# DISCRIMINATIVE FEATURE LEARNING WITH IMPRECISE, UNCERTAIN, AND AMBIGUOUS DATA

By

CONNOR H. MCCURLEY

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2022

To my dad, who taught me that giving up is never an option

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Alina Zare, for all of her guidance and support throughout my studies and research. I will forever be grateful for her logical and reasonable advice, which always reminded me to put things into perspective. I am inspired by her dedication, passion, and creativity in research and teaching.

I would also like to thank my doctoral committee members, Dr. Paul Gader, Dr. Jose Principe, and Dr. Joseph Wilson, for all of their help and valuable suggestions.

Thank you to my former and current labmates. I am particularly grateful for my friends, Dylan Stewart, Josh Peeples, James Bocinksy, and Matt Cook for their constant encouragement, insights, discussion, and laughter.

Thank you to my mother, Kathy McCurley, my twin brother, Colton McCurley, and my girlfriend, Nicole Vitt, for their abundant love, support, and understanding. Finally, many thanks to my late father, Barry McCurley, for his seemingly endless wisdom and encouragement to pursue engineering.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

Figures page

LIST OF ABBREVIATIONS

| | |
|---|---|
| AE | Autoencoder |
| APR | Axis-Parallel Rectangles |
| BFM | Binary Fuzzy Measure |
| CA | Competency Aware |
| CCA | Canonical Component Analysis |
| ChI | Choquet Integral |
| CLFDA | Citation Local Fisher Linear Discriminant Analysis |
| CRF | Conditional Random Field |
| CT | Computed Tomography |
| CHL | Competitive Hebbian Learning |
| DD | Diverse Density |
| DM | Diffusion Maps |
| DR | Dimensionality Reduction |
| DSIAC | Defense Systems Information Analysis Center |
| EM | Expectation-Maximization |
| EM-DD | Expectation-Maximization Diverse Density |
| FA | Factor Analysis |
| fc | Fully Connected |
| FI | Fuzzy Integral |
| FM | Fuzzy Measure |
| FN | False Negative |
| FP | False Positive |
| FPR | False Positive Rate |

| | |
|---|---|
| FPS | Frames per Second |
| FOV | Field of View |
| GAE | Graph Autoencoder |
| GAP | Global Average Pooling |
| GLVM | General Latent Variable Model |
| GNG | Growing Neural Gas |
| GTM | Generative Topographic Mapping |
| GPS | Global Positioning System |
| HAC | Hierarchical Agglomerative Clustering |
| HS | Hyperspectral |
| HSI | Hyperspectral Image |
| ICA | Independent Component Analysis |
| IID | Independently and Identically Distributed |
| iPALM | Intertial Proximal Alternating Linearized Minimization |
| Isomap | Isometric Feature Mapping |
| KPCA | Kernel Principal Component Analysis |
| LDA | Fisher's Linear Discriminant Analysis |
| LE | Laplacian Eigenmaps |
| LiDAR | Light Detection and Ranging |
| LFW | Labeled Faces in the Wild Dataset |
| LLE | Locally Linear Embedding |
| LMNN | Large-Margin K-Nearest Neighbors |

| | |
|---|---|
| LFDA | Local Fisher Discriminant Analysis |
| LPP | Locality Preserving Projections |
| LVM | Latent Variable Model |
| LVQ | Learning Vector Quantization |
| MI-MTM | Multiple Instance Manifold Template Matching |
| MDS | Multi-dimensional Scaling |
| MI | Multiple Instance |
| MI-ALM | Multiple Instance Augmented Lagrangian Multiplier |
| MICI | Multiple Instance Choquet Integral |
| MIDA | Multiple-Instance Discriminant Analysis |
| MidLABS | Multi-Instance Dimensionality reduction by Learning a mAximum Bag margin Subspace |
| MI-FEAR | Multiple-Instance Feature Ranking |
| MIL | Multiple Instance Learning |
| MIDR | Multiple Instance Dimensionality Reduction |
| MILES | Multiple-Instance Learning via Embedded Instance Selection |
| MLE | Maximum Likelihood Estimation |
| MLP | Multilayer Perceptron |
| MoFA | Mixture of Factor Analyzers |
| MoPPCA | Mixture of Probabilistic Principal Component Analysis |
| MWIR | Mid-wave Infrared |
| MVU | Maximum Variance Unfolding |
| NLPCA | Nonlinear Principal Component Analysis |

| | |
|---|---|
| PC | Principal Curve |
| PAC | Probably Approximately Correct |
| pAUC | Partial Area Under the Curve |
| PCA | Principal Component Analysis |
| PGM | Probabilistic Graphical Model |
| QP | Quadratic Programming |
| RBF | Radial Basis Function |
| RKHS | Reproducing Kernel Hilbert Space |
| ROC | Receiver-Operating Characteristic |
| ROI | Region of Interest |
| S-LE | Supervised Laplacian Eigenmaps |
| SOA | State-of-the-Art |
| SOM | Self-organizing Feature Map |
| SVD | Singular Value Decomposition |
| SVM | Support Vector Machine |
| TG | Truncated Gaussian |
| TN | True Negative |
| TP | True Positive |
| TPR | True Positive Rate |
| t-SNE | t-Distributed Stochastic Neighbor Embedding |
| UMAP | Uniform Manifold Approximation and Projection |
| VAE | Variational Autoencoder |

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

DISCRIMINATIVE FEATURE LEARNING WITH IMPRECISE, UNCERTAIN, AND AMBIGUOUS DATA

By

Connor H. McCurley

August 2022

Chair: Alina Zare
Major: Electrical and Computer Engineering

Target detection is a paramount task in remote sensing which aims to detect points of interest from a set of data. A crucial aspect attributed to the success of target detection methods is the representation of the data which goes into them. Consequently, feature representation learning has been explored extensively in the literature [1, 2, 3, 4]. Discriminative feature learning approaches in the literature, however, have two common pitfalls. First, traditional approaches require vast quantities of precisely labeled groundtruth to guide training, which is typically difficult or impossible to acquire in practice. Second, current feature learning approaches which address label imprecision are limited in the features they can represent.

In this work, the Multiple Instance Choquet Integral (MICI) [5] was explored as a method for discriminative feature selection and fusion. Specifically, binary and regular fuzzy measures were investigated for their utilities in fusing activation feature maps. The explored approaches learn solely from imprecisely annotated data. Experimental results on both synthetic and real-world remote sensing datasets indicate the utility of the MICI as an approach for down-selecting sources form a set of features. A MICI fusing the sources provided by the discriminative feature selection approach demonstrated competitive target detection performance compared to class activation mapping methods in the literature.

# CHAPTER 1
# INTRODUCTION

## 1.1 Groundtruth Imprecision, Ambiguity, and Uncertainty

Target detection is a paramount area of research in the field of remote sensing which aims to locate an object or region of interest while suppressing unrelated objects and information [6, 7]. Target detection can be formulated as a two-class classification problem where samples belonging to a class of interest are discriminated from a background or non-target distribution [8, 6, 7]. The goal of target detection in remote sensing is to correctly identify every true positive instance (TP) in a given scene while indicating no false positives (FP) (non-target samples predicted as targets), and to identify every true negative instance (TN) without indicating any false negatives (FN) (true target samples predicted as non-target). Traditional supervised learning approaches require extensive amounts of highly precise, sample- or pixel-level groundtruth to guide algorithmic training. However, acquiring large quantities of accurately labeled training data can be expensive both in terms of time and resources, and in some cases, may even be infeasible to obtain [9]. To demonstrate these inherent labeling problems, consider the following real-world remote sensing examples, beginning with the hyperspectral target detection scenario described by Du [5] and Bocinsky [10].

Hyperspectral (HS) sensors collect spatial and spectral information of a scene by receiving radiance data in hundreds of contiguous wavelengths [11]. Due to their inherent properties, HS cameras can provide a broad range of spectral information about the materials present in a scene, and are thus useful for detecting targets whose spectral signatures vary from the background. Such examples include airplanes on a tarmac or weeds in a cornfield. While HS provides nice properties for target detection, there are significant challenges when utilizing this modality. First, the spatial resolution of HS cameras can be low. As an example, some HS cameras have spatial resolution of $30m^2$ when capturing scenes from the air. This implies that objects of interest in a scene, such as an airplane, will actually be contained in a single pixel along with other materials, such as asphalt. When performing target detection/recognition on that pixel, the HS

spectra will not be distinguishable as a single, pure material, but as a sub-pixel mixture where the actual materials present as well as their corresponding proportions are unknown. Second, assuming pure target pixels are available, accurate positioning at the desired resolution may not be. For example, when analyzing a scene from an airplane or satellite, it is necessary to denote the true locations of targets on the ground using a global positioning system (GPS). It is not uncommon, however, to experience GPS error of greater magnitude than the HS pixel-level spatial resolution. This implies that a halo of uncertainty potentially surrounds every target pixel in the hyperspectral image (HSI), making labeling on the pixel-level difficult.

This example demonstrates inherent infeasibility to obtain accurate sample-level labels due to sensor restrictions on both resolution and accuracy. Furthermore, label imprecision and ambiguity can often be presented from subjectivity between annotators. Many applications such as medical diagnosis and wildlife identification require domain experts to provide accurate data labels [12, 13]. However, there might not always be agreement between expert annotators and humans are prone to making mistakes. For example, when looking at computed tomography (CT) scans for malignant/benign tumors, many doctors would likely determine different pixel-level boundaries denoting a tumor, and in some cases, might even misclassify the detriment of the growth. Similarly, expert wildlife ecologists determining the identity of birds solely from their songs might be uncertain of a species due to corruptive background noise in the audio segment.

Finally, consider the scenarios shown in Figures 1-1 and 1-2. These figures show frames taken from the Defense Systems Information Analysis Center (DSIAC) MS-003-DB dataset [14] which demonstrates mid-wave infrared (MWIR) video segments of moving military vehicles taken at approximately 30 frames per second (FPS). Many computer vision algorithms have already been developed to perform target detection using canonical bounding boxes (shown in green in Figure 1-1) [15]. However, drawing tight boxes around targets in each video frame can be extremely tedious and time

consuming. Labeling burden can be exponentially reduced if a more ambiguous type of label can be used in training, such as relaxed bounding boxes (shown in blue in Figure 1-1 and bottom left in Figure 1-2), dots or scribbles as shown in Figure 1-2, or image-level binary indicators, as shown in Figure 1-3.

Techniques which can address these forms of label ambiguity while achieving comparable or better target detection than standard supervised methods can greatly ease the burdens associated with many remote sensing labeling tasks and allow for the application of pattern recognition techniques which would otherwise be infeasible.

## 1.2 Multiple Instance Learning

One paradigm for learning from uncertain, imprecise and ambiguous data has been an active area of research since the late 1990s and is known as multiple instance learning (MIL) [10]. While supervised learning assumes that each training sample is paired with a corresponding classification label, in multiple instance learning, the label of each sample is not necessarily known. Instead, MIL approaches learn from groups of data points called bags, and each bag concept is paired with a label [16]. Under the two-class classification scenario the bags are labeled as negative if all data points (or instances) are known to belong to the background class (not the class of interest). While the actual number of positive and negative instances may be unknown, bags are labeled positive if at least one instance is known to belong to the target class (also called a "true positive") [8] or to contain a proportion of true target. The goal of learning under the MIL framework is to train a model which can classify a bag as positive or negative (bag-level classification) or to predict the class labels of individual instances (instance-level classification). Consider again the example shown in Figure 1-3. The figure labeled as "Target" can be considered as a positive bag because, while the image is not accompanied with pixel-level labels, it is known that a true target pixel exists somewhere within the set. Additionally, the image denoted as "Background" can be considered as a negative bag, since it does not contain any pure target pixels. Another way to formulate

this problem could be to consider sets of these image patches, where a negative bag would only include samples of background patches, while a positive bag would contain at least one patch that held target pixels or part of a target. Given data of this type, multiple instance learning can be used to discover target and/or background class representatives which can be used for class discrimination, or a classifier can be trained to label and unseen test image as containing or excluding target pixels (bag-level classification) or to label each individual pixel as belonging to the target class (instance-level classification). As with this example, the MIL problem formulation fits many remote sensing scenarios and is thus an important area of investigation [5].

Multiple instance learning approaches in the literature can be broadly generalized into two categories: learning a generative model which effectively describes the positive and/or negative classes, or training a classifier to discriminate between target and background samples or bags [17]. The success of both learning frameworks, however, is highly dependent on the feature representations of the data and the metrics used to measure dissimilarity. Many feature learning approaches in the literature use supervised learning to obtain discriminative feature representations, or use supervised methods to fine-tune unsupervised feature extraction. However, this cannot be done directly in MIL because of the uncertainty on the labels [18].

### 1.3 Manifold and Feature Representation Learning

A popular approach to obtain discriminative feature representations is through the application of manifold learning, also commonly referred to as dimensionality reduction (DR), feature embedding, geometric machine learning or representation learning in the literature. The goal of manifold learning can be posed as discovering intrinsic (often lower-dimensional) features from the data which meet an overarching objective, such as: preserving variance, finding compressed representations of data, maintaining global or local structure or promoting discriminability in the embedded space [1, 2, 3, 4]. Manifold learning and dimensionality reduction have been studied extensively in the literature and

have been used for visualization, classification, redundancy removal, compression and data management, improving computational tractability and efficiency, combating the curse of dimensionality and obtaining more appropriate measures of dissimilarity [19, 20, 21, 22, 3, 23, 24, 25, 2].

Feature representation learning is difficult in MIL because of the uncertainty over instance-level labels. As a result, multiple instance manifold learning has scarcely been explored [18, 26, 27, 28, 29, 30, 31]. At the time of this work, all existing MIL manifold learning approaches in the literature suffer from two common pitfalls. Specifically, each method is only capable of learning a linear projection of the data. Additionally, each method relies on a priori target concepts, having no mechanisms to update target class concepts through learning. All of these methods rely on finding linear projections which will adequately separate positive and negative bags in the embedding space. However, these approaches fail when the the target and background data exhibit highly-curved structure in the feature space. For example, consider the data shown in Figure 1-4. This image shows the popular Swiss Roll dataset, which is a 2-dimensional manifold embedded into 3-dimensions according to a nonlinear function. Assume that the data classes lie on separate ends of the manifold, and that samples are governed by a smooth labeling function. In Figure 1-4A, the red samples denote the target class while the blue represent the background class. If using Euclidean (straight line) distance to measure dissimilarity, it would appear that many red samples are (untruly) close to the blue. However, if an alternative distance metric such as geodesic distance (following the curve) is used to measure dissimilarity between samples, the true class distributions are better represented. Figure 1-4B shows the unfolding of the manifold according to geodesic distance. It can be observed that, after the unfolding, the classification problem was transformed from nonlinear to linear, and a dimension of the data was deemed unnecessary.

## 1.4 Preliminary Investigation and Motivation for Approach

In order to show potential for the proposed work to advance SOA in the literature, initial experiments were conducted to demonstrate the utilitarian aspects of manifold information/ representation learning in bag and instance-level classification. Specifically, a multiple instance manifold template matching (MI-MTM) scheme was implemented and tested.

### 1.4.1 Multiple Instance Manifold Template Matching (MI-MTM)

An adaptation of the multiple instance learning algorithm, MI-ACE [8], was implemented where the pairwise geodesic distances between samples were used to select the most-likely target concepts within a set of positive and negative bags. Following the notation for multiple instance learning defined in Section 2.1, the generalized objective function for MI-ACE is given by

$$\arg\max_{\boldsymbol{t}} \frac{1}{N^+} \sum_{k:L_k=1} \mathcal{D}(\boldsymbol{x}_k^*, \boldsymbol{t}) - \frac{1}{N^-} \sum_{k:L_k=0} \frac{1}{N_k^-} \sum_{\boldsymbol{x}_n \in \boldsymbol{B}_k^-} \mathcal{D}(\boldsymbol{x}_n, \boldsymbol{t}) \tag{1-1}$$

where $N^+$ and $N^-$ are the number of positive and negative bags, respectively, $N_k^-$ is the number of instances in the $k^{th}$ negative bag, and $\boldsymbol{x}_k^*$ is the instance from positive bag $\boldsymbol{B}_k^+$ which is most likely a positive sample given $\boldsymbol{t}$. Optimizing the MI-ACE objective returns the target signature $\boldsymbol{t}$ (assumed to be a positive instance concept) which maximizes the detection statistic $\mathcal{D}(\cdot,\cdot)$ for the most-likely target instances in each of the positive bags while also minimizing the statistic across all negative instances. The most likely target signature from each positive bag is determined by

$$\boldsymbol{x}_k^* = \arg\max_{\boldsymbol{x}_n \in \boldsymbol{B}_k^+} \mathcal{D}(\boldsymbol{x}_n, \boldsymbol{t}) \tag{1-2}$$

which implies that $\boldsymbol{x}_k^*$ is the training sample with the maximum detection statistic given a target signature concept, $\boldsymbol{t}$. In order to use manifold information to influence target

concept selection, the detection statistic, $\mathcal{D}(\cdot,\cdot)$, was set as a similarity measure derived from the geodesic distances (Section A.1.3.3) between samples:

$$\mathcal{D} = \frac{1}{1+\mathcal{D}_{geo}} \tag{1-3}$$

which gives samples with smaller dissimilarities detection scores close to $1$, and samples with disparate geodesic distances scores near zero. The single most-likely positive sample in a training set can be selected greedily by setting it as the instance which obtains the maximum objective score. This sample can be viewed as a template target signature which was chosen according to the geodesic distances between instances in the input feature space.

