Chapter 3

Representation Learning for Multi-source Heterogeneous Data

Abstract Representation learning for MSHD is a cutting-edge technology focused on extracting and integrating information from various sources and types of data. Its characteristics include data diversity, high dimensionality, and structural variety, which introduce complexity to data integration. To address these characteristics, MSHD representation learning must solve challenges such as integration complexity, handling large-scale and high-dimensional data, and dealing with noise and inconsistencies. It must also ensure that the dimensionality reduction process retains critical information and that the representations are interpretable. The scope of MSHD representation learning includes data preprocessing, dimensionality reduction, feature extraction, multimodal integration, visualization, and model development. Its objectives are to develop robust methods for integrating and harmonizing MSHD, uncover new insights and patterns, improve decision-making processes, and drive applications in fields such as healthcare, finance, and social sciences. Additionally, it aims to ensure the interpretability and practicality of the representations and support the ability to handle the increasing size and complexity of data sources. Overall, MSHD representation learning holds a significant position in data science and artificial intelligence, aiming to drive innovation and progress across various fields by addressing its unique challenges and leveraging its characteristics.

3.1 Introduction to MSHD Representation Learning

MSHD representation learning focuses on extracting and integrating information from diverse data types collected from multiple sources, such as text, images, and time-series. This diversity provides a comprehensive view but introduces complexity in data processing and analysis. The main challenges include the complexity of data integration, managing high-dimensional data, dealing with noise and inconsistencies, and ensuring that the reduced-dimensional representations retain critical information and are interpretable. As shown in Fig. 3.1, in this section we will explore the

representation learning methods for MSHD, starting with an introduction to figure out the concepts and target of MSHD representation learning, and discuss the challenges in MSHD representation learning. To address these challenges, we will introduce some methods of MSHD representation learning, such as multi-view learning (MVL), cross-modal learning (CML), graph-based representation learning (GRL), and deep learning.

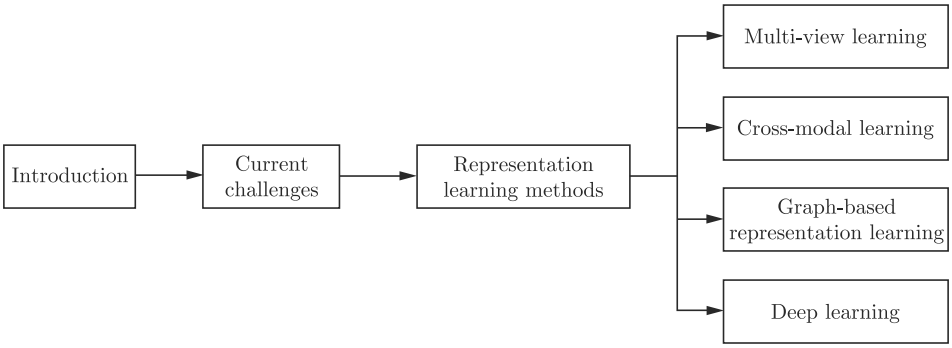

Fig. 3.1 Framework for MSHD representation learning: from foundational concepts and challenges to methodological advancements

First of all, we will introduce the objectives and challenges in MSHD representation learning. The primary objectives are to enhance data integration methods, uncover new insights and patterns, improve decision-making processes, and drive applications in various domains. Additionally, ensuring the interpretability and practicality of the learned representations and supporting the scalability of algorithms are crucial for handling the increasing complexity and size of data sources. MSHD representation learning is essential in data science and AI, aiming to address unique challenges and leverage data characteristics to unlock new insights, enhance decision-making, and drive innovation across multiple fields. To effectively handle the challenges associated with MSHD, several key characteristics must be considered:

(1) **Diversity and complexity of data types**: MSHD involves a variety of data types that differ in structure, format, and semantics. This diversity makes it challenging to create unified representations that capture the essential features of each data type while maintaining overall coherence. Techniques like feature extraction, transformation, and normalization are essential to handle this complexity.

