
Cal20[151]	3766	2386	6	20	24.18
Mfeat[93]	649	600	6	10	1
BBC[249]	13628	2012	2	5	1.34
BBCSport[249]	6386	544	2	5	3.16
NUS-WIDE2[60]	639	442	5	2	1.12
NUS-WIDE3[60]	639	546	5	3	1.43
AWA8[143]	10940	640	6	8	1
AWA15[143]	10940	1200	6	15	1

the nearest neighbor to the test instance. This method can well reflect the quality of the dissimilarity matrix, because the idea behind 1NN is that the most similar instances should belong to the same class. A stratified random splitting strategy has been used to obtain a robust estimate of the performance of these 1NN classifiers. Each dataset has been randomly split 50 times, with 50% of the instances for training and 50% for test. A grid search has been performed on M and δ over the 50 random splits.

2.4.2 Results on real-world datasets

The results on the 71 views are presented in this section as mean and standard deviations of the classification rates over the 50 runs. To better illustrate the results, a 2D color-map is drawn for each dataset, as in Figure 2.5. The warm color (yellow) stands for a relatively high quality of the RFD as measured by the 1NN accuracy while the cold color (blue) stands for relatively low quality. The y-axis corresponds to the number of trees, and the x-axis corresponds

to the tree depth. For clarity concerns, only four examples are shown in Figure 2.5. However, these four color-maps have been chosen as they are good representatives of all the results we have obtained in this experiment.

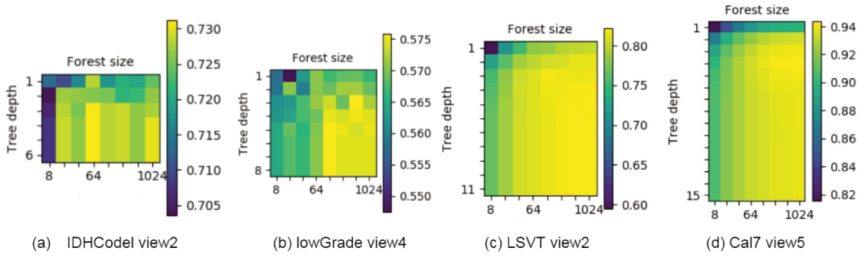


FIGURE 2.5: Color-maps of 4 of the 71 views.

Additionally, a statistical test of significance has been performed over the 71 views to state whether or not the differences observed on these color-maps are statistically significant. However, also for clarity concerns, only nine combinations of M and δ have been considered among all the possible combinations shown on the color-maps. Table 2.2 sums up these nine parameterization settings, that have been chosen according to the conclusions drawn on the toy datasets in the previous section. The statistical test used in this experiment is the Nemenyi post-hoc test with Critical Differences (CD), as recommended in [79].

TABLE 2.2: The 9 combinations chosen for the statistical test

Name	Tree depth	Number of trees
minmin	1	8
minhalf	1	128
minmax	1	1024
halfmin	$\delta_{max}/2$	8
halfhalf	$\delta_{max}/2$	128
halfmax	$\delta_{max}/2$	1024
maxmin	δ_{max}	8
maxhalf	δ_{max}	128
maxmax	δ_{max}	1024

The results of the statistical test are shown in Figure 2.6. Over all the 71 views, the best rank is achieved by the *maxmax* setting, followed by *halfmax*, *maxhalf* and *halfhalf*. The performance of *maxmax*, *halfmax*, *maxhalf* and *halfhalf* are significantly better than the performance of *minmin*, *minhalf*, *minmax*, *halfmin* and *maxmin*, which confirms the observations made from the color-maps in Figure 2.5: the bottom-right corner (maximum trees, maximum depth) globally corresponds to better accuracies than the top-left corner (minimum trees, minimum depth). One can conclude that with a large number of trees, all fully grown to their maximum depth, the quality of the resulting RFD measure is guaranteed to be close to the best possible.

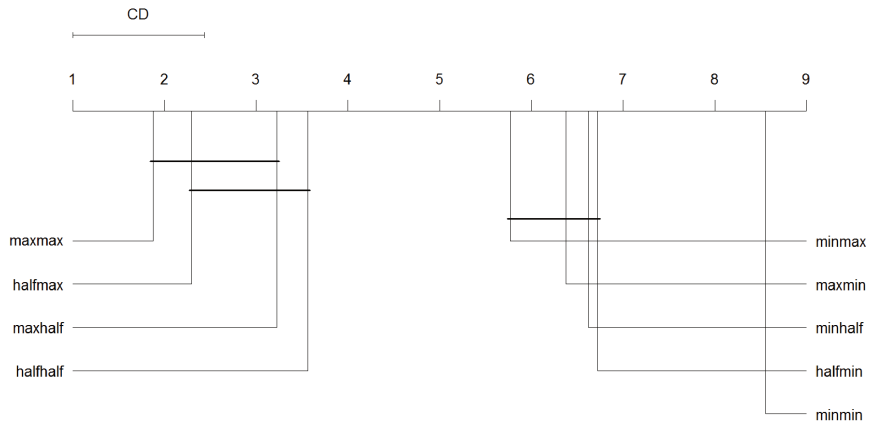


FIGURE 2.6: The CD diagram according to the Nemenyi post hoc test result when $\alpha = 0.05$.

Note however that one can see small differences in some of the views tested, as illustrated in Figure 2.5: from Figures 2.5 (c) and (d), it is clear that the worst RFD measure is obtained from forests with very few trees and minimum tree depth, while the best RFD measure is obtained with very large forests and maximum tree depth. In contrast, this trend is not as clear in Figures 2.5 (a) and (b); the worst results are not clearly located at top-left corner of the color-map, and the best results are neither located exactly at bottom-right corner. This may be due to the fact that the views on the left (Figure (a) and (b)) have a very small number of instances (67 and 75 instances respectively) and that the resulting learning phase is inherently less reliable for these cases. However,

even with very few instances for learning, the trend is still observable on these figures.

2.4.3 Discussion

From this study on the parametrization, one can see that the general trend for all datasets is similar: the RFD measure is more reliable when the RF contains more trees and when these trees are fully grown. From the overall comparison on the real-world datasets, the *maxmax* setting (1024 trees with maximum depth) appeared to be better than the *maxhalf* setting (128 trees with maximum depth) but not statistically significantly, which means that 128 trees already allow to obtain a quite good RFD measure for most of the views. For a better insight into this, Figure 2.7 shows the result of the Nemenyi post-hoc test when focusing on the number of fully grown trees. It shows that the performance gaps for forests from 256 to 1024 trees are not statistically significant. However, these differences in terms of average ranks, observable on this figure, are still important enough from our point of view to consider using more than 256 trees. It is worth noting that the computational cost of learning an RF classifier is directly proportional to the number of trees ([157]). Hence, in all the remaining experiments, all the RF have been learned with 512 trees as a good compromise between reliability and computational costs. And of course, all the trees have been fully grown all along these experiments.

2.5 RFD-based multi-view learning

As far as we know, only one method has already been proposed that perform learning from a joint RF dissimilarity matrix: the method in [99], named RFMDS in this chapter. This method is similar to RFDis except for two aspects. First, the computation of the joint similarity matrix differs in that, for RFSVM and RFDis, it is an unweighted average combination whereas in RFMDS, it is a linear combination with weights optimized through a coarse-grained grid search. Second, RFSVM works in a kernel space and RFDis works in a dissimilarity space, whereas RFMDS works in an embedded space obtained with

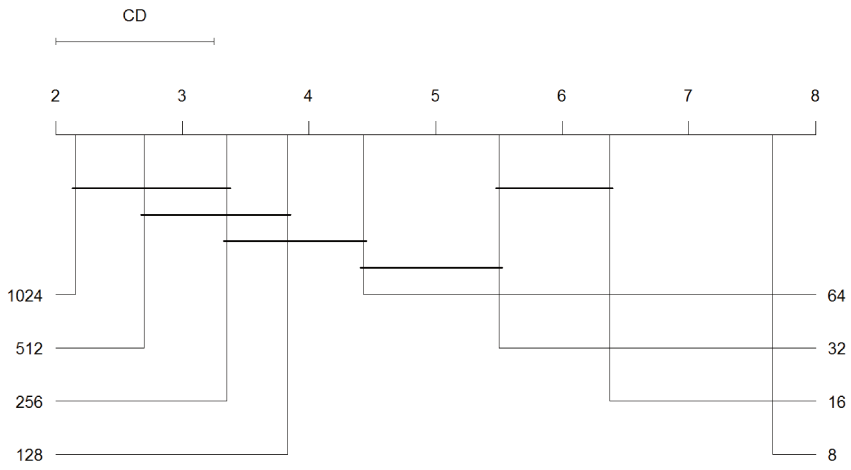


FIGURE 2.7: Comparison of RFD results with different numbers of trees and with maximum tree depth, using Nemenyi test with Critical value (CD) when $\alpha = 0.05$.

the Multi-Dimensional Scaling (MDS). MDS projects data from the original feature space to a low dimensional embedded space where the distances between instances are preserved, which is often used for data visualization and dimensionality reduction. These three methods, RFSVM (kernel space), RFDis (dissimilarity space) and RFMDS (embedded space) are compared to other state-of-the-art methods on different HDLSS multi-view learning problems in the next section.

2.5.1 Experimental protocol

In this second set of experiments, the two RFD-based multi-view learning methods presented in the previous section are compared to several state-of-the-art methods, from multi-view learning literature. The datasets used in this experiment are the real-world datasets used in the preliminary study (cf. Table 2.1) but the data are now treated as multi-view. Let us recall that all these datasets are multi-view datasets, that is to say they are made up with several views (description spaces) of the same instances.

For this experimental validation, six methods are compared: one state-of-the-art Radiomics solution based on feature selection, namely SVM-RFE ([238, 257]); four intermediate integration methods, i.e. the proposed RFSVM and RFDis presented in section 2.5, the RFMDS method proposed in [99] and the MKL method EasyMKL ([6]); and one RFD-based late integration method, namely LateRFDis, from [44]. This latter method is a basic MCS architecture, which firstly builds an RFD matrix on each view, then trains an RF classifier on each of these dissimilarity matrices, and finally combines these RF classifiers by majority voting (However, we also explore the potential of some more complex late integration methods and the results can be found in Appendix B). All these methods are sum up in Table 2.3.

TABLE 2.3: An overview of all the methods compared in this work

Methods	**Integration Method**	**Learning space**
SVM-RFE [257]	Early	Reduced feature space
RFSVM	Intermediate	Kernel space
RFDis	Intermediate	Dissimilarity space
RFMDS [99]	Intermediate	Embedded dissimilarity space
EasyMKL [6]	Intermediate	Kernel space
LateRFDis	Late	Dissimilarity space

For the SVM-RFE method, the number of features to select, which is a hyperparameter of the method, is set according to the total number of features following the rules described in [30]. An RF classifier is then built from the selected features. For all the RF classifiers used in this experiment, the number of trees is set to 512 as explained in section 2.4.3, while the other parameters are set by default as proposed in the *Scikit-learn* machine learning framework ([178]). As for the SVM based method, the usual hyperparameter C is used to define the penalty factor. Its value is classically set using a grid search with cross-validation. For EasyMKL, a similar grid search with cross-validation strategy is used to find the best combination of kernels among a pool of linear kernels, gaussian kernels and polynomial kernels with different hyperparameters, following the protocol given in [195].

Finally, similar to the preliminary study, a stratified random splitting procedure is repeated 10 times, with 50% of the instances for training, 50% for testing. In order to compare the methods, the mean and standard deviations of accuracy are evaluated over 10 runs.

2.5.2 Results and discussions

The results of this experimental comparison are given and discussed in the following, firstly for non-Radiomic datasets and secondly on the Radiomic datasets.

Results on non-Radiomic datasets

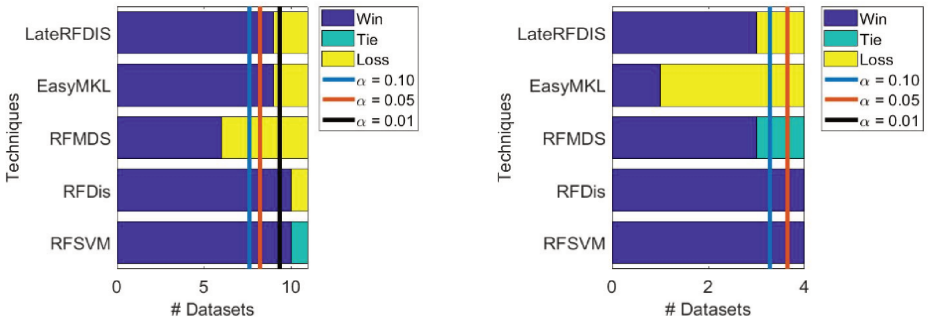
The average classification rates over the 10 repetitions, along with standard deviations, are shown in Table 2.4. From the average ranking, it can be seen that the RFSVM method performs globally the best among the six methods, followed by the RFDis and EasyMKL methods, while the state-of-the-art Radiomics solution (i.e. SVM-RFE) is ranked the worst.

Comparison of the multi-view solutions and the feature selection method: From Table 2.4, one can see that all the multi-view methods are globally better than the feature selection method SVM-RFE. To better assess the difference, a pairwise analysis based on the Sign test is computed on the number of wins, ties and losses as in [73]. The result is shown in Figure 2.8 (a). All the multi-view solutions are compared to SVM-RFE: each vertical line indicates the critical value corresponding to a confidence level α equal to 0.10 and 0.05. If the number of wins is above these lines, the corresponding method can be considered to be significantly better than the baseline method. Figure 2.8 (a) shows that except for RFMDS, all the methods are significantly better than SVM-RFE with α = 0.05 and 0.01. RFSVM and RFDis are the ones that win the most against SVM-RFE.

Comparison of the RFD-based methods (RFSVM, RFDis, RFMDS): Let us recall that each of these methods exploits the RFD measure in the three different possible ways for dissimilarity-based classification according to [83, 180]:

TABLE 2.4: Classification accuracy with 50% training data and 50% test data for non-Radiomic datasets. The bold number of each row represents the best performing method.

Dataset	SVM-RFE	RFSVM	RFDis	RFMDS	EasyMKL	LateRFDis
LSVT	84.12% ±3.48	84.12% ±2.93	83.33% ±3.97	**85.07**% ±2.61	80.95% ±3.40	81.42% ±2.66
Metabolomic	63.54% ±7.53	**68.75**% ±5.10	67.71% ±5.12	67.29% ±6.94	58.33% ±5.89	64.37% ±6.55
Cal7	94.57% ±0.86	96.36% ±0.47	95.21% ±0.67	86.95% ±0.85	**96.40**% ±0.87	93.98% ±0.73
Cal20	85.06% ±1.98	88.39% ±0.34	89.12% ±0.69	64.94% ±0.52	**92.33**% ±0.59	86.15% ±0.58
Mfeat	91.13% ±4.12	**97.83**% ±0.95	97.56% ±0.99	87.90% ±2.66	95.66% ±1.54	96.56% ±1.26
BBC	74.19% ±1.76	95.63% ±0.39	92.82% ±0.67	93.37% ±1.09	**97.98**% ±0.73	88.88% ±0.50
BBCSport	78.05% ±2.88	95.56% ±0.81	81.75% ±2.70	90.69% ±2.07	**96.98**% ±0.31	81.61% ±1.96
NUS-WIDE2	91.61% ±1.26	**93.77**% ±0.85	92.49% ±2.01	92.30% ±1.50	93.15 ±0.98	91.94% ±1.28
NUS-WIDE3	76.48% ±3.07	**82.56**% ±1.78	79.41% ±1.94	80.63% ±1.34	78.33% ±1.29	78.60% ±2.26
AWA8	38.87% ±3.34	**56.90**% ±1.86	56.06% ±1.35	21.00% ±1.52	43.75% ±3.56	51.31% ±1.05
AWA15	19.31% ±1.38	37.46% ±0.78	**37.90**% ±1.49	8.2% ±0.42	24.98% ±7.90	32.35% ±1.17
Average Rank	5.04	**1.68**	2.72	4.18	3.18	4.18



(a) Multi-view methods compared to SVM-RFE on non Radiomics data

(b) Multi-view methods compared to SVM-RFE on Radiomics data

FIGURE 2.8: Pairwise comparison between multi-view solutions and feature selection methods for non-Radiomic problem. The vertical lines illustrate the critical values considering a confidence level α= {0.10, 0.05, 0.01}.

RFSVM uses a kernel space, RFDis a dissimilarity space, and RFMDS an embedded space. From Table 2.4, it can be seen that RFSVM clearly outperforms the two other methods which indicates that kernel space seems to be the best approach. Note that the RFMDS method is quite accurate on 2-class problems, but not so much for multi-class problems, while the proposed RFSVM and RFDis are as accurate for both. The reason may be that the RFMDS is based on an embedded space as explained in section 3, which may suffer from a loss of information as the dimension is reduced. This information loss is more obvious for multi-class data (Cal7: 7 classes, Cal20: 20 classes, Mfeat: 10 classes, AWA8: 8 classes, AWA15: 15 classes).

Comparison of the RFSVM method and the state-of-the-art MKL method (EasyMKL): Let us recall that RFSVM and EasyMKL both adopt the same kind of kernel-based principle. From average ranking in Table 2.4, one can see that both RFSVM and RFDis globally outperform EasyMKL. EasyMKL is accurate for most datasets except for the two medical datasets (LSVT and Metabolomic) with very small sample size (126 instances and 94 instances respectively). This stresses that the proposed RFSVM method, as well as the RFDis method, manage to better handle HDLSS datasets than the state-of-the-art MKL approach. Let us also recall that, contrary to EasyMKL, the RFSVM method does not require a greedy optimization of the kernel combination, neither requires to choose a priori the different kernel to use in the combination.

Results on Radiomic datasets

In the following, the previous analysis is confirmed on the real-world Radiomic datasets. The results are gathered in Table 2.5. By looking at the average ranking, the RFSVM method is still ranked first and the RFMDS method is ranked second. As for the feature selection method SVM-RFE, it is still ranked last.

Comparison of the multi-view solutions and the state-of-the-art Radiomics solution: From Table 2.5, one can see that all the multi-view solutions are generally better than the Radiomics solution SVM-RFE. Similar to the analysis on non-Radiomics problems, a pairwise analysis based on the Sign test

TABLE 2.5: Classification accuracy with 50% training instances and 50% test instances for Radiomic datasets

Dataset	SVM-RFE	RFSVM	RFDis	RFMDS	EasyMKL	LateRFDis
nonIDH1	76.28% ±4.39	80.69% ±2.76	79.53% ±3.57	**82.55**% ±4.55	76.04% ±2.37	80.93% ±2.51
IDHcodel	73.23% ±5.50	**76.76**% ±4.52	76.47% ±3.95	73.82% ±4.26	72.35% ±2.35	76.17% ±2.06
lowGrade	62.55% ±3.36	63.95% ±4.56	63.48% ±3.76	62.55% ±5.53	64.65% ±4.26	**65.11**% ±5.20
progression	62.36% ±3.73	**65.52**% ±4.47	63.42% ±6.49	65.00% ±5.95	59.73% ±6.00	58.94% ±6.02
Average Rank	4.875	**2.000**	3.250	3.125	4.750	3.000

is also given in Figure 2.8 (b), and the same conclusion holds: the two proposed methods, RFSVM and RFDis, significantly outperform SVM-RFE with $\alpha = 0.05$.

Comparison of the RFD-based methods (RFSVM, RFDis, RFMDS): Here again, the same conclusion goes with the Radiomic datasets: the RFSVM is still the best method, followed by the RFMDS method. The RFMDS method is still slightly better than RFDis on Radiomic datasets that are all 2-class problems, which also confirms the previous conclusion that RFMDS works well for 2-class problems.

Comparison of the RFSVM method and the state-of-the-art MKL method (EasyMKL): Table 2.5 shows in particular that the EasyMKL method has much worse performance on Radiomic problems than on non-Radiomic problems, which seems to confirm that EasyMKL hardly handles very small datasets like those found in the medical field. The proposed RFSVM and RFDis on the other side still work well for both Radiomics or non-Radiomic problems.

Discussion

From the results on both non-Radiomics and Radiomic problems, the following conclusions can be drawn: (i) in general, the multi-view solutions outperform the state-of-the-art Radiomics solution that uses feature selection in the concatenated feature space (early integration), and the differences are always statistically significant for the two proposed RFD-based methods; (ii) by comparing three different possibilities of learning with dissimilarity, learning

in kernel space seems to be a better choice for multi-view learning problems, while one can note that the RFMDS method, that uses an embedded space, is less accurate for multi-class problems; (iii) by comparing RFSVM to MKL method, one can see that even though both methods use kernels, the RFSVM method is better than MKL, especially for very small datasets like in the Radiomics application. These results also stress that the RF kernel outperforms the traditional gaussian, linear, and polynomial kernel in the HDLSS context. Let us finally recall that the RFSVM method has the strong advantage to not require the optimization of the kernel combination, neither to choose the different kernels to use beforehand.

2.6 Conclusion

In this chapter, we proposed to use RFD as the intermediate representation for multi-representation fusion method and showed its potential compared to other state-of-the-art methods. Because multi-view datasets share the class information among views, learning a proper dissimilarity measure that can reflect the class becomes necessary. Among various dissimilarity learning methods, RFD was chosen thanks to the following two reasons. Firstly, it can take advantage of both feature and class dissimilarities due to the tree structure. Secondly, it can deal well with the HDLSS problem, which makes it a better choice compared to other dissimilarity learning methods such as distance metric learning or kernel learning.

A preliminary experiment has been proposed to better understand how the RFD measure behaves according to the most important hyperparameters. We have shown that when there are more trees in the forest, and the trees are deeper, the resulting RFD estimate is more accurate. In the second set of experiments, five multi-view methods (four intermediate integration methods and one late integration method) have been compared to the Radiomics state-of-the-art method, on several HDLSS multi-view datasets, including four Radiomic datasets. The results have shown that the multi-view solutions are globally better, but only the two proposed intermediate integration methods,

namely RFSVM and RFDis, significantly outperform the state-of-the-art Radiomics solution SVM-RFE. These proposed approaches, that use two well-known principles of dissimilarity-based classification (kernel and dissimilarity spaces), have also been compared to a different kind of RFD-based method, RFMDS, that uses a third dissimilarity-based representation (embedded space), and the result shows that RFDis and RFSVM are globally better options in the HDLSS multi-view setting. Finally, the comparison has been extended with a Multiple Kernel Learning method, namely EasyMKL, known to be efficient and straightforward for application to multi-view learning. The results show that the RFSVM method is more accurate than EasyMKL while avoiding a greedy optimization for the combination of a pool of different predefined kernels. The results show that the RFD based intermediate integration methods are the most promising for the HDLSS multi-view problem. Hence, we focus more on intermediate integration in the following of this thesis.

For the two proposed methods, the RFD measure used for each view shares the same parameter settings, while we think it could be further fruitful to make the RFD measure suit to the specificities of each view. Therefore, we propose in the next two chapters some improvements of the proposed methods. Firstly, we will focus on how to better capture the dissimilarity information using RF in the next chapter. For the current RFD measure, each tree in the forest gives binary information (i.e. 1 or 0). However, we believe that values with higher precision would be more appropriate to measure the dissimilarities at the tree level. It would also be important to take into account the tree confidence in the dissimilarity measure. Different improvements will be proposed in the next chapter. Furthermore, the two dissimilarity-based intermediate integration methods treat all the views with the same importance, while a weighted combination could also be used to generate a better joint dissimilarity matrix. These view combination solutions will be presented in chapter 4.

Chapter 3

Towards a better RFD measure

Contents

3.1 Introduction

From the experimental results in the previous chapter, we showed the potential of the multi-representation fusion approach for HDLSS multi-view learning. This approach is a two-step process: i) project each view in a new intermediate representation space, based on dissimilarities and ii) combine the resulting dissimilarity spaces to form a merged representation from which a new classifier can be learned. However, the multi-view learning literature mainly focuses on the second step only ([6, 98, 128, 133]), considering that the dissimilarity measure is problem-dependent and supplied beforehand. Yet, as explained in the 'related works' section of the previous chapter, the RFD measurement does not require choosing a dissimilarity formulation beforehand, since it is fully learned from the data. As a consequence, we think that this first step deserves to be further studied in this context, and in particular, that the RFD computation should be modified to suit better to our task. This is the purpose of this chapter.

The process of construction, the mathematical properties and the parametrization of RFD measure were discussed in details in the previous chapter. The dissimilarity values between instances are firstly generated from each decision tree and then combined by averaging over all the trees in the Random Forest. However, it is possible to make the RFD measure more accurate. Firstly, estimating the dissimilarity between two instances with a binary value may not be accurate enough and the precision of dissimilarity value depends on the number of trees. We believe that if each tree is able to provide a real value between 0 and 1, the final averaged dissimilarity value would be more accurate and the precision would depend less on the number of trees. Secondly, the dissimilarity value provided by a forest is calculated with a simple average. Yet, it is known for a RF that all the trees are not as reliable as each others for prediction, and one could think that the same phenomenon holds for the dissimilarity estimation: the RFD measure could then benefit from a weighting process to make the trees contribute differently to the final dissimilarity computation.

With the objective of proposing more elaborate dissimilarity measures based on Random Forest, the rest of this chapter is organized as follows: in section

2, the binary dissimilarity output of each decision tree is modified using information from the tree structure. A simple dissimilarity measure namely RFD_{NC} is proposed by taking advantage of terminal node confidence. In section 3, a more elaborated dissimilarity measure namely RFD_{IH} is proposed using instance hardness information beyond the tree structure. Different methods are tested and compared on real world datasets. Finally, in section 4 the conclusion and future work are discussed.

3.2 An accurate RFD measure from the tree structure

Random Forest based similarity and dissimilarity measures have been used in several works. For example, RFD is used in [16] for the change point detection in time series data and is also used in [99] for the classification of Alzheimer's disease. Several studies also show the use of unsupervised version of Random Forest similarity for data clustering [4, 209, 210]. However, very few works have been proposed to modify the RFD measure introduced in Chapter 2.

In [85], the authors state that the binary dissimilarity values provided by each tree is a very simple binary measure, which needs to be more refined. They propose a novel approach to estimate the Random Forest similarity for small sized forest. They try to make the similarity value non-binary for each tree by taking into account the distance between terminal nodes. Let us assume $\mathbf{x}_i$ ends in node n_i and $\mathbf{x}_j$ ends in node n_j. The distance g_{ijk} between n_i and n_j of the kth tree is the number of tree branches between n_i and n_j. The similarity between instances in n_i and n_j of tree k is calculated with:

$$s_{ij}^{(k)} = \frac{1}{e^{wg_{ijk}}} \tag{3.1}$$

where the parameter w controls the influence of the distance between two terminal nodes on the final dissimilarity. When w is very big (over ten for example), the similarity between two instances in different nodes is close to 0. In this case, g_{ijk} has no effect on the dissimilarity measure. However, if w is very small, g_{ijk} has a very strong influence on the final similarity value. One

example of this novel approach is given in Figure 3.1: in tree 1, the distance g_{131} between leaf nodes n_1 and n_3 is 3; while the distance g_{151} between leaf nodes n_1 and n_5 is 7.