Performing target detection on unseen test samples is simple with the implemented MI-Manifold Template Matching (MI-MTM) scheme. Essentially, the geodesic distances of test samples to the estimated target point are approximated as the distances of their closest corresponding training points, or keypoints, plus the Euclidean distances to their keypoints. This approximation of geodesic distance assumes that the keypoints are representative of the data and are sufficiently sampled such that the distances between test points and their closest keypoints are locally Euclidean. Detection scores in $[0,1]$ are given to the test instances according to Equation 1-3.

The implemented approach was meant to investigate the efficacy of incorporating manifold information as an approach for feature representation learning in instance-level discrimination tasks. By definition, however, the current approach acts only in the input feature space and does not take advantage of the intrinsic dimensionality of the data, nor does it exploit the discriminative power of supervised dimensionality reduction methods, as demonstrated in Section 1.4.2.

### 1.4.2 MI-MTM Results

Previous label refinement experiments showed that assigning instances their bag-level labels and refining before training is not always effective [29, 30]. Thus,

methods which explicitly learn positive/negative concepts for use in instance-level classification were investigated. The MI-MTM algorithm (Section 1.4.1) was compared with alternative similarity measures for instance-level classification on the Binary Swiss Roll, Separable Quadratic Surfaces and Non-separable Quadratic Surfaces synthetic datasets. Each dataset was perturbed with $\epsilon = 0.2$ Gaussian noise. The datasets were randomly partitioned into sets of bags where each bag was allowed to contain up to $40$ instances. Figure 1-5 shows one of the random bagging partitions of the data into positive and negative bags as well as the instances' true class labels on the Binary Swiss Roll synthetic dataset.

The multiple instance template selection approach proposed by Zare et al. [8] was compared for four different similarity metrics: cosine similarity, inverse Euclidean distance, inverse Mahalanobis distance and inverse geodesic distance (as described in Section 1.4.1). The resulting positive instance proposals were qualitatively analyzed. Figure 1-6 shows the selected template instances as well as the top-scoring points in each positive bag according to the selected positive template instance for one random bagging partition of the Binary Swiss Roll dataset. As can be seen from the figure, the objective using cosine similarity failed to select a template instance in the positive class. Additionally, all but the inverse geodesic distance selected some negative instances as positive bag representatives. This result shows that the geodesic distance measure was able to effectively utilize manifold information to detect positive instances on the nonlinear data manifold. Instance-level classification on the Binary Swiss Roll and Quadratic surfaces datasets using the multiple instance objective was also compared. Each of the three datasets was partitioned into random bag subsets and the multiple instance objective was trained and tested on a hold-out test set using each of the four objective/similarity metrics. ROC curves were used to provide a quantitative comparison between the four similarity metrics. The methods were trained and tested on $100$ random bagging partitions for each of the datasets to provide a notion of performance variability.

Figures 1-7, 1-8 and 1-9 show the resulting mean ROC curves along with variation bounds representing one standard deviation across the $100$ trials. Statistics were computed across the true positive rate. The AUC of the mean curves is also presented to show the average total classification ability for each objective. Figure 1-7 shows that all four combinations outperformed random chance. However, the objective using inverse geodesic distance significantly outperformed the alternatives. The manifold metric achieved a TPR of $1.0$ at a FPR of $0.2$, whereas the second quickest converging, Mahalanobis distance, did not get to a TPR of $1.0$ until a FPR of $0.8$. Additionally, the manifold metric obtained a mean AUC of $0.94$ while the second best was at $0.82$. This result was as expected on the Binary Swiss Roll dataset simply from the understanding of the data geometry and how the other metrics compute their notions of similarity. As can be seen from the ROC curves on the quadratic surfaces dataset, the objectives using Euclidean and cosine metrics often failed to predict better than chance. Both the Mahalanobis and manifold metrics were above random guess, with the manifold similarity ultimately rising to a TPR of $1.0$ before the Mahalanobis. While the manifold metric ultimately achieved better classification results than the Mahalanobis, it was more sensitive to variations in the training bags.

### 1.4.3 Conclusions from Initial Experiments

Preliminary results suggest that the incorporation of manifold information/feature representation learning may significantly impact the inference ability of multiple instance models for some datasets. However, the implemented method suffers from a few drawbacks. Specifically, MI-MTM learns a single target signature which may not be able to sufficiently describe complex objects. Additionally, a common drawback of methods/datasets is that the input features are assumed to be representative for the learning task. Yet, data often needs to be transformed to uncover the governing class distributions or to be amenable for the task (see Sections A.1 and A.2 for an extensive review of manifold and metric learning). Similarly, MI-MTM does not use bag-level

information to transform data into representations more amenable for instance-level classification. While the incorporation of nonlinear structural information proved useful for the synthetic datasets, there remains an open question concerning nonlinear feature learning under the MIL paradigm (Section 2.2 reviews current approaches in the literature for manifold learning/dimensionality reduction under the multiple instance learning paradigm).

## 1.5 Overview of Research

Discriminative feature learning approaches in the literature have two common pitfalls. First, traditional learning approaches require vast quantities of precisely labeled groundtruth to guide training. Data of this type, however, is typically difficult or impossible to acquire for real-world applications. Second, current feature learning approaches which address label imprecision are all linear, meaning they are inherently limited in the structure they can represent.

To address these two problems, this study examines methods for nonlinear feature learning and selection which promote class discriminability and are simultaneously capable of addressing uncertainty and imprecision in training data groundtruth. Specifically, this dissertation investigates the Multiple Instance Choquet Integral (MICI) [5] as a method for both feature selection and fusion for instance-level target detection. Specifically, the MICI with a binary fuzzy measure was investigated as a method to select discriminative instance-level features. The MICI with a full measure was then explored as an instance-level classifier on the selected sources.

Experiments were conducted to characterize the ability of the Choquet integral to select discriminative sources from multiple instance learning annotated data. Experiments were conducted on both synthetic data and real-world applications such target detection and semantic scene understanding in remote sensing imagery. Results indicate that the Choquet integral with binary fuzzy measure can successfully down-sample a collection of sources while promoting inclusion of target-discriminative

features. Results also indicate that the Choquet integral trained on the reduced source set is competitive with class activation map methods in the literature for target detection. All of the approaches investigated in this work learn from imprecise data annotations.

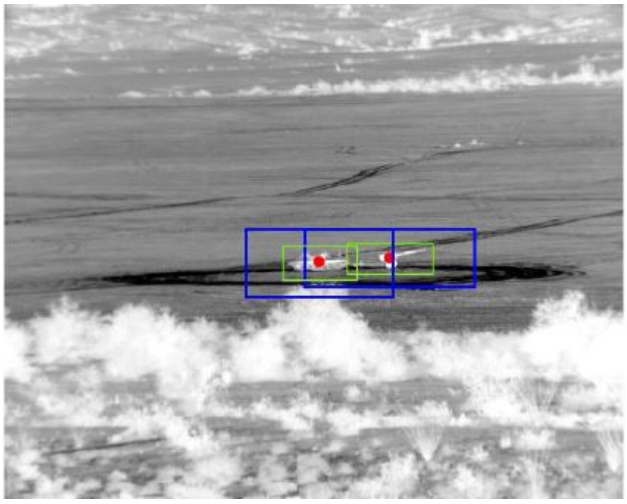


Figure 1-1. A sample frame from the DSIAC MS-003-DB MWIR dataset. Two targets are shown with canonical bounding boxes (green) and relaxed bounding boxes (blue). Red dots represent the centers of the target objects.

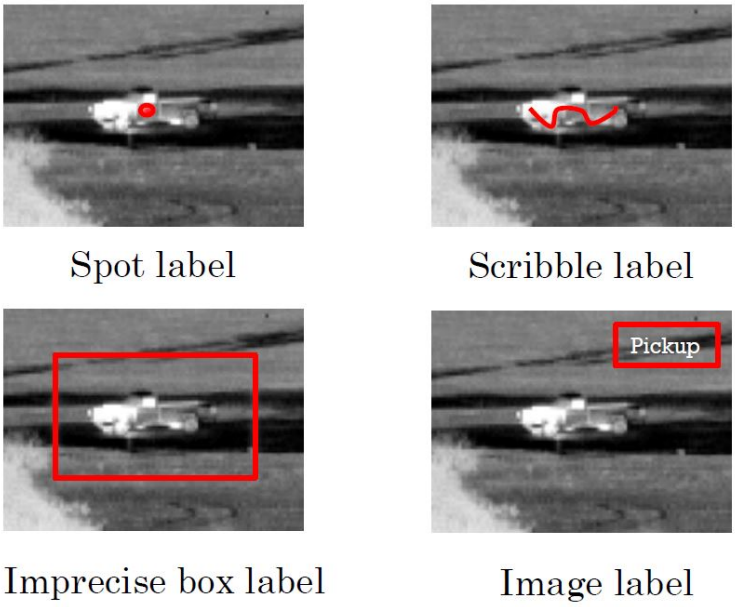
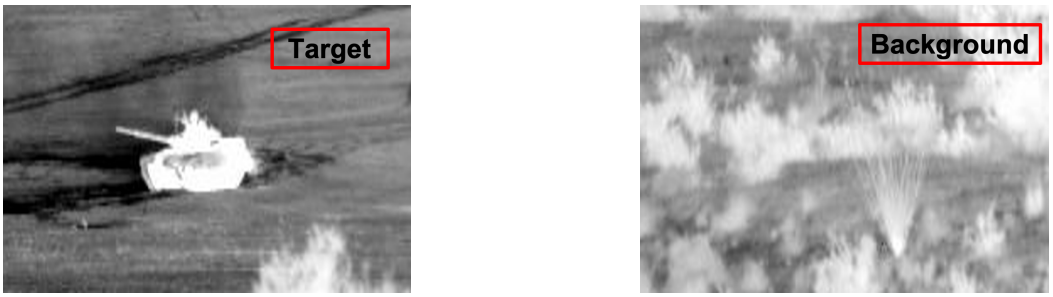
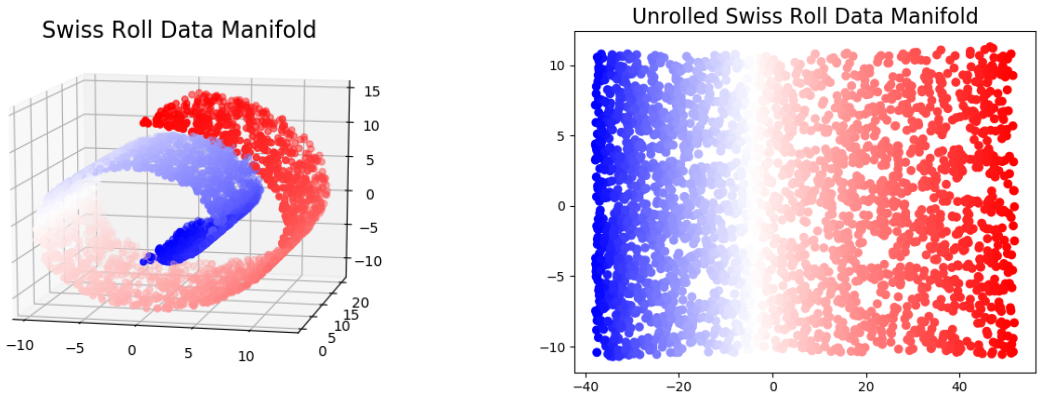



Figure 1-2. Examples of weakly-labeled infrared imagery. The images demonstrate various forms of weak groundtruth around a pickup truck taken with a mid-wave infrared camera. The images show spot, scribble, imprecise bounding box and image-level labels, respectively.



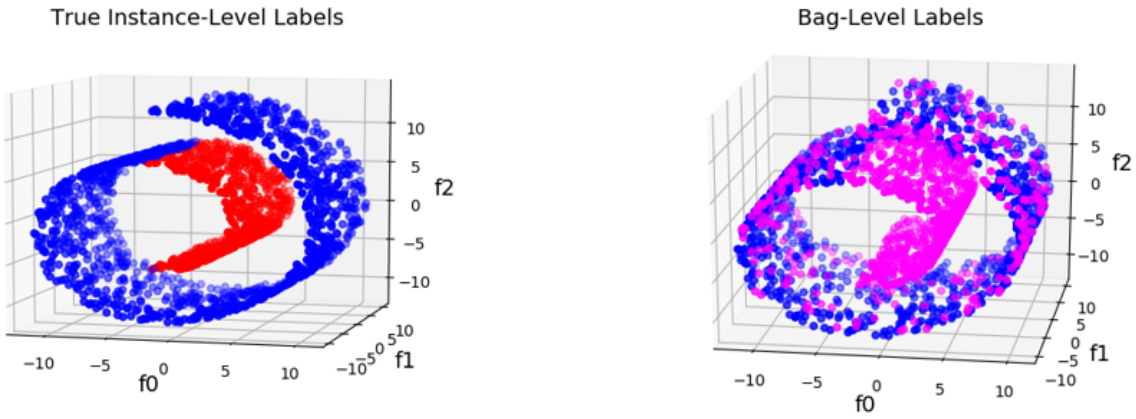

(A) (B)

Figure 1-3. Example of image-level labels for binary target detection. Image A) is denoted to contain pixels belonging to the target class somewhere within the image while image B) clearly contains samples solely from the background distribution.




(A) (B)

Figure 1-4. Swiss Roll manifold unfolding. A) This dataset is known as the Swiss Roll and it depicts a 2-dimensional manifold embedded in 3 dimensions. B) The Swiss Roll unfolded according to the geodesic path around the manifold.




(A) (B)

Figure 1-5. Binary Swiss Roll bags. A) Binary Swiss Roll where instances are colored by their corresponding true label. Red denotes the target class while blue represents the background class. B) Binary Swiss Roll where instances are colored by the label of the bag the are in. Blue denotes the instances in negative bags, while magenta shows the instances in positive bags. Negative bags are comprised of only negative instances while positive bags must contain at least one true positive instance, but can also contain negative instances.