(2) **High dimensionality**: Data from multiple sources often result in high-dimensional feature spaces. Techniques such as PCA, t-distributed stochastic neighbor embedding (t-SNE), and uniform manifold approximation and projection (UMAP) are employed to reduce dimensionality, facilitating efficient data integration and analysis.

(3) **Handling noise and inconsistency**: MSHD often includes noisy and

inconsistent data. Robust preprocessing methods, including filtering, normalization, and outlier detection, are crucial to mitigate the impact of noise and ensure the reliability of the integrated data.

(4) **Interpretability and explainability**: Ensuring that the learned representations are interpretable and explainable is critical, especially in fields like healthcare and finance where decisions must be transparent and justifiable. Techniques like self-explaining models and post-hoc interpretability methods help achieving this goal.

(5) **Scalability**: As data volumes continue to grow, scalable algorithms and systems are necessary to handle the increasing complexity and size of MSHD. Distributed computing frameworks like Apache Hadoop and Apache Spark are often employed to achieve scalability.

To effectively address these challenges, several methods have been proposed to uncover potential patterns, enhance decision-making processes, and drive applications across various domains. Next, we will introduce three typical methods, such as MVL, CML, and GRL, and explore how they play a role in the MSHD representation learning framework to tackle the aforementioned challenges.

3.2 Multi-view Learning for MSHD

3.2.1 Concepts of Multi-view Learning

MVL is a subfield of ML that deals with data coming from multiple sources or perspectives. It aims to leverage complementary information from these diverse views to improve learning performance and generalization. MVL is a type of problem that involves applying ML to data represented by multiple different sets of features. This emerging learning mechanism is largely driven by the properties of data in practical applications, where samples are described by different sets of features or different "views". For example, in multimedia content understanding, a multimedia segment can be described by both its video and audio signals. In web page classification, a web page can be described by the document text and also by the anchor text of hyperlinks pointing to that page. Another example is in content-based web image retrieval, where an object is described by both the visual features of the image and the surrounding text of the image. Moreover, a noteworthy fact in MVL is that even when a natural feature split does not exist, using an artificial manual split can still improve performance. Therefore, MVL is a very promising topic with wide applicability.

The following are key concepts in MVL help in gaining a deeper understanding of MVL.

(1) **Views**: In MVL, views mean different sets of features or representations that describe the same data instances. Each view provides a unique perspective or type of

information. Views can come from different data modalities (e.g., text, image, audio) or different feature sets within the same modality.

(2) **Consistency and complementarity**: Ensures that the predictions or representations from different views agree with each other. This means that if one view predicts a certain outcome, the other views should support or corroborate this prediction. Different views provide additional, non-redundant information that, when combined, enhances the overall learning process. This is crucial for improving model accuracy and robustness.

Up to now, the concept of MVL has influenced various branches of ML, leading to the development of numerous algorithms in this area, as follows:

(1) **Co-training**: A semi-supervised learning method where multiple classifiers are trained on different views[1]. These classifiers iteratively label unlabeled data and retrain each other, assuming the views are conditionally independent given the class label.

(2) **Canonical correlation analysis (CCA)**: A statistical method that finds the linear relationship between two views of data by maximizing the correlation between their projections[2].

(3) **Multiple kernel learning (MKL)**: It combines kernels from different views into a composite kernel, often used in SVMs to capture the combined information from multiple views[3].

(4) **Multi-view clustering**: Data are clustered by simultaneously considering multiple views, ensuring the coherence of clustering across all views[4].

3.2.2 Methods for Multi-view Representation Learning

3.2.2.1 Co-training

Co-training is an ML algorithm used when there are only small amounts of labeled data and large amounts of unlabeled data. One of its uses is in text mining for search engines. The core idea is to leverage the complementary information provided by different views to enhance the learning process and achieve better classification results[5]. The general process of co-training is shown in Fig. 3.2.

The process of co-training includes three steps.

(1) **Multiple views**: Co-training assumes that the data can be represented by at least two distinct and complementary views. Each view should be informative enough to build a classifier independently but should provide different types of information. The views are assumed to be conditionally independent given the class label, meaning that each view offers unique information about the class that is not redundant with the other view.