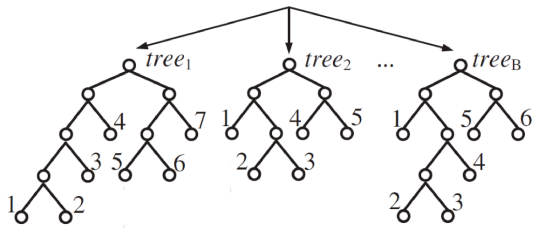
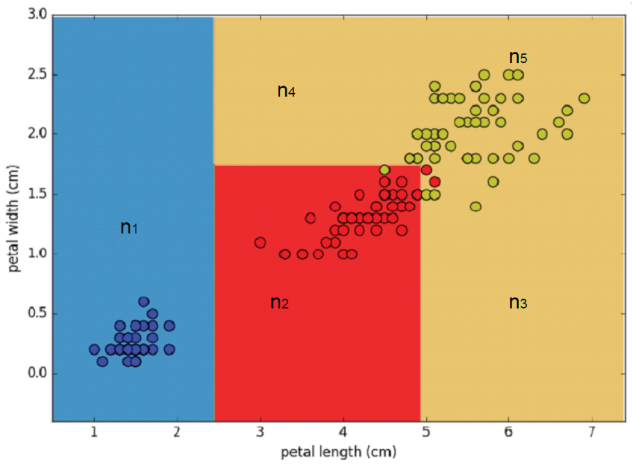


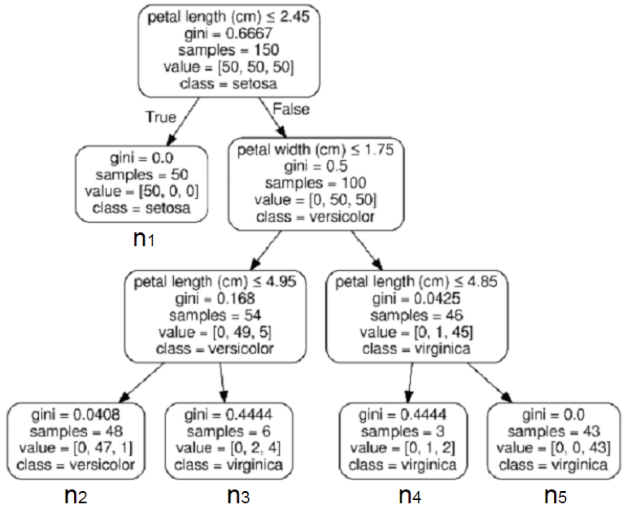
FIGURE 3.1: The example given in [85] for the measure of g_{ijk}.

However, from our point of view, there are several limitations of the method proposed in [85], noted RFD_{PB}. Firstly, even though the RFD_{PB} measure makes the dissimilarity measure from each tree non-binary, the scale of similarity values depends on the size of each tree (for a tree with more terminal nodes, the measure is more refined). In terms of class similarity, apart from the instances in the same terminal node, the smallest value of g_{ijk} is 2, from two terminal nodes sharing the same parent node. But when two terminal nodes have the same parent node, they are likely to be labeled with different classes. For example in Figure 3.2 (b), the distance between leaf node n_2 and leaf node n_3 is 2 and the distance between n_3 and n_4 is 4, which means that compared to n_4, n_2 is more similar to n_3. But the fact is that n_4 and n_3 gather instances from the same class while n_4 and n_3 gather instances from different classes. In terms of feature similarity, leaf nodes sharing more intermediate nodes should be more similar. Each split separates the instances in the same parent node into two non-overlapping sub-regions. When more intermediate nodes are shared by two leaf nodes, it means that their corresponding sub-regions are closer and hence more similar. However, with the proposed method in [85], leaf nodes sharing more intermediate nodes are not necessarily more similar. For example in Figure 3.2 (b), the distance g_{13} between leaf node n_1 and leaf node n_3 is 4, while the distance g_{35} between leaf node n_5 and leaf node n_3 is also 4. From Figure 3.2 (a) it can be seen that the sub-regions of n_3 and n_5 are

closer than the sub-regions of n_3 and n_1.



(a)



(b)

FIGURE 3.2: An example of two dimensional Iris data with three classes: (a) training instances in feature space: each color corresponds to a class; (b) the decision tree that splits instances into multiple partitions.

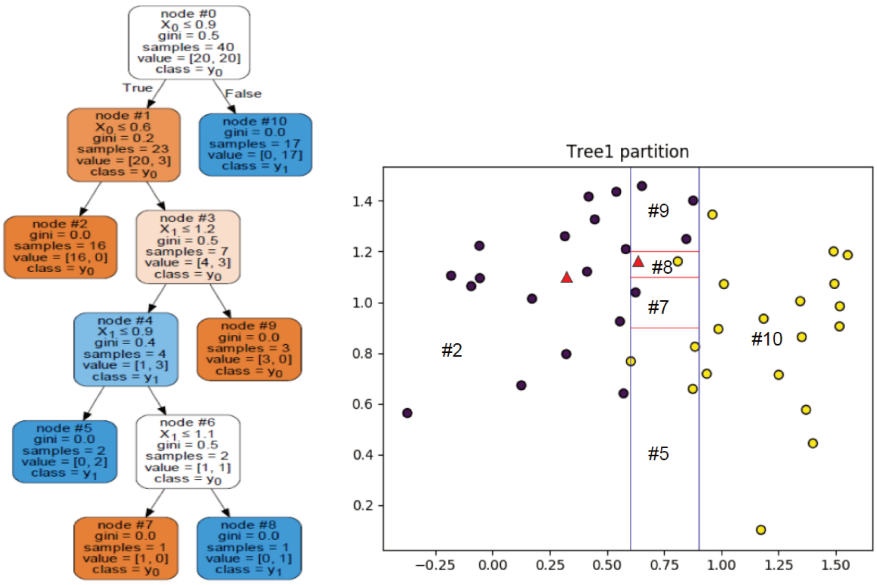
Secondly, their experimental results show that their approach can improve the

proximity estimate, especially when RF is made of a small number of trees. However, in the previous chapter, we have shown that for HDLSS problem, a bigger number of trees is necessary. Thirdly, the proposed method is controlled by a new hyperparameter, potentially difficult to tune in the HDLSS setting, due to the lack of validation datasets. In the next section, we propose to focus on the information provided by each terminal node to obtain a more accurate RFD measure.

3.2.1 New RFD measure with terminal node confidence

The motivation of using the node confidence is that each tree is a weak learner and not all terminal nodes can provide useful dissimilarity information. To illustrate this, another example is given in Figure 3.3a and 3.3b. The decision tree is built on a synthetic dataset with two classes and two features. All terminal nodes from Figure 3.3a are shown as partitions of the feature space in Figure 3.3b, where training instances are represented by circles while red triangles represent test instances. These six terminal nodes are very different: node #2 and node #10 are the two biggest sub-regions while other nodes occupy much smaller sub-regions; node #10 has the tree depth of 1 while node #8 has the tree depth of 5; node #2 contains a lot of training instances while node #7 contains only 1 training instance. Due to the diversity in terminal nodes of a decision tree, we have different confidences on different nodes too. For example, when a test instance lands in node #2 (the red triangle in node #2 shown in Figure 3.3b), we are quite confident that it belongs to class 1 (blue circles). However, if a test instance lands in node #8 (the red triangle in node #8 shown in Figure 3.3b), it is difficult to tell which class it belongs to because the node #8 is surrounded by many instances from other classes and it is easy to make a mistake.

The main task here is to identify these difficult terminal nodes and to assign them a lower confidence. However, it is hard to evaluate the quality of leaf nodes only according to the tree structure (instances in the node, node depth, etc.) because the structure of the tree can be easily affected by noise or outliers when the decision trees are fully grown ([35]). One advantage of using Bagging for building a Random Forest, as it is done in the reference method used



(a) (b)

FIGURE 3.3: An example of tree partition (a) the decision tree and (b) the visualization of partition in feature space. The decision tree is built on a synthetic dataset with two classes (class 1: blue circles, class 2: yellow circles). Two red triangles represent two test instances.

in this work, is that it offers the Out-Of-Bag (OOB) mechanism. The Bagging principle consists in drawing the training instances at random with replacement to form the bootstrap instances from which the trees are grown. This ensures that an average of 63,2 % of the training instances will be used in each bootstrap. The remaining 36,8% are called the OOB instances. These instances are very useful, since they efficiently replace validation datasets. For example, this is widely used to estimate the performance of a Random Forest, and has actually been proven to be a good estimator of the accuracy ([33, 41, 42]). This can also be used to estimate posterior probabilities for any test instance one wants to predict with the forest ([26]).

Here, we propose to use the OOB instances for estimating the "quality" of each leaf node, in the sense that it has been explained earlier. Typically, this would be done with a validation set in any other context, but let us recall that the

HDLSS setting makes often impossible to obtain additional instances for that purpose. When a decision tree is fully grown, all its terminal nodes are "pure", that is to say contain instances from only one class. As OOB instances are not seen during the training process, each decision tree may make mistakes on OOB instances. Hence, the weight of the kth tree for any test instance $\mathbf{x}_t$ can be defined as:

$$w^{(k)}(\mathbf{x}_t) = \frac{\sum_{\mathbf{x}_i \in N^{(k)}(\mathbf{x}_t)} I(h^{(k)}(\mathbf{x}_i) = y_i)}{\sum_{\mathbf{x}_i \in N^{(k)}(\mathbf{x}_t)} 1} \tag{3.2}$$

where $N^{(k)}(\mathbf{x}_t)$ is the neighborhood of test $\mathbf{x}_t$ on tree k, which includes all the instances (both training and OOB instances) landing in the same terminal node as $\mathbf{x}_t$. $I()$ equals to 1 if the expression is true and 0 otherwise. Hence, $w^{(k)}$ represents the posterior probability assigned to the terminal node where the test instance lands in using OOB instances. This weighting procedure is summarized in Algorithm 2.

Algorithm 2 RFD_{NC}: RFD with node confidence

1: Input: Test instance $\mathbf{x}_t$, trained RF (**H** <- RandomForest($mtry = \sqrt{m}, n_{trees} = M, \mathcal{T}$))
2: **for** each training instance $\mathbf{x}_i$ in $\mathcal{T}$ **do**
3: **for** each tree h_k in **H do**
4: If $\mathbf{x}_t$ and $\mathbf{x}_i$ land in different leaf nodes, $s^{(k)}(\mathbf{x}_t, \mathbf{x}_i) = 0$
5: If $\mathbf{x}_t$ and $\mathbf{x}_i$ land in the same leaf node, $s^{(k)}(\mathbf{x}_t, \mathbf{x}_i) = 1$
6: Measure the weight of each tree $w^{(k)}(\mathbf{x}_t)$ (as in Equation 3.2)
7: The final dissimilarity: $d(\mathbf{x}_t, \mathbf{x}_i) = 1 - \frac{\sum_{k=1}^{M} w^{(k)}(\mathbf{x}_t) \times s^{(k)}(\mathbf{x}_t, \mathbf{x}_i)}{\sum_{k=1}^{M} w^{(k)}(\mathbf{x}_t)}$
8: Output: The dissimilarity vector $\mathbf{d}(\mathbf{x}_t)$ between $\mathbf{x}_t$ and all training instances.

It can be seen that the node confidence does not depend on the entire tree, but only depends on the terminal node the test instance ends in. This procedure enables a "personalized weighting": each test instance will have different weights over M decision trees because it is very unlikely that two different instances always end in the same terminal node for all M trees, especially when M is big and the feature dimension is very high.

According to the results in the previous chapter, two RFD based methods both work very well with SVM and RF as the classifier. However, when the RFD

measure is modified as proposed, the resulting RFD matrix is not p.s.d. anymore. Indeed, let us recall that this property has been proven for the binary dissimilarity values given by a single tree. Since these values are not binary anymore, the p.s.d. property can not be proven anymore. Nevertheless, even if the RFSVM can not be used here because it requires the RFD matrix to be p.s.d., the RFDis methods is still usable, and will be used in the following to test the new RFD measure and compare it to state-of-the-art methods.

3.2.2 Experiments and results

Protocol

In this section, we compare the proposed method RFD_{NC} to two Random Forest based dissimilarity measures: the RFD measure which is introduced in the previous chapter and proved to work very well in HDLSS case and the RFD_{PB} proposed in [85]. We also compare the proposed dissimilarity learning method to the classic Euclidean distance measure and a metric learning measure based on the recommendation of the very recent survey ([148]): Large Margin Nearest Neighbors (LMNN [80]) along with PCA. LMNN is one of the most widely-used Mahalanobis distance learning methods. The constraints of LMNN are defined for each training instance: the "target neighbors" of a training instance are its k nearest neighbors based on Euclidean distance while the "impostors" are the instances from other classes. The objective is to make the target neighbors belong to the correct class of the training instance and keep impostors away with a margin. According to the suggestions in [148]: the number of components for PCA is chosen as 300; the size of neighborhood for LMNN is set to 25, but for datasets with less than 25 instances for some class, the size of neighborhood is set as the number of training minority class size.

For all the RF classifiers used in this experiment, the number of trees is set to 512 following the conclusions from the previous chapter, while other parameters are set to the default values proposed in the *Scikit-learn* machine learning framework ([178]).

With RFDis, all the five chosen dissimilarity measures (summarized in Table 3.1) are firstly used to build the dissimilarity matrix for each view. Then these dissimilarity matrices are averaged to form the joint dissimilarity matrix, which is then used as the input of the Random Forest for the classification task. Two non-RFD based dissimilarity measures (Euclidean distance and LMNN) provide unbounded distance values. Hence, they are re-scaled to the [0,1] interval by dividing each dissimilarity vector by its maximum value before the averaging. The results are presented in the following for multi-view datasets only, but single-view results are also reported in Appendix B, which share the same conclusions as the multi-view results.

TABLE 3.1: An overview of all the methods compared in this chapter

Methods	**Dissimilarity measure**
$EUDis$	Euclidean distance
$LMNNDis$	PCA+LMNN
$RFDis$	RFD
$RFDis_{PB}$	RF based dissimilarity in [85]
$RFDis_{NC}$	Proposed RF based measure using node confidence

The datasets used in this chapter are the same as in the previous chapter. We recall that a stratified random splitting procedure is repeated 10 times, with 50% of the instances for training, 50% for testing. In order to compare the methods, the mean and standard deviations of accuracy are evaluated over 10 runs.

Results

The average classification rates over the 10 repetitions, along with standard deviations, are shown in Table 3.2. Bold numbers on each row show the best classification results among all five methods. The proposed method $RFDis_{NC}$ with node confidence wins 6 times the first place among 15 datasets. The method $RFDis_{PB}$ proposed in [85] wins 5 times the first place among 15 datasets.

From the average ranking, it can be seen that the proposed method $RFDis_{NC}$ and $RFDis_{PB}$ are the best among all the methods. Among all three Random Forest based dissimilarity measures, the original RFD is the worst ranked;

TABLE 3.2: The classification accuracies of multi-view intermediate integration with different dissimilarity measures. Random Forest is built on the joint dissimilarity matrix (average over views) to evaluate the performance.

	EUDis	*LMNNDis*	*RFDis*	$RFDis_{PB}$	$RFDis_{NC}$
awa8	39.22% ± 2.55	42.28% ± 3.13	56.06% ± 1.35	**56.38% ± 1.47**	56.34% ± 1.68
awa15	24.80% ± 0.97	28.25% ± 1.60	37.90% ± 1.49	37.62% ± 1.40	**37.93% ± 1.50**
Metabo	**69.38% ± 2.29**	67.08% ± 4.04	67.71% ± 5.12	67.50% ± 5.76	67.08% ± 6.31
mfeat	96.00% ± 1.45	96.87% ± 0.79	97.56% ± 0.99	**97.63% ± 0.95**	97.63% ± 1.00
nus2	89.52% ± 1.44	90.33% ± 1.55	92.49% ± 2.01	92.49% ± 1.81	**92.67% ± 1.47**
bbc	85.89% ± 1.33	**93.02% ± 1.29**	92.82% ± 0.67	93.00% ± 0.67	92.33% ± 0.49
lowGrade	63.72% ± 5.12	62.33% ± 7.04	63.48% ± 3.76	63.72% ± 4.67	**63.95% ± 3.64**
nus3	73.92% ± 2.40	78.02% ± 2.69	79.41% ± 1.94	79.64% ± 2.19	**79.91% ± 2.14**
progression	58.42% ± 4.82	62.63% ± 5.86	63.42% ± 6.49	63.42% ± 7.48	**63.95% ± 6.56**
LSVT	82.86% ± 2.11	**85.24% ± 2.84**	83.33% ± 3.97	82.70% ± 3.44	83.49% ± 3.56
IDHCodel	73.53% ± 5.42	71.47% ± 2.30	**76.47% ± 3.95**	76.47% ± 4.16	76.18% ± 3.82
nonIDH1	79.07% ± 3.45	73.26% ± 3.49	79.53% ± 3.57	79.53% ± 3.72	**79.77% ± 3.46**
bbcsport	80.11% ± 1.69	73.77% ± 5.45	81.75% ± 2.70	**82.56% ± 2.85**	79.93% ± 3.11
Cal20	84.04% ± 0.82	87.50% ± 0.78	89.12% ± 0.69	**89.27% ± 1.01**	89.06% ± 1.19
Cal7	92.67% ± 0.63	95.09% ± 0.66	95.21% ± 0.67	**95.51% ± 0.50**	95.34% ± 0.48
Avg rank	4.23	3.90	2.60	2.13	2.13

while among all five methods, there is no surprise that *EUDis* is ranked as the worst solution because it is the only learning free dissimilarity measure. Many studies have shown that the Euclidean distance measure can suffer from the curse of dimensionality, especially when the feature dimension is bigger than the sample size ([13, 88]). For the highest feature dimension datasets such as BBC (13628 features), the performance gap between learning based and learning free dissimilarity measures is thus important.

The result of Nemenyi post-hoc test is shown in Figure 3.4: there are no significant difference among the three RF based methods, but $RFDis_{NC}$ and $RFDis_{PB}$ are significantly better than *EUDis* and *LMNNDis*.

The performance gap between proposed $RFDis_{NC}$ and *RFDis* is quite small. It may be due to the following limitations of the proposed $RFDis_{NC}$ method:

- When sample size is small, the OOB sample size is of course also small, which may result in the case that some terminal nodes have no OOB instances. In this case, the node posterior probability is 1 but it does not mean that the corresponding sub-region is relevant for learning the dissimilarity.

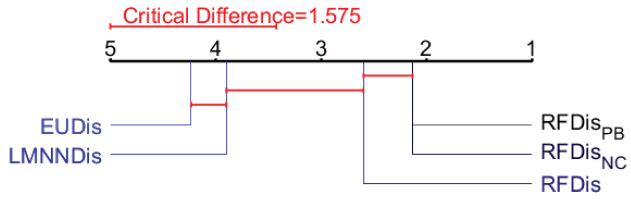


FIGURE 3.4: The CD diagram according to the Nemenyi post hoc test result when $\alpha = 0.05$ (based on the results of Table 3.2).

- All the instances in the same node share the same weight. For each decision tree, all the test instances that land in the same terminal node have the same posterior probability. But instances in the same terminal node do not necessarily share the same relevance for learning the dissimilarity. For example, #node 10 in Figure 3.3b has many instances, some of them (eg, the one near #node 9) are close to the class border, but others are far away from the class border. This sub-region is then more relevant for the instances far from the class border and less relevant for the instances close to the border.

In the next section, to avoid these limitations of RFD_{NC}, a new method based on instance hardness is introduced.

3.3 New RFD measure based on instance hardness

We recall that the node confidence proposed in the previous section works as follows: if the instances in the same leaf node (including bootstrap instances and OOB instances) contain more than one class, the node is considered as "hard" (difficult to be classified correctly) and we are less confident on the dissimilarity induced from this node. However, instances may not have the same confidence even though they land in the same terminal node. In this case, the node level confidence is no more appropriate and an instance level confidence

needs to be used instead. To give a more refined weight, the concept of Instance Hardness (IH) is introduced in 3.3.1, which can be used to measure the confidence at instance level. Then in 3.3.2, a new dissimilarity measure based on IH, namely RFD_{IH}, is proposed to overcome the limitations of RFD_{NC}. The comparison with all the methods tested in the previous experiment and a further result analysis are given in 3.3.3 and 3.3.4 respectively.

3.3.1 Instance hardness

In [216], the authors provide an empirical definition of instance hardness. Generally speaking, the IH measure helps to identify which instances are likely to be misclassified and why they are hard to classify ([156, 187]). High values of instance hardness usually indicate difficult instances such as outliers or instances close to a decision boundary. The advantage of IH measure is that it reveals the difficulty of a problem at an instance level rather than at the dataset level ([127, 155]).

A summary of different hardness measures is given in Table 3.5. The "+" and "-" symbols indicates whether the corresponding measure is positively or negatively correlated to IH ([216]). All hardness measures are designed to understand why an instance is hard (a positive correlation with instance hardness) or easy (a negative correlation with instance hardness) to classify. For example, Class Likelihood (CL) measures the probability of an instance belonging to a certain class. For "harder" instances, their CL values are lower.

Among the different IH measures listed in Table 3.5, the most used is the k-Disagreeing Neighbors (kDN) measure. The kDN value is simply the percentage of instances from other classes in the neighborhood of a given instance:

$$kDN(\mathbf{x}_i) = \frac{|\mathbf{x}_k : \mathbf{x}_k \in KNN(\mathbf{x}_i) \cap y_k \neq y_i|}{K} \tag{3.3}$$

where $KNN(\mathbf{x}_i)$ is the K nearest neighbors of $\mathbf{x}_i$. The distance measure chosen here is usually the Euclidean distance. kDN is straightforward and easy to understand. If the nearest neighbors of an instance are all from the same class as the given instance, the kDN value is 0 and it is easy to classify. On the other hand, if the nearest neighbors of an instance are all from other classes, the

Abbr	+/−	Measure	Insight
*k*DN	+	*k*-Disagreeing Neighbors	Overlap of an instance using all of the data set features on a subset of the instances
DS	−	Disjunct Size	Complexity of the decision boundary for an instance
DCP	−	Disjunct Class Percentage	Overlap of an instance using a subset of the features and a subset of the instances
TD	+	Tree Depth	The description length of an instance in an induced C4.5 decision tree
CL	−	Class Likelihood	Overlap of an instance using all of the features and all of the instances
CLD	−	Class Likelihood Difference	Relative overlap of an instance using all of the features and all of the instances
MV	+	Minority Value	Class skew
CB	−	Class Balance	Class skew

FIGURE 3.5: The table of different hardness measures extracted from [216], where "+" and "-" symbols distinguish which hardness measures are positively and negatively correlated with instance hardness.

kDN value is 1, which means that this given instance is very hard to classify. kDN has been successfully used in many recent works due to its simplicity and interpretability such as in [72, 235].

3.3.2 Instance hardness for RFD

Before introducing the proposed method based on IH, we firstly use a simple example to visualize how IH value can guide us for a more accurate dissimilarity measure. In Figure 3.6, the Iris dataset is projected into two of its 2D subspaces. The relative position of the same training instances is different from one subspace to the other. For example, the blue circle pointed by an arrow in subspace1 is close to the green class. The same training instance in subspace2 has a very different location: it is very close to its own class and far away from other classes. With the definition of kDN, we can say that this instance is more difficult to classify in subspace1 than in subspace2 in terms of the classification task. When we want to measure the dissimilarity of a test instance to this instance, we should trust the result from subspace2 more than the result from subspace1.

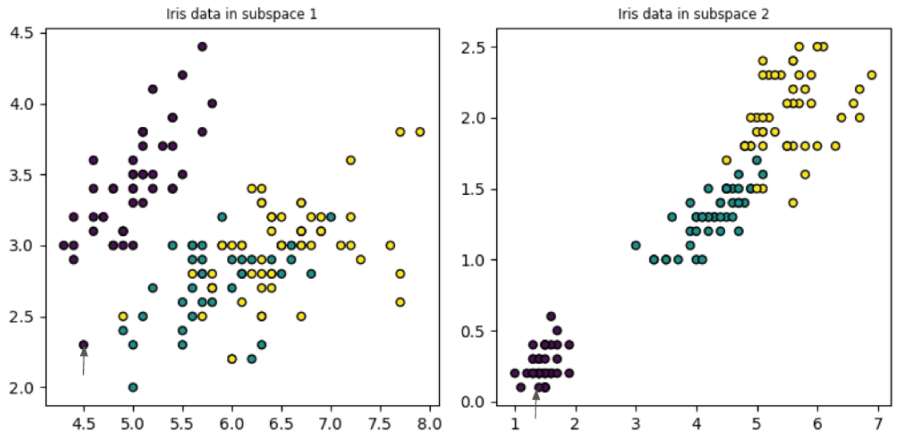


FIGURE 3.6: Scatter plot of the Iris dataset into two of its 2D subspaces. The two data points pointed by the arrow in each subspace are the same training instance.

Similarly, due to the bagging process and the other randomization principles of RF, different decision trees result in different data partitions. For a given instance, it may end in some "easy region" such as the pointed instance in subspace2 with tree i and may also end in some "hard region" such as the pointed instance in subspace1 with tree j. In this case, IH measure can help us to identify these "easy regions" that can provide more useful dissimilarity information.

From Equation 3.3, it can be seen that the class information is needed by the kDN IH measure. As we do not have class information for test instances, the IH measure can only be applied on training instances. We propose to use kDN to estimate the relevance of dissimilarity induced from the decision tree for each training instance.

However, kDN measures the neighborhood based on the Euclidean distance, which is known to suffer from the concentration of pairwise distance [5] and the asymptotic effect [109] in high dimensional space. Furthermore, it is not very meaningful to use the kDN in the original feature space as each tree can typically use a small subset of the features for HDLSS problems. For a training set with n instances, each tree can have a maximum of n leaf nodes and $n-1$

split nodes (this case happens only if each leaf node contains one instance). Hence, for each tree, there are maximum $n - 1$ features that are used to split the nodes. For classification tasks, the number of leaf nodes is usually much smaller than n because some leaf nodes may contain much more than one instance. Hence, for HDLSS problem, only a small subset of features is used for building each tree. Each decision tree may use different feature subsets due to the diversity in the RF. This fact makes the IH of the same instance differs among trees and makes it necessary to measure the kDN on the feature subsets.