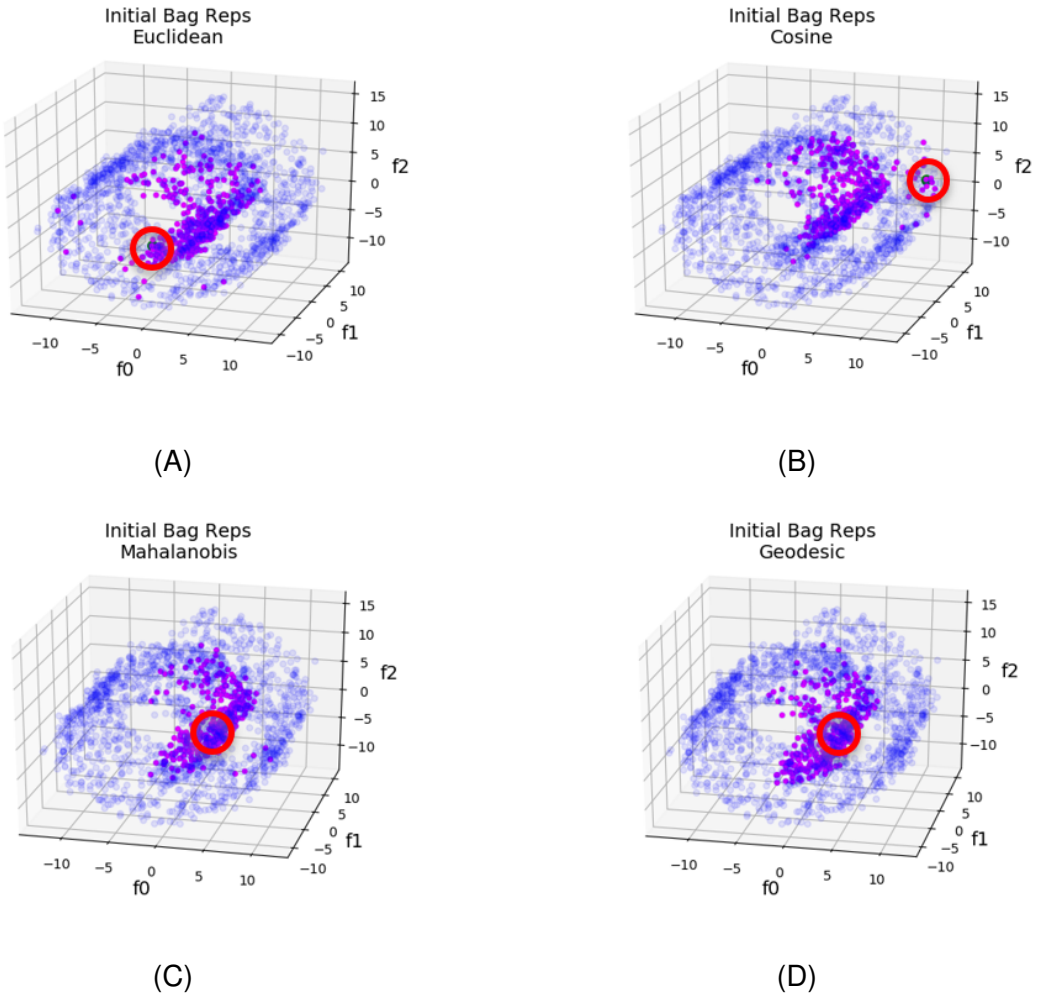



Figure 1-6. Positive instance template selection for A) Euclidean, B) cosine, C) Mahalanobis, and D) geodesic similarities on the Binary Swiss Roll dataset. Green instances circled in red are the selected positive template points. Magenta samples are the corresponding positive bag representatives.

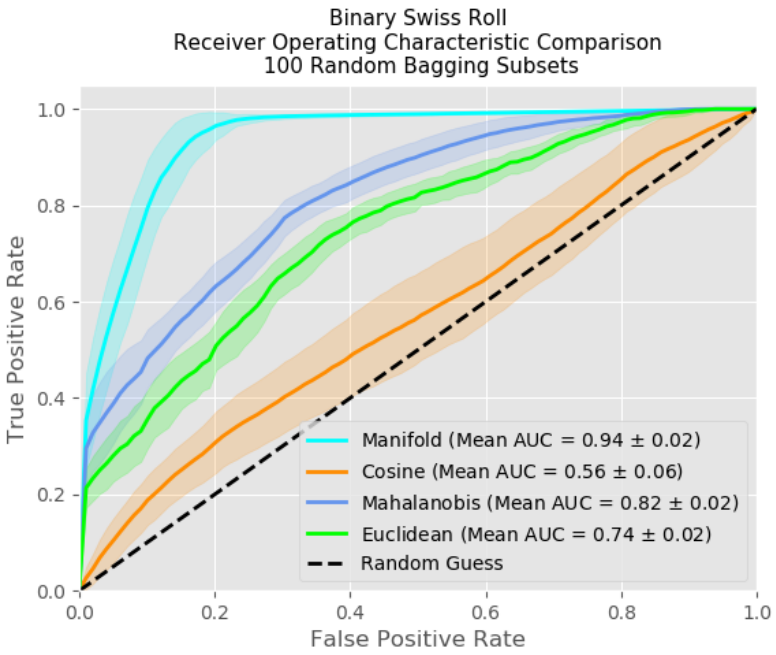



Figure 1-7. Receiver Operating Characteristic curves for test prediction of the multiple instance objective for four metrics: manifold (geodesic), cosine, Mahalanobis and Euclidean. Each algorithm was trained and tested on $100$ random bagging subsets. The curves show performance on the Binary Swiss Roll synthetic dataset.

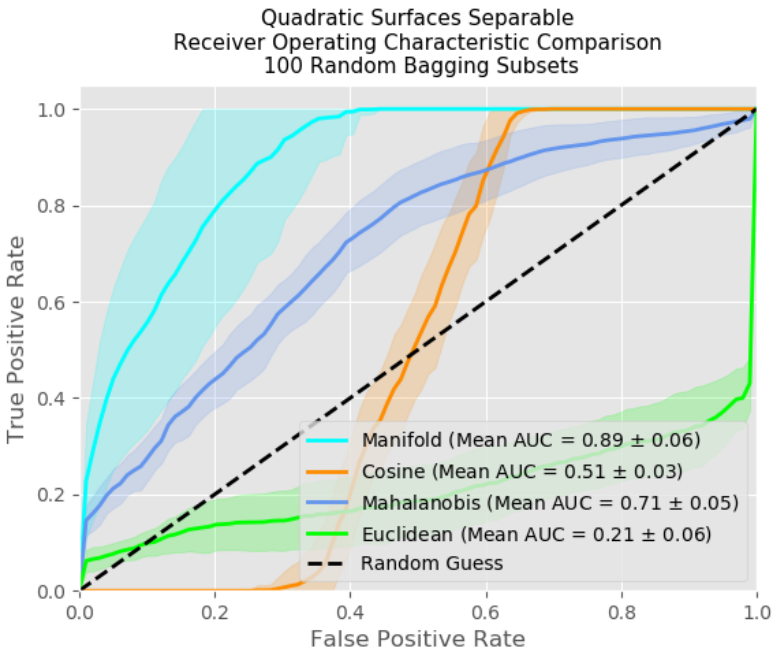



Figure 1-8. Receiver Operating Characteristic curves for test prediction of the multiple instance objective for four metrics: manifold (geodesic), cosine, Mahalanobis and Euclidean. Each algorithm was trained and tested on $100$ random bagging subsets. The curves show performance on the Separable Quadratic Surfaces synthetic dataset.

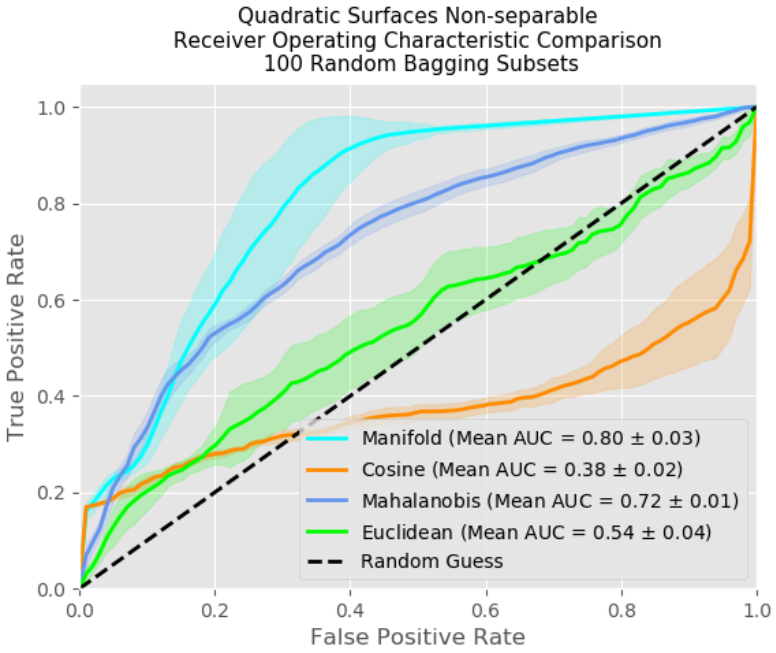



Figure 1-9. Receiver Operating Characteristic curves for test prediction of the multiple instance objective for four metrics: manifold (geodesic), cosine, Mahalanobis and Euclidean. Each algorithm was trained and tested on $100$ random bagging subsets. The curves show performance on the Non-separable Quadratic Surfaces synthetic dataset.

# CHAPTER 2
# BACKGROUND

This chapter provides a comprehensive review of literature pertinent to the work proposed in this document. First, a review is given in Section 2.1 on the multiple instance learning framework for learning from weak and ambiguous annotations. Notation is given and summaries of methods used in bag and instance-level classification and ranking are discussed. A section dedicated to reviewing manifold learning and dimensionality reduction techniques specifically tailored for use with weak labels is provided in Section 2.2. Extensive review on manifold learning, including classic approaches as well as supervised and semi-supervised adaptations is given in Appendix A.1 and related work on metric embedding, focusing heavily on the utilization of contrastive and triplet-based loss evaluation, is provided in Appendix A.2. Following Section 2.2, Section 2.3 reviews methods for estimating attention in a neural network for use in pixel-level classification. Section 2.4 reviews the Choquet Integral for combining sources of information. Reviews describe basic terminology and definitions. Foundational approaches are elaborated and advances are addressed.

## 2.1 Multiple Instance Learning

Multiple Instance Learning (MIL) was originally proposed by Dietterich [32] as a method to handle inherent observation difficulties associated with drug activity prediction. This problem, among many others, fits well into the framework of MIL where training labels are associated with sets of data points, called bags instead of each individual data points, or instances. Under the standard MIL assumption, a bag is given a "positive" label if it is known that at least one sample in the set represents pure or partial target. Alternatively, a bag is labeled as "negative" if does not contain any positive instances [18]. Let $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{D \times N}$ be training data where $D$ is the dimensionality of an instance, $\boldsymbol{x}_n$, and $N$ is the total number of training instances.

The data is grouped into K bags, $\boldsymbol{B} = \{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, with associated binary bag-level labels, $\mathcal{L} = \{L_1, \ldots, L_K\}$ where

$$L_k = \begin{cases} +1, & \exists \boldsymbol{x}_{kn} \in \boldsymbol{B}_k^+ \ni l_{kn} = +1 \\ -1, & l_{kn} = -1 \quad \forall \boldsymbol{x}_{kn} \in \boldsymbol{B}_k^- \end{cases} \tag{2-1}$$

and $\boldsymbol{x}_{kn}$ denotes the $n^{th}$ instance in positive bag $\boldsymbol{B}_k^+$ or negative bag $\boldsymbol{B}_k^-$ and $l_{kn} \in \{-1, +1\}$ denotes the instance-level label on instance $\boldsymbol{x}_{kn}$. Figure 2-1 demonstrates the concept of MIL bags. The objective of learning under MIL is, given only bag-level label information, to fit a model which can perform one of the following tasks: classification, regression, ranking or clustering [18].

### 2.1.1 Multiple Instance Learning with Manifold Bags

It should be noted that while the aforementioned MIL formulation is standard in the literature, Babenko et al. [33] introduced the framework of MIL using manifold bags . Their work claimed that instead of finite sets of instances, bags are inherently better represented as low-dimensional manifolds in a high-dimensional feature space. They considered the scenario of classifying whether or not an image contained a face. In this scenario, the entire image was considered to be a bag and patches of the image represented individual instances. It was assumed that the collection of instances collectively formed a low-dimensional manifold. The representation of bags as low-dimensional manifolds over the domain of instances is related to the Multiple-Instance Learning via Embedded Instance Selection (MILES) algorithm [34], which embeds bags into a feature space defined by similarities between instances. By giving more importance to the features representing samples in local neighborhoods around the most-likely-cause concept, MILES can be interpreted as representing bags according to local, manifold approximations centered on the instances which most-likely provide the bag-level labels. While related to bag-embedding methods such as MILES, the work by Babenko et al. [33] laid the groundwork for learning with manifold bags and

proved probably approximately correct (PAC) learnability under this (more general) framework. While it was worth mentioning this work as it aligns with the objective of manifold learning discussed in the proposed work, the remainder of this literature review focuses solely on the standard MIL formulation.

### 2.1.2 Tasks

Multiple instance learning in the literature can be broadly categorized into four tasks: classification, regression, ranking and clustering [18]. MIL classification can be performed at either the bag or instance level. The goal is to assign a class label to either the set of instances or the individual instances themselves. MIL regression consists of assigning a real-valued label to a bag (or instance) instead of a class label. A few methods have been proposed to rank bags or instances instead of assigning a class label or score. This problem differs from regression because the goal is not to obtain an exact real-valued label, but to compare the magnitude of scores to perform sorting. The clustering task can also be performed at the bag or instance-level. Bag-level clustering involves separating a set of unlabeled bags into distinct groupings based on a measure of similarity. Alternatively, clustering is often performed on the instances within individual bags in attempt to quantize the data into positive and negative concepts. Classification and ranking are the most pertinent tasks to the proposed work, and will thus be discussed in detail.

### 2.1.3 Multiple Instance Classification

The standard supervised learning task is to learn a classifier based on a training set of feature vectors, where each feature vector is paired with and associated class label. In the multiple instance classification task, the goal is to learn a classifier based on a training set of bags, where each bag is a set of feature vectors known as instances. In this setting, each bag is paired with an associated binary class label; however, the labels of each instance in the sets are unknown.

#### 2.1.3.1 Space paradigms

The multiple instance classification problem has been formulated under three paradigms: instance-space, bag-space and embedded-space [35]. Each paradigm is categorized according to how information presented in the MI data is exploited. In the instance-space paradigm, the discriminative information is considered to lie at the instance-level. An instance-level classifier is trained to separate the true positive instances from the true negative instances. Given an instance-level classifier, a bag-level classifier can be developed by simply aggregating the instance-level scores in a test bag. This paradigm is based on local, instance-level information. In the bag-space paradigm, the discriminative information is considered to lie at the bag-level. Under this paradigm, each bag is treated as a whole entity, and the learning process discriminates between entire bags. This paradigm is based on global, bag-level information. Considering that the bag space is a non-vector space, current BS methods make use of non-vectorial learning techniques which define distance metrics as a way to compare bags. In the embedded-space paradigm, each bag is mapped to a single feature vector which captures the relevant information about the entire bag. Consequently, the learning problem transforms into a standard supervised problem, where each feature vector is paired with an associated (bag-level) label. Similar to the bag-space paradigm, the embedded-space paradigm is also based on global, bag-level information. However, the difference between the two paradigms lies in the way bag-level information is extracted. In the bag-space paradigm, information is extracted implicitly through the definition of the distance or kernel function which maps two or more bags to a real number measuring the similarity between them. Alternatively, information is extracted explicitly in the embedded-space paradigm through the definition of the mapping function, which aggregates the instances contained in an individual bag to form a single real-valued vector representing the bag in the embedded vector space. Figure 2-2 demonstrates the differences between standard supervised classification and the three MIL classification

space paradigms. Apart from space paradigm, MIL classification methods in the literature can be organized according to the their primary approaches toward learning. A review of prominent MIL classification methods in the literature is provided, categorized according to learning approach.

#### 2.1.3.2 MIL classification approaches

MIL classification approaches in the literature can be categorized by the underlying principle used for learning. The categories discussed in this review are: Axis-Parallel Concepts, Maximum Likelihood, Distance-Based, Maximum Margin, Deep Learning, Probabilistic Graphical Methods, Dictionary Learning and Ensembles of Classifiers. Each approach is reviewed in the following.

**Learning axis-parallel concepts** Axis-Parallel Concepts are among the first group of methods used to solve MIL problems [32]. The foundation of this category is based on the method of Axis-Parallel Rectangles (APR), proposed by Dietterich et al. in the 1990's. An axis-parallel hyper-rectangle is a set of thresholds (one for each feature dimension) that is used to discriminate between two classes. It can also be viewed as an overlap or aggregation region of true positive instances in the feature space. The goal of APR is to find an axis-parallel hyper-rectangle in the feature space to represent the target concept [17, 36, 37]. A disadvantage of these approaches is that most of the data is ignored when working with large bags.

**Maximum likelihood** Similar to traditional Maximum Likelihood Estimation (MLE), the objective of maximum likelihood in the MIL setting is to train a classifier which maximizes the likelihood of the data. The most prominent of these methods is Diverse Density (DD) [38], which considers each bag as a manifold describing the instances. The goal of learning is to discover a prototype from the training data which maximizes the DD measure. Diverse Density is essentially a measure of the intersection of positive bags minus the union of the negative bags. By maximizing DD, a point of intersection can be found which is close to at least one instance from the positive bags while being as far as

possible from all points in the negative bags [17]. Zhang et al. later proposed Expectation-Maximization Diverse Density (EM-DD) [39]. EM-DD extends DD by viewing the relationships between instances and their corresponding bag's labels as latent variables. In other words, it asks which instance in the set corresponds to the bag's label [5]. In the E-step, an the instance from each bag which is the most probable for providing the bag its label is selected. Then in the M-step, a new concept point is found by maximizing DD with gradient ascent. This process is iterated until a stopping criteria is met, however, it will stop naturally as there only a finite number of instance combination that the algorithm can pick. Multiple instance maximum likelihood approaches have been used in a variety of problems, such as: stock selection, person identification, scene classification, hyperspectral target classification and explosive hazard detection [38, 40, 8, 41, 36, 42].