(2) **Initial training and pseudo-label generation**: Two or more classifiers

are initially trained separately on different views of the labeled data. Each classifier learns to predict the class labels based on its respective view. The trained classifiers then predict labels for the unlabeled data. These predictions, known as pseudo-labels, are added to the labeled dataset.

(3) **Iterative exchange and re-training**: The classifiers are retrained using the expanded labeled dataset, which now includes the pseudo-labeled examples. This process is repeated iteratively, with each classifier providing labels for the other view's data. During the iterative process, the classifiers' agreement on pseudo-labels helps to ensure that the newly labeled examples are likely accurate.

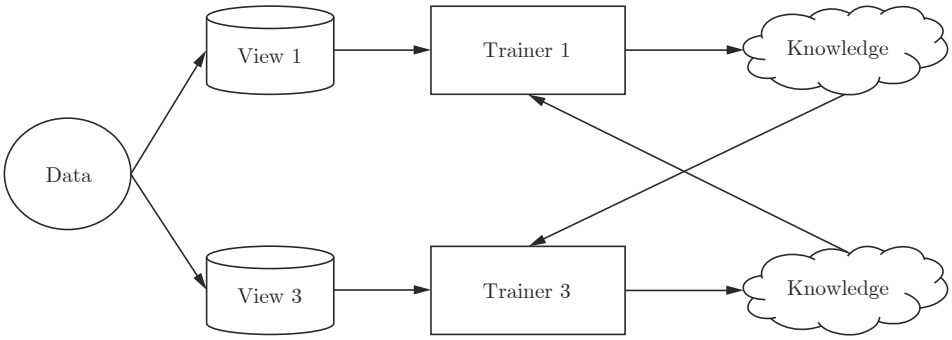

Fig. 3.2 General process of co-training

Co-training is widely applied in various domains, such as text classification and web mining. In text classification, co-training leverages different views such as the textual content of a document and the hyperlink structure linking to or from the document. This method is particularly effective in email spam detection, where features like email content and sender information are analyzed to identify spam. In web mining, co-training is utilized for document categorization by incorporating multiple features, including document content and web page metadata. Additionally, it considers both content and context, deriving different views from web content and user interaction data to enhance classification accuracy.

3.2.2.2 Canonical Correlation Analysis

CCA is a statistical method used to understand the relationship between two sets of variables[2]. As shown in Fig. 3.3, it finds linear combinations of the variables in each set such that the correlation between these combinations is maximized. CCA is commonly used in multivariate statistics, signal processing, and ML to identify and measure the associations between two data sets. The following is the algorithm design of CCA.

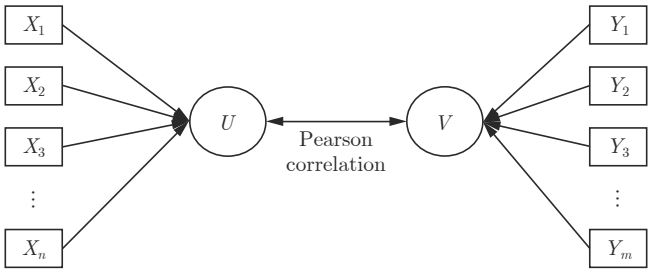

Fig. 3.3 The mechanism of CCA

(1) **Input data**: Two sets of variables $\boldsymbol{X}$ and $\boldsymbol{Y}$. $\boldsymbol{X}$ is an $n \times p$ matrix (n samples, p variables). $\boldsymbol{Y}$ is an $n \times q$ matrix (n samples, q variables).

(2) **Objective**: Find linear combinations $\boldsymbol{u} = \boldsymbol{X}\boldsymbol{a}$ and $\boldsymbol{v} = \boldsymbol{Y}\boldsymbol{b}$ such that the correlation ρ between $\boldsymbol{u}$ and $\boldsymbol{v}$ is maximized.