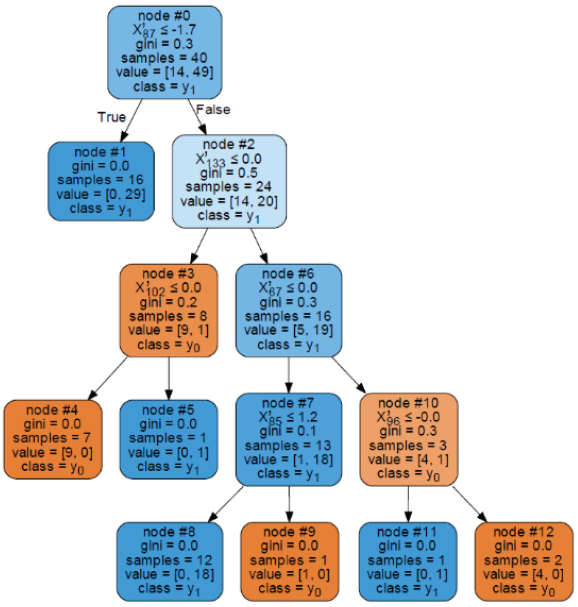


FIGURE 3.7: An example of decision tree constructed on the LSVT dataset.

However, even in the same decision tree, different decision paths use different features. For example in Figure 3.7, a decision tree built on LSVT data is shown. The decision path for leaf node #1 is very simple: from node #0 to node #1 with only one feature used. In contrast, for leaf node #8, the corresponding decision path passes through node #0, node #2, node #6 and node #7 with four different features being selected. These selected features define different

subspaces for leaf nodes. This example shows that different leaf nodes have different decision paths, which normally leads to different selected feature subsets. This feature subset defines the feature subspace for all the instances landing in the corresponding leaf node. From the example of Iris dataset (3.6), we have shown the importance of the neighborhood in different subspaces. Hence, the kDN measure can be used to measure the instance hardness in this low dimensional subspace to solve the problems presented in the previous paragraph. Compared to the node confidence weighting scheme proposed in the previous section, the IH provides more refined weights on instance level instead of node level.

The IH weighting scheme is summarized in Algorithm 3. For each training instance, we firstly find the corresponding decision path and leaf node. Then, we project all training instances into the subspace defined by the decision path and use kDN to measure the instance hardness in the subspace. According to the suggestion of [72], the neighborhood size of kDN can be set to 7. It can be seen that IH assigns different weights to the training instances even though they are in the same terminal node.

Algorithm 3 RFD_{IH}: RFD with instance hardness

1: Input: Test instance $\mathbf{x}_t$, trained RF $\mathbf{H}$ <- RandomForest($mtry = \sqrt{m}, n_{trees} = M, \mathcal{T}$)
2: **for** each training instance $\mathbf{x}_i$ in $\mathcal{T}$ **do**
3: **for** each tree h_k in $\mathbf{H}$ **do**
4: If $\mathbf{x}_t$ and $\mathbf{x}_i$ land in different leaf nodes, $s^{(k)}(\mathbf{x}_t, \mathbf{x}_i) = 0$
5: If $\mathbf{x}_t$ and $\mathbf{x}_i$ land in the same leaf node, $s^{(k)}(\mathbf{x}_t, \mathbf{x}_i) = 1$
6: Measure the instance hardness of $\mathbf{x}_i$ in the corresponding subspace $IH^{(k)}(\mathbf{x}_i)$ based on kDN defined in Equation 3.3.
7: The final dissimilarity: $d(\mathbf{x}_t, \mathbf{x}_i) = 1 - \frac{\sum_{k=1}^{M}(1-IH^{(k)}(\mathbf{x}_i))\times s^{(k)}(\mathbf{x}_t,\mathbf{x}_i)}{\sum_{k=1}^{M}(1-IH^{(k)}(\mathbf{x}_i))}$
8: Output: The dissimilarity vector $\mathbf{d}(\mathbf{x}_t)$ between $\mathbf{x}_t$ and all training instances.

3.3.3 Experiments and results

The experimental protocol in this section is the same as in the previous section. We add the results of the proposed RFD_{IH} to Table 3.2. The average

classification rates with standard deviations are shown in Table 3.3.

TABLE 3.3: The classification accuracies of multi-view intermediate integration with different dissimilarity measures. Random Forest is built on the joint dissimilarity matrix (average over views) to evaluate the performance.

	EUDis	*LMNNDis*	*RFDis*	$RFDis_{PB}$	$RFDis_{NC}$	$RFDis_{IH}$
AWA8	39.22% ± 2.55	42.28% ± 3.13	56.06% ± 1.35	**56.38% ± 1.47**	56.34% ± 1.68	56.22% ± 1.01
AWA15	24.80% ± 0.97	28.25% ± 1.60	37.90% ± 1.49	37.62% ± 1.40	37.93% ± 1.50	**38.23% ± 0.83**
Metabo	**69.38% ± 2.29**	67.08% ± 4.04	67.71% ± 5.12	67.50% ± 5.76	67.08% ± 6.31	69.17% ± 5.80
Mfeat	96.00% ± 1.45	96.87% ± 0.79	97.56% ± 0.99	**97.63% ± 0.95**	97.63% ± 1.00	97.53% ± 1.00
NUS-WIDE2	89.52% ± 1.44	90.33% ± 1.55	92.49% ± 2.01	92.49% ± 1.81	92.67% ± 1.47	**92.82% ± 1.93**
BBC	85.89% ± 1.33	93.02% ± 1.29	92.82% ± 0.67	93.00% ± 0.67	92.33% ± 0.49	**95.46% ± 0.65**
lowGrade	63.72% ± 5.12	62.33% ± 7.04	63.48% ± 3.76	63.72% ± 4.67	**63.95% ± 3.64**	63.95% ± 5.62
NUS-WIDE3	73.92% ± 2.40	78.02% ± 2.69	79.41% ± 1.94	79.64% ± 2.19	79.91% ± 2.14	**80.32% ± 1.95**
progression	58.42% ± 4.82	62.63% ± 5.86	63.42% ± 6.49	63.42% ± 7.48	63.95% ± 6.56	**65.79% ± 4.71**
LSVT	82.86% ± 2.11	**85.24% ± 2.84**	83.33% ± 3.97	82.70% ± 3.44	83.49% ± 3.56	84.29% ± 3.51
IDHCodel	73.53% ± 5.42	71.47% ± 2.30	76.47% ± 3.95	76.47% ± 4.16	76.18% ± 3.82	**76.76% ± 3.59**
nonIDH1	79.07% ± 3.45	73.26% ± 3.49	79.53% ± 3.57	79.53% ± 3.72	79.77% ± 3.46	**80.70% ± 3.76**
BBCSport	80.11% ± 1.69	73.77% ± 5.45	81.75% ± 2.70	82.56% ± 2.85	79.93% ± 3.11	**90.18% ± 1.96**
Cal20	84.04% ± 0.82	87.50% ± 0.78	89.12% ± 0.69	89.27% ± 1.01	89.06% ± 1.19	**89.76% ± 0.80**
Cal7	92.67% ± 0.63	95.09% ± 0.66	95.21% ± 0.67	95.51% ± 0.50	95.34% ± 0.48	**96.03% ± 0.53**
Avg rank	5.20	4.83	3.67	2.83	2.93	1.53

The average ranking of all the methods shows that the proposed $RFDis_{IH}$ is the best among all methods, followed by $RFDis_{PB}$ and $RFDis_{NC}$. Among all four Random Forest based dissimilarity measures, the original RFD is ranked the worst. More detailed comparison is given in the following paragraphs.

Comparison to RFDis

As RFDis is our baseline from the previous chapter, to better assess the progression between the proposed methods and the baseline, a pairwise analysis based on the Sign test is computed on the number of wins, ties and losses between $RFDis$ and all the other methods. The result is shown in Figure 3.8, which demonstrates that $RFDis$ is a strong baseline as most methods are not able to be significantly better. With $\alpha = 0.10$, the proposed two methods $RFDis_{NC}$ and $RFDis_{IH}$ are significantly better than $RFDis$. With $\alpha = 0.05$ or $\alpha = 0.01$, only $RFDis_{IH}$ is significantly better than $RFDis$. Compared to $RFDis_{NC}$, $RFDis_{IH}$ has better performance, which confirms our hypothesis that more refined weighting can lead to better result.

The Nemenyi post-hoc test with Critical Differences (CD) is also done to have an overall statistical comparison. The result is shown in Figure 3.9: only

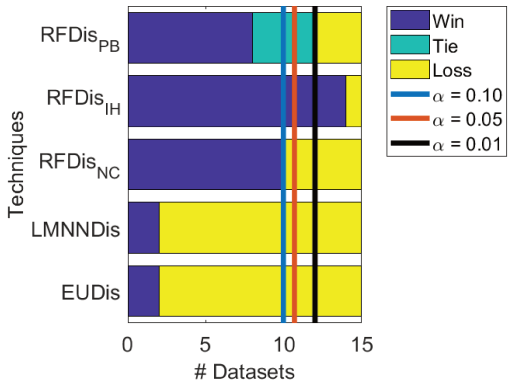


FIGURE 3.8: Pairwise comparison between RFDis and all the other methods. The vertical lines illustrate the critical values considering a confidence level $\alpha = \{0.10, 0.05, 0.01\}$.

$RFDis_{IH}$ is significantly better than $RFDis$, but the difference among RFD_{IH}, RFD_{NC} and RFD_{PB} is not significant. There is no significant difference among $RFDis_{PB}$, $RFDis_{NC}$, $RFDis$ and $LMNNDis$ neither.

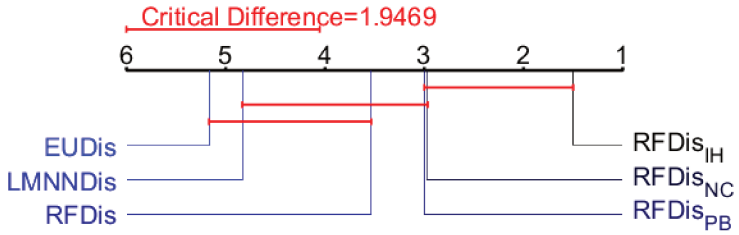


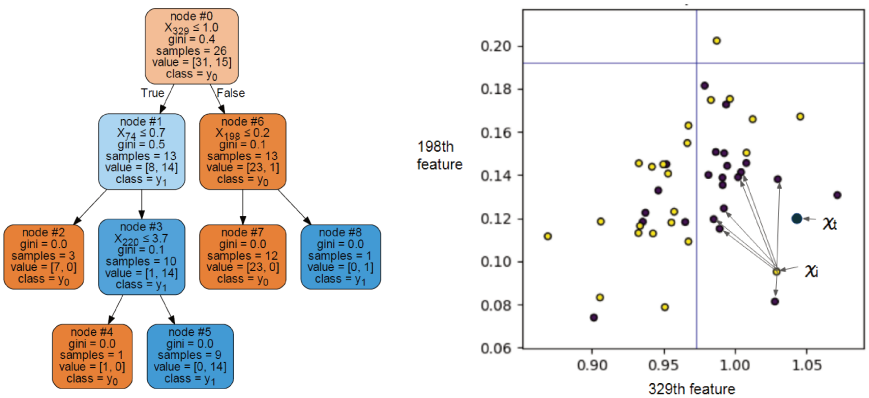
FIGURE 3.9: The CD diagram according to the Nemenyi post hoc test result when $\alpha = 0.05$ (based on the results in Table 3.3).

3.3.4 Illustrative example

From the experimental results in the previous section, we confirmed our hypothesis of the proposed method RFD_{IH}. To make it easier to understand how

IH can help to improve the quality of the dissimilarity measure, we present some detailed analysis from the experiments of the previous section.

For a given instance, if all the trees in the Random Forest can provide correct dissimilarity information, then the IH based weighting can not influence the quality of the final aggregated dissimilarity measure. However, each decision tree of a Random Forest is called a weak learner in the ensemble, indicating that each tree does not have good global performance but does have some local expertise. In this case, RFD_{IH} can help to put more focus on the trees that can provide correct dissimilarity information. The kDN in the subspaces defined by each decision path is used to measure the confidence of each tree. To visualize this procedure, we selected one of the 15 datasets used in the previous experiment and plotted the decision path of a instance and its corresponding subspace.



(a) (b)

FIGURE 3.10: An example of a training instance considered hard: (a) the decision tree and (b) the visualization of partition in sub feature space of node #7. There are two classes (class 0: blue circles, class 1: yellow circles). Two blue lines show the corresponding split threshold of each feature.

The example from Metabo dataset is shown in Figure 3.10a and 3.10b: both $\mathbf{x}_t$ and $\mathbf{x}_i$ land in leaf node #7. Two features, feature 329 and feature 198, are used by leaf node #7. The original feature space is projected into the subspace of node #7 shown in Figure 3.10b: it can be seen that $\mathbf{x}_i$ belongs to class 1 but

node #7 is associated with class 0. As all the trees are fully grown, each leaf node is pure (all the bootstrap instances are correctly classified), which shows that $\mathbf{x}_i$ is a misclassified OOB instance. It results in wrong information when we try to measure the dissimilarity between other instances and $\mathbf{x}_i$. From its neighborhood, it can be told that all the 7 nearest neighbors in the projected subspace (pointed with arrow lines) belong to class 0, which makes the IH value of $\mathbf{x}_i$ equal to 1. The test instance $\mathbf{x}_t$ is from class 0. With the classic RFD measure, their similarity on this tree is 1 because they land in the same leaf node. However, with RFD_{IH}, their similarity is 0 because $\mathbf{x}_i$ is considered as a very hard instance with IH = 1. From this example, it confirms our hypothesis that RFD_{IH} can help to identify if a tree provides a reliable dissimilarity measure for a given instance.

3.4 Conclusion

RFD based multi-representation fusion methods have been shown to be a successful solution for HDLSS multi-view problem in the previous chapter. In this chapter, with the aim to address the first possibility to improve the classification performance, the limitations of classic RFD have been analyzed. We argued that the original binary dissimilarity value provided by each decision tree can be refined with more accurate values. Based on the idea that not all trees can provide reliable dissimilarity information for a given instance, two different weighting schemes have been proposed: the first one is to weight each leaf node of the decision tree using the node confidence because the leaf node contains all the information of the dissimilarity induced from the decision tree; the second one provides a more refined weighting scheme based on instance hardness to address the disadvantages of the first proposal.

The experiments on 15 multi-view datasets have been realized to compare the proposed two measures to other dissimilarity measures including learning free measure and learning based measures. The results have shown that: 1. compared to learning free measure, learning based measures have better performance for multi-representation fusion on HDLSS multi-view problem; 2.

Random Forest based dissimilarity measures are better than the metric learning method LMNN; 3. among all Random Forest based measures, the proposed method RFD_{IH} is significantly better than the classic RFD measure. The two weighting methods for RFD lead to more accurate dissimilarity measures without the limitations of being metric or p.s.d. However, this also limits the use of the proposed measures as they are not symmetric and not p.s.d, which means that they can not be used as a kernel for SVM classifier. But they can always be used as new feature vectors to train the classifiers such as Random Forest.

In this chapter, the kDN has been chosen as the instance hardness measure. The parameter k was fixed for all the datasets according to the suggestion from other studies. The effect of k can be studied in short term to see its influence on the resulting dissimilarity measure on different datasets, especially those datasets with very small sample size. Usually, larger k can provide more refined weights and differentiate better the confidences of different trees. Hence, an adaptive k for different datasets may help to improve the classification performance instead of using a fixed number. According to [159], kDN does not take class imbalance into account and may not be a good choice for imbalanced data. The authors propose a Bayes Imbalance Impact Index, which can be studied and tested for the imbalanced datasets.

In the long term, the instance hardness for test instances can be proposed. One disadvantage of most instance hardness measures is that the class information is needed. In this case, we can only measure the IH on training instances and measure the confidence only based on training instances. But dissimilarity reflects the pairwise information, the instance hardness of both training instances and the given test instances should be taken into account. In [126], the authors have tried to estimate the classifier's trust score for each test instance by measuring the agreement between the classifier and the modified nearest neighbor classifier on the test instance with some data density requirement. However, how to adapt the measure for HDLSS problem remains a challenge.

Chapter 4

Towards a better combination of dissimilarity matrices

Contents

4.1 Introduction

The second chapter of this manuscript showed the great potential of RFD based multi-representation fusion methods for HDLSS multi-view learning. It also introduced two areas for improvement: (i) proposing a finer measure of dissimilarity that better suits to our problem and (ii) proposing a mechanism for merging representations that better takes into account the complementarities of views. As the first goal has been explored in the previous chapter, it is the second goal that we are now focusing on in this chapter.

The RFD based multi-view methods proposed so far are merging the view-specific representations in a quite naive way: by simply averaging them. Let us recall that these view-specific representations are built by describing each instance through its dissimilarities with each of the training instances. For example, any instance $\mathbf{x}$ is represented by Q vectors:

$$\{d^{(q)}(\mathbf{x}, \mathbf{x}_1), d^{(q)}(\mathbf{x}, \mathbf{x}_2), \ldots, d^{(q)}(\mathbf{x}, \mathbf{x}_N)\}, \forall q = 1, 2, \ldots Q \tag{4.1}$$

where $d^{(q)}(\mathbf{x}, \mathbf{x}_i)$ is the dissimilarity between the instance $\mathbf{x}$ and the training instance $\mathbf{x}_i$, computed from the qth view. These values are fully comparable from a view to another and as a consequence, can be straightforwardly averaged over the views. The merged representation is given by :

$$\{d_m(\mathbf{x}, \mathbf{x}_1), d_m(\mathbf{x}, \mathbf{x}_2), \ldots, d_m(\mathbf{x}, \mathbf{x}_N)\} \tag{4.2}$$

where

$$d_m(\mathbf{x}, \mathbf{x}_i) = \frac{1}{Q} \sum_{q=1}^{Q} d^{(q)}(\mathbf{x}, \mathbf{x}_i) \tag{4.3}$$

The averaged dissimilarity is a simple, yet meaningful way to merge the representations built from all the views. However, it intrinsically considers that all views are equally reliable with regard to the task, and that the resulting dissimilarities are as important as each other. This is likely to be wrong, from our point of view. In multi-view learning problems, the different views are meant to be complementary in some ways, that is to say to vehicle different

types of information regarding the classification task. These different types of information may not have the same contribution to the final predictions. That is the reason why it may be important to differentiate these contributions, for example by weighting them according to some criterion taking into account their reliability.

Generally speaking, there are two ways of weighting dissimilarity matrices: static weighting and dynamic weighting. Static weighting methods assign a fixed weight to each view with the assumption that the importance of each view is the same for all test instances, while dynamic weighting methods assign "personalized" view weights depending on the instance to predict. In section 4.2, different static weighting methods are reviewed and a weighting method based on the OOB accuracy is proposed. In section 4.3, several dynamic weighting methods are reviewed. A dynamic view selection method is proposed in section 4.4 to select the best combination of views for each instance to predict.

4.2 Static view combination

4.2.1 Static weight calculation

Given a set of dissimilarity matrices $\{\mathbf{D}^{(1)}, \mathbf{D}^{(2)}, \ldots, \mathbf{D}^{(Q)}\}$ built from Q different views, our goal here is to find the best non-negative weight set $\{w^{(1)}, w^{(2)}, \ldots, w^{(Q)}\}$, so that the joint dissimilarity matrix is:

$$\mathbf{D} = \sum_{q=1}^{Q} w^{(q)} \mathbf{D}^{(q)} \tag{4.4}$$

where $w^{(q)} \geq 0$ and $\sum_{q=1}^{Q} w^{(q)} = 1$.

In the literature, there exists many approaches that allow to find such an optimal or near-optimal set of weights, to combine dissimilarity matrices. The most natural one is to deduce the weights from a quality score measured on each view. For example in [150] this principle is used for multi-scale image classification where each view is a version of the image at a given scale. For

this application, the weights are derived directly from the scale factor associated with the view. Obviously, this only makes sense with regard to the application, for which the scale factor gives an indication of the reliability for each view.

Another, more generic and classification-specific approach, is to evaluate the quality of the dissimilarity matrix using the performance of a classifier. This makes it possible to estimate whether a dissimilarity matrix sufficiently reflects class membership ([83, 150]). For example, in [150], the authors propose to train a linear SVM from each view and to use the cross-validation performance for the estimation of the weight of each view:

$$\alpha^{(q)} = max(acc^{(q)} - rnd^{(q)}, 0) \tag{4.5}$$

where $acc^{(q)}$ is the cross validation accuracy on view q and $rnd^{(q)}$ is the random classification accuracy. In this case, if the cross validation accuracy is somehow random, the weight assigned to the corresponding view is zero and the view is not taken into account in the combination.

KNN is also very often used to evaluate the quality of a dissimilarity matrix due to its easy interpretation: a good dissimilarity measure should propose good neighborhoods (i.e, the most similar instances should belong to the same class). For example, a 1 Nearest Neighbor (1NN) classifier is used to assess the performance of dissimilarity matrices in [83]. In the field of metric learning, 3NN is widely used to evaluate the quality of learned metrics ([130, 148, 190, 243]).

In the field of kernel combination, a different heuristic rule is used. In [68], the notion of Kernel Alignment (KA) is defined to measure the similarity between two kernels with the following formulation:

$$A(\mathbf{K}_1, \mathbf{K}_2) = \frac{\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F}{\sqrt{\langle \mathbf{K}_1, \mathbf{K}_1 \rangle_F \langle \mathbf{K}_2, \mathbf{K}_2 \rangle_F}} \tag{4.6}$$

where $\langle \cdot, \cdot \rangle_F$ is the Frobenius norm which is given by:

$$\langle \mathbf{K}_1, \mathbf{K}_2 \rangle_F = \sum_{i=1}^{N} \sum_{j=1}^{N} K_1(\mathbf{x}_i, \mathbf{x}_j) K_2(\mathbf{x}_i, \mathbf{x}_j) \tag{4.7}$$

Transposed to vectors instead of matrix, this KA measure can be interpreted as the angle between two vectors, which reflects their correlation. To compute the KA, an ideal target matrix must be defined beforehand. This ideal target matrix is an optimal theoretical similarity matrix, in regards to the task. For example, for binary classification, the ideal target matrix is usually defined as $\mathbf{K}^* = \mathbf{y}\mathbf{y}^T$, where $\mathbf{y} = \{y_1, y_2, \ldots, y_N\}$ is the label vector and $y_i \in \{-1, +1\}$. Thus, each value in $\mathbf{K}^*$ is:

$$\mathbf{K}^*_{ij} = \begin{cases} 1, & \text{if } y_i = y_j \\ -1, & \text{otherwise} \end{cases} \tag{4.8}$$

In other words, instances are considered similar ($\mathbf{K}^*_{ij} = 1$) if and only if they belong to the same class. In [233], the authors proposed an adaptation of this matrix to multi-class classification:

$$\mathbf{K}^*_{ij} = \begin{cases} 1, & \text{if } y_i = y_j \\ \frac{-1}{\mathcal{C}-1}, & \text{otherwise} \end{cases} \tag{4.9}$$

where $\mathcal{C}$ is the number of classes. To use the KA method in our experiments, the dissimilarity are transformed into kernel as explained in chapter 2 (section 2.3.3).

In [191], the authors propose an heuristic rule with KA to weight kernel matrices for binary classification problem. The weight assigned to the kernel matrix $\mathbf{K}^{(i)}$ of the ith view is given by:

$$w^{(q)} = \frac{A(\mathbf{K}^{(q)}, \mathbf{y}\mathbf{y}^T)}{\sum_{h=1}^{Q} A(\mathbf{K}^{(h)}, \mathbf{y}\mathbf{y}^T)} \tag{4.10}$$

In [1, 68, 191], the authors proved that if a kernel matrix is p.s.d, the corresponding alignment to the target kernel is non-negative, and the corresponding $w^{(q)}$ is also non-negative. However, as our similarity matrix generated by RFD_{IH} is not p.s.d. anymore, we cannot make sure that $w^{(q)}$ in Equation 4.10 is non-negative. To avoid negative weights, we propose to use the softmax function to normalize the weights. The corresponding weights for each view are measured as in Equation 4.11.

$$w^{(q)} = \frac{exp(A(\mathbf{K}^{(q)}, \mathbf{K}^*))}{\sum_{h=1}^{Q} exp(A(\mathbf{K}^{(h)}, \mathbf{K}^*))} \tag{4.11}$$

Both classification accuracy (Equation 4.5) and KA (Equation 4.11 and 4.9) could be used in our framework to estimate the quality of the dissimilarity matrix $D^{(q)}$. The disadvantage of using the classification accuracy is that a validation dataset is required for many classifiers to guarantee a good performance, which is less adapted to the HDLSS problem. In the next section, we propose another way of measuring the quality of each dissimilarity matrix without using a validation dataset for static weighting.

4.2.2 Weighting with OOB accuracy

All the static weighting methods calculate the importance of each view from the dissimilarity matrices only, which may not generalize well on the test instances if there is no validation dataset. To overcome this problem, we propose to use OOB estimation in this section.

In the previous chapter, OOB instances were used to evaluate the confidence of each terminal node. Here, the OOB instances are used to estimate the performance of each Random Forest. For each OOB instance, the trees that have not used this instance as training data are chosen to form a sub-forest to predict its label. When the labels of all OOB instances are predicted, they are compared to the ground truth labels to give a classification accuracy. This estimate is known to be a reliable estimate of the generalization performance [33].

We propose to use the Out-Of-Bag accuracy of the Random Forest classifier from each view as the weight of the corresponding dissimilarity matrix. The intuition behind this choice is that if the trained Random Forest classifier has good performance, the dissimilarity measure generated from this Random Forest should be good too. The main advantages of using OOB accuracy to weight each view in the combination is that, in terms of computational cost, OOB measure adds no extra cost as it can be obtained during the training process. OOB measure does not need extra validation neither, which better suits the HDLSS problem.

4.2.3 Experiments and Results

Experimental protocol

In the previous chapter, we showed that $RFDis_{IH}$ achieved the best results with simple averaging over all the dissimilarity matrices, and therefore, it is used as the baseline method in this experiment. The proposed method based on OOB weighting (SW_{OOB}) is compared to two additional methods from the literature: 1. Using kNN classifiers to estimate the quality of each dissimilarity matrix and 2. Using Kernel Alignment. For the first method, a 3NN classifier has been chosen as it is the most popular method in the metric learning literature ([130, 148, 190, 243]). The training accuracy rate of these classifiers is used as weight directly. For the second method, dissimilarity matrices are transformed into similarity matrices as previously explained, and the Equation 4.11 is used to compute the weights; this method is called Dissimilarity Alignment (DA) in the following. A summary of all these Static Weighting (SW) methods are shown in Table 4.1.