**Distance-based** A simple approach for classification in the supervised learning paradigm is to compare distances of test points to samples in the training set. Gartner et al. [43] defined the MI-Kernel by regarding each bag as a set of features and applying a set kernel directly to compare similarity. The Citation-$k$NN algorithm [5, 18, 44, 29] is a nearest-neighbor style classifier which borrows the idea of scientific citers and references when considering a bag's label. References are simply the nearest neighbors of a bag, while citers are the bags which have the query bag in it's $C$-nearest neighbors. The vanilla citation-$k$NN uses the minimal Hausdorff distance to measure bag similarity. The Hausdorff distance between two bags $\boldsymbol{B}$ and $\boldsymbol{B}'$ is defined as:

$$H(\boldsymbol{B}, \boldsymbol{B}') = \max\{h(\boldsymbol{B}, \boldsymbol{B}'), h(\boldsymbol{B}', \boldsymbol{B})\} \tag{2-2}$$

and

$$h(\boldsymbol{B}, \boldsymbol{B}') = \max_{\boldsymbol{b} \in \boldsymbol{B}} \min_{\boldsymbol{b}' \in \boldsymbol{B}'} ||\boldsymbol{b} - \boldsymbol{b}'|| \tag{2-3}$$

where $\boldsymbol{b}$ and $\boldsymbol{b}'$ are instances in bags $\boldsymbol{B}$ and $\boldsymbol{B}'$, respectively. Intuitively, the minimal

Hausdorff distance is the smallest value $H$ such that every instance in $\boldsymbol{B}$ has a point of $\boldsymbol{B}'$ within distance $H$, and every instance in $\boldsymbol{B}'$ has an instance in $\boldsymbol{B}$ within a distance of $H$. Variants of citation-$k$NN include Bayesian citation-$k$NN and fuzzy citation-$k$NN [5]. Additionally, the minimal Hausdorff distance has been substituted for the Earth Mover's Distance (EMD), Chamfer distance and specialized bag-distance kernels [35].

Two alternative distance-based MIL classifiers are MIGraph and miGraph [45]. Both methods consider bags as graphs to capture interdependence between instances. The distance measures used to compare bags is what separates the two methods. MIGraph explicitly maps every bag to an undirected graph and uses a graph kernel or metric such as graph edit distance to discriminate between positive and negative bags. Alternatively, miGraph implicitly constructs graphs by defining affinity matrices between instances and uses clique information to help distinguish positive from negative bags.

**Maximum margin** The concept of weakly-supervised maximum margin learning has been explored under a variety of techniques, the primary being Support Vector Machines (SVM) and metric embedding, which will be discussed in Section A.2. Andrews et al. [46] proposed two SVM methods under the MIL framework, namely, mi-SVM and MI-SVM, for instance-level and bag-level classification, respectively. The goal of a SVM is to train a classifier to maximize the margin between a small subset of training examples (called support vectors) and the decision hyper-plane [47]. Both MIL SVM methods follow a similar procedure to EM-DD. In MI-SVM, each positive bag is represented by a single instance which is considered to be the "most positive instance" in the bag. Optimization alternates between learning a decision boundary with SVM and selecting the positive bag representatives given the new classifier. In contrast, mi-SVM considers all points in the positive bags while learning the decision boundary. Every point in the positive bags is provided a positive label. The instance labels are refined iteratively under the constraint of the standard MIL assumption, that at least one instance from each positive bag must lie on the positive side of the decision hyper-plane [48]. MissSVM

is a semi-supervised max-margin approach which considers the instances in positive bags as unlabeled, and enforces a constraint that at least one of them is positive [49]. DD-SVM and Multiple-Instance Learning via Embedded Instance Selection (MILES) are both embedding-space methods which convert MIL into a standard supervised problem [34]. DD-SVM trains an SVM in a feature space constructed from a mapping defined by the local maximizers and minimizers of the DD function. MILES maps each bag into a feature space defined by the instances in the training bags via an instance similarity measure. A 1-norm SVM is applied to simultaneously select the important features and construct classifiers [13]. Additionally, Xiao et al. [50] developed an ensemble method in which the base classifier enforces a margin between optimal hyper-spheres while enclosing at least one instance from each positive bag inside the ball.

**Neural networks and deep learning** Recent developments in deep learning have also made their way into the MIL literature under the assumption that useful features can be learned by the networks using only bag-level labels. Gao et al. [51] used convolutional neural networks (CNN) with count-based region selection to perform weakly-supervised object localization. Ilse et al. [52] modelled the MIL problem as learning the Bernoulli distribution over the bag labels, where the label probability was parameterized by a neural network. Ilse's approach employs attention-based MIL pooling as a way to visualize which instances the network selects as being positive. A multi-instance multi-scale CNN to detect regions of interest in medical images was proposed by Li et al. [53]. Li's work introduced "top-k pooling" to aggregate feature maps of varying scales and spatial dimensions, allowing the model to be trained using weak, MIL annotations. Wang et al. [54] explored the use of a U-Net segmentation network architecture to obtain pixel-level ground-cover classification from image-level labels. A discriminative variational autoencoder (VAE) was used by Ghaffarzadegan [17] to maximize the difference between latent representations of positive and negative instances. Tu et al. [55] developed an end-to-end graph neural network which treats

each bag as a graph. Each bag is passed through the network to obtain a feature representation encapsulating the structural information present in the bag. Deep learning MIL approaches have been explored in a wide variety of applications, including: retinal image classification [55], histopathology classification [52], object localization [51] and region-of-interest proposal in medical images [53].

**Probabilistic graphical methods** Probabilistic graphical models (PGMs) are powerful tools used to capture inter-relations between random variables and learn structured models. In some problems, data exhibits an underlying structure between instances or bags that is more complex than simple co-occurrence. Capturing this structure may lead to better classification performance [18]. Deselaers and Ferrari [56] proposed a multi-instance conditional random field, MI-CRF. In this method, bags are modeled as nodes in a conditional random field (CRF), where each node can take one of the instances in the bag as its state. Classification corresponds to selecting one instance (positive bag) or selecting no instances (negative bag). Instance selection is formulated as inference in the CRF. This lets all bags to be considered jointly in training and testing. Thus, bags are jointly classified based on unary instance classifiers and pairwise dissimilarity measurements. Hajimirsadeghi and Mori [57] introduced a max-margin classification scheme using Markov networks. Yuksel et al. [58, 59] developed a multiple instance Hidden Markov Model (MI-HMM) for use in landmine detection. In this scenario, each bag is associated with a label, however, the bags can be composed of time sequences of variable length. A noisy-OR relationship is assumed between the sequences within each bag and the joint probability of the bags of sequences and the corresponding labels for the bags is maximized with a stochastic expectation maximization. PGMs have proven to work well on MIL problems exhibiting time series data and data with structural dependence.

**Dictionary learning** Dictionary Learning is a method of learning how to reconstruct a dataset from a much smaller set of building blocks called atoms [16]. Given a training

data set, the goal is to learn the set of atoms and sparse weights which reconstruct the data. At test, a bag or instance can be classified based on the atoms which effectively reconstruct the sample. Multi-Instance Dictionary Learning (MIDL) uses bag-level information with a least angle regression to alternatively learn a dictionary which represents the training data and regression weights for classification. Max-Margin Multiple Instance Dictionary Learning (MMDL) adopts the idea of the bag of words (BoW) model and trains a set of linear SVMs as codebooks [37]. Functions of Multiple Instances (FUMI) is a supervised technique which tries to learn target and background dictionaries such that a target instance can be written as a linear combination of a single target concept and multiple background atoms . This formulation considers target instances that may contain portions of background signature, as with sub-pixel target detection in hyperspectral imagery. A known problem with FUMI is that it does not work well with noisy labels. To account for this problem, extended Functions of Multiple Instances (eFUMI), uses bag-level labels to identify the data [37, 16]. A function is built in to determine whether a point labeled as target actually contains a portion of the target dictionary atoms. Task-driven extended Functions of Multiple Instances (TD-eFUMI) adopts the MI aspect of eFUMI and Task-Driven Dictionary learning to simultaneously learn target and background dictionaries in conjunction with a classifier [60]. Zare et al. [8] proposed the Multiple Instance Adaptive Cosine/Coherence Estimator and Spectral Matched Filter (MI-ACE and MI-SMF). These methods learn discriminative prototypes under the multiple instance learning framework to classify instances. However, these methods inherently consider only a single target concept. In order to capture intra-class variation among target instances, Multi-Target MI-ACE/SMF were proposed by Bocinsky [10]. Finally, Jiao et al. [61] presented Multiple Instance Hybrid Estimator (MI-HE) to learn multiple target and background concepts by maximizing the probability that positive bags are labeled as positive and negative bags are labeled as negative under a

noisy-OR model. Multiple instance dictionary learning problems have been applied successfully to sub-pixel hyperspectral target detection and landmine detection tasks [10, 41, 8, 60, 61].

**Ensembles of classifiers** Ensemble learning paradigms train multiple versions of a base classifier and aggregate the results to achieve a stronger classifier than any of the individuals. Ensemble methods are typically broken into two realms: bagging and boosting. Bagging employs bootstrap sampling to generate several training subsets from the original training set, then trains a learner on each data subset. The predictions from each component learner are aggregated in order to provide a final class score. Zhou and Zhang [44] studied whether ensemble learning paradigms could be used to enhance MI learners by applying bagging on base MI learners, namely, Diverse Density and Citation K-NN. Random Forest classifiers operate under the bagging paradigm and use a technique called divide-and-conquer. These classifiers deploy an ensemble of decision trees which iteratively divide the feature space and make simple thresholding decisions. The predicted class labels provided by each tree in the forest are combined to give a final class label. Leistner et al. [62] proposed MIForest which combines the random forest learning algorithm with MIL. Since only bag-level labels are known, MIForest treats the label of each instance as a random variable defined over a space of probability distributions. The instance labels are disambiguated by iteratively searching for distributions which minimize the overall classification error.

The other paradigm of ensemble learning is called boosting. The goal of boosting is, using the entire training set, to find a weighted combination of weak learners (that may perform only slightly better than chance), such that the combination produces a strong classifier with high classification accuracy [63]. Several MI boosting approaches have been proposed in the literature. Section 2.1.4 discusses a popular variant, MIL-Boost, in detail.

### 2.1.4 Multiple Instance Boosting (MIL-Boost)

In the standard supervised learning setting, the goal of binary classification is to learn a classification function $h : \mathcal{X} \rightarrow \mathcal{L}$ which maps data in $\mathcal{X}$ to a binary class label $\mathcal{L} \in \{-1, +1\}$. The objective of boosting is to train a classifier of the form

$$\boldsymbol{h}(\boldsymbol{x}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}) \tag{2-4}$$

where each $h_t : \mathcal{X} \rightarrow \mathcal{L}$ is a weak learner whose performance may only be slightly above chance, and the weights $\alpha_t$ are each weak learners' relative importance [64]. Boosting is a specific case of ensembling which combines multiple weak learners into a single strong classifier with low classification error. In each phase of training, samples classified incorrectly are given more weight in order to improve classification performance in the next iteration. The response of each weak classifier $h_t$ is given as the maximum response over all instances in the training set:

$$h_t = \arg\max_{h} \sum_{n=1}^{N} w_n h(\boldsymbol{x}_n) \tag{2-5}$$

Boosting under the MIL framework was originally proposed by Zhang et al. [63] and is known as MIL-Boost. Under MIL, it is assumed that every instance has a true label $l_{kn} \in \{-1, +1\}$. A bag is labeled positive if at least one of its instances are positive, and a bag is labeled negative if every instance is negative. This means that the label of a bag is provided as the max label over all instances in the bag:

$$L_k = \max_{n}(l_{kn}) \tag{2-6}$$

In this setting, the goal is to learn a classifier $\boldsymbol{h}$ using only bag-level labels such that $\max_n(\boldsymbol{h}(\boldsymbol{x}_{kn})) = L_k$. The score given to a sample is $l_{kn} = \boldsymbol{h}(\boldsymbol{x}_{kn})$ and $\boldsymbol{h}(\boldsymbol{x}_{kn}) = \sum_{t=1}^{T} \alpha_t h_t(\boldsymbol{x}_{kn})$ which is a weighted sum of predicted labels from each of the weak classifiers. Obviously, the sign of the prediction from the strong classifier provides the class label for the instance.

A natural objective is to minimize the negative log-likelihood between instances and their predicted labels. This can also be done at the bag-level. The probability that an instance is positive is given by the standard logistic function

$$p_{kn} = \frac{1}{1 + \exp(-l_{kn})} \tag{2-7}$$

and the probability that bag $k$ is positive is given by a "noisy OR", $p_k = 1 - \prod_{n \in k}(1 - p_{kn})$. Therefore, the likelihood assigned to a set of training bags is given by:

$$\mathcal{L}(\boldsymbol{h}) = \prod_{k=1}^{K} p_k^{L_k}(1 - p_k)^{(1-L_k)} \tag{2-8}$$

where $L_k \in \{0, 1\}$ is the actual label of the bag.

Following the idea of gradient boosting, the weight on each training instance is given as the derivative of the log-likelihood with respect to a change on the score given to the instance. Thus gradient descent can be used to update the the strong classifier. Pseudo-code for MIL-Boost is provided in Algorithm 2-1.

Each round of boosting consists of updating the weak learners according to the instances they misclassified and updating the strong classifier according to the new weights calculated over the weak learners. The goal of MIL-Boost is to be able to correctly classify each instance in a test bag as being positive or negative such that the label of the bag is given as the maximum label over all instances in the bag.

**Algorithm 2-1** MIL-Boost

**Input:** Dataset $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$

1: **for** $t = 1$ to $T$ **do**
2: Compute weights $w_{kn} = -\frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_{kn}}$
3: Train weak classifier $h_t$ using weights $|w_{kn}|$
$h_t = \arg\min_h \sum_{kn} \mathbf{1}(h(\boldsymbol{x}_{kn} \neq L_k))|w_{kn}|$
4: Find $\alpha_t$ via line search to minimize $\mathcal{L}(\boldsymbol{h})$
$\alpha_t = \arg\min_\alpha \mathcal{L}(\boldsymbol{h} + \alpha h_t)$
5: Update strong classifier $\boldsymbol{h} \leftarrow \boldsymbol{h} + \alpha_t h_t$
6: **end for**

### 2.1.5 Multiple Instance Ranking

Another primary task in MIL is ranking [18]. The ranking problem is different from classification because, instead of providing a binary class label, the objective is to order a set of bags [65, 66] or instances [67] according to preference for a particular task. Ranking is a key task under Preference Learning, or learning to predict preferences on a set of alternatives, which are often represented in the form of an order relation [68]. The statement that $\boldsymbol{x}$ is preferred to $\boldsymbol{x}'$ can be simply expressed as an inequality relation $f(\boldsymbol{x}) > f(\boldsymbol{x}')$, where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are instances, and $f$ defines a preference function. For example, given a news story about the Olympics, one might prefer to give it the label "sports" rather than "politics" or "weather". Alternatively, one might prefer to label one bag or instance as "positive" over another. In machine learning, the preference learning problem is often analyzed in two cases: learning instance preference and learning label preference [69]. Under the scenario of learning instance preferences, the training set consists of a set of pairwise preferences between instances. The objective is to learn the underlying ordering from the set of pairwise distances (such as ordering bags from the "most positive" to "least positive"). Alternatively, the goal of label preference learning is to order a pre-defined set of labels for each individual instance (such as ordering the preference of "positive" or "negative" on each individual bag or instance) [70, 71].