(3) **Mathematical formulation**: The canonical correlations are obtained by solving the eigenvalue problem:

$$(\boldsymbol{C}_{XX}^{-1}\boldsymbol{C}_{XY}\boldsymbol{C}_{YY}^{-1}\boldsymbol{C}_{YX} - \rho^2\boldsymbol{I})\boldsymbol{a} = 0 \tag{3.1}$$

$$(\boldsymbol{C}_{YY}^{-1}\boldsymbol{C}_{YX}\boldsymbol{C}_{XX}^{-1}\boldsymbol{C}_{XY} - \rho^2\boldsymbol{I})\boldsymbol{b} = 0 \tag{3.2}$$

where $\boldsymbol{C}_{XX}$, $\boldsymbol{C}_{YY}$, $\boldsymbol{C}_{XY}$, and $\boldsymbol{C}_{YX}$ are the covariance matrices between the variables in $\boldsymbol{X}$ and $\boldsymbol{Y}$.

(4) **Procedure**

- Compute the covariance matrices $\boldsymbol{C}_{XX}$, $\boldsymbol{C}_{YY}$, $\boldsymbol{C}_{XY}$, and $\boldsymbol{C}_{YX}$.
- Solve the eigenvalue problem to obtain the canonical correlations ρ and the canonical weights $\boldsymbol{a}$ and $\boldsymbol{b}$.
- The canonical variables are given by $\boldsymbol{u} = \boldsymbol{X}\boldsymbol{a}$ and $\boldsymbol{v} = \boldsymbol{Y}\boldsymbol{b}$.

CCA is widely applied in various fields for analyzing the relationships between two multivariate data sets. In multivariate data analysis, CCA is commonly used in psychology, economics, and social sciences to explore how different variables are related across these data sets. In the field of signal processing, CCA helps analyzing the relationships between different signal channels, such as in EEG or audio processing, by examining how the signals correlate across different channels. Additionally, in ML, CCA plays a crucial role in MVL, where it integrates and correlates information from different data sources. It is also employed in dimensionality reduction techniques to identify correlated subspaces between data sets, facilitating the extraction of meaningful patterns.

3.2.2.3 Multiple Kernel Learning

MKL is an ML technique that combines multiple kernels to improve the performance of algorithms, particularly in cases where the data have diverse representations or

domains. MKL aims to learn a weighted combination of kernels that best fits the data, enhancing the model's ability to capture different aspects of the data. A general procedure of MKL is shown in Fig. 3.4.

The process of MKL includes three steps:

(1) **Kernel selection**: MKL assumes that multiple kernels are available for different feature spaces or representations of the data. These kernels could represent different aspects, such as raw features, similarity measures, or various transformations of the input data. The kernels are assumed to capture different perspectives or types of information. The learning process involves finding the most appropriate combination of these kernels.

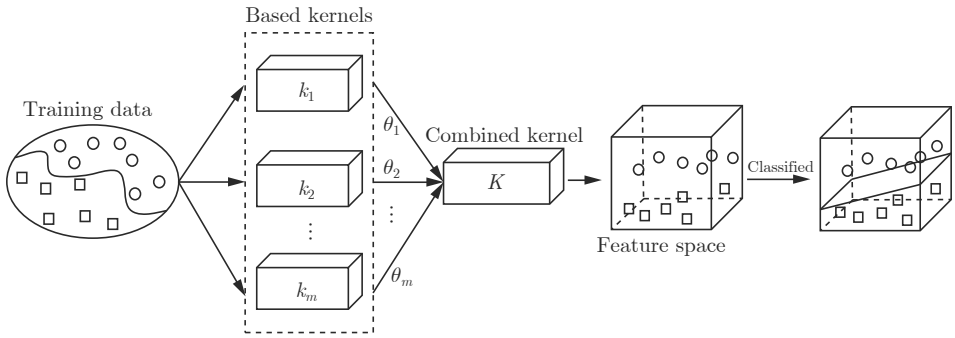

Fig. 3.4 General procedure of MKL

(2) **Model training with combined kernels**: A model is initially trained using the available kernels, either independently or as a combination of the kernels, to make predictions based on the data. The core of MKL in this step is to learn an optimal weight for each kernel, enabling an adaptive, weighted combination of all kernel functions to form a unified kernel representation for the learning task.