TABLE 4.1: An overview of all the methods compared in this section.

Name	Combination method
Avg	Simple average over all matrices
SW_{3NN}	Static weighting by 3NN performance
SW_{DA}	Static weighting by kernel alignment (Equation 4.11)
SW_{OOB}	Static weighting by OOB accuracy (Section 4.2.2)

The experimental protocol and datasets are the same as the previous chapter. We recall that for the RF classifiers used in this experiment, the number of trees is set to 512 according to the recommendation of previous chapters, while the other parameters are set to the default values as proposed in the *Scikit-learn* machine learning framework ([178]).

Results

The average classification rates over the 10 repetitions, along with standard deviations, are shown in Table 4.2. The bold numbers are the best results for each dataset. The proposed approach SW_{OOB} is the best for 9 of the 15 datasets, while SW_{DA} wins 5 times and SW_{3NN} wins 2 times over 15 datasets. For the nus3 dataset, no static weighting method is able to improve the classification performance over the baseline method *Avg*.

TABLE 4.2: The average classification rates of multi-view intermediate integration with different static weighting methods. The dissimilarity measure is RFD_{IH} proposed in the previous chapter. Random Forest is built on the joint dissimilarity matrix to evaluate the performance.

	Avg	SW_{3NN}	SW_{DA}	SW_{OOB}
awa8	56.22% ± 1.01	56.22% ± 0.99	56.12% ± 1.42	**56.59% ± 1.41**
awa15	38.23% ± 0.83	38.13% ± 0.87	**38.27% ± 1.05**	38.23% ± 1.26
Metabo	69.17% ± 5.80	68.54% ± 5.85	**70.00% ± 4.86**	**70.00% ± 6.12**
mfeat	97.53% ± 1.00	97.53% ± 1.09	97.53% ± 1.09	**97.57% ± 1.01**
nus2	92.82% ± 1.93	92.86% ± 1.88	92.60% ± 2.12	**92.97% ± 1.72**
bbc	95.46% ± 0.65	**95.52% ± 0.64**	95.36% ± 0.74	95.46% ± 0.60
lowGrade	63.95% ± 5.62	62.56% ± 6.10	**63.95% ± 3.57**	**63.95% ± 5.01**
nus3	**80.32% ± 1.95**	79.95% ± 2.40	80.09% ± 2.07	80.14% ± 2.20
progression	65.79% ± 4.71	65.79% ± 4.71	65.79% ± 4.99	**66.32% ± 4.37**
LSVT	84.29% ± 3.51	84.29% ± 3.65	84.60% ± 3.54	**84.76% ± 3.63**
IDHCodel	76.76% ± 3.59	77.06% ± 3.43	**77.35% ± 3.24**	76.76% ± 3.82
nonIDH1	80.70% ± 3.76	80.47% ± 3.32	80.00% ± 3.15	**80.93% ± 4.00**
bbcsport	90.18% ± 1.96	**90.29% ± 1.83**	90.26% ± 1.78	90.26% ± 1.95
Cal20	89.76% ± 0.80	89.88% ± 0.82	89.77% ± 0.68	**90.00% ± 0.71**
Cal7	96.03% ± 0.53	96.10% ± 0.57	**96.11% ± 0.60**	96.10% ± 0.60
Avg rank	2.93	2.77	2.57	1.73

The average ranking results show that SW_{OOB} has the best results, followed by SW_{DA} and SW_{3NN}. All three static weighting methods are better than the

simple averaging. Even though the difference of average ranking between the best method (SW_{OOB}) and the worst method (Avg) is obvious, the accuracy differences on each dataset are quite small, rarely above 1%. The difference is even below 0.1% for 7 of the datasets. To see more clearly if any static weighting method is significantly better than the baseline, the pairwise comparison result is shown in Figure 4.1. None of these three static weighting methods is significantly better than Avg.

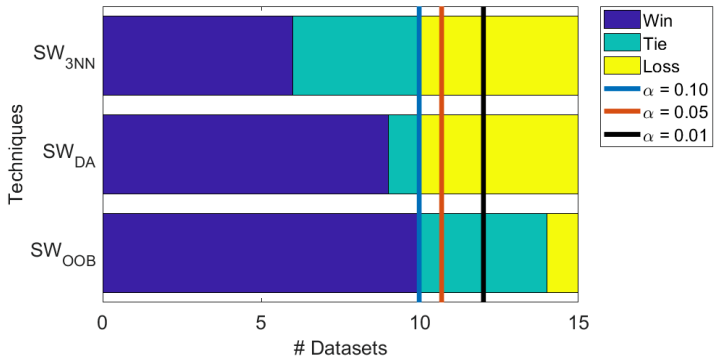


FIGURE 4.1: Pairwise comparison between different combination methods and the simple averaging. The vertical lines illustrate the critical values considering a confidence level $\alpha = \{0.10, 0.05, 0.01\}$.

In a nutshell, these results show that the best performing method SW_{OOB} is not able to have the significant performance improvement. One possible reason is that the views in these datasets are all equally informative over all the instances, which may lead to very similar OOB weights in the combination. And as a consequence, the corresponding combination is very similar to our baseline method, with a simple average. To verify this, the differences between SW_{OOB} and Avg are analyzed in three different ways (Table 4.3). Firstly, the $Weight_{diff}$ values are reported, as the average difference of weight values between SW_{OOB} and Avg; Secondly, $Dissimilarity_{diff}$, as the average difference of dissimilarity values between the joint dissimilarity matrices generated from SW_{OOB} and Avg; and Thirdly, $Prediction_{diff}$, as the average percentage of test instances that have different prediction results between SW_{OOB} and Avg. The results in Table 4.3 show that all three indicators are very low in a

TABLE 4.3: The average difference between SW_{OOB} and Avg in terms of: weight value difference, the corresponding joint dissimilarity matrices difference and the percentage of differently predicted test instances.

	$Weight_{diff}$	$Dissimilarity_{diff}$	$Prediction_{diff}$
awa8	$1.26\% \pm 1.16$	$0.05\% \pm 0.01$	$3.44\% \pm 1.42$
awa15	$1.72\% \pm 1.18$	$0.03\% \pm 0.01$	$9.65\% \pm 1.88$
Metabo	$1.85\% \pm 1.47$	$0.43\% \pm 0.34$	$1.67\% \pm 1.56$
mfeat	$1.62\% \pm 0.48$	$0.15\% \pm 0.01$	$0.70\% \pm 0.48$
nus2	$0.52\% \pm 0.43$	$0.20\% \pm 0.05$	$0.37\% \pm 0.57$
bbc	$0.19\% \pm 0.12$	$0.00\% \pm 0.00$	$0.30\% \pm 0.15$
lowGrade	$1.61\% \pm 0.97$	$0.42\% \pm 0.14$	$1.86\% \pm 1.74$
nus3	$1.27\% \pm 0.63$	$0.20\% \pm 0.05$	$1.53\% \pm 0.91$
progression	$1.17\% \pm 0.77$	$0.36\% \pm 0.15$	$1.58\% \pm 1.75$
LSVT	$0.88\% \pm 0.55$	$0.19\% \pm 0.08$	$0.48\% \pm 1.02$
IDHCodel	$0.70\% \pm 0.64$	$0.26\% \pm 0.17$	$0.59\% \pm 1.18$
nonIDH1	$0.75\% \pm 0.49$	$0.39\% \pm 0.18$	$0.23\% \pm 0.70$
bbcsport	$0.48\% \pm 0.28$	$0.03\% \pm 0.02$	$0.37\% \pm 0.23$
Cal20	$1.09\% \pm 0.37$	$0.19\% \pm 0.01$	$1.56\% \pm 0.32$
Cal7	$0.57\% \pm 0.20$	$0.30\% \pm 0.02$	$0.31\% \pm 0.21$
Avg	1.05%	0.22%	1.64%

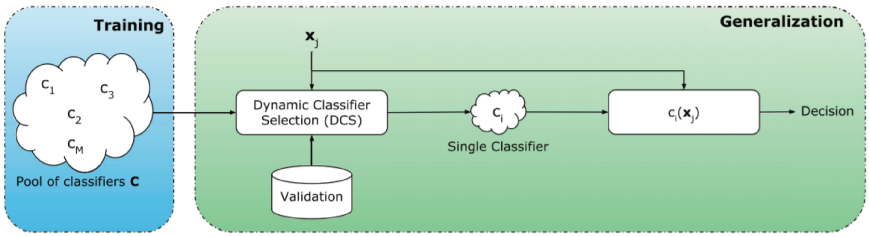
large majority of cases, which indicates that the best static weighted combination found is always very close to the simple average combination. The most plausible interpretation of these results is that no single view is more or less informative in general than all others with regard to the classification task. They all contribute to the good predictions, at least for some of the instances in the problems. However, since the weights provided by the static weighting methods are very close to the uniform weights of the simple average combination, this does not allow us say whether or not some instances could be better predicted if we relied on some views rather than others. This only makes it possible to say that no single view is undoubtedly better or worse than all the others. In the next section, a dynamic view combination method is proposed to enforce larger weight differences in each weighted combination, by using different matrices combinations for different test instances.

4.3 Dynamic view combination

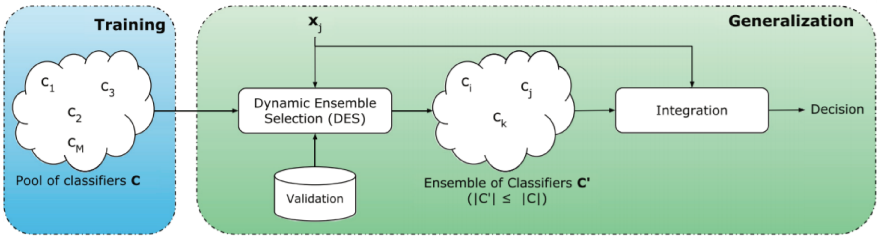
In opposition to static weighting, dynamic weighting aims at assigning different weights to each view for each instance to predict ([164, 189]). The intuition behind dynamic weighting is that, due to the heterogeneity among instances, different instances may rely on different information sources. In Radiomics problem for example, for a patient A, there may be more useful information in one view (e.g. texture or shape features) with regards to the classification task while for a patient B, there may be more useful information in another view (e.g. intensity or wavelet features).

Nevertheless, dynamic weighting is particularly complex in our framework. Let us recall that the multi-representation fusion approach is made up with two stages: 1. inferring the dissimilarity matrices from each view, and 2. combining these dissimilarity matrices to form a new training set. The weights we want to compute with dynamic weighting are the weights used to compute the final joint dissimilarity matrix in stage 2. As a consequence, if these weights change for each test instance, the joint dissimilarity matrix needs to be entirely re-calculated and a new classifier needs also to be re-trained afterwards. This means that, for every new instance to be predicted, a whole training procedure has to be performed. This is computationally expensive and quite inefficient from our point of view.

To avoid this problem, the dynamic view weighting can be replaced by a dynamic view selection procedure. Dynamic Selection (DS) is one of the most successful approaches in Multiple Classifier Systems (MCS) ([71]). DS aims at selecting the most competent classifier(s) for each test instance. When a single classifier is selected, it is called Dynamic Classifier Selection (DCS). When multiple classifiers are selected and combined, it is called Dynamic Ensemble Selection (DES). In this work, we are more interested in DCS to select the best view combination. There are essentially two steps for DCS ([36]): the generation of a pool of classifiers and the selection of the most adequate classifier, as shown in Figure 4.2. In the following sections, we propose a procedure to generate a pool of classifiers with different combinations of views and a selection criterion for dynamic view selection.



(a) Dynamic classifier selection



(b) Dynamic ensemble selection

FIGURE 4.2: Different steps in dynamic selection. The cylinders represent datasets (training or validation). This figure is extracted from [71].

4.3.1 Classifier generation

The generation of the classifier pool plays a very important role in dynamic selection. As the aim is to select the most competent classifier on the fly for each given test instance, the classifiers in the pool should be diverse and accurate. There are a lot of methods proposed to guarantee the diversity in the pool ([82, 137]), and the two most successful methods consist in generating diverse classifiers using: 1. different feature subspace such as multiple feature extractors ([21, 70, 74, 192]) or random subspaces ([20, 213]); 2. different training sets such as Bagging ([34, 214]) , Boosting ([89, 203]) and clustering-based classifier generation approaches ([193]).

In our case, the challenge is not to generate diversity, because the different classifiers in the pool all come from a different set of weights. The challenge is rather to generate these different weight tuples, used to compute the joint dissimilarity matrix, and from which the classifiers are trained afterward. For

such a task, traditional grid search or random search strategies can be used to generate the candidate weight tuples. However, the number of such tuples increases exponentially with respect to the number of views. For example, suppose we use a grid search strategy with weights discretized with 10 different values in $[0, 1]$. For Q views, it will result in 10^Q different weights tuples. Six views would imply to generate 1 million weight tuples and thus 1 million classifiers to train from each of them. This is too computationally expensive in our point of view.

The alternative approach we propose is to select a subset of views for every candidate in the pool, instead of considering a weighted combination of all of them. By doing so, for each instance to predict, we intend to use only the views that are considered informative enough. The selected views are then combined by averaging, since it has been shown in the previous section that the simple average is a strong baseline. For example, if a problem has three views, there are $2^3 - 1 = 7$ different situations by only considering the presence or absence of views. The weights assigned by simple averaging could thus take their values in the following ensembles: $\{[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}], [0, \frac{1}{2}, \frac{1}{2}], [\frac{1}{2}, 0, \frac{1}{2}], [\frac{1}{2}, \frac{1}{2}, 0], [1, 0, 0], [0, 1, 0], [0, 0, 1]\}$ (the situation where all views are not selected is obviously ignored). Based on these weights, a pool of 7 classifiers $\mathbf{C} = \{C_1, C_2, \ldots, C_7\}$ is formed. From this example, it can be seen that compared to a grid search, dynamic view selection has the advantage of using a pool of much smaller size, which reduces significantly the computational cost. Another advantage of dynamic view selection is that it can deal well with missing views. In this chapter, for a fair comparison with the static weighting methods, the Random Forest classifier is chosen to form the pool.

Once the pool of classifiers is generated, the best classifier needs to be selected for each instance to predict. In the next section, a new classifier selection method for HDLSS problems is proposed.

4.3.2 Selection criteria

The selection of the most competent classifier is a key aspect of DS. Generally speaking, the selection procedure can be divided into two steps: 1. define

the region of competence, 2. compute a selection criterion to select the best classifier.

The region of competence Θ_t of each test instance $\mathbf{x}_t$ is the region used to estimate the competence of each classifier for that instance. There exist four categories of methods for defining the region of competence: Clustering, kNN, Decision Space, Potential Functions ([71]). The Clustering and kNN methods are the most widely used in the literature. For Clustering ([138, 217]), the principle is usually to define the region of competence of an instance $\mathbf{x}$ as the closest cluster, according to the distances of $\mathbf{x}$ to the centroids of all the clusters. As the clusters are fixed, many different instances might share the same region of competence. In contrast, kNN methods give different regions of competence from one instance to another, since they are defined by their neighborhood. This allows for more flexibility but at the expense of a higher computational cost ([78]).

The most important part of the selection process is to define the criterion to measure the competence level of each classifier in the pool. There are a lot of methods according to the type of information used to measure the competence, including ranking ([200, 246]), accuracy ([69, 246]), probabilistic ([95, 139, 245]), behavior ([49, 96]), oracle ([134]), data complexity ([38]) and Meta-learning ([75, 183]). We do not give an exhaustive survey of these methods but detail in the following, one of the most representative work, namely Local Classifier Accuracy (LCA, [246]). LCA is a widely used method in the field of dynamic selection with good and robust classification results ([71]).

LCA uses the local accuracy of classifier C_i with respect to the predicted label $\hat{y}_t$ for a given test instance $\mathbf{x}_t$:

$$w_{i,t} = \frac{\sum_{\mathbf{x}_k \in \Theta_{t,\hat{y}_t}} I(C_i(\mathbf{x}_k) = \hat{y}_t)}{\sum_{\mathbf{x}_k \in \Theta_t} I(y_k = \hat{y}_t)} \tag{4.12}$$

where $\Theta_t = \{\mathbf{x}_1, \dots \mathbf{x}_k, \dots, \mathbf{x}_K\}$ is the competence region of the given test instance. These K instances usually come from a validation dataset instead of the training dataset. $\Theta_{t,\hat{y}_t}$ gathers the instances in Θ_t with label $\hat{y}_t$. $I()$ equals to 1 if the expression is true and 0 otherwise. Equation 4.12 simply represents the percentage of correct classifications within the region of competence, but

considering only those instances where the classifier has given the same class as the one it gives for $\mathbf{x}_t$. Many DS methods share the similar idea as LCA, they mainly differ in how to calculate the competence of the classifier in the region of competence.

However, traditional dynamic selection methods demand a validation dataset ([71]), which is not very suitable for HDLSS data. In the following part, a selection criterion that does not rely on validation data is proposed.

To define the appropriate region of competence, we choose the KNN technique. KNN neighborhood is usually based on Euclidean distance. However, the experimental results from the previous chapter have shown that the three RF based dissimilarity measures are significantly better than the Euclidean distance, among which the best performing method is RFD_{IH}. Hence, we propose to use RFD_{IH} as the dissimilarity measure to define the region of competence θ_t of each test instance $\mathbf{x}_t$. To calculate the local confidence $w_{i,t}^{local}$ for the RF classifier $\mathbf{H}_i$, the OOB accuracy is estimated on the region of competence θ_t:

$$w_{i,t}^{local} = OOB_{accuracy}(\mathbf{H}^i, \theta_t) \tag{4.13}$$

The proposed dynamic view selection method using local confidence is summarized in Algorithm 4.

4.3.3 Experiments and Results

Experimental protocol

For the experimental validation of DS_{local}, the neighborhood size K is set to 7 according to the suggestion in [71]. The base classifier for each view combination is the Random Forest classifier. The other parameters are the same as in the previous section. Table 4.4 gathers the results of DS_{local}, as the average classification rates and standard deviations, and all the results from Table 4.2, for comparison purposes. All the four methods reported in this table have been run on the same datasets, with the same replicates.

Results

Algorithm 4 Dynamic view selection (DS_{local})

Input: Q $N \times N$ training dissimilarity matrices $\{\mathbf{D}^{(1)}, \mathbf{D}^{(2)}, \ldots, \mathbf{D}^{(Q)}\}$
  Q $n \times N$ test dissimilarity matrices $\{\mathbf{D}_T^{(1)}, \mathbf{D}_T^{(2)}, \ldots, \mathbf{D}_T^{(Q)}\}$
  $\mathbf{y}$: the labels of training instances
  T_s: the set of l test instances
$\{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{2^Q-1}\} :=$ { all the Q-sized binary vectors, apart from all zeros } ▷ Generation of the candidate subsets of views
$\mathbf{D}_1^m, \mathbf{D}_2^m, \ldots, \mathbf{D}_{2^Q-1}^m$, where $\mathbf{D}_i^m = \frac{\sum_{j=1}^{Q} w_{i,j} \times \mathbf{D}^{(j)}}{||\mathbf{w}_i||}$ ▷ Generation of the joint dissimilarity matrices
$\mathcal{H} = \{\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_{2^Q-1}\}$, where $\mathbf{H}_i = RandomForest(\mathbf{D}_i^m, \mathbf{y})$ ▷ Generation of the pool of classifiers
for each $\mathbf{x}_t in T_s$ **do**
  for each $\mathbf{H}_i$ in $\mathcal{H}$ **do**
    $\mathbf{d}_t^m = \frac{\sum_{j=1}^{Q} w_{i,j} \times \mathbf{d}_T^{(j)}}{||\mathbf{w}_i||}$
    $\theta_t = KNN(RFD_{IH}(\mathbf{H}_i, \mathbf{d}_t^m))$ ▷ Region of competence for $\mathbf{x}_t$
    $w_{i,t}^{local} = OOB_{accuracy}(\mathbf{H}^i, \theta_t)$
  $\mathbf{w}_t^{local} = \{w_{1,t}^{local}, w_{2,t}^{local}, \ldots, w_{2^Q-1,t}^{local}\}$
  $idx = argmax(\mathbf{w}_t^{local})$
Output: $\mathbf{w}_{idx}$

Table 4.4 firstly shows that the DS_{local} outperforms the four competitors for 10 over the 15 datasets. It also shows that DS_{local} is the best ranked over all the datasets, followed by the static weighting method SW_{OOB}. Even though the difference between the average ranks is very small, these two methods are very differents from each others. DS_{local} usually achieves a much higher improvement in classification accuracy than SW_{OOB}. For example, on datasets such as awa8, Metabo, lowGrade, progression and IDHCodel, DS_{local} improves the average accuracy between 1 to 2 percent, while SW_{OOB} only improves the average accuracy between 0.1 to 0.5 percent. On other datasets such as bbcsport, LSVT, nus2, nus3 and awa15, DS_{local} also has better performance than SW_{OOB}.

TABLE 4.4: The experimental results of multi-view intermediate integration with different weighting methods. The dissimilarity measure is RFD_{IH} proposed in the previous chapter.

	Avg	SW_{3NN}	SW_{DA}	SW_{OOB}	DS_{local}
awa8	56.22% ± 1.01	56.22% ± 0.99	56.12% ± 1.42	56.59% ± 1.41	**57.28% ± 1.49**
awa15	38.23% ± 0.83	38.13% ± 0.87	38.27% ± 1.05	38.23% ± 1.26	**38.82% ± 1.56**
Metabo	69.17% ± 5.80	68.54% ± 5.85	70.00% ± 4.86	70.00% ± 6.12	**70.21% ± 4.85**
mfeat	97.53% ± 1.00	97.53% ± 1.09	97.53% ± 1.09	97.57% ± 1.01	**97.63% ± 0.99**
nus2	92.82% ± 1.93	92.86% ± 1.88	92.60% ± 2.12	92.97% ± 1.72	**93.30% ± 1.58**
bbc	95.46% ± 0.65	**95.52% ± 0.64**	95.36% ± 0.74	95.46% ± 0.60	95.42% ± 0.59
lowGrade	63.95% ± 5.62	62.56% ± 6.10	63.95% ± 3.57	63.95% ± 5.01	**65.81% ± 5.31**
nus3	80.32% ± 1.95	79.95% ± 2.40	80.09% ± 2.07	80.14% ± 2.20	**80.77% ± 2.06**
progression	65.79% ± 4.71	65.79% ± 4.71	65.79% ± 4.99	66.32% ± 4.37	**66.84% ± 5.29**
LSVT	84.29% ± 3.51	84.29% ± 3.65	84.60% ± 3.54	**84.76% ± 3.63**	84.44% ± 3.87
IDHCodel	76.76% ± 3.59	77.06% ± 3.43	77.35% ± 3.24	76.76% ± 3.82	**77.65% ± 3.77**
nonIDH1	80.70% ± 3.76	80.47% ± 3.32	80.00% ± 3.15	**80.93% ± 4.00**	79.77% ± 2.76
bbcsport	90.18% ± 1.96	90.29% ± 1.83	90.26% ± 1.78	90.26% ± 1.95	**90.44% ± 1.89**
Cal20	89.76% ± 0.80	89.88% ± 0.82	89.77% ± 0.68	**90.00% ± 0.71**	89.15% ± 0.97
Cal7	96.03% ± 0.53	96.10% ± 0.57	**96.11% ± 0.60**	96.10% ± 0.60	94.65% ± 1.09
Avg rank	3.67	3.50	3.30	2.40	2.13

The pairwise analysis based on the Sign test is used here again to better assess the difference between the baseline method and the proposed dynamic view selection method. The result is shown in Figure 4.3: with $\alpha = 0.05$, the dynamic view selection method DS_{local} is significantly better than the simple average combination.

From our point of view, these results underlines two essential results: First

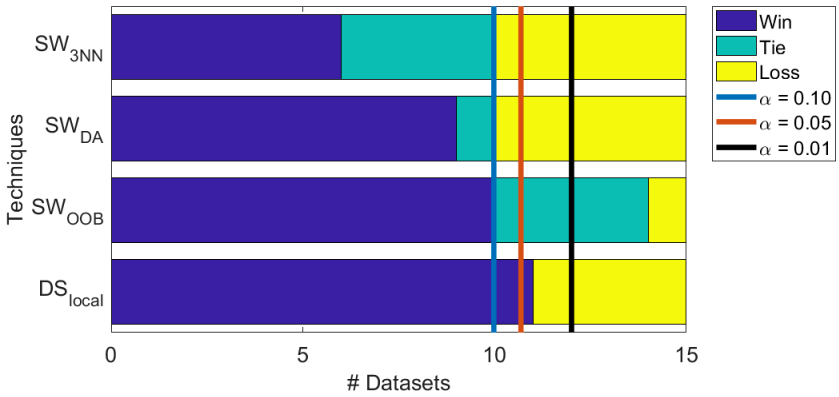


FIGURE 4.3: Pairwise comparison between different combination methods and the simple averaging. The vertical lines illustrate the critical values considering a confidence level $\alpha = \{0.10, 0.05, 0.01\}$.

of all, the simple average combination, when using dissimilarity representation for multi-view learning, is a quite strong baseline. This is consistent with conclusions one can find in the Multiple Kernel Learning literature, as it has already been discussed in the first chapter of this manuscript. Secondly however, the dynamic selection procedure proposed in this chapter allows predominantly to improve the accuracy over this baseline. It shows that all the views do not participate in the same extent to the good prediction of every instance. Some instances are better recognized when the dissimilarities are computed by relying on some views more than the others. These views are certainly not the same from one instance to another, and some instances may need the dissimilarity informations from all the views. Nevertheless, this highlights that the confusion between the classes is not always consistent from one view to another, and we think that this may be amplified for HDLSS problems. In that sense, the views complement each others, and this can be efficiently exploited for multi-view learning provided that we can identify the views that are the most reliable for every instance, one by one.

4.4 Conclusion

In this chapter, we proposed several adaptive ways to combine dissimilarity matrices instead of using the simple average combination as in the previous chapters. As dissimilarity matrices built from different views are comparable, it is more meaningful to combine them in a linear way with their weights representing the importance of each view. Hence different linear combination methods including static weighting and dynamic weighting have been investigated.

Two hypotheses has been tested in this chapter. The first one is that all the views are not globally as informative as each other. A static weighting method SW_{OOB} has been proposed to reduce the influence of the less informative views and to increase the influence of the more informative views. The second hypothesis is that the views do not all contribute in the same way to the good prediction of all instances. With this hypothesis, a dynamic view selection method DS_{local} has been proposed to remove views that are irrelevant for a particular instance to classify.