Ranking under the multiple instance framework was proposed by Bergeron et al. [65] for predicting hydrogen atom grouping in computational chemistry. The method is called MIRank [66]. MI ranking differs from MIC in that the label for each bag is not known. Instead, the MI ranking algorithms are provided preference information between pairs of bags. MIRank considers the partial ranking problem, which inherently exhibits three levels of structure: items (instances) belong to bags and bags belong to boxes. The objective is to learn a ranking function that can identify the preferred bag in each box. A concrete example where this framework can be applied is in learning positive target concepts. For a set of video frames (boxes), one may want to predict the smaller

image chips (bags) that have the highest probability of containing a target object (positive instances). MIRank uses a linear prediction function to rank instances in individual bags. The ranking of instances $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ in bags $I$ and $J$ is guided by the preference information between $I$ and $J$. Work by Hu et al. [67] introduced multiple-instance ranking based on a max-margin framework. In this setting, images were represented by sets of regions and the goal was to rank images according to relevance to a keyword. Assuming the preference relationship that $\boldsymbol{x}_m$ is preferable to $\boldsymbol{x}_n$ is denoted by $\boldsymbol{x}_m \succ \boldsymbol{x}_n$, the goal is to induce a ranking function $f : \mathbb{R}^D \rightarrow \mathbb{R}$ that fulfills the set of constraints

$$\forall \boldsymbol{x}_m \succ \boldsymbol{x}_n : f(\boldsymbol{x}_m) > f(\boldsymbol{x}_n) \tag{2-9}$$

The value of $f(\boldsymbol{x}_n)$ is referred to as the ranking score of $\boldsymbol{x}_n$ and is typically a linear function $f(\boldsymbol{x}_n) = \langle \boldsymbol{w}, \boldsymbol{x}_n \rangle = \boldsymbol{w}^T \boldsymbol{x}_n$. Adding slack variable $\zeta_{mn}$, the optimization problem can be solved with with the following objective:

$$\begin{aligned} \min_{\boldsymbol{w},\boldsymbol{\zeta}} \quad & \frac{1}{2}||\boldsymbol{w}||^2 + \gamma \sum_{m,n} \zeta_{mn} \\ \text{s.t.} \quad & \forall \boldsymbol{x}_m \succ \boldsymbol{x}_n : \langle \boldsymbol{w}, \boldsymbol{x}_m \rangle \geq \langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + 1 - \zeta_{mn} \\ & \forall m, n : \zeta_{mn} \geq 0 \end{aligned} \tag{2-10}$$

which is referred to as a ranking SVM. Assuming the optimal solution is $\boldsymbol{w}^*$, the ranking score of a test point $\boldsymbol{x}'$ is given as $f(\boldsymbol{x}') = \langle \boldsymbol{w}^*, \boldsymbol{x}' \rangle$. This framework assumes each sample $\boldsymbol{x}_n$ is a single instance. In multiple-instance ranking, we are given a set of preference relations between bag pairs and it is assumed that the score of a bag $\boldsymbol{B}_k$ is determined by the scores of the instances it contains

$$h(\boldsymbol{B}_k) = h\left(\{f(\boldsymbol{x}_n)\}_{n=1}^{N_k}\right) \tag{2-11}$$

Under this formulation, the objective can be re-written to consider bag scores as

$$\begin{aligned} \min_{f\in\mathcal{H},\zeta} \quad & \frac{1}{2}||f||_{\mathcal{H}}^2 + \gamma \sum_{m,n} \zeta_{mn} \\ \text{s.t.} \quad & \forall \boldsymbol{B}_m \succ \boldsymbol{B}_n : h(\boldsymbol{B}_m) \geq h(\boldsymbol{B}_n) + 1 - \zeta_{mn} \\ & \forall m, n : \zeta_{mn} \geq 0 \end{aligned} \tag{2-12}$$

A bag's score is determined by the scores of its instances. Different functions for providing a bag score have been investigated, including using the max of instance scores, the mean of instance scores and the softmax of instance scores.

Besides MIRank and multiple-instance ranking, Asif et al. [72] recently proposed pyLEMMINGS, which implements locally linear MI ranking by learning a large margin discriminant function from bags with corresponding integer rankings. While ranking has been successfully applied to text information retrieval, image retrieval [67] and bioinformatics [72], ranking under the MIL framework is still a relatively unexplored area of research.

### 2.2 Weakly Supervised Manifold Learning and Dimensionality Reduction

Although the specific feature vectors may be very high-dimensional, the underlying structure of a given dataset set is usually governed by only a few variables. Manifold learning (see Section A.1 for extensive details) seeks to uncover the governing distribution of data. Either implicitly or explicitly, most learning algorithms exploit this underlying structure to make learning and inference possible. If it is available, relevant information for a specific task can generally be incorporated to provide supervision and improve the performance of unsupervised methods. Supervised methods for dimensionality reduction assume that the samples lie on a manifold parameterized by multiple latent factors. Different from traditional manifold learning (where the goal is to preserve the relationships between samples), these methods find the most discriminative low-dimensional representations for classification tasks [73]. The optimal embedding uses class label information to minimize distances between nearby points

with the same class label while separating samples of different classes. Although supervised manifold learning often outperforms unsupervised methods for classification tasks, this learning cannot be done directly in MIL because of the uncertainty on the labels [18]. Moreover, fully-annotated samples are often difficult or impossible to obtain in many remote sensing applications [8]. Even with the successes of manifold learning, most the of the previous work has mainly focused on either fully supervised or unsupervised learning. Existing work in weakly supervised learning on manifolds has primarily considered the semi-supervised setting [74, 75, 76, 77, 78, 79, 80, 81, 82]. Methods in this category usually incorporate partially provided image labels and propagate the labels over the manifold approximated by the neighborhood graph on the images. In a broad sense, however, different situations of weak supervision (specifically, Multiple Instance Learning), have not been well studied [73]. The existing MIL manifold learning in the literature can be broken into two paradigms: LDA-based approaches and sparse, orthogonal matrix-based techniques [27]. Thus, all existing approaches are linear, meaning they may not work well if the underlying bag manifold exhibits curvature. Alternative weakly-supervised dimensionality reduction approaches have been proposed, however, they do not adhere to the constraints of MIL. The current literature for weakly supervised dimensionality reduction is reviewed in the following sections, beginning with reviews of MIL DR approaches, namely: MIDR, MidLABS, CLFDA, MIDA and MI-FEAR. Approaches using alternative definitions of weak learning are addressed at the end of the section.

### 2.2.1 MIDR

The first true MIL manifold learning experimentation was performed by Sun et al. [26] under the orthogonal matrix-based paradigm. To show the need for MIL-specific methods, Sun showed that Principal Component Analysis (PCA) failed to incorporate bag-level label information and thus provided poor separation between positive and negative bags. Additionally, traditional Linear Discriminant Analysis (LDA) was used to

project bags into a latent space which maximized between-bag separation, while minimizing within-bag dissimilarity. However, LDA often mixed the latent bag representations due to the uncertainty of negative sample distributions in the positive bags. These results have been shown many times in the literature [83, 84], so they were to be expected. However, they motivated the need for specialized manifold learning methods that are directly applicable with MIL. Therefore, Sun et al. [26] proposed Multiple Instance Dimensionality Reduction (MIDR), which optimizes an objective through gradient descent to discover sparse, orthogonal projection vectors in the latent space in conjunction with the Multiple Instance Logistic Regression classifier. The goal of MIDR is to discover a projection matrix $\boldsymbol{W} \in \mathbb{R}^{D \times d}$ which will increase discriminability between positive and negative bags in the latent embedding space. If given the $k^{th}$ training bag, $\boldsymbol{B}_k = \{\boldsymbol{x}_{k,1}, \ldots, \boldsymbol{x}_{k,n_k}\}$ with corresponding binary bag-level label $L_k \subset \{0, 1\}$, MIDR attempts to find a matrix $\boldsymbol{W}$ such that the projection of $\boldsymbol{B}_k \subset \mathbb{R}^D$ by $\boldsymbol{W}^T\boldsymbol{B}_k \subset \mathbb{R}^d$ increases the separation between positive and negative bags. The intuition is that the probability of the $k^{th}$ bag being positive $\Pr(L_k = 1|\boldsymbol{W}^T\boldsymbol{B}_k)$ should be close to one if it is positive and close to zero otherwise. This can be achieved by minimizing the squared loss between the actual and predicted label of each bag

$$\min_{\boldsymbol{W}} \sum_{k=1}^{K} (\Pr(L_k = 1|\boldsymbol{W}^T\boldsymbol{B}_k) - L_k)^2 \tag{2-13}$$

Taking advantage of the standard assumption, the posterior probability of a bag can be written in terms of the posterior probabilities of its instances

$$\Pr(L_k = 1|\boldsymbol{W}^T\boldsymbol{B}_k) = \max_n \Pr(l_{k,n} = 1|\boldsymbol{W}^T\boldsymbol{x}_{k,n}) \tag{2-14}$$

Equation 2-13 then becomes

$$\min_{\boldsymbol{W}} \sum_{k=1}^{K} (\max_n \Pr(l_{k,n} = 1|\boldsymbol{W}^T\boldsymbol{x}_{k,n}) - L_k)^2 \tag{2-15}$$

From the objective, it is clear that in order to minimize the squared loss, the distances between the key positive instances and all negative instances should be as large as possible. Additionally, $\boldsymbol{W}$ is required to be orthogonal in order to guarantee the resulting latent features are uncorrelated (to remove redundancy) as well as sparse (to improve interpretability). This implies that the new feature representations of instances $\boldsymbol{x}_{kn}$ in bag $\boldsymbol{B}_k$ are formed by linear combinations of the features in the input feature space. As defined by Zhu et al. [27], the MIDR optimization problem can be written succinctly as

$$\min_{\boldsymbol{W},\boldsymbol{\beta}} f(\boldsymbol{W},\alpha) + \gamma||\boldsymbol{W}||_1 \quad \text{s.t.} \quad \boldsymbol{W}^T\boldsymbol{W} = \mathbb{I}_d \tag{2-16}$$

where $\gamma$ is a positive number which controls the balance between the sparsity term $||\boldsymbol{W}||_1$ and the fitting term $f(\boldsymbol{W},\alpha) = \sum_{k=1}^{K}(P_k(\boldsymbol{W},\boldsymbol{\beta}) - L_k)^2$. In this case,

$$\begin{aligned} P_k(\boldsymbol{W},\boldsymbol{\beta}) &= softmax_\alpha(P_{k,1}(\boldsymbol{W},\boldsymbol{\beta}),\ldots,P_{k,n_k}(\boldsymbol{W},\boldsymbol{\beta})) \\ &= \frac{\sum_{n=1}^{n_k} P_{k,n} e^{\alpha P_{k,n}(\boldsymbol{W},\boldsymbol{\beta})}}{\sum_{n=1}^{n_k} e^{\alpha P_{k,n}(\boldsymbol{W},\boldsymbol{\beta})}} \end{aligned} \tag{2-17}$$

is the softmax approximation over $n$ of $\max(P_{k,1}(\boldsymbol{W},\boldsymbol{\beta}),\ldots,P_{k,n_k}(\boldsymbol{W},\boldsymbol{\beta}))$. A popular way to estimate the posterior probability is by logistic regression

$$P_{k,n}(\boldsymbol{W},\boldsymbol{\beta}) = \Pr(l_{k,n} = 1|\boldsymbol{W}^T,\boldsymbol{x}_{k,n}) = \frac{1}{1+\exp(-\boldsymbol{\beta}^T\boldsymbol{W}^T\boldsymbol{x}_{k,n})} \tag{2-18}$$

Pseudo-code for MIDR is provided in Algorithm 2-2.

**Algorithm 2-2** MIDR

**Input:** Multiple-instance dataset $\boldsymbol{B} = \{\boldsymbol{B}_1,\ldots,\boldsymbol{B}_K\}$, $\boldsymbol{L} = \{L_1,\ldots,L_K\}$, $L_k \in \{-1,+1\}$, sparsity parameter $\gamma$
**Output:** Projection matrix $\boldsymbol{W}$

```
while Not converged do
    Train multiple instance logistic regression h using data W^T B, L
    for B_k ∈ B do
        P_k ← probability h(B_k)
    end for
    Optimize Equation 2-16 for new W
end while
```

Sun et al. [26] used gradient descent to optimize the MIDR objective. An alternating optimization scheme was employed that switched between estimating the parameters of the MI Logistic Regression and solving for a new sparse, orthogonal embedding matrix. It was found that the newly developed method outperformed unsupervised instance-level dimensionality reduction approaches (applied to bags) for bag-level classification. MIDR was later revisited by Zhu et al. [27] where the optimization problem was reformulated using the inertial proximal alternating linearized minimization (iPALM) method. The advantage of Multiple Instance Augmented Lagrangian Multiplier (MI-ALM) [27] is that the problem variables can be managed separately and updated effectively. Additionally, the global convergence of MIDR was proved.

### 2.2.2 MidLABS

The other existing approaches for dimensionality reduction with multiple instance learning follow a LDA scheme. Multi-Instance Dimensionality reduction by Learning a mAximum Bag margin Subspace (MidLABS) [28] applies LDA to find a projection vector which simultaneously maximizes between-class scattering and minimizes within-class scattering to separate positive and negative bags in the embedding space. While most MIL approaches assume instances are independently and identically distributed (IID), MidLABS represents each bag as a neighborhood graph in order to take advantage of data structure and jointly constructs the scatter matrices by evaluating the scattering between bags. MidLABS optimizes the following objective:

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = \max_{\boldsymbol{w}} \frac{\boldsymbol{w}^T(\sum_{L_i \neq L_j} \boldsymbol{K}_{ij})\boldsymbol{w}}{\boldsymbol{w}^T(\sum_{L_i = L_j} \boldsymbol{K}_{ij})\boldsymbol{w}} \tag{2-19}$$

By defining a bag distance measure, $\boldsymbol{K}$, the the between-class and within-class scatter matrices can be constructed as

$$\boldsymbol{S}_b = \sum_{L_i \neq L_j} \boldsymbol{K}_{ij} \tag{2-20}$$

$$\boldsymbol{S}_w = \sum_{L_i=L_j} \boldsymbol{K}_{ij} \tag{2-21}$$

It can be observed that this problem follows LDA exactly, and can thus be solved as the generalized eigenvalue problem:

$$\max_{\boldsymbol{w}} J(\boldsymbol{w}) = \max_{\boldsymbol{w}} \frac{\boldsymbol{w}^T \boldsymbol{S}_b \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w}} \tag{2-22}$$

In order to take structural information into account, the customized bag distance measurement is defined by:

$$\boldsymbol{K}_{ij} = \frac{\sum_{a=1}^{n_i} \sum_{b=1}^{n_j} (\boldsymbol{x}_{ia} - \boldsymbol{x}_{jb})(\boldsymbol{x}_{ia} - \boldsymbol{x}_{jb})^T}{n_i n_j} + C \frac{\sum_{c=1}^{m_i} \sum_{d=1}^{m_j} (\boldsymbol{e}_{ic} - \boldsymbol{e}_{jd})(\boldsymbol{e}_{ic} - \boldsymbol{x}_{jd})^T}{n_i^2 n_j^2} \tag{2-23}$$

where $\boldsymbol{x}_{ia}$ is the $a^{th}$ instance in the $i^{th}$ bag, $n_i$ is the total number of instances in the $i^{th}$ bag, $\boldsymbol{e}_{ic}$ is the $c^{th}$ edge in the $i^{th}$ bag and $m_i$ is the total number of edges in the $i^{th}$ bag. This kernel represents each bag as an $\epsilon$-graph, where instances are treated as nodes. Edges are defined between two nodes if the Euclidean distance between them is lower than a threshold, $\epsilon$. [31]. Essentially, this measurement compares the closeness of bags by summing over all pairs of instances between the bags. Pseudo-code for MidLABS is provided in Algorithm 2-3.

**Algorithm 2-3** MidLABS

**Input:** Multiple-instance dataset $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$
**Output:** Linear projection vector $\boldsymbol{w}$

1: **for** bag $\boldsymbol{B}_k$ in $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$ **do**
2: $\quad G_k \leftarrow \epsilon$-graph for $\boldsymbol{B}_k$
3: **end for**
4: Define $\boldsymbol{K}_{ij} = \frac{\sum_{a=1}^{n_i} \sum_{b=1}^{n_j} (\boldsymbol{x}_{ia} - \boldsymbol{x}_{jb})(\boldsymbol{x}_{ia} - \boldsymbol{x}_{jb})^T}{n_i n_j} + C \frac{\sum_{c=1}^{m_i} \sum_{d=1}^{m_j} (\boldsymbol{e}_{ic} - \boldsymbol{e}_{jd})(\boldsymbol{e}_{ic} - \boldsymbol{x}_{jd})^T}{n_i^2 n_j^2}$
5: $\boldsymbol{S}_b = \sum_{L_i \neq L_j} \boldsymbol{K}_{ij}$
6: $\boldsymbol{S}_w = \sum_{L_i = L_j} \boldsymbol{K}_{ij}$
7: Solve the generalized eigenvalue problem $\boldsymbol{S}_b \boldsymbol{w} = \lambda \boldsymbol{S}_w \boldsymbol{w}$
8: Sort eigenvectors $\boldsymbol{w}$ by their eigenvalues $\lambda$
9: $\boldsymbol{w} \in \mathbb{R}^{D \times 1} \leftarrow$ top eigenvector

### 2.2.3 MIDA

In 2014, Chia et al. introduced Multiple-Instance Discriminant Analysis (MIDA) [30]. MIDA has the same objective as MidLABS, which is to discover a linear projection basis which separates bags in the embedding space. Both MIDA and MidLABS can be considered as MI extensions of LDA. However, the way these algorithms construct their scatter matrices is very different. While MidLABS constructs its scatter matrices at the bag level by directly evaluating the scattering amongst bags, MIDA uses instance-level information to formulate the within-class and between-class scatter. In other words, MIDA selects a prototype for each bag and utilizes the selected instances as bag representatives for constructing the scatter. The mean of all negative instances is used as the negative class prototype. The difficulty with this approach is the ambiguity on which instances are truly positive. The other major difference between MidLABS and MIDA is that MIDA does not consider structural information of data when formulating the scatter matrices. To select candidate positive prototypes, MIDA initializes a set by selecting the most-likely positive instances as those with the lowest density in a Gaussian likelihood estimated by all instances in the negative bags. An iterative optimization procedure is applied which trades-off between candidate positive instance selection and learning the projection weights into the latent space which maximizes the separation between bags with different labels and minimizes the distances between bags with the same label. Pseudo-code for MIDA is provided in Algorithms 2-4 and 2-5.