(3) **Iterative optimization of weights**: The kernel combination is refined through an iterative optimization process. The weights assigned to each kernel are adjusted to maximize overall model performance, typically guided by a specific optimization objective (e.g., margin maximization in SVMs). Techniques such as gradient descent or convex optimization are commonly employed to iteratively update these weights until convergence, thereby improving the model's accuracy.

The weighted combination of kernels helps the model capture various data characteristics and improve the overall performance. The agreement between the kernels ensures that different aspects of the data are jointly taken into account. The diversity of the kernels plays a critical role in the MKL process. Combining kernels that capture different properties of the data results in a more powerful model.

MKL has a wide range of applications, particularly in domains where the data

are highly diverse and come from multiple sources. In image classification, MKL can combine different kernels representing various image features, such as color, texture, and shape, to improve classification accuracy. Another application of MKL is in NLP, where it is used to combine different linguistic representations, such as syntactic structures, word embeddings, and semantic features, to improve text classification or sentiment analysis performance.

3.2.2.4 Multi-view Spectral Clustering

Multi-view spectral clustering (MVSC) is an extension of MVC. Each view provides different perspectives or features of the data, and MVSC aims to integrate this information to improve clustering performance. By leveraging the complementary information from different views, this method can produce more robust and accurate clusters.

The algorithm design of MVSC is shown in Fig. 3.5: The inputs are multiple views of data $\{\boldsymbol{X}^{(1)}, \boldsymbol{X}^{(2)}, \cdots, \boldsymbol{X}^{(m)}\}$, where each $\boldsymbol{X}^{(i)}$ is an $n \times d_i$ matrix representing the i-th view with n samples and d_i features. The objective is to integrate multiple views to obtain a unified clustering result that captures the intrinsic structure of the data. The procedure of MVSC is as follows:

(1) **Construct similarity matrices**: For each view i, construct a similarity matrix $\boldsymbol{S}^{(i)}$ that captures the pairwise similarities between samples. This can be done using various methods such as Gaussian kernel or KNN.

(2) **Construct laplacian matrices**: Compute the normalized Laplacian matrix $\boldsymbol{L}^{(i)}$ for each similarity matrix $\boldsymbol{S}^{(i)}$:

$$\boldsymbol{L}^{(i)} = \boldsymbol{I} - \boldsymbol{D}^{(i)^{-1/2}} \boldsymbol{S}^{(i)} \boldsymbol{D}^{(i)^{-1/2}} \tag{3.3}$$

where $\boldsymbol{D}^{(i)}$ is the degree matrix corresponding to $\boldsymbol{S}^{(i)}$.

(3) **Integrate multiple views**: Combine the Laplacian matrices from all views into a unified Laplacian matrix $\boldsymbol{L}_{\text{multi}}$. This can be done in several ways, such as averaging the Laplacians:

$$\boldsymbol{L}_{\text{multi}} = \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{L}^{(i)} \tag{3.4}$$

(4) **Perform spectral clustering**: Perform eigen-decomposition on the integrated Laplacian matrix $\boldsymbol{L}_{\text{multi}}$ and obtain the first k eigenvectors (where k is the number of clusters). Form the matrix $\boldsymbol{U}$ by stacking these k eigenvectors as columns. Apply K-means clustering on the rows of $\boldsymbol{U}$ to obtain the final clustering result.