For the first hypothesis, the experimental results show that, even though the proposed static weighting method is better than the baseline methods on most of the datasets, the performance difference is not significant enough. All the views contribute in the similar way to the problem, at least for the problems that we considered in our experiments. It is rare that one view is indisputably better or worse than all the others and particularly on data extracted from real-world applications like those we tested. As for the second hypothesis, in contrast to static weighting methods, the dynamic weighting method based on local accuracy has more potential (DS_{local} got better performance than SW_{OOB} on 10 out of 15 datasets). The results presented in this chapter show that a significant part of the instances can be better recognized if we only consider a well selected subset of views.

However, in its current form, the dynamic selection method proposed in this chapter strongly depends on the number of candidate classifiers in the pool. This pool is limited by the number of views available in the datasets. However, as explained in this chapter, the diversity in the pool is a key property

for the DCS approach to be relevant. It allows in particular to propose many different candidate solutions from which the best one is selected. To allow for more versatility, we think it could be interesting to decompose each view into several sub-views. It could be done for example, by using Bagging and Random Subspaces principles before computing the view-specific dissimilarities. In such a way, the dynamic combination could only select some specific part of each view, instead of considering the views as a whole.

General Conclusion

Multi-view data are now very common in real world applications. Whether they arise from multiple sources or from multiple feature extractors, the multiple views are supposed to provide a more accurate and complete description of objects than a single description would do. The starting point of this thesis work is one of such real-world applications: Radiomics. This medical imaging classification problem is a typical multi-view problem because the purpose of Radiomics is precisely to "describe" the patients through several image modalities and several types of features extracted from these images.

In addition to the multi-view aspect, another machine learning challenge is underlying the Radiomics problems: High feature Dimension, Low Sample Size (HDLSS). As a medical related Pattern Recognition problem, the amount of data available for learning to solve Radiomics problems is always very limited, and at the same time, the features extracted from the medical images are always very numerous.

The goal of the thesis was to provide solutions to HDLSS multi-view learning problems, such as Radiomics problems but not exclusively. The main challenge is that most of the state-of-the-art multi-view learning methods are not always suitable for HDLSS problems. Therefore, our proposal in this work was to address HDLSS multi-view problems using methods based on dissimilarities. Dissimilarity strategies allow to overcome the well-known issues that arise from HDLSS problems, and to give an efficient way to handle the heterogeneity of the multiple views at the same time. Three major contributions have been proposed in this direction, and are summarized below.

The first contribution, introduced in Chapter 2, is the design of a global framework consisting of building an intermediate representation based on dissimilarity for each view, and combining these intermediate representations for learning. The key mechanism is to use Random Forest classifiers to measure the dissimilarities. Random Forests embed a (dis)similarity measure, called the RFD measure, that does not suffer from the high dimensions and that takes the class membership into account in such a way that instances from the same class are similar. The resulting dissimilarity representations are not HDLSS anymore and can efficiently be merged since they are fully comparable from one view to another.

The second contribution, described in Chapter 3, focuses on proposing a new dissimilarity measure, still based on Random Forest, that better deals with HDLSS issues. As our initial RFD measure was based on the Decision Tree dissimilarity estimates that take their values in $\{0, 1\}$, two approaches are proposed to make the RFD value more accurate : the first one (RFD_{NC}) uses leaf node confidence estimates to weight the dissimilarity between instances that land in this node; the second one (RFD_{IH}) uses an *Instance Hardness* estimator, applied on the training instances in the same node to refine their dissimilarity values. For both HDLSS single-view and HDLSS multi-view situations, the proposed methods achieve better performance than the initial RFD measure.

The third and final contribution, presented in Chapter 4, concerns the design of a dynamic view selection method that provides a better way of merging the per-view dissimilarity representations. In the experiments of Chapter 2 and 3, the multi-view dissimilarity matrices were merged with a simple average. Even though the flat average is considered as a strong baseline in the literature, it does not take the view importance into account. The rationale behind using weighted combinations of views is to base the decision on the relevant views primarily, and to ignore as much as possible the irrelevant views. We have shown that the dynamic selection of views we propose achieves significant improvement against the simple average.

In a nutshell, the RFD based intermediate integration method enables to face both the HDLSS and the multiview issues. However, there are still open issues and room for improvement. We detail the most important and promising ones from our point of view in the following.

Even though the high dimensional challenge is overcame by the dissimilarity space, the low sample size challenge may be still an issue for the proposed framework. Indeed the mechanism we propose in this work strongly rely on the OOB measure. However, this measure is very sensitive to the size of the initial training set : the measure is based on about one third of the training instances as explained in Chapter 3 and if there are not enough instances (which is the case when dealing with LSS data), this measure may not be reliable enough. This is even more critical for imbalanced LSS datasets for which there

is still a strong risk with our proposed framework to ignore under-represented classes in the dissimilarity space: only few dimensions of this space would be related to the minority class. Therefore we believe it deserves to be further explored. One possible direction to investigate would be the use of ad-hoc ensemble techniques. For example data augmentation methods such as the Synthetic Minority Over-Sampling (SMOTE) technique ([54]) could be tested. SMOTE was originally proposed to create artificial instances for the minority class of imbalanced datasets. It can also be used to create artificial instances for all the classes to overcome the LSS issue in general. Note however that this would result in higher dissimilarity space dimension since it corresponds to the number of instances in the training set. One possible solution to face this issue would be to select a subset of reference instances (or prototypes) to which dissimilarities are measured, as it is commonly done in the dissimilarity space approach ([67]).

As we pointed out in Chapter 4, our framework seems to be all the more successful when there is heterogeneity and disagreement between views, similar to the diversity concept in ensemble learning. The diversity in the pool of classifiers is a key property for the dynamic selection approach to be relevant, as it allows in particular to propose many different candidate solutions from which the best one is selected. Popular ensemble learning methods could be transposed in our framework to enhance this diversity. For example, before computing the view-specific dissimilarities, we could generate random subviews by using the random subspaces mechanism along with Bagging, instead of considering each view as a whole. Doing so, the final predictions could be given by selecting the more reliable parts of each view and by ignoring the irrelevant ones, using the same dynamic selection mechanism as in our framework.

Finally, as explained in Chapter 1, the approach that has been adopted for this work is the intermediate integration method, the most relevant approach in the literature for dealing with HDLSS multi-view problem. However, in preliminary works on Radiomics datasets we report in Appendix B, we have also shown that there is some potential to use the late integration method. The method proposed in this preliminary work was to first generate dissimilarity space for each view, then learn one classifier per dissimilarity space

and combine these classifiers at last. Compared to the multi-view late integration methods proposed in the literature, this can be seen as a simple and naive procedure since the classifiers do not cooperate together. Indeed, the state-of-the-art methods in multiview learning are co-training based, that is to say classifiers are trained jointly to maximize their agreement. This could be adapted to our framework by "co-generating" the dissimilarity spaces. The main obstacle as explained in Chapter 1 would be the lack of additional instances that could be circumvent by the OOB mechanism.

List of publications

1. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. Dissimilarity-based representation for Radiomics applications. First International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI) (2018).

2. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. Pondération dynamique en apprentissage multi-vues pour des applications radiomics in Conférence sur l'apprentissage automatique (cap) (2018).

3. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. Dynamic voting in multi-view learning for Radiomics applications in Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR) (2018), 32–41.

4. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. Improve the performance of transfer learning without fine-tuning using dissimilarity based multi-view learning for breast cancer histology images in International Conference on Image Analysis and Recognition (2018), 779–787.

5. Cao, H., Bernard, S., Sabourin, R. & Heutte, L. Random forest dissimilarity based multi-view learning for Radiomics application. Pattern Recognition 88, 185–197 (2019).

Appendix A

Transfer learning without fine-tuning for breast cancer histology images

Published as: Cao, H., Bernard, S., Heutte, L. and Sabourin, R. Improve the performance of transfer learning without fine-tuning using dissimilarity based multi-view learning for breast cancer histology images. In *Proc. of ICIAR 2018*, LNCS 10882, Springer, pp. 779-787, 2018.

Abstract: Breast cancer is one of the most common types of cancer and leading cancer-related death causes for women. In the context of ICIAR 2018 Grand Challenge on Breast Cancer Histology Images, we compare one handcrafted feature extractor and five transfer learning feature extractors based on deep learning.We find out that the deep learning networks pretrained on ImageNet have better performance than the popular handcrafted features used for breast cancer histology images. The best feature extractor achieves an average accuracy of 79.30%. To improve the classification performance, a random forest dissimilarity based integration method is used to combine different feature groups together. When the five deep learning feature groups are combined, the average accuracy is improved to 82.90% (best accuracy 85.00%). When handcrafted features are combined with the five deep learning feature groups, the average accuracy is improved to 87.10% (best accuracy 93.00%).

A.1 Introduction

The detection and treatment of cancer are still very challenging. The normal process of cancer detection is from certain signs and symptoms to the further investigation by medical imaging and at last confirmed by biopsy ([3, 64]). The diagnosis of breast cancer usually uses the biopsy tissue. The pathologists can histologically assess the microscopic structure and elements of the tissue from breast tissue biopsies ([14]).

One of the most important method for tumor histological examination in pathology is Hematoxylin and eosin (H&E) staining ([51]). However, manual analysis is experience based, qualitative and always causes intra- or inter-observers variation even for experienced pathologists ([165]). Hence developing a more efficient, accurate, quantitative and automated system is necessary and urgent. Due to the high performance of deep learning networks, more and more studies used deep learning for the classification of breast cancer images ([221]). However, the number of images available has always been an obstacle for the use of deep learning. Many studies divide images into patches for data augmentation, but the new problem is that there are no label information for patches.

In this paper, transfer learning without fine-tuning is proposed to solve the above problems. Six different feature extractors are compared, including five deep learning architectures and a traditional feature extractor combining PFTAS (Parameter-Free Threshold Adjacency Statistics) and GLCM (Gray Level Co-Occurrence Matrices) features. When all features are combined, there are mainly three challenges from the machine learning point of view: (i) small sample size: size: like most other medical applications, the number of breast cancer histology images is very small (400 images); (ii) high dimensional feature space: as six groups of features may be combined, the size of the feature space may be up to 31855, which is over 80 times bigger than the sample size; (iii) multiple feature groups: it may be hard to improve the learning performance by exploiting the complementary information that different groups contain ([44]). To deal with these three challenges, we propose to treat breast cancer histology image classification as a multi-view learning problem.

A multi-view RFSVM method proposed in our previous work ([44]) is then used as a solution.

The remainder of this paper is organized as follows: the six feature extractors are detailed in Section II; in Section III, the dissimilarity based multi-view learning solution is introduced; we describe in Section IV the data sets chosen in this study and provide the protocol of our experimental method; we analyze in Section V the results of our experiments; the final conclusion and future works are drawn in Section VI.

A.2 Feature extractors

In total six different feature extractors are used in this work: handcrafted features, ResNet-18, ResNeXt, NASNet-A, ResNet-152 and VGG16. In this section, a brief introduction of each feature extractor is given. The handcrafted features include PFTAS and GLCM and have been chosen due to their good performance on breast cancer histology image classification ([220]). The five deep learning networks have been chosen for their performance and because they are built on different structures with different depths, and the pre-trained models are available online[1][2].

Handcrafted features:

Two kinds of feature extractors are combined together to form the handcrafted feature group: PFTAS and GLCM. TAS (Threshold Adjacency Statistics) is a simple and fast morphological measure for cell phenotype image classification presented by Hamilton et al. in [110]. Similar to the work of [220], we use the Parameter-Free Threshold Adjacency Statistics (PFTAS) from the python library Mahotas ([62]) to build a 162-dimensional PFTAS-feature vector. GLCM features are widely used to describe the texture of tumor in cancer applications. Same as PFTAS, the library Mahotas is used to calculate the GLCM features leading to a 175-dimensional GLCM-feature vector.

[1] https://github.com/Cadene/pretrained-models.pytorch
[2] https://github.com/pytorch/vision/tree/master/torchvision

ResNet-18 and ResNet-152:

ResNet is one of the deepest deep learning architectures proposed by Microsoft researchers. The deep residual nets based methods have won the first places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation as well as the first place on the ILSVRC 2015 classification task ([115]). We use two ResNet in this work: ResNet-18 and ResNet-152. Both networks take as input a {3, 224, 224} RGB image and are pretrained on ImageNet with 1000 classes[3][4]. Features are extracted from the average pool layer (i.e. before the last classification layer), which results in 512 features for ResNet-18 and 2048 features for ResNet-152.

ResNeXt:

ResNeXt is one of the state-of-the-art techniques for object recognition. It builds upon the concepts of repeating layers while exploiting the split transform merge strategy to bring about a new and improved architecture ([250]). The input space of ResNeXt is a {3, 224, 224} RGB image and we use the network pretrained on ImageNet with 1000 classes[5]. 2048 features are extracted from the average pool layer (i.e. before the last classification layer).

NASNet-A:

In the work of [262], the authors proposed to search for an architectural building block on a small dataset and then transfer the block to a larger dataset to reduce the computation cost and improve the efficiency. They used NAS (Neural Architecture Search) framework from [261] as the main search method for their NASNets. The three networks constructed from the best three searches are named NASNet-A, NASNet-B and NASNet-C respectively. In this work, a NASNet-A pretrained on ImageNet is used. The input space of NASNet-A is a {3, 331, 331} RGB image and we use the network pretrained on ImageNet

[3]https://download.pytorch.org/models/resnet18-5c106cde.pth
[4]http://data.lip6.fr/cadene/pretrainedmodels/fbresnet152-2e20f6b4.pth
[5]http://data.lip6.fr/cadene/pretrainedmodels/resnext101-64x4d-e77a0586.pth

with 1001 classes (ImageNet+background)[6]. 4032 features are extracted from the last layer before the classification layer.

VGG16:

The VGG Network was introduced by the researchers at Visual Graphics Group at Oxford ([212]). This network is specially characterized by its pyramidal shape. VGG16 takes as input a {3, 224, 224} RGB image and we use the network pretrained on ImageNet with 1000 classes[7]. Features are extracted from the last max pooling layer, which results in 512x7x7 features.

A.3 Dissimilarity-based learning

In our previous work ([44]), we proposed to use RFSVM to integrate information from different views together (each feature group is a view in multi-view learning framework). We have shown that RFSVM offers a good performance on Radiomics data. The RFSVM method can deal well with high dimensional low sample size multi-view data because: (i) RFSVM uses random forest dissimilarity measure to transfer each view of the data to a dissimilarity matrix so that the data dimension is reduced without feature selection, and at the same time the data in each view become directly comparable; (ii) RFSVM can take advantage of the complementary information contained in each view by combining the dissimilarity matrices together. We now recall the RFSVM method.

Random forest: Given a training set $\mathcal{T}$, a Random Forest classifier $\mathbf{H}$ is a classifier made up of M trees denoted as in Equation (A.1):

$$\mathbf{H}(\mathbf{x}) = \{h_k(\mathbf{x}), k = 1, \ldots, M\} \tag{A.1}$$

where $h_k(\mathbf{x})$ is a random tree grown using the Bagging and the Random Feature Selection techniques as in [26]. For predicting the class of a given query point $\mathbf{x}$ with such a tree, $\mathbf{x}$ goes down the tree structure, from its root till its

[6]https://data.lip6.fr/cadene/pretrainedmodels/nasnetalarge-a1897284.pth
[7]https://download.pytorch.org/models/vgg16-397923af.pth

terminal node. The prediction is given by the terminal node (or leaf node) in which $\mathbf{x}$ has landed. We refer the reader to [26] for more information about this process. Hence if two query points land in the same terminal node, they are likely to belong to the same class and they are also likely to share similarities in their feature vectors, since they have followed the same descending path.

Random Forest Dissimilarity (RFD): the RFD measure is inferred from a RF classifier $\mathbf{H}$, learned from $\mathcal{T}$. Let us firstly define a dissimilarity measure inferred by a decision tree $d^{(k)}$: let L_k denote the set of leaves of the kth tree, and let $l_k(\mathbf{x})$ denote a function from $\mathbb{X}$ to L_k that returns the leaf node of the kth tree where a given instance $\mathbf{x}$ lands when one wants to predict its class. The dissimilarity measure $d^{(k)}$, inferred by the kth tree in the forest is defined as in Equation (A.2): if two training instances $\mathbf{x}_i$ and $\mathbf{x}_j$ land in the same leaf of the kth tree, then the dissimilarity between both instances is set to 0, else set to 1.

$$d^{(k)}(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 0, & \text{if } l_k(\mathbf{x}_i) = l_k(\mathbf{x}_j) \\ 1, & \text{otherwise} \end{cases} \tag{A.2}$$

The RFD measure $d^{(\mathbf{H})}$ consists in calculating the $d^{(k)}$ value for each tree in the forest, and to average the resulting dissimilarity values over the M trees, as in Equation (A.3):

$$d^{(\mathbf{H})}(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{M} \sum_{k=1}^{M} d^{(k)}(\mathbf{x}_i, \mathbf{x}_j) \tag{A.3}$$

Multi-view learning dissimilarities: For multi-view learning tasks, the training set $\mathcal{T}$ is composed of K views: $\mathcal{T}^{(k)} = \{(\mathbf{x}_1^{(k)}, y_1), \ldots, (\mathbf{x}_N^{(k)}, y_N)\}$, k=1..K. Firstly, for each view $\mathcal{T}^{(k)}$, the RFD matrix is computed and noted as $\{\mathbf{D}_{\mathbf{H}}^k, k = 1..K\}$. In multi-view learning, the joint dissimilarity matrix can typically be computed by averaging over the Q matrices as in Equation (A.4):

$$\mathbf{D}_{\mathbf{H}} = \frac{1}{Q} \sum_{i=1}^{K} \mathbf{D}_{\mathbf{H}}^i \tag{A.4}$$

Multi Random Forest kernel SVM (RFSVM): From the joint RFD matrix $\mathbf{D}_{\mathbf{H}}$

of Equation (A.4), one can calculate the joint similarity matrix $\mathbf{S_H}$ as in Equation (A.5):

$$\mathbf{S_H} = \mathbf{1} - \mathbf{D_H} \tag{A.5}$$

where $\mathbf{1}$ is a matrix of ones. SVM is one of the most successful classifier. Apart from the most used gaussian kernel, a lot of custom kernels can also be used: we use the joint similarity matrix $\mathbf{S_H}$ inferred from the RF classifier $\mathbf{H}$ as a kernel in a SVM classifier.

A.4 Experiments

The dataset used in this work is from ICIAR 2018 Grande Challenge on BreAst Cancer Histology images[8]. It is composed of Hematoxylin and eosin stained breast histology microscopy images. Microscopy images are labeled as normal, benign, in situ carcinoma or invasive carcinoma according to the predominant cancer type in each image. It is a balanced dataset with in total 400 images.

The protocol of the experiments is as follows:

- First, the 6 feature extractors described in Section 2 are used to extract features from histology image data. As there is no patch label provided, to simplify the feature extracting process, all images are rescaled to the network input size.
- Second, for each group of features, a random forest with 500 trees is built. The performance of each feature group is measured by the classification accuracy of the random forest. The random forest dissimilarity matrix is calculated for each group too.
- Finally, the RFSVM method described in Section 3 is used to combine all the groups together. Two RFSVMs are used: *RFSVM (DL only)* combines the five deep learning based feature groups; *RFSVM-All* combines all the six feature groups. For RFSVM, the search range of parameter C for SVM is {0.01, 0.1, 1, 10, 100, 1000}.

[8] https://iciar2018-challenge.grand-challenge.org/dataset/

Note that in [28], the authors found that when dealing with high dimensional low sample size data, stratification of the sampling is central for obtaining minimal misclassification. In this work, the stratified random splitting procedure is repeated 10 times, with 75% as training data and 25% as testing data. In order to compare the methods, the mean and standard deviations of accuracy were evaluated over the 10 runs. However, for the contest, only one model can be submitted. Hence the best performance among the 10 runs is also presented and chosen as the model for the contest.

A.5 Results

The results of the experiments are shown in Table A.1. We can tell that the best feature extractor is ResNet-152 with an average accuracy of 79.30% and best accuracy of 83.00%. Followed by ResNeXt with an average accuracy of 78.60% and the best accuracy of 81.00%. Surprisingly, the worst feature extractor is handcrafted features with PFTAS and GLCM with an average accuracy of 67.00%. In the work of [220], PFTAS and GLCM are the best features for breast cancer histology image classification. By comparing the performance of the six feature extractors, we can see that even though the deep learning networks are pretrained on ImageNet dataset, which is very different from histology images, they still have a better performance as a feature extractor for breast cancer data than the best handcrafted feature extractor used in the field of breast cancer histology image classification.

With *RFSVM (DL only)* integrating all the five deep learning based feature groups together, the average accuracy is improved to 82.90% and the best performance is improved to 85.00%. However, when all feature groups are combined with *RFSVM-All*, the average accuracy is improved to 87.10% and the best performance is improved to 93.00%. It shows that even though the handcrafted features do not have a very good performance individually, they can still provide useful complementary information for breast cancer classification when combined with deep learning based feature groups.

TABLE A.1: The image wise classification results with 75% training data and 25% test data. *Average* is the average accuracy over 10 runs, *Best* is the best accuracy among the 10 runs.

	Average	Best
Handcrafted	$67.00\% \pm 5.46$	76.0%
ResNet-18	$75.10\% \pm 5.46$	78.0%
ResNeXt	$78.60\% \pm 1.74$	81.0%
NASNet-A	$74.70\% \pm 2.33$	78.0%
ResNet-152	$79.30\% \pm 3.20$	83.0%
VGG16	$68.00\% \pm 5.04$	78.0%
RFSVM(DL only)	$82.90\% \pm 1.37$	85.0%
RFSVM-All	**87.10**$\% \pm 2.17$	**93.0**%

TABLE A.2: The confusion matrix, sensitivity and specificity of our best model.

	Benign	InSitu	Invasive	Normal
Benign	23	1	0	1
InSitu	0	23	0	2
Invasive	1	0	24	0
Normal	2	0	0	23
Sensitivity	92%	92%	96%	92%
Specificity	85%	96%	100%	85%

The confusion matrix, sensitivity and specificity of our best model are shown in Table A.2. From the results we can see that our model has very high sensitivity on all four classes, and very high specificity too for two classes, i.e. *InSitu* and *Invasive*.

Note that the state of the art performance on this dataset is considered to be from [14]. In this work, the authors used CNN patch-wise training on a previous version of the dataset with 249 images for training and 20 images for testing (7.4% of the whole dataset as test data). They obtained as best performance an accuracy of 85.00%. In our work, 300 images are used for training and 100 images are used as test data. Hence, even if the results are not directly comparable with [14], the accuracy of our best model is 8% higher than the accuracy reported in [14] while using 25% of the whole dataset as test data, which is much more than 7.4% in [14].

A.6 Conclusion

In this work, we firstly compared the popular handcrafted features used in breast cancer histology image classification with five deep learning based feature extractors pretrained on ImageNet. Not surprisingly, the experimental results show that the deep learning based features are better than the handcrafted. To improve the performance of transfer learning, we tackled the problem of breast cancer histology image classification as an HDLSS multi-view learning task and applied an RFSVM method previously proposed for the classification of Radiomics data. The results obtained with *RFSVM (DL only)* show that the performance of transfer learning can be improved by combining multiple feature extractors together. The results obtained with *RFSVM-All* show that even though deep learning based features have better performance than handcrafted features for breast cancer histology image classification, the accuracy can be improved significantly when they are combined together and surpass the state of the art performance on the dataset used.

Acknowledgment

This work is part of the DAISI project, co-financed by the European Union with the European Regional Development Fund (ERDF) and by the Normandy Region.

Appendix B

Dynamic voting in multi-view learning for Radiomics applications

Published as: Cao, H., Bernard, S., Heutte, L. and Sabourin, R. Dynamic voting in multi-view learning for radiomics applications. In *Proc. of S+SSPR 2018*, LCNS 11004, Springer, pp. 32-41, 2018.

Abstract: Cancer diagnosis and treatment often require a personalized analysis for each patient nowadays, due to the heterogeneity among the different types of tumor and among patients. Radiomics is a recent medical imaging field that has shown during the past few years to be promising for achieving this personalization. However, a recent study shows that most of the state-of-the-art works in Radiomics fail to identify this problem as a multi-view learning task and that multi-view learning techniques are generally more efficient. In this work, we propose to further investigate the potential of one family of multi-view learning methods based on Multiple Classifier Systems where one classifier is learnt on each view and all classifiers are combined afterwards. In particular, we propose a random forest based dynamic weighted voting scheme, which personalizes the combination of views for each new patient to classify. The proposed method is validated on several real-world Radiomics problems.

B.1 Introduction

One of the biggest challenges of cancer treatment is the inter-tumor heterogeneity and intra-tumor heterogeneity. It demands for more personalized treatment. In Radiomics, a large amount of features from standard-of-care images obtained with CT (computed tomography), PET (positron emission tomography) or MRI (magnetic resonance imaging) are extracted to help the diagnosis, prediction or prognosis of cancer [44]. Many medical image studies like [215, 219] have already tried to use quantitative analysis before the existence of Radiomics. However, with the development of medical imaging technology and more and more available softwares allowing for more quantification and standardization. Radiomics focuses on improvements of image analysis, using an automated high-throughput extraction of large amounts of quantitative features [142]. Radiomics has the advantage of using more useful information to make optimal treatment decisions (personalized medicine) and make cancer treatment more effective and less expensive.

Radiomics is a promising research field for oncology, but it is also a challenging machine learning task. In the work [44], the authors identify Radiomics as a challenge in machine learning for the three following reasons: **(i) small sample size**: due to the difficulty in data sharing, most of Radiomics data sets have no more than 200 patients; **(ii) high dimensional feature space**: the feature space for Radiomics data is always very high dimensional compared to the sample size; **(iii) multiple feature groups**: different sources and different feature extractors are used in Radiomics - the most used features include tumor intensity, shape, texture, and so on[3] - and it may be hard to exploit the complementary information brought by these different views [44].

When the three challenges are encountered in a classification task, it can be seen as an HDLSS (High dimension low sample size) Multi-View learning task. Now most studies in Radiomics ignore the third challenge and propose to simply concatenate different feature groups and to use a feature selection method to reduce the dimension. However, a lot of useful information may be lost when only a small subset of features is retained [44], and the complementary information that different feature groups can offer may be ignored [46].

In contrast to the current studies that treat Radiomics data as a single-view machine learning task, we have proposed in our previsous work to cope with Radiomics complexity using an HDLSS multi-view paradigm [44]: we have used a naive MCS (Multiple Classifier Systems) based method which turns out to work well for Radiomics data but not significantly better than the state of the art methods used in Radiomics. Here we want to further investigate the potential of the MCS multi-view approach. Hence we propose several less simplistic MCS based methods including static voting and dynamic voting methods to combine classification results from different views. Our main contribution in this paper is thus to propose a new dynamic voting scheme to give a personalized diagnosis (decision) from Radiomics data. This dynamic voting method is designed for small sample sized dataset like Radiomics data and uses a large number of trees in random forest to provide OOB (Out Of Bag) samples to replace the validation dataset.