**Algorithm 2-4** Initialization of MIDA

**Input:** Multiple-instance dataset $\boldsymbol{B} = \{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$, parameter $\beta$

**Output:** Initialized positive prototypes $\boldsymbol{x}^+ = \{\boldsymbol{x}_1^+, \ldots, \boldsymbol{x}_{N^+}^+\}$

1: **for** bag $\boldsymbol{B}_k$ in $\boldsymbol{B}^+$ **do**

2: $\quad \boldsymbol{x}_k^+ \leftarrow \arg\min_{\boldsymbol{x}_{kn}} C \sum_{m=1}^{N^-} \sum_{o=1}^{N_m^-} \exp\left(\frac{||\boldsymbol{x}-\boldsymbol{x}_{mo}^-||_2^2}{\beta}\right) \forall n = 1, \ldots, N_k^+$

3: **end for**

**Algorithm 2-5** Projection vector calculation process of MIDA

**Input:** Multiple-instance dataset $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$, Initialized positive prototypes $\boldsymbol{x}^+ = \{\boldsymbol{x}_1^+, \ldots, \boldsymbol{x}_{N^+}^+\}$

**Output:** Linear projection vector $\boldsymbol{w}$

$\boldsymbol{\mu}^+ \leftarrow \frac{1}{N^+} \sum_{k=1}^{N^+} \boldsymbol{x}_k^+$

$\boldsymbol{\mu}^- \leftarrow C \sum_{m=1}^{N^-} \sum_{o=1}^{N_m^-} \boldsymbol{x}_{mo}^-$

$\boldsymbol{S}_w^+ \leftarrow \sum_{k=1}^{N^+} (\boldsymbol{x}_k^+ - \boldsymbol{\mu}^+)(\boldsymbol{x}_k^+ - \boldsymbol{\mu}^+)^T$

$\boldsymbol{S}_w^- \leftarrow \sum_{m=1}^{N^-} (\boldsymbol{x}_m^- - \boldsymbol{\mu}^-)(\boldsymbol{x}_m^- - \boldsymbol{\mu}^-)^T$

$\boldsymbol{S}_w \leftarrow \boldsymbol{S}_w^+ + \boldsymbol{S}_w^-$

$\boldsymbol{S}_b \leftarrow \sum_{k=1}^{N^+} \sum_{m=1}^{N^-} (\boldsymbol{x}_k^+ - \boldsymbol{x}_m^-)(\boldsymbol{x}_k^+ - \boldsymbol{x}_m^-)^T$

Solve the generalized eigenvalue problem $\boldsymbol{S}_b \boldsymbol{w} = \lambda \boldsymbol{S}_w \boldsymbol{w}$

Sort eigenvectors $\boldsymbol{w}$ by their eigenvalues $\lambda$

$\boldsymbol{w} \in \mathbb{R}^{D \times 1} \leftarrow$ top eigenvector of $\frac{\boldsymbol{w}^T \boldsymbol{S}_b \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w}}$

### 2.2.4 CLFDA

Finally, Citation Local Fisher Linear Discriminant Analysis (CLFDA) [29] incorporates citation and reference information into local Fisher Discriminant Analysis, thus it can also be treated as a MI extension to LDA. Contrary to the previously-mentioned approaches, CLFDA operates at the instance-level, attempting to find a subspace where positive and negative instances are maximally-separable [31]. CLFDA can be viewed as complimentary to MIDA [30]. Whereas, MIDA tries to seek true positive instances in positive bags, CLFDA attempts to detect incorrectly labeled instances (false positives) in positive bags. CLFDA pre-labels all instances with the label of their bag $l_{kn} = L_k, \forall n = 1, \ldots, n_k$, then utilizes neighborhood information to detect false positive instances. In other words, it uses the assumption that if an instance in a positive bag is close to many points in the negative bags, it is likely also negative. CLFDA defines references simply as the $R$ nearest neighbors to $\boldsymbol{x}_n$, while $citers$ are defined as the samples which have $\boldsymbol{x}_n$ in their $C$-nearest neighbors, or $citers(\boldsymbol{x}_n) = \{\boldsymbol{x}_m | \boldsymbol{x}_n \in CNN(\boldsymbol{x}_m), m = 1, \ldots, N\}$. The steps of CLFDA are to 1.) construct a $\max(R, C)$-NN graph, where the $\max(R, C)$-NN graph is a $K$-NN graph with $K = \max(R, C)$. This is used to detect false positives and re-label them as negative. If

the ratio of instances from negative bags versus instances from positive bags in the references and citers of $\boldsymbol{x}_n$ exceeds a threshold $\tau$, then $\boldsymbol{x}_n$ is given a negative label. This is only done for instances from positive bags, as the SMI assumptions states that all instances in negative bags should be negative. 2.) Construct scatter matrices using the provided positive and negative instances. 3.) Find projection weights which simultaneously maximize the distances between positive and negative bags and minimizes the distances between bags with the same class labels using Local Fisher Discriminant Analysis (LFDA). LFDA is a version of LDA designed to address multi-modal data distributions. The primary difference between LDA and LFDA is that when maximizing and minimizing the between-class and within-class scatter matrices, respectively, the scatter contribution of a single instance pair is weighted by the locality of the pair. This prevents instances from different clusters sharing the same label from being forced into the same space. This also allows multiple dimensions to be analyzed in the embedding space, whereas LDA is limited to a single dimension for binary classification. Pseudo-code for CLFDA is given in Algorithm 2-6 [31]. A potential downside of CLFDA, however, is that it has been shown that pre-labeling all instances in positive bags as positive often leads to poor results, especially when there are large numbers of negative instances in the positive bags [30].

As previously mentioned, all existing MIL dimensionality reduction approaches in the literature are solely linear. However, many modalities exhibit nonlinear variation. Therefore, nonlinear manifold learning approaches should be developed which operate under the multiple instance learning framework.

### 2.2.5 MI-FEAR

While the previously mentioned approaches were based on feature extraction, multiple instance dimensionality reduction has also been investigated under the feature selection paradigm. Specifically, Latham [31] used feature ranking to determine the most important features for instance-level classification. Feature ranking considers each

**Algorithm 2-6** CLFDA

**Input:** Multiple-instance dataset $\boldsymbol{B} = \{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$, parameters $C, R, \tau$
**Output:** Projection matrix $\boldsymbol{W}$

$\boldsymbol{X} \leftarrow$ instances in $\boldsymbol{B}$
$G \leftarrow \max(R, C)$-nearest neighbor graph of $\boldsymbol{X}$
**for** $n = 1 \rightarrow N$ **do**
    $R_n \leftarrow R$-nearest references of instance $\boldsymbol{x}_n$
    $C_n \leftarrow C$-nearest citers of instance $\boldsymbol{x}_n$
    $N_n^- \leftarrow$-number of instances from negative bags in $R_n + C_n$
    $N_n^+ \leftarrow$-number of instances from positive bags in $R_n + C_n$
    **if** $\frac{N_n^-}{N_n^+} \geq \tau$ or $\boldsymbol{x}_n$ is from a negative bag **then**
        instance label $l_n \leftarrow -1$
    **else**
        $l_n \leftarrow +1$
    **end if**
**end for**
$\boldsymbol{A}^b \leftarrow$ between-class affinity matrix
$\boldsymbol{A}^w \leftarrow$ within-class affinity matrix
Between-class scatter matrix $\boldsymbol{S}_b = \frac{1}{2}\sum_{m,n=1}^{N} \boldsymbol{A}_{mn}^b (\boldsymbol{x}_m - \boldsymbol{x}_n)(\boldsymbol{x}_m - \boldsymbol{x}_n)^T$
Within-class scatter matrix $\boldsymbol{S}_w = \frac{1}{2}\sum_{m,n=1}^{N} \boldsymbol{A}_{mn}^w (\boldsymbol{x}_m - \boldsymbol{x}_n)(\boldsymbol{x}_m - \boldsymbol{x}_n)^T$
$\boldsymbol{W} \in \mathbb{R}^{D \times d} \leftarrow$ top $d$ eigenvectors of $\frac{\boldsymbol{w}^T \boldsymbol{S}_b \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w}}$

feature independently according to a scoring function, thus revealing properties of the individual features by which they may be compared and ranked. The challenge of learning a good feature ranking under an instance-space metric is that the instance labels are not available under the MIL framework. Multiple-Instance Feature Ranking (MI-FEAR) [31] assumes one-sided noise by giving every instance the label of its corresponding bag. This essentially transforms the weakly-supervised problem into a supervised one. Pseudo-code for MI-FEAR is provided in Algorithm 2-7.

Each feature $\theta$, is given a score based on the performance of a learned hypothesis $h_\theta$ operating on a single feature. The performance is determined an evaluation metric $\mathcal{L}$. Once a score has been assigned to every feature, the features are ranked according to relative importance and the top $d$ features are retained as the feature set. As stated in Section 2.2.4, the characteristic accuracy of MI-FEAR is based on noisy labels, where

**Algorithm 2-7** MI-FEAR

**Input:** Multiple-instance dataset $\boldsymbol{B} = \{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, $\boldsymbol{L} = \{L_1, \ldots, L_K\}$, $L_k \in \{-1, +1\}$, evaluation metric $\mathcal{L}$, learning algorithm $\mathcal{A}$, dimensionality of reduced-dimensional space $d$

**Output:** $[\theta_i$ for $\theta_i$ in $V[1, \ldots, d]]$

1: $\boldsymbol{X}, \boldsymbol{Y}^{\gamma} \leftarrow$ supervised dataset of bag-labeled instances using $(\boldsymbol{B}, \boldsymbol{L})$
2: $V \leftarrow$ [ ]
3: **for** feature $\theta_i$ in feature set $\Theta$ **do**
4: $\quad \boldsymbol{X}^{\theta} \leftarrow \boldsymbol{X}$ with all features, excluding $\theta_i$
5: $\quad h_{\theta} \leftarrow$ output of $\mathcal{A}$ when trained on input $(\boldsymbol{X}^{\theta}, \boldsymbol{Y}^{\gamma})$
6: $\quad V[i] \leftarrow$ performance of $h_{\theta}$ on input $(\boldsymbol{X}^{\theta}, \boldsymbol{Y}^{\gamma})$ according to $\mathcal{L}, \theta_i$
7: **end for**
8: $V \leftarrow$ SORTDECREASING$(V)$

each instance is given its corresponding bag-level label. This approach has proven non-ideal in the literature for determining accurate instance-level classification. However, a benefit of feature selection is that, unlike feature extraction, the reduced-dimensional space retains its interpret-ability. In other words, the representations of individual features do not change, meaning they keep their real-world relevance, if applicable. On the other hand, feature selection requires that a subset of useful features for classification already exist in the training set. Empirical results showed that consistently achieved lower instance-level classification accuracy than CLFDA and MidLABS, but had a significantly lower run-time.

### 2.2.6 Comparison Table of MI Dimensionality Reduction Methods

Table 2-1 shows a comparison between the multiple instance dimensionality reduction methods reviewed in Section 2.2.

### 2.2.7 General Weak Supervision

Alternative approaches to weakly supervised dimensionality reduction that do not follow the MIL framework have also been proposed. For example, Wu studied weakly supervised manifold learning for manifold factorization using image-level labels, where the labels were the variation of interest. The primary idea was to use weak labels to find image pairs that should be more similar after removing unwanted image variation. Wu

[73] used an alignment method constrained by Hessian regularization to learn a manifold regression, such that new test images would be projected smoothly into a space near neighboring input images. Gaur and Manjunath [85] used weakly supervised manifold learning to perform dense semantic object correspondence. The objective of the semantic object correspondence problem is to compute dense association maps for a pair of images such that the same object parts get matched, even for very differently appearing object instances. The goal of Gaur and Manjunath's work was to learn a manifold such that features belonging to the same semantic object parts were projected closer to each other on the manifold. This was achieved by re-purposing deep convolutional features from a classification network, where the labels were weak segmentations of object parts. These features were then projected onto a manifold using LDA. Hierarchical Agglomerative Clustering (HAC) was performed in the embedding space and the labels were refined with respect to geodesic distance on the manifold. This method was inspired by the manifold assumption, which states that similar objects (even though disparate in the input feature space), should share an intrinsic manifold. In this way, feature embeddings are learned by an optimization process which is rewarded for projecting features closer on the manifold if they have low feature-space dissimilarity. Additionally, the optimization penalizes feature clusters whose geometric structure is inconsistent with the observed geometric structures of object parts.

An alternative approach for discriminative dimensionality reduction with weak labels is through the application of metric embedding, which is discussed in detail in Section A.2. While they can be fully supervised, metric embedding techniques often consider groups of samples (usually two or three), jointly, to learn an embedding function. Under this type of weak supervision, only a notion of semantic similarity is needed, as compared to direct class labels.

## 2.3 Visual Attention for Weakly Supervised Semantic Segmentation

Weakly Supervised Semantic Segmentation (WSSS) aims at learning pixel-level classification labels from uncertain, imprecise, or ambiguous labels such as spots [86, 87, 88, 89], scribbles [90, 91, 92, 93, 94, 95], bounding boxes [96, 97], or image-level annotations [97]. WSSS using image-level annotations fits the paradigm of multiple instance learning perfectly, where a common task is to learn instance-level (pixel) classification labels form bag-level (image) groundtruth information. While many approaches have been proposed for weakly supervised semantic segmentation with image-level labels, a common theme is to utilize localization information derived from gradient flow through a convolutional neural network. This approach, known as salience or attention.

### 2.3.1 Attention

Attention is the primary scheme used to perform WSSS in the literature [98]. Essentially, attention is an approach for visualizing the regions in an image which are important for determining a particular class score. Generally, there are two primary approaches for computing attention - a trainable attention mechanism or post-hoc network analysis. This dissertation focuses on the later. In the following sections, Class Saliency Maps, Guided Backgropagation, Class Activation Maps, and approaches which utilize post-hoc attention for WSSS are reveiwed.

### 2.3.2 Class Saliency Maps

Class Saliency Maps (CSM) [99] is an approach for visualizing the spatial support of a particular class within a classification CNN. A classification CNN can be considered as a function that maps the input image to a class score. CSM assigns the class importance of a pixel as the magnitude of the backpropagated gradient for a particular class score. Essentially, the gradients of a class score with respect to the input image are computed. The single class saliency map over the input image is computed as the

max absolute value in each pixel location over the input channels. A larger gradient at a pixel location denotes a higher possibility for the presence of the class.

### 2.3.3 Guided Backpropagation

Guided Backpropagation was proposed by Springenberg et al. [100], and combines the idea of CSM with the deconvolutional network (deconvnet) [101]. Essentially, guided backpropagation adds an additional guidance signal from higher layers to normal backpropagation. This addition prevents backward flow of negative gradients corresponding to the neurons which decrease the activation of the higher layer unit to be visualized (i.e. negative gradients are not backpropagated). This has the benefit of allowing for the visualization of gradients with respect to the image where negative gradients are suppressed when backpropagated through ReLU layers. Guided backpropagation has been shown to produce qualitatively better object localization than using plain gradient information (CSM).