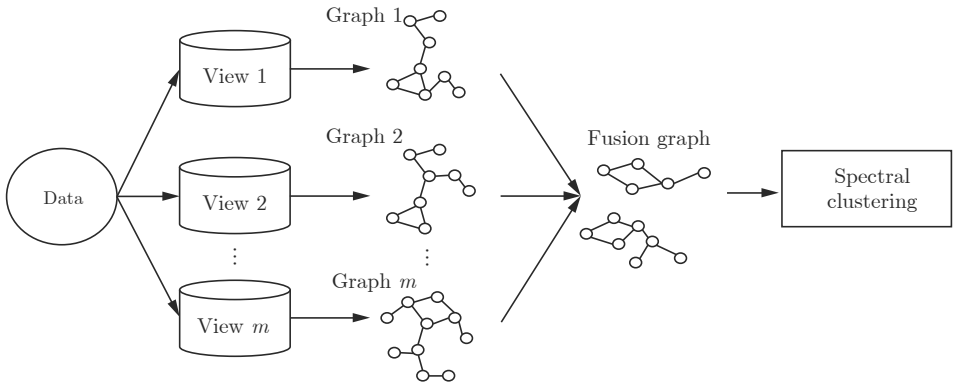

Fig. 3.5 Algorithm design of MVSC

MVSC is widely used to improve the accuracy and robustness of data clustering by integrating information from multiple views or perspectives. It is particularly useful in scenarios where data can be captured from different sources or modalities, such as in image processing, bioinformatics, and social network analysis. By leveraging complementary and consistent information across various views, MVSC helps in producing more precise and reliable clustering results compared to single-view approaches.

3.3 Cross-modal Learning for MSHD

CML is a ML approach that focuses on integrating and learning from data coming from different modalities or sources. In the context of MSHD, CML means combining and leveraging information from various data types, such as text, images, audio, and other sensor data, to enhance learning performance. CML aims to bridge the gap between different modalities and extract meaningful insights that are not possible when each modality is analyzed in isolation.

CML finds extensive applications across various fields by integrating different types of data to improve system performance. Cross-modal is widely utilized in search and retrieval systems, healthcare, autonomous driving, etc[6-7].

3.3.1 Concepts of Cross-modal Learning

CML is an advanced ML approach that involves integrating and learning from data across different modalities or data sources[8]. Modalities can include text, images, audio, and sensor data, among others. The goal of CML is to bridge the gap among these diverse data types, leveraging their complementary information to improve overall learning performance and gain deeper insights that would not be possible when each modality is analyzed in isolation.

Key concepts of CML which help in the following for comprehensive understanding.

(1) **Modality**: Different types of data, such as textual data, visual data (images, videos), auditory data (speech, sounds), and sensor data (temperature, location). Each modality offers unique and complementary information, providing a richer understanding when combined.

(2) **Alignment**: Mapping data from different modalities into a common space or representation, establishing correspondences between modalities.

(3) **Fusion**: Combining aligned information from different modalities to create a unified representation or decision. Fusion can be performed at different stages: early (combining raw features), late (combining decision outputs), or hybrid (combining both features and decisions).

3.3.2 Methods for Cross-modal Learning

3.3.2.1 Deep Cross-modal Learning

Deep CML is a technique that enhances model performance and generalization capabilities by leveraging complementary information across different modalities. With the rapid growth of multimedia and multi-modal data, effectively integrating and utilizing this diverse information has become a significant research direction. Deep CML designs effective model architectures and training strategies (as shown in Fig. 3.6) to enable knowledge transfer and fusion between different modalities. Deep CML utilizes deep learning techniques to process and fuse data from different modalities. The core idea is to train models capable of understanding and handling multi-modal data, allowing these models to use information from other modalities when one modality's information is insufficient, thus improving accuracy and robustness.

Deep CML typically involves the following key steps:

(1) **Feature extraction**: Extract features from each modality, such as using convolutional neural networks (CNNs) for images and recurrent neural networks (RNNs) for text[9-10].

(2) **Modality alignment**: Align features from different modalities to enable comparison and fusion in a common space.

(3) **Modality fusion**: Fuse the aligned features using methods like weighted averaging, concatenation, or attention mechanisms.

(4) **Joint learning**: Conduct joint learning through multi-task learning or specialized network structures based on the fused features.

(5) **Knowledge transfer**: Transfer information from one modality to another to enhance the performance of a single modality.