The remainder of this paper is organized as follows. Related works in Radiomics and multi-view learning are discussed in Section 2. In section 3, the proposed dynamic voting solution is introduced. Before turning to the result analysis (Section 5), we describe the data sets chosen in this study and provide the protocol of our experimental method in Section 4. We conclude and give some future works in Section 6.

B.2 Related Works

In the state of the art of Radiomics, groups of features are most often concatenated into a single feature vector, which results in an HDLSS machine learning problem. In order to reduce the high dimensionality, some feature selection methods are used : in the work of [174] and [3], they used feature stability as a criterion for feature selection While in the work of [218], they used a SVM (Support Vector Machine) classifier as a criterion to evaluate the predictive value of each feature for pathology and TNM clinical stage. Different filter feature selection methods have also been compared along with reliable machine learning methods to find the optimal combination [174]. Generally speaking,

the embedded feature selection method SVMRFE shows good performance on different Radiomics applications [44].

A lot of studies have been done on multi-view learning and according to the work of [204], there are three main kinds of solutions: early integration, intermediate integration and late integration. Early integration concatenates information from different views together and treats it as a single-view learning task[204]. The Radiomics solutions discussed above all belong to this category. Intermediate integration combines the information from different views at the feature level to form a joint feature space. Late integration method firstly builds individual models based on separate views and then combines these models. Compared to intermediate and late integration methods, early integration always leads to high dimensional problems and the feature selection methods used in the state of the art of Radiomics can easily filter a lot of useful information.

In [44], MCS based late integration methods (with simple majority voting) have shown a big potential and a lot of flexibility on Radiomics data. In this work, to further investigate the potential of MCS for Radiomics applications, both static and dynamic combinations are tested. The intuition behind static weighted voting is that different views have different importances for a classification task. While the intuition behind proposing dynamic voting methods is that, due to the heterogeneity among patients, different patients may rely on different information sources. For example, for a patient A, there may be more useful information in one view (e.g. texture or shape features) while for a patient B, there may be more useful information in another view (e.g. intensity or wavelet features). Three dynamic integration methods were considered in the work of [227]: DS (Dynamic Selection), DV (Dynamic Voting), and DVS (Dynamic Voting with Selection). The difficulty in multi view combination is that the number of views is fixed and usually very small. In this case, dynamic selection methods may not be applicable. Hence, we focus on dynamic voting method in this work. However, traditional dynamic voting methods demand a validation dataset [71]. In Radiomics, the data size is too small to have a validation dataset. In the next section, we propose a dynamic voting method based on the random forest dissimilarity measure and the Out-Of-Bag (OOB) measure, without the need of validation dataset.

B.3 Proposed MCS based solutions

As explained in the Introduction, the simple MCS based late integration used in [44] has shown a good potential for Radiomics. In this section, we use several more intelligent voting methods including static voting and dynamic voting to test if they can get significantly better.

For multi-view learning tasks, the training set $\mathbf{T}$ is composed of Q views: $\mathbf{T}^{(q)} = \{(\mathbf{X}_1^{(q)}, y_1), \ldots, (\mathbf{X}_N^{(q)}, y_N)\}, q = 1..Q$. Generally speaking, the MCS based late integration method builds a classifier $C^{(q)}$ for each view $\mathbf{T}^{(q)}$. During test time, for each test data $\mathbf{X}_t$, $C^{(q)}$ will predict the class label $label_t^{(q)}$ of $\mathbf{X}_t$. Finally, the predicted labels from all the views $\{label_t^{(1)}, label_t^{(2)}, \ldots, label_t^{(Q)}\}$ can be combined either by majority voting or weighted voting.

Here Random forest is chosen as the classifier for each view $\mathbf{T}^{(q)}$ because it can deal well with different data types, mixed variables and high dimensional data [44]. Random forest can also offer the OOB measure, which can be used as a measure for static weight and also to replace extra validation dataset for dynamic voting methods. In addition, random forest also provides a proximity measure, which can be used to calculate the neighborhood of a test sample[227].

Firstly, for each view q, a Random Forest $\mathbf{H}^{(q)}$ is built with M decision trees, and is denoted as in Equation (B.1):

$$\mathbf{H}(\mathbf{X}) = \{h_k(\mathbf{X}), k = 1, \ldots, M\} \tag{B.1}$$

where $h_k(\mathbf{X})$ is a random tree grown using bagging and random feature selection. We refer the reader to [26, 35] for more details about this procedure.

For a J-class problem with $label_t^{(q)} = i$, where $i \in \{1, 2, \ldots, J\}$, a weight $W^{(q)}$ is used for each view q (for the case of majority voting, all $W^{(q)} = 1$). The final decision is made by:

$$y_t = \underset{j \in \{1,2,\ldots,J\}}{Max} \left(\sum_{q=1}^{Q} I(label_t^{(q)} = j) \times W^{(q)} \right) \tag{B.2}$$

$I()$ is an indicator function, which equals to 1 when the condition in the parenthesis is fulfilled and 0 otherwise.

B.3.1 WRF (Static Weighted Voting)

To calculate the weights for static voting, we need a measure to reflect the importance of each view to give a final decision. Usually, the prediction accuracy over a validation dataset can be used for that. However, Radiomics data have very small sample size, and it is impossible to have extra validation data. Hence we propose to use the OOB accuracy of each random forest $\mathbf{H}^{(q)}$ as the static weight $W^{(q)}$ for each view:

$$W^{(q)}_{static} = OOB_{accuracy}(\mathbf{H}^{(q)}) \tag{B.3}$$

When Bagging is used in a random forest, each bootstrap sample used to learn a single tree is typically a subset of the initial training set. This means that some of the training instances are not used in each bootstrap sample (37% in average; see [33] for more details). For a given decision tree of the forest, these instances, called the Out-of-bag (OOB) samples, can be used to estimate its accuracy. To use OOB to measure the accuracy of a random forest, the concept of sub-forest is used. When the forest size is big, all training data have a high probability to be an OOB sample at least once. Hence, for each OOB sample $\mathbf{X}_{OOB}$, the trees that did not use this data as training sample are grouped together as a sub-forest $\mathbf{H}_{sub(\mathbf{X}_{OOB})}$ (which can be seen as a representative of the complete random forest $\mathbf{H}$) to give a prediction on $\mathbf{X}_{OOB}$. The overall accuracy of the sub-forests predictions on all OOB samples is then used as OOB accuracy for a random forest $\mathbf{H}$. We refer the reader to the work of [33] for further information about OOB measure.

B.3.2 GDV (Global Dynamic Voting)

In static voting, we believe that different views have different importances for classification. However, with dynamic voting, we can personalize this importance with an assumption that the importances of views are different for

different patients. One easy access to this kind of "personalized" information is the prediction probability of each test sample as it shows generally how confident the classifier C^q is on the test data.

The predicted class probabilities of a test sample $\mathbf{X}_t$ for random forest are computed as the mean predicted class probabilities of the trees in the forest. The class probabilities of a single tree is the fraction of samples of the same class in a leaf. The global weight $W_{global}^{(q)}$ of view q for each test data $\mathbf{X}_t$ is simply the predicted probability (posterior probability obtained from $\mathbf{H}^{(q)}$) for the most confident class of random forest, which measures the overall confidence rate of label prediction based on all the training data:

$$W_{global}^{(q)} = P(label_t^{(q)} \mid \mathbf{X}_t, \mathbf{H}^{(q)}) \tag{B.4}$$

$W_{global}^{(q)}$ generally reflects how confident the classifier $\mathbf{H}^{(q)}$ is when predicting the label of a test sample. But it also means the global measure is not very personalized. To capture more personalized information, we propose in the next subsection the local weight measure.

B.3.3 LDV (Local Dynamic Voting)

A local weight usually means the performance or confidence of a classifier in a smaller neighborhood in validation data of a test sample. It usually demands two measures: firstly, a distance measure to find the neighborhood; secondly the competence measure to evaluate the performance of the classifier in the neighborhood. RFD (random forest dissimilarity) in this work is used as a distance measure to find the neighborhood of a given test sample, while OOB measure is used to replace the validation dataset.

The RFD measure $\mathbf{D_H}$ is inferred from a RF classifier $\mathbf{H}$, learned from training data $\mathbf{T}$. For each tree in the forest, if two samples end in the same terminal node, their dissimilarity is 0 otherwise 1. This process goes over all trees in the forest, and the average value is the RFD value (more details are given in [44]). It can be told that compared to other dissimilarity measures, RFD takes the advantage of class information to measure the distance [44].

To calculate the local weight $W_{local}^{(q)}$, RFD is used to find the neighborhood $\theta_{\mathbf{X}}$ of each test instance $\mathbf{X}$ by choosing the most $n_{neighbor}$ similar instances in training data. The OOB measure over $\theta_{\mathbf{X}}$ is then used to calculate the local weight. Unlike in the work of [227] using OOB to measure the individual tree accuracy, here OOB is used to measure the performance of the RF classifier. With $\theta_{\mathbf{X}}$, the local weight can be easily calculated with OOB measure:

$$W_{local}^{(q)} = OOB_{accuracy}(\mathbf{H}^{(q)}, \theta_{\mathbf{X}}) \tag{B.5}$$

The idea of local weight here is similar to OLA (Overall Local Accuracy) used in dynamic selection [71]. There are two main differences: firstly, LDV uses the random forest dissimilarity as a distance measure which carries both feature information and class label information while OLA uses Euclidean distance which may suffer from the concentration of pairwise distance [5] in high dimensional space; secondly, OLA requires a validation dataset while LDV does not.

B.3.4 GLDV (Global&Local Dynamic Voting)

From the previous two subsections, we can see that $W_{global}^{(q)}$ uses global information from all training data and measures the confidence of the classifier. But it has also the risk of being too generalized and lacks of personalized information. On the other hand, $W_{local}^{(q)}$ uses information on the neighborhood of the test sample to give a more personalized measure which can better represent the heterogeneity among cancer patients but may lose the global vision at the same time. Hence we propose a measure that takes both measures into account.

With each $\mathbf{H}^{(q)}$, the global weight $W_{global}^{(q)}$ and the local weight $W_{local}^{(q)}$ are calculated respectively and the combined weight $W_{GL}^{(q)}$ is calculated by taking advantage of both global and local information together:

$$W_{GL}^{(q)} = W_{global}^{(q)} \times W_{local}^{(q)} \tag{B.6}$$

The reason why we choose to multiply global weight and local weight for deriving a combined weight, is that, as it is explained previously, W_{global} lacks personalized information, but it can be counter-balanced by W_{local} to give more preference in some situations. For example, when $W^{(q)}_{global}$ agrees with $W^{(q)}_{local}$ on a particular view q, if both weights are small, then $W^{(q)}_{GL}$ becomes even smaller as we do not have confidence on this view; if both weights get bigger and bigger, then $W^{(q)}_{GL}$ gets closer and closer to both weights, especially local weight. On the contrary, when $W^{(q)}_{global}$ disagrees with $W^{(q)}_{local}$, it is hard to make a decision with a disagreement (as we need prior knowledge to decide to choose global or local weight); hence we penalize $W^{(q)}_{GL}$ as long as there is a disagreement ($W^{(q)}_{GL}$ is smaller than 0.5) but still with a preference to $W^{(q)}_{local}$.

B.4 Experiments

In this study, we use several publicly available Radiomics datasets. A general description of all datasets can be found in Table B.1 where IR stands for the imbalance ratio of the dataset. More details about these datasets can be found in the work of [258].

	#features	#samples	#views	#classes	IR
nonIDH1	6746	84	5	2	3
IDHcodel	6746	67	5	2	2.94
lowGrade	6746	75	5	2	1.4
progression	6746	75	5	2	1.68

TABLE B.1: Overview of each dataset.

The main objective of the experiment is to compare the state of the art Radiomics methods to static and dynamic voting methods. In total six methods are compared: one state of the art Radiomics method, i.e. SVMRFE; two static weighting methods, i.e. MVRF (combines RF results with majority voting as in [44]) and WRF (combines RF results with weights as in Section 3.1, the weights are the OOB accuracy of each $\mathbf{H}^{(q)}$); three dynamic weighted voting methods, i.e. GDV, LDV and GLDV as described in the previous section.

For the two dynamic voting methods that use local weights, LDV and GLDV, the neighborhood size $n_{neighbor}$ is set to 7 according to the work of [71]. For SVMRFE, the number of selected features is defined as in [44] according to the experiments of [30] and a Random forest classifier is then built on the selected features. For all random forest classifiers, the tree number is set to 500 while the other parameters are set to the default values given by the Scikit-Learn package for Python.

Similar to our previous work [44, 46], a stratified repeated random sampling approach was used to achieve a robust estimate of the performance. The stratified random splitting procedure is repeated 10 times, with 50% sample rate in each subset. In order to compare the methods, the mean and standard deviations of accuracy are evaluated over 10 runs.

B.5 Results

Dataset	SVMRFE +RF	MVRF	WRF	GDV	LDV	GLDV
nonIDH1	76.28% ± 4.39	82.79% ± 2.37	82.79% ± 2.37	82.79% ± 2.37	76.98% ± 1.93	77.44% ± 2.33
IDHcodel	73.23% ± 5.50	76.76% ± 2.06	76.76% ± 2.06	76.76% ± 2.06	74.11% ± 1.17	74.41% ± 1.34
lowGrade	62.55% ± 3.36	64.41% ± 3.76	64.41% ± 3.76	64.41% ± 3.76	64.41% ± 3.45	66.05% ± 3.32
progression	62.36% ± 3.73	61.31% ± 4.25	61.31% ± 4.25	61.57% ± 4.27	62.63% ± 4.37	62.89% ± 4.62
Average Rank	5.250	3.250	3.250	2.875	3.875	**2.500**

TABLE B.2: Experiment results with 50% training data 50% test data for Radiomics data

The results of mean accuracies, along with the corresponding standard deviation, over the 10 repetitions are shown in Table B.2. GDV and the two static voting methods have almost the same results over the four datasets, but these results are different from the two dynamic weighted voting methods LDV and GLDV. It is not surprising that there is no difference between MVRF and WRF because the datasets we use in this work have only five views, which means that there is no situation like even votes (the worst case would be 3 against

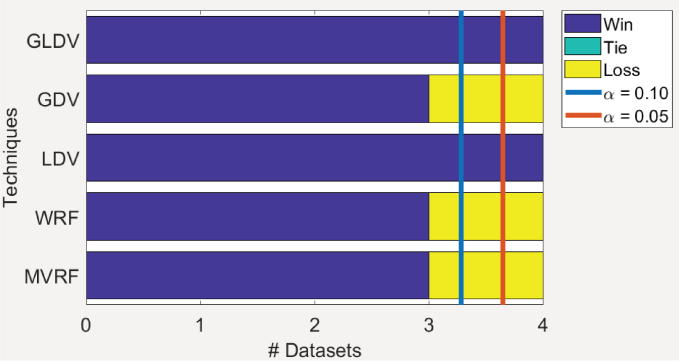


FIGURE B.1: Pairwise comparison between MCS solutions and SVMRFE. The vertical lines illustrate the critical values considering a confidence level α= {0.10 , 0.05}.

2). Hence as long as there is no extremely big difference among performance of different views, the two static voting methods should have similar results. And the result of GDV confirms our assumption in the previous section that the global weight alone does not contain a lot of personalized information. We can also see that there is a benefit of combining global and local weights as the performance of GLDV is always better than LDV. From the average ranking value, it can be told that the best method is the proposed GLDV method, followed by GDV. The state of the art solution SVMRFE is ranked at the last place.

To see more clearly the difference between MCS based methods and SVMRFE, a pairwise analysis based on the Sign test is computed on the number of wins, ties and losses as in the work of [71]. Figure B.1 shows that, when compared to SVMRFE, only the proposed methods LDV and GLDV are significantly better than SVMRFE with α = 0.10 and 0.05. These results show that the MCS based late integration methods can also be significantly better than the state-of-art Radiomics solutions.

When we compare GDV, LDV and GLDV, it can be seen that for nonIDH1 and IDHCodel data, the performance of GLDV is between LDV and GDV (LDV is the worst while GDV is the best). However for the two other datasets,

GLDV is always better than both LDV and GDV, which means that for different datasets, the best combination of LDV and GDV should be different. To further study the preference of global weight W_{global} and local weight W_{local} for different datasets, a new combination is formed as:

$$W_{GLnew}^{(q)} = (W_{global}^{(q)})^{1-a} \times (W_{local}^{(q)})^{a} \tag{B.7}$$

From Equation B.7 it can be told that when $a = 1$, the combination is only affected by local accuracy while when $a = 0$ the combination is only affected by global accuracy. The results of $W_{GLnew}^{(q)}$ are shown in Table B.3, from which we can confirm our conclusion that for IDHCodel1 and nonIDH data, they get better results when they use more global weight. For lowGrade and progression data, they get better results when they use more local weight.

TABLE B.3: The results of new combinations $W_{GLnew}^{(q)}$ with different a value.

Dataset	a=0 (GDV)	a=0.1	a=0.2	a=0.3	a=0.4	a=0.5	a=0.6	a=0.7	a=0.8	a=0.9	a=1 (LDV)
nonIDH	**82.79**% ± 2.37	**82.79**% ± 2.37	**82.79**% ± 2.37	82.32% ± 2.13	81.16% ± 3.02	80.23% ± 2.80	79.99% ± 3.15	79.30% ± 2.42	77.90% ± 2.38	77.44% ± 2.33	76.98% ± 1.93
IDHCodel1	**76.76**% ± 2.06	**76.76**% ± 2.06	**76.76**% ± 2.06	75.88% ± 1.76	75.58% ± 1.34	75.29% ± 1.44	75.29% ± 1.44	75.29% ± 1.95	75.00% ± 1.97	75.00% ± 1.97	74.11% ± 1.17
lowGrade	64.41% ± 3.75	64.41% ± 3.75	64.41% ± 3.75	**64.65**% ± 3.57	64.41% ± 3.45	64.41% ± 3.45	**64.65**% ± 3.72	64.18% ± 4.18	63.48% ± 3.75	63.48% ± 3.45	64.415% ± 3.45
progression	61.57% ± 4.27	61.57% ± 4.27	61.84% ± 3.57	62.10% ± 3.56	62.36% ± 3.91	62.10% ± 4.43	62.36% ± 4.41	**63.42**% ± 4.62	62.89% ± 4.77	62.63% ± 4.37	62.36% ± 4.56

In general, all MCS based late integration methods are better than feature selection methods. Majority voting is simple and efficient. GLDV is only better than majority voting on two datasets. But LDV and GLDV are preferable for Radiomics applications in the following three ways: (i) they give different weights of each view to each test sample, so that each test sample uses a different combination of classifiers to give a personalized decision; (ii) they are significantly better than the state of art work in Radiomics; (iii) the performance of GLDV can be further improved by adjusting the proportion of local weight and global weight. Note that other parameters like the neighborhood size can also be adjusted to optimize the performance. Compared to static voting, the disadvantage of dynamic voting is that it is more complex and less efficient.

B.6 Conclusions

In the state of art works of Radiomics, most studies used feature selection methods as a solution for the HDLSS problem. In this work, we have treated Radiomics as a multi-view learning problem and investigated the potential of MCS based late integration methods, proposed earlier in [44]. In particuler, we have investigated some dynamic voting based MCS methods, that can give each patient a personalized prediction by dynamically integrating the classification result from each view. We believe these methods have a great potential and can significantly outperform early integration methods that make use of feature selection in the concatenated feature space.

To confirm our hypothesis, a representative early integration method, five MCS methods including three dynamic voting methods and two static voting methods, have been compared on four Radiomics datasets. We conclude from our experiments that all MCS based late integration methods are generally better than the state of art Radiomics solution, but only LDV and GLDV are significantly better, which shows the potential of MCS based late integration methods of being a better solution than the state-of-art Radiomics solutions.

Acknowledgment

This work is part of the DAISI project, co-financed by the European Union with the European Regional Development Fund (ERDF) and by the Normandy Region.

Appendix C

Single view results for the experiment in Chapter 3

C.1 Datasets and protocol

A general description of all datasets can be found in Table C.1. There are in total 21 HDLSS datasets used in this work. All these datasets are publicly available. 15 of these datasets are multi-view data (with #views >1). More details of these datasets can be found in Chapter 2. Six other public single view HDLSS datasets are added. When comparing proposed methods with other distance metrics, the multi-view data are treated as single view by concatenating all the views.

All the methods compared in this section are the same as in Chapter 3 and the parameter settings are the same too. To test the performance of the proposed dissimilarity measure, the classification accuracy is evaluated with KNN following the suggestion in many dissimilarity learning studies. The choice of K is 3 here as in many works in metric learning [130, 148, 190, 243]. A stratified random splitting procedure is repeated 10 times, with 50% of the instances for training, 50% for testing. In order to compare the methods, the mean and standard deviations of accuracy are evaluated over the 10 runs.

TABLE C.1: Overview of the real-world datasets used in our experiments. IR (imbalanced ratio) is the number of instances of the majority class over the number of instances of the minority class.

	#features	**#samples**	**#views**	**#classes**	**IR**
AWA8[143]	10940	640	6	8	1
AWA15[143]	10940	1200	6	15	1
Bio[231]	1776	1000	1	2	1
CNAE[93]	856	540	1	8	1
Game[231]	1000	600	1	2	1
Metabolomic[37]	476	94	3	2	1
Mfeat[93]	649	600	6	10	1
LU[231]	10937	250	1	2	1.02
NUS-WIDE2[60]	639	546	5	2	1.12
arcene[93]	10000	200	1	2	1.27
BBC[249]	13628	2012	2	5	1.34
lowGrade[258]	6746	84	5	2	1.4
NUS-WIDE3[60]	639	442	5	3	1.43
progression[258]	6746	75	5	2	1.68
LSVT[226]	309	126	4	2	2
GAs[93]	1396	76	1	3	2.67
IDHcodel[258]	6746	67	5	2	2.94
nonIDH1[258]	6746	84	5	2	3
BBCSport[249]	6386	544	2	5	3.16
Cal20[151]	3766	2386	6	20	24.18
Cal7[151]	3766	1474	6	7	25.74

C.2 Results

The results of average accuracy over 10 repetitions as well as the standard deviation on 21 datasets are shown in Table C.2. From the average ranking, it can be seen that RFD_{IH} is the best, followed by RFD_{NC} and then RFD_{PB}. Four RFD based methods are better than LMNN and Euclidean distance. There is no surprise that Euclidean distance is ranked at the last place.

RFD_{IH} wins the first place on 13 datasets, while RFD_{NC} wins the first place on 5 datasets. However, Euclidean distance and LMNN win the first place once each. It can be seen that RFD based methods are the best for the majority of data sets.

TABLE C.2: The result over 10 repetitions comparing different distance measures. First 2 methods are Euclidean distance and metric learning method PCA+LMNN. RFD is the classic RFD measure, RFD_{NC} is RFD weighted by posterior probability in the leaf node. RFD_{IH} is weighted by instance hardness. The best result of each data set is presented in bold.

	EU	*LMNN*	*RFD*	RFD_{PB}	RFD_{NC}	RFD_{IH}
awa8	36.28% ± 2.33	42.34% ± 3.72	53.56% ± 1.90	**54.25% ± 1.70**	53.16% ± 2.32	53.28% ± 1.75
awa15	17.00% ± 1.67	25.75% ± 1.26	**35.35% ± 1.06**	35.25% ± 1.29	34.48% ± 1.12	34.98% ± 1.14
Bio	66.12% ± 2.51	68.82% ± 2.46	74.74% ± 1.84	74.86% ± 1.97	74.80% ± 1.63	**75.32% ± 1.93**
CNAE	78.56% ± 2.59	84.59% ± 2.28	83.74% ± 0.99	83.52% ± 1.02	**86.44% ± 1.29**	85.11% ± 1.66
Game	**50.67% ± 2.63**	48.73% ± 2.41	48.67% ± 2.35	48.53% ± 1.60	48.53% ± 1.97	49.20% ± 2.25
Metabo	58.75% ± 5.73	56.67% ± 6.37	63.12% ± 5.59	63.12% ± 6.66	**63.33% ± 6.54**	**63.33% ± 5.91**
mfeat	96.37% ± 1.19	97.13% ± 0.86	97.70% ± 1.00	97.73% ± 1.01	97.60% ± 1.03	**97.87% ± 0.93**
LU	91.04% ± 1.55	92.40% ± 0.96	93.60% ± 1.52	93.68% ± 1.49	93.68% ± 1.49	**93.84% ± 1.43**
nus2	89.05% ± 1.78	92.34% ± 1.08	92.75% ± 1.13	92.67% ± 1.13	92.78% ± 1.11	**92.86% ± 1.22**
arcene	80.20% ± 3.43	**80.70% ± 3.07**	76.80% ± 4.89	77.10% ± 5.15	76.40% ± 4.52	77.50% ± 5.16
bbc	62.94% ± 9.37	89.82% ± 1.53	93.71% ± 0.52	93.65% ± 0.50	**94.00% ± 0.64**	93.98% ± 0.66
lowGrade	57.44% ± 6.16	60.47% ± 6.49	63.72% ± 4.19	**64.19% ± 3.63**	63.49% ± 3.46	**64.19% ± 3.15**
nus3	69.23% ± 2.46	79.05% ± 2.03	80.36% ± 1.98	80.36% ± 1.97	80.14% ± 2.00	**80.54% ± 2.01**
progression	66.58% ± 5.27	66.84% ± 6.78	66.84% ± 3.76	66.58% ± 5.00	**67.11% ± 3.95**	65.53% ± 4.77
LSVT	75.71% ± 4.49	80.16% ± 4.27	82.38% ± 3.51	**82.86% ± 3.16**	82.54% ± 3.76	**82.86% ± 3.24**
GAs	55.90% ± 3.77	56.67% ± 6.12	60.77% ± 3.81	61.03% ± 4.10	61.28% ± 4.05	**61.79% ± 4.51**
IDHCodel	71.47% ± 3.73	70.29% ± 5.00	74.41% ± 2.30	74.12% ± 2.20	74.41% ± 1.88	**75.00% ± 2.71**
nonIDH1	77.44% ± 4.54	76.98% ± 5.55	83.72% ± 4.03	83.49% ± 3.95	**83.95% ± 3.52**	83.49% ± 3.95
bbcsport	65.49% ± 3.43	85.46% ± 7.70	90.66% ± 2.22	90.44% ± 2.25	92.60% ± 1.59	**96.92% ± 0.84**
Cal20	74.56% ± 0.65	75.51% ± 0.77	86.49% ± 0.35	86.61% ± 0.31	85.03% ± 0.44	**88.32% ± 0.36**
Cal7	88.69% ± 0.69	89.28% ± 0.83	94.55% ± 0.43	94.65% ± 0.46	93.93% ± 0.60	**95.24% ± 0.44**
Avg rank	5.33	4.64	3.19	3.05	3.00	1.79

Compare proposed methods to RFD: The proposed two methods RFD_{NC} and RFD_{IH} are both based on RFD, to see more clearly if they are significantly better than classic RFD, pairwise comparison based on win tie loss is tested.