### 2.3.4 Class Activation Maps

A Class Activation Map (CAM) [102] is a post-hoc method for visualizing attention in a convolutional artificial neural network (CNN). Essentially, a CAM for a particular class label indicates the discriminative image regions used by the CNN to identify the category. As shown if Figure 2-3, the neural network structure for a CAM commonly consists of stacked convolutional layers, a global pooling layer, a fully connected layer (fc), and the output layer. Formally, let $f$ denote the image classifier parameterized by $\theta$. For a given image $I \in \mathbb{R}^{u \times v \times w}$, the predicted score $y^c$ of the target category $c$ before input to the softmax is given by

$$y^c = f^c(I, \theta). \tag{2-24}$$

Let $\boldsymbol{A}^k \in \mathbb{R}^{u \times v}$ be the $k$-th feature map in the final convolutional layer. The input to the softmax is the sum of the activations scaled by their relative importances, $\alpha_k^c$, toward

each class label. The localization map for CAM, $\boldsymbol{L}^c_{\text{CAM}}$, for class $c$ is obtained by applying a ReLU operation on the summation to remove negative responses

$$\boldsymbol{L}^c_{\text{CAM}} = \text{ReLU}\left(\sum_k \alpha^c_k \boldsymbol{A}^k\right) \tag{2-25}$$

Grad-CAM [103] and Grad-CAM++ [104] generalize CAM by allowing for computation in any CNN architecture. In these approaches, activation maps are weighted by their average gradients. The gradient of prediction score $y^c$ with respect to spatial location $(i, j)$ in feature map $\boldsymbol{A}^k$ is given as

$$g^{kc}_{ij} = \frac{\partial y^c}{\partial \boldsymbol{A}^k_{ij}}. \tag{2-26}$$

Grad-CAM obtains the channel-wise importance weighting by averaging the gradients over all locations in the feature map as

$$\alpha^c_k = \frac{1}{N}\sum_i \sum_j g^{kc}_{ij} \tag{2-27}$$

where $N$ is the number of spatial locations in feature map $\boldsymbol{A}_k$. For Grad-CAM++, the channel-wise importance weights are found as

$$\alpha^c_k = \sum_i \sum_j \beta^{kc}_{ij} \cdot \text{ReLU}(g^{kc}_{ij}), \tag{2-28}$$

where $\beta^{kc}_{ij}$ is computed as

$$\beta^{kc}_{ij} = \frac{(g^{kc}_{ij})^2}{2(g^{kc}_{ij})^2 + \sum_a \sum_b A^k_{ab}(g^{kc}_{ij})^3} \tag{2-29}$$

with $(a, b)$ denoting the spatial location in $\boldsymbol{A}^k$. The primary differences between Grad-CAM and Grad-CAM++ is that the latter uses second order gradient information to determine the feature importance weights. As a result, Grad-CAM++ has been shown to produce better object localization ability when multiple instances occur in the same image. Both Grad-CAM and Grad-CAM++ assign a single weight for each spatial location in an activation feature map. However, in shallow layers, the variances of activation maps

are often large. Thus, a global weight cannot adequately represent the importance of different spatial locations toward particular categories. To address this problem, Jiang et al. [105] proposed LayerCAM. It was empirically verified that a positive gradient corresponding to a location in a feature map indicates that increasing the intensity of this location would have a positive influence on the prediction score of the target class. Thus, in LayerCAM, importance weights are assigned to each spatial location as the positive class-specific gradient at the location. Negative gradients are set to zero. Formally, the importance weight for spatial location $(i, j)$ in the $k$-th feature map can be written as

$$\alpha_{ij}^{kc} = \text{ReLU}(g_{ij}^{kc}). \tag{2-30}$$

To obtain the class activation map for a particular layer, LayerCAM multiplies the activation value of each location by its importance weight

$$\hat{A}_{ij}^{kc} = \alpha_{ij}^{kc} \cdot A_{ij}^{kc}. \tag{2-31}$$

As with alternative approaches, the results $\hat{A}^k$, are linearly combined along the channel dimension to obtain the final CAM

$$\boldsymbol{L}_{\text{LayerCAM}}^{c} = \text{ReLU}\left(\sum_k \hat{\boldsymbol{A}}^k\right). \tag{2-32}$$

LayerCAM has been shown to generate reliable class activation maps in shallow layers which capture fine-grained localization information.

ScoreCAM [106] rids itself of dependence on gradients by finding importance weights of activation maps by a forward passing score in the network. Essentially, each activation map is used as a mask on the input image, and the importance weights for the $c$ classes are given as the output scores for the masked image.

AblationCAM [107] also attempts to remedy the problem of diminishing gradients by avoiding them entirely. In their work, Desai and Ramaswamy [107] showed that removing certain feature map units had a severe impact on the accuracy of certain classes. As a

result, AblationCAM considers the performance drop to be an indicator of feature importance. The importance weight for activation feature map, $\boldsymbol{A}^k$, is computed as the performance drop between the output score $y^c$, and the output score with $k$-th activation map removed, denoted as $y^c_k$. The importance weights are formally given as

$$\alpha^c_k = \frac{y^c - y^c_k}{y^c}. \tag{2-33}$$

Similarly to ScoreCAM, AblationCAM uses forward class activation information to visualize attention in convolutional neural networks. These approaches have the advantage of using the inherent flow through a neural network, instead of relying on gradients which have lost spatial information through pooling.

Eigen-CAM [108] is a class non-discriminative approach for identifying salience information in the input space. Similarly to ScoreCAM and AblationCAM, Eigen-CAM does not rely on gradients, but obtains a saliency map by projecting the input image onto the first eigenvector of the convolutional feature map weights at a particular layer. This approach has the benefit of providing salience information irrespective of model accuracy. Additionally, it has been shown that Eigen-CAM is more robust to adversarial noise than alternative CAM approaches in the literature.

More recently, a variety of approaches have been developed in attempt to obtain better visual explanations for WSSS [109, 110, 111, 112, 113, 114]. An interpretable basis decomposition approach was explored by Zhou et al. [114] which decomposes the penultimate representation for a sample into a linear combination of interpretable basis elements. CAMERAS [109] drastically improves saliency map realization by systematically performing multi-scale accumulation and fusion of activation maps and backpropagated gradients. LFI-CAM [112] is a hybrid combination of and attention branch network and Score-CAM [106]. Essentially, the importance weights on activation maps are learned through a novel “feature importance network” during training. Zhang et al. [113] proposed A-FMI, which assigns activation maps importances using a

"difference-from-reference" approach. In A-FMI, activation maps for an input image are subtracted from feature maps for a reference (e.g. a black image) to provide an importance value.

Jung et al. [110] explored the use of SHAP values as important weights on activation maps. This method takes advantage of the intrinsic linearity of CAMs with respect to its activation maps. Essentially, an explanation model of CNN is constructed as a linear function of binary variables that denote the existence of the corresponding activation maps. The explanation model can be determined by additive feature attribution methods (i.e. SHAP, fuzzy integeral, etc.). Since exact SHAP values are unattainable, Jung et al. [110] introduced an approximation method called LIFT-CAM.

SHAP [115] is a model agnostic feature attribution method which determines the importance of a feature by the marginal differences in prediction. Jung et al. [110] derived the SHAP values of activation maps with respect to the $c^{th}$ class as:

$$\alpha^c_{k-SHAP} = \sum_{a' \subset A'} \frac{(K - |a'|)!(|a' - 1|)!}{K!} [y^c(h_A(a')) - y^c(h_A(a' \setminus k))] \tag{2-34}$$

where $K$ is the total number of activation maps, $A = h_A(a')$ is a mapping to $A$, $A'$ is a vector of ones, and $a'$ is an indicator variable denoting the existence of an activation map such that $a'_k = 1$ is mapped to $A_k$ and $a'_k = 0$ is mapped to $\mathbf{0}$, having the same dimensionality as $A_k$. The SHAP value for $A_k$ is $\alpha^c_k$, and $y^c$ denotes the output class-specific output. Equation 2-34 implies that $\alpha^c_k$ can be obtained by averaging marginal prediction differences between the inclusion and exclusion of $A_k$ across $\Pi_{\text{total}}$ which denotes the set of all possible orderings of $\{1, \ldots, K\}$.

Since calculating the exact SHAP values for a set of activation maps is typically intractable, Jung et al. [110] consider an approximation derived from DeepLIFT [116].

Specifically, LIFT-CAM defines the contribution score of a specific activation map as an aggregation of all pixels in that activation map:

$$\alpha^c_{k-LIFT} = C_{\Delta A_k} \Delta y^c = \sum_{(u,v)\in\Lambda} C_{\Delta A_k(u,v)} \Delta y^c \tag{2-35}$$

where $\Lambda = \{1, \ldots, U\} \times \{1, \ldots, V\}$ is a discrete activation dimension and $A_k(u, v)$ is the activation value at location $(u, v)$. The term $\Delta$ denotes the difference-from-reference. The reference values (values corresponding to the excluded feature) of all activation maps are set to zero. Consequently, LIFT-CAM can estimate $\alpha_{SHAP}$ in a single backward pass.

Within this framework, the importance weights of AblationCAM [107] can be re-written as

$$\alpha^c_{k-Ablation} = \frac{y^c(h_A(A')) - y^c(h_A(A' \setminus k))}{y^c(h_A(A'))}. \tag{2-36}$$

Since AblationCAM uses this specific marginal difference as the coefficient, it can also be viewed as an alternative approximation method for $\alpha_{SHAP}$. However, AblationCAM requires $K$ forward simulations, and is thus computationally expensive.

### 2.3.5 CAM-Informed Weakly-Supervised Semantic Segmentation

A plethora of approaches in the literature use class activation maps as stimuli for weakly supervised object localization and semantic segmentation. A new approach developed by Yu et al. [117], described in detail in the following section, learns to compute pixel-level segmentation from image level labels. First, the model learns the parameters to compute CAMs using a novel sampling approach. Pixel-level segmentation labels are estimated from the learned CAMs and a semantic segmentation network is trained. Wei et al. [118] investigated dilated convolution as a method for generating dense attention maps. The activation maps are converted into pseudo-labels which are used to train a semantic segmentation network. Laradji et al. [119] used the peak local maxima of class activation maps to train a classification network. Simultaneously, a segmentation network was trained with pseudo-labels informed by the

peak maxima. Ahn and Kwak [120] proposed AffinityNet to propagate local discriminative label information from CAMs to cover the entire target object. The work by Kolesnikov and Lampert [121] performs semantic segmentation using three principles: 1) seed using weak localization (CAM), 2) expand objects based on the information about which classes can occur in an image, and 3) constrain the segmentations to coincide with object boundaries. Hong et al. [122] proposed a novel encoder-decoder architecture attention which uses applies a separate classifier to estimate the labels present in an image, then computes class saliency maps using the inferred labels. The saliency maps are used to obtain image foreground segmentation masks from the decoder. A semantic mining graph neural network [123] was used to compute CAMs and propagate segmentation information. The work of Fend et al. [124] uses CAMs to seed segmentation masks and inform graph cuts to refine the inferred pixel-level labels. Alternative approaches for CAM-informed semantic segmentation have been proposed using co-attention [125], online attention accumulation [126], probabilistic latent label discovery [127], pseudo-label masking [128], and region refinement [129, 130, 131, 132, 133, 134].

#### 2.3.5.1 MIL-CAM

Yu et al. [117] proposed a multiple instance learning class activation map (MIL-CAM) approach for learning pixel-level segmentation for minirhizotron imagery. MIL-CAM learns to perform semantic segmentation in two stages. In the first stage, the parameters needed to compute attention maps for the background/target classes are computed. The attention maps are estimated using the softmax outputs of a weighted linear combination of features maps, extracted from the various layers of a trained CNN as

$$\boldsymbol{L}^{c}_{\text{MIL-CAM}} = \frac{\exp\left(\sum_k \alpha^c_k A_k + b^c\right)}{\sum_q \exp\left(\sum_k \alpha^q_k A_k + b^q\right)} \tag{2-37}$$

where each activation map, $A_k$, has been interpolated to the original image input size, and $b^c$ is an estimated bias term. In the second step, a segmentation network is trained. After training, the segmentation network maps input test imagery to pixel-level segmentation maps. The details of each stage are as follows.

**Attention map estimation** MIL-CAM estimates the parameters needed to obtain attention maps using a combination of three key components: 1) a pixel-level feature extraction component; 2) a pixel sampling component used to form bags for MIL analysis; and 3) a linear model that performs MIL-based segmentation. The sampled pixels with features extracted from the image classification network are used to train the linear model. In step one, a binary classification CNN is trained with cross-entropy loss to predict bag-level labels, where a negative bag is an image which has only background pixels, while a positive bag is an image with at least one pixel on target. Once the classification network is trained, the CNN feature maps are up-sampled to match the size of the input image. Each pixel is represented by the corresponding feature vector obtained from the collection of up-sampled feature maps. Next, a sampling approach is used to identify representative pixels from each image. Finally, the weights and biases needed to compute attention maps are computed from the sampled instances by Equation 2-38. Under the MIL constraints, at least one instance in each positive bag must be labeled as positive, and all negative instances must be labeled as negative,

$$\min_{\boldsymbol{l}_{kn}} \min_{\boldsymbol{\alpha},\boldsymbol{b}} = \frac{-1}{\sum_k N_k} \sum_{\boldsymbol{x}_{kn}} \sum_q l_{knq} \log g_q(\boldsymbol{x}_{kn}; \boldsymbol{\alpha}, \boldsymbol{b}) \tag{2-38}$$

where $N_k$ is the number of instances in bag $k$, $\boldsymbol{l}_{kn}$ is the one-hot encoded label of instance $\boldsymbol{x}^{kn}$, and $g_q(\boldsymbol{x}_{kn}; \boldsymbol{\alpha}, \boldsymbol{b})$ is the $q^{th}$ element of the softmax output of the MIL-CAM network with parameters $(\boldsymbol{\alpha}, \boldsymbol{b})$.

**Image segmentation** After MIL-CAM attention maps can be determined, an image segmentation network is trained. First, target class attention maps are estimated for positively-labeled images. The attention maps are thresholded to obtain pixel-level

segmentation groundtruth. All pixels in negative images are labeled as non-target/background. The pseudo-labels are used to train a U-Net segmentation network [135].

In this way, MIL-CAM is able to estimate pixel-level segmentation masks given image-level, multiple instance learning labels. Yu et al. [117] showed that MIL-CAM is able to obtain better activation maps and semantic segmentation performance than alternative approaches on minirhizotron root segmentation datasets.

## 2.4 Choquet Integral

The Choquet Integral (ChI) is a well-used aggregation operator for pattern recognition and information fusion [136, 137, 138, 139, 140]. The Choquet integral is a type of fuzzy integral (FI), which rely on the use of fuzzy measures (FM) [141, 142]. In practice, the FI can act as a wide class of operators; the max, min, linear order statistics, etc [143, 5]. Depending on the values of each fuzzy measure, the CI can represent a variety of relationships and combinations among the information sources. Thus, a primary aspect when using the CI for information fusion is learning the fuzzy measures [144, 145, 140]. This section provides definitions for fuzzy measures and fuzzy integrals, focusing on the Choquet integral. Methods for learning the fuzzy measures within the Choquet integral are also discussed.

### 2.4.1 Fuzzy Measure

The FI aggregates data from sources - people, sensors, algorithms, and combinations thereof. As described by Du [5], let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M\}$ be a finite set of $M$ sources for fusion. The set $\boldsymbol{X}$ contains $2^M - 1$ non-empty subsets. The power set of all (crisp) subsets of $\boldsymbol{X}$ is defined as $2^X$.

**Definition 2.4.1.** A monotonic and discrete fuzzy measure, $\mu$, on $\boldsymbol{X}$ is a real-valued function that maps $\mu : 2^{\boldsymbol{X}} \to \mathbb{R}^+$. It satisfies the following properties [5, 140, 146, 147]:

1. (boundary condition): $\mu(\emptyset) = 0$ and $\mu(X) > 0$

2. (monotonicity property): If $A, B \subseteq \boldsymbol{X}$ and $A \subseteq B$, then $\mu(A) \leq \mu(B)$

An additional constraint is often imposed on the FM to limit the upper bound to $1$, i.e. $\mu(\boldsymbol{X}) = 1$. For a subset $A \subseteq \boldsymbol{X}$, $\mu(A)$ is often considered as the worth of this subset of information.

A special case of fuzzy measure is the Sugeno $\lambda$-measure [140, 148]. To maintain conventional notion, this class of measure is denoted using $g$ instead of $\mu$. It is defined as follows.