The comparison result is shown in Figure C.1, It can be seen that, among all the other methods, only proposed RFD_{IH} is significantly better than RFD.

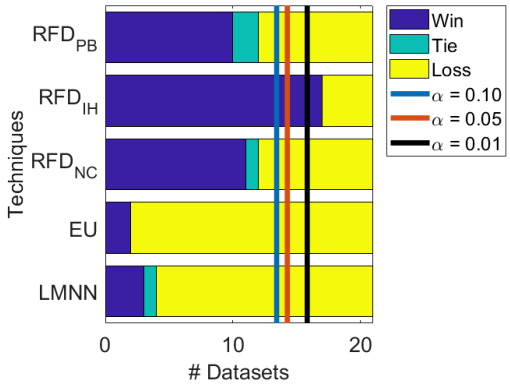


FIGURE C.1: Pairwise comparison between classic RFD and other methods on overall accuracy. The vertical lines illustrate the critical values considering a confidence level α= {0.10, 0.05, 0.01}.

From the experiment results, it can be seen that all RF based methods are better than learning free method Euclidean distance and classic metric learning method LMNN on HDLSS data. Comparing to classic metric learning methods, RF based measures have the advantage of dealing with high feature dimension efficiently in the subspace without using PCA. Among four RF based measures, proposed RFD_{IH} is the best, which is significantly better than classic RFD. This single view result corresponds to the multi-view result in Chapter 3.

Bibliography

1. Abbasnejad, M. E., Ramachandram, D. & Mandava, R. A survey of the state of the art in learning the kernels. *Knowledge and information systems* **31,** 193–221 (2012).
2. Adankon, M. M. & Cheriet, M. Optimizing resources in model selection for support vector machine. *Pattern Recognition* **40,** 953–963 (2007).
3. Aerts, H., Velazquez, E. R., Leijenaar, R., Parmar, C., Grossmann, P., Cavalho, S., Bussink, J., Monshouwer, R., Haibe-Kains, B., Rietveld, D., *et al.* Decoding tumour phenotype by noninvasive imaging using a quantitative radiomics approach. *Nature Communications* **5,** 1–8 (2014).
4. Afanador, N. L., Smolinska, A., Tran, T. N. & Blanchet, L. Unsupervised random forest: a tutorial with case studies. *Journal of Chemometrics* **30,** 232–241 (2016).
5. Aggarwal, C. C., Hinneburg, A. & Keim, D. A. *On the surprising behavior of distance metrics in high dimensional space* in *International Conference on Database Theory (ICDT)* (2001), 420–434.
6. Aiolli, F. & Donini, M. EasyMKL: a scalable multiple kernel learning algorithm. *Neurocomputing* **169,** 215–224 (2015).
7. Aldous, D. J. in *École d'Été de Probabilités de Saint-Flour XIII—1983* 1–198 (Springer, 1985).
8. Alexandre, L. A., Campilho, A. C. & Kamel, M. *Combining independent and unbiased classifiers using weighted average* in *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000* **2** (2000), 495–498.
9. Alexandridis, A., Patrinos, P., Sarimveis, H. & Tsekouras, G. A two-stage evolutionary algorithm for variable selection in the development of RBF neural network models. *Chemometrics and Intelligent Laboratory Systems* **75,** 149–162 (2005).

10. Amari, S.-i. & Wu, S. Improving support vector machine classifiers by modifying kernel functions. *Neural Networks* **12,** 783–789 (1999).
11. Amerise, I. L., Tarsitano, A., *et al. Weighting Distance Matrices Using Rank Correlations* tech. rep. (2012).
12. Andrew, G., Arora, R., Bilmes, J. & Livescu, K. *Deep canonical correlation analysis* in *International Conference on Machine Learning* (2013), 1247–1255.
13. Angiulli, F. On the Behavior of Intrinsically High-Dimensional Spaces: Distances, Direct and Reverse Nearest Neighbors, and Hubness. *Journal of Machine Learning Research* **18,** 170–1 (2017).
14. Araújo, T., Aresta, G., Castro, E., Rouco, J., Aguiar, P., Eloy, C., Polónia, A. & Campilho, A. Classification of breast cancer histology images using Convolutional Neural Networks. *PloS one* **12,** e0177544 (2017).
15. Argyriou, A., Hauser, R., Micchelli, C. A. & Pontil, M. *A DC-programming algorithm for kernel selection* in *Proceedings of the 23rd International Conference on Machine Learning* (2006), 41–48.
16. Auret, L. & Aldrich, C. Change point detection in time series data with random forests. *Control Engineering Practice* **18,** 990–1002 (2010).
17. Bach, F. R., Lanckriet, G. R. & Jordan, M. I. *Multiple kernel learning, conic duality, and the SMO algorithm* in *Proceedings of the twenty-first International Conference on Machine Learning* (2004), 6.
18. Balagurunathan, Y., Gu, Y., Wang, H., Kumar, V., Grove, O., Hawkins, S., Kim, J., Goldgof, D. B., Hall, L. O., Gatenby, R. A., *et al.* Reproducibility and prognosis of quantitative features extracted from CT images. *Translational Oncology* **7,** 72–87 (2014).
19. Bar-Hillel, A., Hertz, T., Shental, N. & Weinshall, D. Learning a mahalanobis metric from equivalence constraints. *Journal of Machine Learning Research* **6,** 937–965 (2005).
20. Barandiaran, I. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell* **20,** 1–22 (1998).
21. Bashbaghi, S., Granger, E., Sabourin, R. & Bilodeau, G.-A. Robust watchlist screening using dynamic ensembles of svms based on multiple face representations. *Machine vision and applications* **28,** 219–241 (2017).
22. Bay, H., Ess, A., Tuytelaars, T. & Van Gool, L. Speeded-up robust features (SURF). *Computer Vision and Image Understanding* **110,** 346–359 (2008).

23. Bellet, A. Supervised metric learning with generalization guarantees. *arXiv preprint arXiv:1307.4514* (2013).
24. Bellet, A., Habrard, A. & Sebban, M. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709* (2013).
25. Bernard, S., Adam, S. & Heutte, L. *Using random forests for handwritten digit recognition* in *Ninth International Conference on Document Analysis and Recognition (ICDAR)* **2** (2007), 1043–1047.
26. Biau, G. & Scornet, E. A random forest guided tour. *Test* **25,** 197–227 (2016).
27. Biggio, B., Fumera, G. & Roli, F. *Bayesian analysis of linear combiners* in *International Workshop on Multiple Classifier Systems* (2007), 292–301.
28. Bill, J. & Fokoué, E. A Comparative Analysis of Predictive Learning Algorithms on High-Dimensional Microarray Cancer Data. *Serdica Journal of Computing* **8,** 137–168 (2014).
29. Blum, A. & Mitchell, T. *Combining labeled and unlabeled data with co-training* in *11th Conference on Computational Learning Theory (CCLT)* (1998), 92–100.
30. Bolón-Canedo, V., Sánchez-Maroño, N. & Alonso-Betanzos, A. A review of feature selection methods on synthetic data. *Knowledge and Information Systems* **34,** 483–519 (2013).
31. Bosch, A., Zisserman, A. & Munoz, X. *Representing shape with a spatial pyramid kernel* in *Proceedings of the 6th ACM International Conference on Image and video retrieval* (2007), 401–408.
32. Branden, K. V. & Hubert, M. Robust classification in high dimensions based on the SIMCA method. *Chemometrics and Intelligent Laboratory Systems* **79,** 10–21 (2005).
33. Breiman, L. Out-of-bag estimation. *Tech. rep., Statistics Department, University of California* (1996).
34. Breiman, L. Bagging predictors. *Machine Learning* **24,** 123–140 (1996).
35. Breiman, L. Random forests. *Machine Learning* **45,** 5–32 (2001).
36. Britto Jr, A. S., Sabourin, R. & Oliveira, L. E. Dynamic selection of classifiers—a comprehensive review. *Pattern Recognition* **47,** 3665–3680 (2014).
37. Bro, R., Nielsen, H. J., Savorani, F., Kjeldahl, K., Christensen, I. J., Brünner, N. & Lawaetz, A. J. Data fusion in metabolomic cancer diagnostics. *Metabolomics* **9,** 3–8 (2013).

38. Brun, A. L., Britto, A. S., Oliveira, L. S., Enembreck, F. & Sabourin, R. *Contribution of data complexity features on dynamic classifier selection* in *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), 4396–4403.
39. Brunelli, R. & Falavigna, D. Person identification using multiple cues. *IEEE transactions on pattern analysis and machine intelligence* **17,** 955–966 (1995).
40. Bucak, S. S., Jin, R. & Jain, A. K. Multiple kernel learning for visual object recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **36,** 1354–1369 (2013).
41. Bylander, T. Estimating generalization error on two-class datasets using out-of-bag estimates. *Machine Learning* **48,** 287–297 (2002).
42. Bylander, T. & Hanzlik, D. *Estimating generalization error using out-of-bag estimates* in *AAAI/IAAI* (1999), 321–327.
43. Cameron, A., Khalvati, F., Haider, M. A. & Wong, A. MAPS: a quantitative radiomics approach for prostate cancer detection. *IEEE Transactions on Biomedical Engineering* **63,** 1145–1156 (2016).
44. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. Dissimilarity-based representation for radiomics applications. *First International Conference on Pattern Recognition and Artificial Intelligence (ICPRAI)* (2018).
45. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. *Dynamic voting in multi-view learning for radiomics applications* in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)* (2018), 32–41.
46. Cao, H., Bernard, S., Heutte, L. & Sabourin, R. *Improve the performance of transfer learning without fine-tuning using dissimilarity-based multi-view learning for breast cancer histology images* in *International Conference Image Analysis and Recognition* (2018), 779–787.
47. Cao, H., Bernard, S., Sabourin, R. & Heutte, L. Random forest dissimilarity based multi-view learning for Radiomics application. *Pattern Recognition* **88,** 185–197 (2019).
48. Carneiro, G., Oakden-Rayner, L., Bradley, A. P., Nascimento, J. & Palmer, L. *Automated 5-year mortality prediction using deep learning and radiomics features from chest computed tomography* in *14th International Symposium on Biomedical Imaging (ISBI)* (2017), 130–134.

49. Cavalin, P. R., Sabourin, R. & Suen, C. Y. Dynamic selection approaches for multiple classifier systems. *Neural Computing and Applications* **22,** 673–688 (2013).
50. Chakraborty, S., Tomsett, R., Raghavendra, R., Harborne, D., Alzantot, M., Cerutti, F., Srivastava, M., Preece, A., Julier, S., Rao, R. M., *et al. Interpretability of deep learning models: a survey of results* in *2017 IEEE Smart-World, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation* (2017), 1–6.
51. Chan, J. K. The Wonderful Colors of the Hematoxylin–Eosin Stain in Diagnostic Surgical Pathology. *International Journal of Surgical Pathology* **22,** 12–32 (2014).
52. Chandrashekar, G. & Sahin, F. A survey on feature selection methods. *Computers & Electrical Engineering* **40,** 16–28 (2014).
53. Chatzis, V., Bors, A. G. & Pitas, I. Multimodal decision-level fusion for person authentication. *IEEE transactions on systems, man, and cybernetics-part a: systems and humans* **29,** 674–680 (1999).
54. Chawla, N. V., Bowyer, K. W., Hall, L. O. & Kegelmeyer, W. P. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* **16,** 321–357 (2002).
55. Chen, B., Liu, H. & Bao, Z. A kernel optimization method based on the localized kernel Fisher criterion. *Pattern Recognition* **41,** 1098–1109 (2008).
56. Chen, N., Zhu, J. & Xing, E. P. *Predictive subspace learning for multi-view data: a large margin approach* in *Advances in Neural Information Processing Systems (NIPS)* (2010), 361–369.
57. Chen, X. & Ishwaran, H. Random forests for genomic data analysis. *Genomics* **99,** 323–329 (2012).
58. Chipman, H. A., George, E. I., McCulloch, R. E., *et al.* BART: Bayesian additive regression trees. *The Annals of Applied Statistics* **4,** 266–298 (2010).
59. Chu, L. C., Park, S., Kawamoto, S., Fouladi, D. F., Shayesteh, S., Zinreich, E. S., Graves, J. S., Horton, K. M., Hruban, R. H., Yuille, A. L., *et al.* Utility of CT Radiomics Features in Differentiation of Pancreatic Ductal Adenocarcinoma From Normal Pancreatic Tissue. *American Journal of Roentgenology*, 1–9 (2019).

60. Chua, T.-S., Tang, J., Hong, R., Li, H., Luo, Z. & Zheng, Y.-T. *NUS-WIDE: A Real-World Web Image Database from National University of Singapore* in *ACM Conference on Image and Video Retrieval (CIVR)* (Santorini, Greece., 2009).
61. Chung, A. G., Shafiee, M. J., Kumar, D., Khalvati, F., Haider, M. A. & Wong, A. Discovery Radiomics for Multi-Parametric MRI Prostate Cancer Detection. *arXiv preprint arXiv:1509.00111* (2015).
62. Coelho, L. P. Mahotas: Open source software for scriptable Computer Vision. *Journal of Open Research Software* **1.** doi:10.5334/jors.ac. http://dx.doi.org/10.5334/jors.ac (2013).
63. Cordón, O., Damas, S. & Santamaría, J. Feature-based image registration by means of the CHC evolutionary algorithm. *Image and Vision Computing* **24,** 525–533 (2006).
64. Coroller, T. P., Grossmann, P., Hou, Y., Velazquez, E. R., Leijenaar, R. T., Hermann, G., Lambin, P., Haibe-Kains, B., Mak, R. H. & Aerts, H. J. CT-based radiomic signature predicts distant metastasis in lung adenocarcinoma. *Radiotherapy and Oncology* **114,** 345–350 (2015).
65. Coroller, T. P., Agrawal, V., Narayan, V., Hou, Y., Grossmann, P., Lee, S. W., Mak, R. H. & Aerts, H. J. Radiomic phenotype features predict pathological response in non-small cell lung cancer. *Radiotherapy and Oncology* **119,** 480–486 (2016).
66. Cortes, C., Mohri, M. & Rostamizadeh, A. Algorithms for learning kernels based on centered alignment. *Journal of Machine Learning Research* **13,** 795–828 (2012).
67. Costa, Y. M. G., Bertolini, D., Britto, A. S., Cavalcanti, G. D. C. & Oliveira, L. E. S. The dissimilarity approach: a review. *Artificial Intelligence Review* (Aug. 2019).
68. Cristianini, N., Shawe-Taylor, J., Elisseeff, A. & Kandola, J. S. *On kernel-target alignment* in *Advances in Neural Information Processing Systems* (2002), 367–373.
69. Cruz, R. M., Cavalcanti, G. D. & Ren, T. I. *A method for dynamic ensemble selection based on a filter and an adaptive distance to improve the quality of the regions of competence* in *The 2011 International Joint Conference on Neural Networks* (2011), 1126–1133.

70. Cruz, R. M., Cavalcanti, G. D. & Ren, T. I. *An ensemble classifier for offline cursive character recognition using multiple feature extraction techniques* in *The 2010 International Joint Conference on Neural Networks (IJCNN)* (2010), 1–8.
71. Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. Dynamic classifier selection: Recent advances and perspectives. *Information Fusion* **41,** 195–216 (2018).
72. Cruz, R. M., Zakane, H. H., Sabourin, R. & Cavalcanti, G. D. *Dynamic Ensemble Selection VS K-NN: why and when Dynamic Selection obtains higher classification performance?* in *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)* (2017), 1–6.
73. Cruz, R. M., Oliveira, D. V., Cavalcanti, G. D. & Sabourin, R. FIRE-DES++: Enhanced online pruning of base classifiers for dynamic ensemble selection. *Pattern Recognition* **85,** 149–160 (2019).
74. Cruz, R. M., Cavalcanti, G. D. & Ren, T. I. *Handwritten digit recognition using multiple feature extraction techniques and classifier ensemble* in *17th International Conference on Systems, Signals and Image Processing* (2010), 215–218.
75. Cruz, R. M., Sabourin, R. & Cavalcanti, G. D. *META-DES. H: a dynamic ensemble selection technique using meta-learning and a dynamic weighting approach* in *2015 International Joint Conference on Neural Networks (IJCNN)* (2015), 1–8.
76. Davies, A. & Ghahramani, Z. The random forest kernel and other kernels for big data from random partitions. *arXiv preprint arXiv:1402.4293* (2014).
77. Davis, J. V., Kulis, B., Jain, P., Sra, S. & Dhillon, I. S. *Information-theoretic metric learning* in *Proceedings of the 24th International Conference on Machine Learning* (2007), 209–216.
78. De Souto, M. C., Soares, R. G., Santana, A. & Canuto, A. M. *Empirical comparison of dynamic classifier selection methods based on diversity and accuracy for building ensembles* in *2008 IEEE International Joint Conference on Neural Networks (IEEE world congress on computational intelligence)* (2008), 1480–1487.
79. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research* **7,** 1–30 (2006).

80. Domeniconi, C., Gunopulos, D. & Peng, J. Large margin nearest neighbor classifiers. *IEEE transactions on Neural Networks* **16,** 899–909 (2005).
81. Dudoit, S. & Fridlyand, J. Bagging to improve the accuracy of a clustering procedure. *Bioinformatics* **19,** 1090–1099 (2003).
82. Duin, R. P. *The combining classifier: to train or not to train?* in *Object recognition supported by user interaction for service robots* **2** (2002), 765–770.
83. Duin, R. P. & Pekalska, E. The dissimilarity space: Bridging structural and statistical Pattern Recognition. *Pattern Recognition Letters* **33,** 826–832 (2012).
84. El Naqa, I., Brock, K., Yu, Y., Langen, K. & Klein, E. E. On the fuzziness of Machine Learning, neural networks, and artificial intelligence in radiation oncology. *International Journal of Radiation Oncology Biology Physics* **100,** 1–4 (2018).
85. Englund, C. & Verikas, A. A novel approach to estimate proximity in a random forest: An exploratory study. *Expert Systems with Applications* **39,** 13046–13050 (2012).
86. Ester, M., Kriegel, H.-P., Sander, J., Xu, X., *et al. A density-based algorithm for discovering clusters in large spatial databases with noise.* in *KDD* **96** (1996), 226–231.
87. Farhidzadeh, H., Kim, J. Y., Scott, J. G., Goldgof, D. B., Hall, L. O. & Harrison, L. B. *Classification of progression free survival with nasopharyngeal carcinoma tumors* in *SPIE Medical Imaging* (2016), 97851I–97851I.
88. Feldbauer, R. & Flexer, A. A comprehensive empirical comparison of hubness reduction in high-dimensional spaces. *Knowledge and Information Systems* **59,** 137–166 (2019).
89. Feng, J., Wang, L., Sugiyama, M., Yang, C., Zhou, Z.-H. & Zhang, C. Boosting and margin theory. *Frontiers of Electrical and Electronic Engineering* **7,** 127–133 (2012).
90. Fernández-Delgado, M., Cernadas, E., Barro, S. & Amorim, D. Do we need hundreds of classifiers to solve real world classification problems. *Journal of Machine Learning Research* **15,** 3133–3181 (2014).
91. Fierrez, J., Morales, A., Vera-Rodriguez, R. & Camacho, D. Multiple classifiers in biometrics. Part 1: Fundamentals and review. *Information Fusion* **44,** 57–64 (2018).

92. Florez, E., Fatemi1, A., Claudio, P. P. & Howard, C. M. Emergence of Radiomics: Novel Methodology Identifying Imaging Biomarkers of Disease in Diagnosis, Response, and Progression. *SM Journal of Clinical and Medical Imaging* **4,** 1019 (2018).
93. Frank, A., Asuncion, A., *et al. UCI Machine Learning repository* 2010.
94. Freund, Y. & Schapire, R. E. A decision-theoretic generalization of online learning and an application to boosting. *Journal of computer and system sciences* **55,** 119–139 (1997).
95. Giacinto, G. & Roli, F. Design of effective neural network ensembles for image classification purposes. *Image and Vision Computing* **19,** 699–707 (2001).
96. Giacinto, G. & Roli, F. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition* **34,** 1879–1881 (2001).
97. Goldberg, D. E. Genetic algorithms in search. *Optimization, and Machine-Learning* (1989).
98. Gönen, M. & Alpaydın, E. Multiple kernel learning algorithms. *Journal of Machine Learning Research* **12,** 2211–2268 (2011).
99. Gray, K. R., Aljabar, P., Heckemann, R. A., Hammers, A., Rueckert, D., Initiative, A. D. N., *et al.* Random forest-based similarity measures for multi-modal classification of Alzheimer's disease. *NeuroImage* **65,** 167–175 (2013).
100. Gu, Y., Chanussot, J., Jia, X. & Benediktsson, J. A. Multiple kernel learning for hyperspectral image classification: A review. *IEEE Transactions on Geoscience and Remote Sensing* **55,** 6547–6565 (2017).
101. Gu, Y., Zydek, D. & Jin, Z. in *Progress in Systems Engineering* 273–278 (Springer, 2015).
102. Guillaumin, M., Verbeek, J. & Schmid, C. *Is that you? Metric learning approaches for face identification* in *ICCV 2009-International Conference on Computer Vision* (2009), 498–505.
103. Gunduz, N. & Fokoué, E. Robust classification of high dimension low sample size data. *arXiv preprint arXiv:1501.00592* (2015).
104. Gutman, D. A., Cooper, L. A., Hwang, S. N., Holder, C. A., Gao, J., Aurora, T. D., Dunn Jr, W. D., Scarpace, L., Mikkelsen, T., Jain, R., *et al.* MR imaging predictors of molecular profile and survival: multi-institutional study of the TCGA glioblastoma data set. *Radiology* **267,** 560–569 (2013).

105. Gutschoven, B. & Verlinde, P. *Multi-modal identity verification using support vector machines (SVM)* in *Proceedings of the Third International Conference on Information Fusion* **2** (2000), THB3–3.
106. Guyon, I. & Elisseeff, A. An introduction to variable and feature selection. *Journal of Machine Learning Research* **3,** 1157–1182 (2003).
107. Guyon, I., Weston, J., Barnhill, S. & Vapnik, V. Gene selection for cancer classification using support vector machines. *Machine Learning* **46,** 389–422 (2002).
108. Haasdonk, B. & Bahlmann, C. *Learning with distance substitution kernels* in *Pattern Recognition - Proceedings of the 26th DAGM Symposium* (2004), 220–227.
109. Hall, P., Marron, J. S. & Neeman, A. Geometric representation of high dimension, low sample size data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **67,** 427–444 (2005).
110. Hamilton, N. A., Pantelic, R. S., Hanson, K. & Teasdale, R. D. Fast automated cell phenotype image classification. *BMC bioinformatics* **8,** 110 (2007).
111. Hansen, L. K. & Salamon, P. Neural network ensembles. *IEEE Transactions on Pattern Analysis & Machine Intelligence,* 993–1001 (1990).
112. Hardoon, D. R. & Shawe-Taylor, J. Convergence analysis of kernel canonical correlation analysis: theory and practice. *Machine Learning* **74,** 23–38 (2009).
113. Hardoon, D. R., Szedmak, S. & Shawe-Taylor, J. Canonical correlation analysis: An overview with application to learning methods. *Neural computation* **16,** 2639–2664 (2004).
114. Hawkins, S. H., Korecki, J. N., Balagurunathan, Y., Gu, Y., Kumar, V., Basu, S., Hall, L. O., Goldgof, D. B., Gatenby, R. A. & Gillies, R. J. Predicting outcomes of nonsmall cell lung cancer using CT image features. *IEEE Access* **2,** 1418–1426 (2014).
115. He, K., Zhang, X., Ren, S. & Sun, J. *Deep residual learning for image recognition* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), 770–778.
116. Herbster, M., Pontil, M. & Wainer, L. *Online learning over graphs* in *Proceedings of the 22nd International Conference on Machine Learning* (2005), 305–312.

117. Hoi, S. C. & Jin, R. *Active kernel learning* in *Proceedings of the 25th International Conference on Machine Learning* (2008), 400–407.
118. Hoi, S. C., Jin, R. & Lyu, M. R. *Learning nonparametric kernel matrices from pairwise constraints* in *Proceedings of the 24th International Conference on Machine Learning* (2007), 361–368.
119. Huang, J., Zhu, Q. & Xing, Y. *Multi-target detection and positioning in crowds using multiple camera surveillance* in *Ninth International Conference on Graphic and Image Processing (ICGIP 2017)* **10615** (2018), 106150C.
120. Huynh, B. Q., Li, H. & Giger, M. L. Digital mammographic tumor classification using transfer learning from deep convolutional neural networks. *Journal of Medical Imaging* **3,** 034501 (2016).
121. Ibba, A., Duin, R. P. & Lee, W.-J. *A study on combining sets of differently measured dissimilarities* in *2010 20th International Conference on Pattern Recognition* (2010), 3360–3363.
122. Itakura, H., Ikeda, D. M., Okamoto, S., Chen, S.-T., Rister, B., Gude, D., Mattonen, S. A., Alkim, E., Todderud, J., Schueler, E., *et al. Radiomics features to identify distinct subtypes of triple-negative breast cancers.* 2019.
123. Jaakkola, T. & Haussler, D. *Exploiting generative models in discriminative classifiers* in *Advances in Neural Information Processing Systems* (1999), 487–493.
124. Jain, P., Kulis, B., Dhillon, I. S. & Grauman, K. *Online metric learning and fast similarity search* in *Advances in Neural Information Processing Systems* (2009), 761–768.
125. Jia, T.-Y., Xiong, J.-F., Li, X.-Y., Yu, W., Xu, Z.-Y., Cai, X.-W., Ma, J.-C., Ren, Y.-C., Larsson, R., Zhang, J., *et al.* Identifying EGFR mutations in lung adenocarcinoma by noninvasive imaging using radiomics features and random forest modeling. *European radiology,* 1–9 (2019).
126. Jiang, H., Kim, B., Guan, M. & Gupta, M. *To trust or not to trust a classifier* in *Advances in Neural Information Processing Systems* (2018), 5541–5552.
127. Kabir, A., Ruiz, C. & Alvarez, S. A. *Mixed Bagging: A Novel Ensemble Learning Framework for Supervised Classification Based on Instance Hardness* in *2018 IEEE International Conference on Data Mining (ICDM)* (2018), 1073–1078.

128. Kaden, M, Nebel, D & Villmann, T. *Adaptive dissimilarity weighting for prototype-based classification optimizing mixtures of dissimilarities* in *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)* (2016), 135–140.
129. Kang, Z., Peng, C. & Cheng, Q. Kernel-driven similarity learning. *Neurocomputing* **267,** 210–219 (2017).
130. Kedem, D., Tyree, S., Sha, F., Lanckriet, G. R. & Weinberger, K. Q. *Nonlinear metric learning* in *Advances in Neural Information Processing Systems* (2012), 2573–2581.
131. Kettenring, J. R. Canonical analysis of several sets of variables. *Biometrika* **58,** 433–451 (1971).
132. Kittler, J. & Alkoot, F. M. Sum versus vote fusion in multiple classifier systems. *IEEE transactions on pattern analysis and machine intelligence* **25,** 110–115 (2003).
133. Kloft, M. & Blanchard, G. *The local rademacher complexity of lp-norm multiple kernel learning* in *Advances in Neural Information Processing Systems (NIPS)* (2011), 2438–2446.
134. Ko, A. H., Sabourin, R. & Britto Jr, A. S. From dynamic classifier selection to dynamic ensemble selection. *Pattern Recognition* **41,** 1718–1731 (2008).
135. Kulis, B. *et al.* Metric learning: A survey. *Foundations and Trends in Machine Learning* **5,** 287–364 (2013).
136. Kumar, V., Gu, Y., Basu, S., Berglund, A., Eschrich, S. A., Schabath, M. B., Forster, K., Aerts, H. J., Dekker, A., Fenstermacher, D., *et al.* Radiomics: the process and the challenges. *Magnetic Resonance Imaging* **30,** 1234–1248 (2012).
137. Kuncheva, L. Combining pattern classifiers methods and algorithms. john wiley&sons. *Inc. Publication, Hoboken* (2004).
138. Kuncheva, L. I. *Clustering-and-selection model for classifier combination* in *KES'2000. Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No. 00TH8516)* **1** (2000), 185–188.
139. Kurzynski, M., Woloszynski, T. & Lysiak, R. *On two measures of classifier competence for dynamic ensemble selection-experimental comparative analysis*

in *2010 10th International Symposium on Communications and Information Technologies* (2010), 1108–1113.

140. Lai, P. L. & Fyfe, C. Kernel and nonlinear canonical correlation analysis. *International Journal of Neural Systems* **10,** 365–377 (2000).
141. Lakshminarayanan, B., Roy, D. M. & Teh, Y. W. *Mondrian forests: Efficient online random forests* in *Advances in Neural Information Processing Systems* (2014), 3140–3148.
142. Lambin, P., Rios-Velazquez, E., Leijenaar, R., Carvalho, S., van Stiphout, R. G., Granton, P., Zegers, C. M., Gillies, R., Boellard, R., Dekker, A., *et al.* Radiomics: extracting more information from medical images using advanced feature analysis. *European Journal of Cancer* **48,** 441–446 (2012).
143. Lampert, C. H., Nickisch, H. & Harmeling, S. *Learning to detect unseen object classes by between-class attribute transfer* in *IEEE Conference on Computer Vision and Pattern Recognition* (2009), 951–958.
144. Lanckriet, G. R., Cristianini, N., Bartlett, P., Ghaoui, L. E. & Jordan, M. I. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research* **5,** 27–72 (2004).
145. Lao, J., Chen, Y., Li, Z.-C., Li, Q., Zhang, J., Liu, J. & Zhai, G. A deep learning-based radiomics model for prediction of survival in glioblastoma multiforme. *Scientific reports* **7,** 10353 (2017).
146. Lee, G., Lee, H. Y., Park, H., Schiebler, M. L., van Beek, E. J., Ohno, Y., Seo, J. B. & Leung, A. Radiomics and its emerging role in lung cancer research, imaging biomarkers and clinical management: State of the art. *European Journal of Radiology* **86,** 297–307 (2017).
147. Letchford, A. N. & Sørensen, M. M. Binary positive semidefinite matrices and associated integer polytopes. *Mathematical Programming* **131,** 253–271 (2012).
148. Li, D. & Tian, Y. Survey and experimental study on metric learning methods. *Neural Networks* (2018).
149. Li, T., Zhu, S., Li, Q. & Ogihara, M. *Gene functional classification by semi-supervised learning from heterogeneous data* in *ACM Symposium on Applied Computing* (2003), 78–82.
150. Li, Y., Duin, R. P. & Loog, M. *Combining multi-scale dissimilarities for image classification* in *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)* (2012), 1639–1642.

151. Li, Y., Nie, F., Huang, H. & Huang, J. *Large-Scale Multi-View Spectral Clustering via Bipartite Graph* in *29th AAAI Conference on Artificial Intelligence* (2015), 2750–2756.
152. Lim, D., Lanckriet, G. & McFee, B. *Robust structural metric learning* in *International Conference on Machine Learning* (2013), 615–623.
153. Lin, Z. & Davis, L. S. *Learning pairwise dissimilarity profiles for appearance recognition in visual surveillance* in *International Symposium on Visual Computing (ISVC)* (2008), 23–34.
154. Liu, R., Hall, L. O., Goldgof, D. B., Zhou, M., Gatenby, R. A. & Ahmed, K. B. *Exploring deep features from brain tumor magnetic resonance images via transfer learning* in *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), 235–242.
155. Lorena, A. C., Garcia, L. P., Lehmann, J., Souto, M. C. & Ho, T. K. How Complex is your classification problem? A survey on measuring classification complexity. *arXiv preprint arXiv:1808.03591* (2018).
156. Lorena, A. C. & de Souto, M. C. *On measuring the complexity of classification problems* in *International Conference on Neural Information Processing* (2015), 158–167.
157. Louppe, G. *Understanding random forests: From theory to practice* PhD thesis (University of Liège, Belgium, 2014).
158. Lowe, D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* **60,** 91–110 (2004).
159. Lu, Y., Cheung, Y. & Tang, Y. Y. Bayes Imbalance Impact Index: A Measure of Class Imbalanced Dataset for Classification Problem. *arXiv preprint arXiv:1901.10173* (2019).
160. MacKay, D. J. & Mac Kay, D. J. *Information theory, inference and learning algorithms* (Cambridge university press, 2003).
161. Mahalanobis, P. C. *On the generalized distance in statistics* in (1936).
162. Maiora, J., Ayerdi, B. & Graña, M. Random forest active learning for AAA thrombus segmentation in computed tomography angiography images. *Neurocomputing* **126,** 71–77 (2014).
163. Marron, J. S., Todd, M. J. & Ahn, J. Distance-weighted discrimination. *Journal of the American Statistical Association* **102,** 1267–1271 (2007).
164. Mejri, D., Limam, M. & Weihs, C. A new dynamic weighted majority control chart for data streams. *Soft Computing* **22,** 511–522 (2018).

165. Meyer, J. S., Alvarez, C., Milikowski, C., Olson, N., Russo, I., Russo, J., Glass, A., Zehnbauer, B. A., Lister, K. & Parwaresch, R. Breast carcinoma malignancy grading by Bloom–Richardson system vs proliferation index: reproducibility of grade and advantages of proliferation index. *Modern pathology* **18,** 1067 (2005).
166. Mohandes, M., Deriche, M. & Aliyu, S. O. Classifiers combination techniques: A comprehensive review. *IEEE Access* **6,** 19626–19639 (2018).
167. Mosallam, A. *Remaining useful life estimation of critical components based on Bayesian Approaches.* PhD thesis (Université de Franche-Comté, 2014).
168. Moutafis, P., Leng, M. & Kakadiaris, I. A. An overview and empirical comparison of distance metric learning methods. *IEEE transactions on cybernetics* **47,** 612–625 (2017).
169. Mukhopadhyay, A., Singh, P., Sarkar, R. & Nasipuri, M. A study of different classifier combination approaches for handwritten Indic Script Recognition. *Journal of Imaging* **4,** 39 (2018).
170. Neisius, U., El-Rewaidy, H., Nakamori, S., Rodriguez, J., Manning, W. J. & Nezafat, R. Radiomic analysis of myocardial native T1 imaging discriminates between hypertensive heart disease and hypertrophic cardiomyopathy. *JACC: Cardiovascular Imaging* (2019).
171. Nigam, K. & Ghani, R. *Analyzing the effectiveness and applicability of co-training* in *9th International Conference on Information and Knowledge Management* (2000), 86–93.
172. Nioche, C., Orlhac, F., Boughdad, S., Reuzé, S., Goya-Outi, J., Robert, C., Pellot-Barakat, C., Soussan, M., Frouin, F. & Buvat, I. LIFEx: a freeware for radiomic feature calculation in multimodality imaging to accelerate advances in the characterization of tumor heterogeneity. *Cancer Research* **78,** 4786–4789 (2018).
173. Parmar, C., Grossmann, P., Bussink, J., Lambin, P. & Aerts, H. J. Machine Learning methods for quantitative radiomic biomarkers. *Scientific Reports* **5,** 13087 (2015).
174. Parmar, C., Grossmann, P., Rietveld, D., Rietbergen, M. M., Lambin, P. & Aerts, H. J. Radiomic machine-learning classifiers for prognostic biomarkers of head and neck cancer. *Frontiers in Oncology* **5,** 272 (2015).

175. Parmezan, A. R. S., Lee, H. D. & Wu, F. C. Metalearning for choosing feature selection algorithms in data mining: Proposal of a new framework. *Expert Systems with Applications* **75,** 1–24 (2017).
176. Paul, R., Hawkins, S. H., Balagurunathan, Y., Schabath, M. B., Gillies, R. J., Hall, L. O. & Goldgof, D. B. Deep feature transfer learning in combination with traditional features predicts survival among patients with lung adenocarcinoma. *Tomography* **2,** 388 (2016).
177. Pavlidis, P., Weston, J., Cai, J. & Grundy, W. N. *Gene functional classification from heterogeneous data* in *Fifth Annual International Conference on Computational Biology (ICCB)* (2001), 249–255.
178. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12,** 2825–2830 (2011).
179. Peerlings, J., Woodruff, H. C., Winfield, J. M., Ibrahim, A., Van Beers, B. E., Heerschap, A., Jackson, A., Wildberger, J. E., Mottaghy, F. M., DeSouza, N. M., *et al.* Stability of radiomics features in apparent diffusion coefficient maps from a multi-centre test-retest trial. *Scientific reports* **9,** 4800 (2019).
180. Pekalska, E. & Duin, R. P. *Classifiers for dissimilarity-based Pattern Recognition* in *15th International Conference on Pattern Recognition (ICPR)* **2** (2000), 12–16.
181. Pekalska, E., Paclik, P. & Duin, R. P. A generalized kernel approach to dissimilarity-based classification. *Journal of Machine Learning Research* **2,** 175–211 (2001).
182. Pekalska, E. M. *Dissimilarity representations in Pattern Recognition: concepts, theory and applications* PhD thesis (Delft University of Technology, Netherlands, 2005).
183. Pinto, F., Soares, C. & Mendes-Moreira, J. *Chade: Metalearning with classifier chains for dynamic combination of classifiers* in *Joint european Conference on Machine Learning and knowledge discovery in databases* (2016), 410–425.
184. Pitman, J., Yor, M., *et al.* The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator. *The Annals of Probability* **25,** 855–900 (1997).

185. Porro Muñoz, D., Talavera, I., W Duin, R. P. & Orozco Alzate, M. Combining dissimilarities for three-way data classification (2011).
186. Premebida, C., Carreira, J., Batista, J. & Nunes, U. *Pedestrian detection combining rgb and dense lidar data* in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2014), 4112–4117.
187. Prudêncio, R. B., Hernández-Orallo, J. & Martınez-Usó, A. *Analysis of instance hardness in Machine Learning using item response theory* in *Second International Workshop on Learning over Multiple Contexts in ECML 2015. Porto, Portugal, 11 September 2015* **1** (2015).
188. Pudil, P., Novovičová, J. & Kittler, J. Floating search methods in feature selection. *Pattern Recognition letters* **15,** 1119–1125 (1994).
189. Qadeer, A. & Qamar, U. *A Dynamic Ensemble Selection Framework Using Dynamic Weighting Approach* in *Proceedings of SAI Intelligent Systems Conference* (2019), 330–339.
190. Qi, G.-J., Tang, J., Zha, Z.-J., Chua, T.-S. & Zhang, H.-J. *An efficient sparse metric learning in high-dimensional space via l 1-penalized log-determinant regularization* in *Proceedings of the 26th Annual International Conference on Machine Learning* (2009), 841–848.
191. Qiu, S. & Lane, T. A framework for multiple kernel support vector regression and its applications to siRNA efficacy prediction. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* **6,** 190–199 (2009).
192. Rahman, A. & Verma, B. Effect of ensemble classifier composition on offline cursive character recognition. *Information Processing & Management* **49,** 852–864 (2013).
193. Rahman, A. & Verma, B. Novel layered clustering-based approach for generating ensemble of classifiers. *IEEE Transactions on Neural Networks* **22,** 781–792 (2011).
194. Raina, R., Battle, A., Lee, H., Packer, B. & Ng, A. Y. *Self-taught learning: transfer learning from unlabeled data* in *Proceedings of the 24th International Conference on Machine Learning* (2007), 759–766.
195. Rakotomamonjy, A., Bach, F. R., Canu, S. & Grandvalet, Y. SimpleMKL. *Journal of Machine Learning Research* **9,** 2491–2521 (2008).
196. Reunanen, J. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research* **3,** 1371–1382 (2003).

197. Rokach, L. Decision forest: Twenty years of research. *Information Fusion* **27,** 111–125 (2016).
198. Romero, E. & Sopena, J. M. Performing feature selection with multilayer perceptrons. *IEEE Transactions on Neural Networks* **19,** 431–441 (2008).
199. Roy, D. M., Teh, Y. W., *et al. The Mondrian Process.* in *Advances in Neural Information Processing Systems (NIPS)* (2008), 1377–1384.
200. Sabourin, M., Mitiche, A., Thomas, D. & Nagy, G. *Classifier combination for hand-printed digit recognition* in *Proceedings of 2nd International Conference on Document Analysis and Recognition (ICDAR'93)* (1993), 163–166.
201. Saeys, Y., Inza, I. & Larrañaga, P. A review of feature selection techniques in bioinformatics. *Bioinformatics* **23,** 2507–2517 (2007).
202. San Biagio, M., Crocco, M., Cristani, M., Martelli, S. & Murino, V. *Heterogeneous auto-similarities of characteristics (hasc): Exploiting relational information for classification* in *IEEE International Conference on Computer Vision (ICCV)* (2013), 809–816.
203. Schapire, R. & Freund, Y. *A decision-theoretic generalization of on-line learning and an application to boosting* in *Second European Conference on Computational Learning Theory* (1995), 23–37.
204. Serra, A., Fratello, M., Fortino, V., Raiconi, G., Tagliaferri, R. & Greco, D. MVDA: a multi-view genomic data integration methodology. *BMC Bioinformatics* **16,** 261 (Aug. 2015).
205. Setiono, R. & Liu, H. Neural-network feature selector. *IEEE transactions on Neural Networks* **8,** 654–662 (1997).
206. Shakhnarovich, G. *Learning task-specific similarity* PhD thesis (Massachusetts Institute of Technology, 2005).
207. Shechtman, E. & Irani, M. *Matching Local Self-Similarities across Images and Videos.* in *Conference on Computer Vision and Pattern Recognition (CVPR)* **2** (2007), 3.
208. Shental, N., Hertz, T., Weinshall, D. & Pavel, M. *Adjustment learning and relevant component analysis* in *European Conference on Computer Vision* (2002), 776–790.
209. Shi, T. & Horvath, S. Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics* **15,** 118–138 (2006).

210. Shi, T., Seligson, D., Belldegrun, A. S., Palotie, A. & Horvath, S. Tumor classification by tissue microarray profiling: random forest clustering applied to renal cell carcinoma. *Modern Pathology* **18,** 547 (2005).
211. Shi, Z., Zhang, C., Welch, M., Kalendralis, P., Wee, L. & Dekker, A. *CT-based Radiomics Predicting HPV Status in Head and Neck Squamous Cell Carcinoma* in *Radiotherapy and Oncology* **132** (2019), 18–19.
212. Simonyan, K. & Zisserman, A. Very deep convolutional networks for large scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).
213. Skurichina, M. & Duin, R. P. Bagging, boosting and the random subspace method for linear classifiers. *Pattern Analysis & Applications* **5,** 121–135 (2002).
214. Skurichina, M. & Duin, R. P. Bagging for linear classifiers. *Pattern Recognition* **31,** 909–930 (1998).
215. Sluimer, I., Schilham, A., Prokop, M. & Van Ginneken, B. Computer analysis of computed tomography scans of the lung: a survey. *IEEE Transactions on Medical Imaging* **25,** 385–405 (2006).
216. Smith, M. R., Martinez, T. & Giraud-Carrier, C. An instance level analysis of data complexity. *Machine Learning* **95,** 225–256 (2014).
217. Soares, R. G., Santana, A., Canuto, A. M. & de Souto, M. C. P. *Using accuracy and diversity to select classifiers to build ensembles* in *The 2006 IEEE International Joint Conference on Neural Network Proceedings* (2006), 1310–1316.
218. Song, J., Liu, Z., Zhong, W., Huang, Y., Ma, Z., Dong, D., Liang, C. & Tian, J. Non-small cell lung cancer: quantitative phenotypic analysis of CT images as a potential marker of prognosis. *Nature Research Scientific Reports* **6,** 38282 (2016).
219. Sorensen, L., Shaker, S. B. & De Bruijne, M. Quantitative analysis of pulmonary emphysema using local binary patterns. *IEEE transactions on medical imaging* **29,** 559–569 (2010).
220. Spanhol, F. A., Oliveira, L. S., Petitjean, C. & Heutte, L. A dataset for breast cancer histopathological image classification. *IEEE Transactions on Biomedical Engineering* **63,** 1455–1462 (2016).
221. Spanhol, F. A., Oliveira, L. S., Petitjean, C. & Heutte, L. *Breast cancer histopathological image classification using convolutional neural networks* in

2016 International Joint Conference on Neural Networks (IJCNN) (2016), 2560–2567.

222. Svetnik, V., Liaw, A., Tong, C., Culberson, J. C., Sheridan, R. P. & Feuston, B. P. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences* **43,** 1947–1958 (2003).
223. Tang, J., Alelyani, S. & Liu, H. Feature selection for classification: A review. *Data classification: algorithms and applications,* 37 (2014).
224. Theodoridis, S. & Koutroumbas, K. *Pattern Recognition* 4th ed. (2018).
225. Tixier, F., Um, H., Bermudez, D., Iyer, A., Apte, A., Graham, M. S., Nevel, K. S., Deasy, J. O., Young, R. J. & Veeraraghavan, H. Preoperative MRI-radiomics features improve prediction of survival in glioblastoma patients over MGMT methylation status alone. *Oncotarget* **10,** 660 (2019).
226. Tsanas, A., Little, M. A., Fox, C. & Ramig, L. O. Objective automatic assessment of rehabilitative speech treatment in Parkinson's disease. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* **22,** 181–190 (2014).
227. Tsymbal, A., Pechenizkiy, M. & Cunningham, P. *Dynamic integration with random forests* in *European Conference on Machine Learning (ECML)* (2006), 801–808.
228. Tulyakov, S., Jaeger, S., Govindaraju, V. & Doermann, D. in *Machine Learning in Document Analysis and Recognition* 361–386 (Springer, 2008).
229. Van de Sande, K. E., Gevers, T. & Snoek, C. G. *Color descriptors for object category recognition* in *Conference on Colour in Graphics, Imaging, and Vision* **2008** (2008), 378–381.
230. Van Der Maaten, L. *Learning Discriminative Fisher Kernels.* in *ICML* **11** (2011), 217–224.
231. Vanschoren, J., van Rijn, J. N., Bischl, B. & Torgo, L. OpenML: Networked Science in Machine Learning. *SIGKDD Explorations* **15,** 49–60 (2013).
232. Verlinde, P., Chollet, G. & Acheroy, M. Multi-modal identity verification using expert fusion. *Information Fusion* **1,** 17–33 (2000).
233. Vert, R. Designing a m-svm kernel for protein secondary structure prediction. *Master's thesis, DEA informatique de Lorraine* (2002).

234. Viswanath, S. E., Tiwari, P., Lee, G. & Madabhushi, A. Dimensionality reduction-based fusion approaches for imaging and non-imaging biomedical data: concepts, workflow, and use-cases. *BMC Medical Imaging* **17,** 2 (2017).
235. Walmsley, F. N., Cavalcanti, G. D., Oliveira, D. V., Cruz, R. M. & Sabourin, R. *An ensemble generation method based on instance hardness* in *2018 International Joint Conference on Neural Networks (IJCNN)* (2018), 1–8.
236. Wang, F. & Sun, J. Survey on distance metric learning and dimensionality reduction in data mining. *Data Mining and Knowledge Discovery* **29,** 534–564 (2015).
237. Wang, J., Kato, F., Oyama-Manabe, N., Li, R., Cui, Y., Tha, K. K., Yamashita, H., Kudo, K. & Shirato, H. Identifying triple-negative breast cancer using background parenchymal enhancement heterogeneity on dynamic contrast-enhanced MRI: a pilot radiomics study. *PLoS One* **10,** e0143308 (2015).
238. Wang, J., Wu, C.-J., Bao, M.-L., Zhang, J., Wang, X.-N. & Zhang, Y.-D. Machine Learning-based analysis of MR radiomics can help to improve the diagnostic performance of PI-RADS v2 in clinically relevant prostate cancer. *European Radiology* **27,** 4082–4090 (Oct. 2017).
239. Wang, T., Zhao, D. & Tian, S. An overview of kernel alignment and its applications. *Artificial Intelligence Review* **43,** 179–192 (2015).
240. Wang, W. & Zhou, Z.-H. *Analyzing co-training style algorithms* in *European Conference on Machine Learning* (2007), 454–465.
241. Wang, W., Arora, R., Livescu, K. & Srebro, N. *Stochastic optimization for deep cca via nonlinear orthogonal iterations* in *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)* (2015), 688–695.
242. Wei, L., Rosen, B., Vallières, M., Chotchutipan, T., Mierzwa, M., Eisbruch, A. & El Naqa, I. Automatic recognition and analysis of metal streak artifacts in head and neck computed tomography for radiomics modeling. *Physics and Imaging in Radiation Oncology* **10,** 49–54 (2019).
243. Weinberger, K. Q. & Saul, L. K. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research* **10,** 207–244 (2009).

244. Weinberger, K. Q. & Saul, L. K. *Fast solvers and efficient implementations for distance metric learning* in *Proceedings of the 25th International Conference on Machine Learning* (2008), 1160–1167.
245. Woloszynski, T. & Kurzynski, M. A probabilistic model of classifier competence for dynamic ensemble selection. *Pattern Recognition* **44,** 2656–2668 (2011).
246. Woods, K., Kegelmeyer, W. P. & Bowyer, K. Combination of multiple classifiers using local accuracy estimates. *IEEE transactions on pattern analysis and machine intelligence* **19,** 405–410 (1997).
247. Woźniak, M., Graña, M. & Corchado, E. A survey of multiple classifier systems as hybrid systems. *Information Fusion* **16,** 3–17 (2014).
248. Wu, W., Parmar, C., Grossmann, P., Quackenbush, J., Lambin, P., Bussink, J., Mak, R. & Aerts, H. J. Exploratory study to identify radiomics classifiers for lung cancer histology. *Frontiers in Oncology* **6,** 71 (2016).
249. Xia, R., Pan, Y., Du, L. & Yin, J. *Robust Multi-View Spectral Clustering via Low-Rank and Sparse Decomposition.* in *28th Conference on Artificial Intelligence* (2014), 2149–2155.
250. Xie, S., Girshick, R., Dollár, P., Tu, Z. & He, K. *Aggregated residual transformations for deep neural networks* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017), 1492–1500.
251. Xu, C., Tao, D. & Xu, C. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634* (2013).
252. Yang, L. & Jin, R. Distance metric learning: A comprehensive survey. *Michigan State Universiy* **2,** 4 (2006).
253. Yao, D., Zhao, P., Pham, T.-A. N. & Cong, G. *High-dimensional Similarity Learning via Dual-sparse Random Projection.* in *IJCAI* (2018), 3005–3011.
254. Ypsilantis, P.-P., Siddique, M., Sohn, H.-M., Davies, A., Cook, G., Goh, V. & Montana, G. Predicting response to neoadjuvant chemotherapy with PET imaging using convolutional neural networks. *PloS one* **10,** e0137036 (2015).
255. Zampieri, G., Van Tran, D., Donini, M., Navarin, N., Aiolli, F., Sperduti, A. & Valle, G. Scuba: scalable kernel-based gene prioritization. *BMC bioinformatics* **19,** 23 (2018).

256. Zhang, L. & Lin, X. Some considerations of classification for high dimension low-sample size data. *Statistical Methods in Medical Research* **22,** 537–550 (2013).
257. Zhang, X., Xu, X., Tian, Q., Li, B., Wu, Y., Yang, Z., Liang, Z., Liu, Y., Cui, G. & Lu, H. Radiomics assessment of bladder cancer grade using texture features from diffusion-weighted imaging. *Journal of Magnetic Resonance Imaging* **46,** 1281–1288 (2017).
258. Zhou, H., Vallières, M., Bai, H. X., Su, C., Tang, H., Oldridge, D., Zhang, Z., Xiao, B., Liao, W., Tao, Y., *et al.* MRI features predict survival and molecular markers in diffuse lower-grade gliomas. *Neuro-Oncology* **19,** 862–870 (2017).
259. Zhou, M., Scott, J., Chaudhury, B., Hall, L., Goldgof, D., Yeom, K. W., Iv, M., Ou, Y., Kalpathy-Cramer, J., Napel, S., *et al.* Radiomics in brain tumor: image assessment, quantitative feature descriptors, and machine-learning approaches. *American Journal of Neuroradiology* **39,** 208–216 (2018).
260. Zhu, X., Huang, Z., Shen, H. T., Cheng, J. & Xu, C. Dimensionality reduction by mixed kernel canonical correlation analysis. *Pattern Recognition* **45,** 3003–3016 (2012).
261. Zoph, B. & Le, Q. V. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578* (2016).
262. Zoph, B., Vasudevan, V., Shlens, J. & Le, Q. V. *Learning transferable architectures for scalable image recognition* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), 8697–8710.