**Definition 2.4.2.** Let $\lambda \in (-1, \infty)$. A Sugeno $\lambda$-measure, $g$, is a real-valued function that maps $g : 2^{\boldsymbol{X}} \to [0, 1]$ with the following properties [140, 148]:

1. $g(\boldsymbol{X}) = 1$ (normalized)
2. If $A, B \subseteq \boldsymbol{X}$ and $A \bigcap B = \emptyset$, then $g\left(A \bigcup B\right) = g(A) + g(B) + \lambda g(A) g(B)$

In general, FM model the relationship or "interactions" (e.g. statistical correlations) among the sources. Each element value represents the "power", "importance", "worth", etc. of a particular combination of the sources. In this dissertation, notation follows as defined by Du [5]. Elements within a FM are given a subscript matching its corresponding subset. For example, element $g_1$ corresponds to the subset $\boldsymbol{x}_1$, element $g_{12}$ corresponds to the subset $\boldsymbol{x}_1, \boldsymbol{x}_2$, and so on. Note that $g$ has a total of $2^M - 1$ elements and $g_{123\ldots M}$ is always equal to 1 (by Definition 2.4.1). All other measure elements hold real values in $[0, 1]$ and satisfy monotonicity (Definition 2.4.1). While non-monotonic FMs have been investigated [149, 150, 151], this dissertation focuses solely on monotonic and normalized fuzzy measures. Thus, the term "fuzzy measures" refers only to monotonic and normalized fuzzy measures hereon.

Figure 2-5, as demonstrated by Du [5], illustrates the monotonicty property among FM elements given four sources. As an example, FM element $g_{23}$ has subsets $g_2$ and $g_3$ and supersets $g_{123}$ and $g_{234}$. Thus, element $g_{23}$ satisfies $g_{23} \geq g_2$, $g_{23} \geq g_3$, $g_{23} \leq g_{123}$, and $g_{23} \leq g_{234}$. The blue arrow shows one path to "climb the trellis" of the FM elements. The

FM element at the top $g_{1234}$ corresponding to the full set is equal to $1$. All other elements take values in $[0, 1]$.

Fuzzy measures have been used extensively among a variety of applications, including image segmentation and enhancement [152, 153, 154, 155], medical applications [156, 157, 158], linguistics [159], multi-criteria decision making [138, 160], and multi-resolution sensor fusion [161, 148, 162].

### 2.4.2 Choquet Integral

Fuzzy measures are used to define fuzzy integrals, such as the Sugeno, non-direct, shape preserving, general, and thte Shilkret fuzzy integrals [143]. In this dissertation, focus is placed on the Choquet fuzzy integral (ChI) [153, 149]. The ChI has many desirable properties, e.g. it recovers the Lebesgue integral for an addative/probability measure [141, 163], and has thus been long used an an effective aggration operator [140].

**Definition 2.4.3.** The Choquet integral, denoted as $(C)\int$, of a measurable function $h$ with respect to fuzzy measure $\mu$ is defined as: $(C)\int h\,d\mu = \int_0^\infty \mu(h > \alpha)\,d\alpha$.

When the mapping function $h \in [0, 1]$, the $\infty$ is replaced by $1$ (integrating from $0$ to $1$). To fuse evidence supplied by different sources from a discrete fuzzy set of $\boldsymbol{X}$, the discrete Choquet integral can be used.

**Definition 2.4.4.** Let $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M\}$ be $M$ information sources, (e.g. classifier outputs, humans, sensors, etc.), and $\boldsymbol{h} = (h_1, h_2, \ldots, h_M)^T$ be their inputs to be aggregated. (Note: $h_m$ is often thought as the "support" to the "question" being ask.) The discrete Choquet integral of integrand $\boldsymbol{h}$ on finite $\boldsymbol{X}$ for FM $\mu$ is [143, 144, 145]

$$\int_C \boldsymbol{h} \circ \mu = C_\mu(\boldsymbol{h}) = \sum_{m=1}^{M} h(\boldsymbol{x}_{\pi_m})[\mu(\boldsymbol{A}_m) - \mu(\boldsymbol{A}_{m-1})] \tag{2-39}$$

where $\pi$ is a permutation of $\boldsymbol{X}$ such that $h(\boldsymbol{x}_{\pi(1)}) \geq h(\boldsymbol{x}_{\pi(2)}) \geq \cdots \geq h(\boldsymbol{x}_{\pi(M)})$, $\boldsymbol{A}_m = \{\boldsymbol{x}_{\pi(1)}, \ldots, \boldsymbol{x}_{\pi(m)}\}$, and $\mu(\boldsymbol{A}_0) = 0$.

In total, there are $M!$ unique sort orders and $2^M$ FM variables. Each sort is associated with a unique walk up the lattice. Consider, as an example, if $h_2 > h_1 > h_3$, then the Choquet Integral is computed as $h_2(\mu(\{\boldsymbol{x}_2\}) - 0) + h_1(\mu(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}) - \mu(\{\boldsymbol{x}_2\})) + h_3(\mu(\{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3\}) - \mu(\{\boldsymbol{x}_1, \boldsymbol{x}_2\}))$. It can be noted that $\sum_{m=1}^{3}[\mu(\boldsymbol{A}_m) - \mu(\boldsymbol{A}_{m-1})] = 1$ as $\mu(\emptyset) = 0$ and $\mu(\boldsymbol{X}) = 1$. Thus, each walk is a linear convex sum.

Depending on the selection of $\mu$, the ChI turns into a specific aggregation operator. For example, when $\mu(A) = 1, \forall A \in 2^{\boldsymbol{X}} \setminus \emptyset$, the maximum operator is obtained. When $\mu(A) = \frac{|A|}{N} \forall A$ the arithmetic mean operator is obtained. Moreover, when $\mu(A) = \mu(B) \forall A, B \in 2^{\boldsymbol{X}}$ s.t. $|A| = |B|$, a familiar class of operators is obtained, e.g. min, max, soft min, soft max, mean, median, trimmed statistics, etc [164].

The Choquet Integral has been used in many information fusion applications – providing effective performance in pattern recognition and classification tasks [140, 138, 165]. Gader et al. [166, 167, 137] applied the ChI to multi-algorithm, multi-sensor explosive ordinance detection. Wang et al. [168] used the ChI to combine models for Landsat image classification. Additionally, the Choquet fuzzy integral has been used for handwriting recognition [169, 170], text classification [171], temperature prediction [155], medical applications [172, 173], gesture recognition [174], crowdsourcing aggregation [175], morphological image filtering [138, 176, 177], and distance measuring [178]. Recently, Murray et al. [143, 142, 164] explored use of the ChI to explain neural model decisions.

### 2.4.3 Learning the Fuzzy Measure

In a classifier or regressor problem, the training data $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ and corresponding labels $L = \{l_1, l_2, \ldots, l_N\}$ are known. Suppose the sources to be fused are the outputs from a set of $M$ classifiers, regressors, sensors, algorithms, etc. $C = \{c_1, c_2, \ldots, c_M\}$. The output for the $m^{th}$ source, $c_m$, on the $n^{th}$ data point/instance $\boldsymbol{x}_n$ is $h(c_m; \boldsymbol{x}_n)$. In a classification/regression problem, these source values are also known.

Thus, a critical aspect of using the ChI to perform fusion is to learn all the FM elements from the training data [5]. One method for determining the FM values is to have an expert define them. However, this is often impractical. Another possibility is to determine the values of the singletons (densities) and to use a formula to calculate the remaining variables; e.g. Sugeno $\lambda$-FM or the S-Decomposable FM [143]. A variety of approaches for inferring the FM values from the data have been explored in the literature - including least-square based approaches, gradient descent algorithms, and evolutionary algorithms [141, 139, 144, 137, 140, 179, 180, 181, 182]. In the following, a least-square based approach is reviewed in detail.

Similarly to the general discrete ChI described in Section 2.4.1, the discrete Choquet Integral on instance $\boldsymbol{x}_n$ given FM, $\mu$, and sources, $C$, can be defined as a difference between source values

$$C_\mu(\boldsymbol{x}_n) = \sum_{m=1}^{M} \mu(\boldsymbol{A_m})[h(c_m; \boldsymbol{x}_n) - h(c_{m+1}; \boldsymbol{x}_n)] \tag{2-40}$$

where $C$ is sorted such that $h(c_1; \boldsymbol{x}_n) \geq h(c_2; \boldsymbol{x}_n) \geq \cdots \geq h(c_M; \boldsymbol{x}_n)$. Since there are only $M$ sources, $h(c_{M+1}; \boldsymbol{x}_n) = 0$. The FM element value corresponding to the subset $\boldsymbol{A}_m = \{c_1, c_2, \ldots, c_m\}$ is $\mu(\boldsymbol{A}_m)$.

**Least squares** For $M$ input sources, there are $M \times M!$ FM elements tied to the underlying $2^M$ FM variables. A quadratic programming (QP) approach to solve for the FM values was explored by Anderson et al. [144]. Assuming the desired label for the $n^{th}$ data point/instance $\boldsymbol{x}_n$ is $l_n$, the objective of the least-squares criteria is to learn a fuzzy measure $\mu$ such that the squared errors between the ChI outputs of all training samples and their desired labels is minimized:

$$\min_{\mu} E^2 = \sum_{n=1}^{N} \left(C_\mu(\boldsymbol{x}_n) - l_n\right)^2 = \sum_{n=1}^{N} \left(\boldsymbol{c}_n^T \boldsymbol{u} - l_n\right)^2 = ||\boldsymbol{D}\boldsymbol{u} - \boldsymbol{l}||_2^2 \tag{2-41}$$

where $\boldsymbol{u} = [\mu_1, \mu_2, \ldots, \mu_M, \mu_{12}, \mu_{13}, \ldots, \mu_{123M-1}, \ldots, \mu_{12\ldots M}]$ (lexiographic vector of size $2^M - 1$), $\boldsymbol{D} = [\boldsymbol{c}_1\, \boldsymbol{c}_2\, \ldots\, \boldsymbol{c}_N]^T$ (full dataset), $\boldsymbol{l} = [l_1\, l_2\, \ldots\, l_N]^T$, and $\boldsymbol{c}_n$ holds the coefficients

of $\boldsymbol{u}$ for the observation $\boldsymbol{x}_n$, e.g., for $N = 3$ and $h(c_2; \boldsymbol{x}_n) \geq h(c_1; \boldsymbol{x}_n) \geq h(c_3; \boldsymbol{x}_n)$, $\boldsymbol{c}_n = [0, h(c_2; \boldsymbol{x}_n) - h(c_1; \boldsymbol{x}_n), 0, h(c_1; \boldsymbol{x}_n) - h(c_3; \boldsymbol{x}_n), 0, 0, h(c_3; \boldsymbol{x}_n)]$.

The regularized SSE optimization problem is

$$\min_{\boldsymbol{u}} f(\boldsymbol{u}) = ||\boldsymbol{D}\boldsymbol{u} - \boldsymbol{l}||_2^2 + \beta v(\boldsymbol{u}), \tag{2-42}$$

where $\beta \in \mathbb{R}^{\geq 0}$ is a regularization parameter and $v(\boldsymbol{u})$ is a model complexity term (e.g. k-additive, Mobius, $\ell-$norm, etc.), subject to the FM boundary and monotonicity constraints. Anderson et al. [145] showed how to pack these conditions into a linear algebra expression which can be solved via QP.

As outlined by Du [5], there are other optimization approaches to learn the FM element values from the data. A maximum split approach, proposed by Marichal and Roubens [183], models the pairs and interactions of the FM elements subject to the monotonicity and normalization constraints, and solves the maximization problem with linear programming. While the maximum split method is simple, it introduces many hyperparameters which must be chosen carefully. Kojadinovic et al. [184, 185] proposed a minimum variance approach based on the principle of maximum entropy. The assumption for this approach is that the optimal solution will maximize the entropy, and thus minimize the variance of the FM elements. A less constrained approach was introduced by Meyer et al. [186]. This method writes the optimization objective as the least squares criteria subject to the FM constraints and solves for the FM element values using a convex quadratic program. However, this methods relaxes some of the constraints on the FM values. Thus, the relaxed approach is considered a generalization of the standard least-squares QP method [187].

**Gradient descent** An alternative class of algorithms for learning the FM element values is derived from the gradient-descent algorithm. Specifically, Mendez-Vasquez et al. [137] used gradient-descent to learn a Sugeno $\lambda$-measure (Section 2.4.2). By deriving the partial derivative of the Choquet Integral and using inherent properties of the

Sugeno $\lambda$-measure, Mendez-Vasquez et al. [137] showed that gradient-descent could be used to solve the FM element values from any differentiable objective, such as the aforementioned least-squares criteria [145, 187]. Keller and Osborn [141] proposed a neural model and a "reward-punishment" training objective to learn fuzzy measures for multi-class decision making. In their approach, a neuron is used to represent each data class. FM element values are initialized and the ChI is calculated and compared to the true class labels. If the ChI output does not match the true class label, the density values are decreased ("punished"). If the output is correct, however the values are increased ("rewareded"). This process continues until an acceptable amount of the data are correctly classified. Neural network approaches have also been used to determine Sugeno FM element values [180, 188].

**Evolutionary algorithms** Fuzzy measure element values have been determined in the literature by Evolutionary Algorithms [179], specifically, Genetic Algorithms [189, 148, 162, 144] and Particle Swarm Optimization [190]. Genetic Algorithms [181, 191, 182, 192, 193] are special instances of Evolutionary Algorithms. Genetic Algorithms represent FM elements as chromosomes and generate populations of potential measures. A pre-defined function measures the "fitness" of a population (essentially the error/accuracy). Measures from the previous iteration are selected based on their fitness, and a new population is created through application of a "mutation". The generation/mutation process continues until a stopping criterion is met (i.e. maximum number of iterations or small mis-classification rate). The measure which provides the best fitness value is considered as the FM solution.

Alternative methods, such as Particle Swarm Optimization [190], have also been used to solve non-linear mulit-regression based on generalized Choquet integrals [190, 5, 148, 162]. It was shown by Kennedy and Eberhart [190] that Particle Swarm Optimization can speed up convergence as compared to Genetic Algorithms.

### 2.4.4 Multiple Instance Choquet Integral

As with standard supervised learning approaches, the Choquet integral requires precise groundtruth/label information to learn the fuzzy measure element values from the data. Exact labels, as mentioned, are often unavailable in practice. As a solution, Du and Zare [162] developed the Multiple Instance Choquet Integral (MICI), which is trained to fuse classifiers or regressors given imprecise label information. Specifically, Du and Zare [5, 148, 162] introduced three variants of the ChI objective, i.e. noisy-or, min-max, and generalized-mean models, for the classifier fusion framework.

In standard multiple instance learning (Section 2.1), a bag is given a negative label if all the instances in the bag are negative and a bag is labeled positive if at least one instance in the bag is a true positive instance. The noisy-or model is often used to express these assumptions [161, 189, 38]:

$$\begin{aligned}\ln p(\boldsymbol{X}|\boldsymbol{\theta}) = &\sum_{j=1}^{B_J^-}\sum_{m=1}^{N_j^-} \ln\left(1-\mathcal{N}(C_{\boldsymbol{g}}(\boldsymbol{x}_{jm})|\psi,\beta)\right) \\ &+\sum_{k=1}^{B_K^+} \ln\left(1-\prod_{n=1}^{N_k^+} 1-\mathcal{N}(C_{\boldsymbol{g}}(\boldsymbol{x}_{kn})|\psi,\beta)\right)\end{aligned} \tag{2-43}$$

where $B_J^-$ and $B_K^+$ are the total number of positive and negative bags, respectively, $N_j^-$ is the total number of instances in negative bag $j$, and $N_k^+$ is the total number of instances in positive bag $k$. The parameters $\psi$ and $\beta$ are the mean and width of a radial basis function $\mathcal{N}(\cdot)$. Given a two-class classifier fusion problem, the positive class (target) is given a label of "$+1$", while the negative class (non-target or background) is marked as "$0$". The parameter $\psi$ can be set, in this case, to $1$, to encourage the Choquet integral values of positive instances to be $1$ and Choquet integral values of negative instances to be far from $1$. Here the model parameter vector $\boldsymbol{\theta}$ consists of the RBF bandwidth, $\beta$, and the fuzzy measure $\boldsymbol{g}$ used to compute the Choquet integral.

Du and Zare [162] later introduced two models to supplement the MICI framework. The min-max model substitutes the noisy-or with simple "min" and "max" operators to

enforce the standard MIL assumption. Through optimization, the min-max model encourages the ChI of all instances in negative bags to zero and the ChI of at least one instance in each positive bag to one. In the min-max case, the objective function relies solely on the training data and labels and does not require any user-set parameters, as needed in the noisy-or model. A generalized mean model was also employed to enforce the MIL assumption instead of hard “min” and “max” operations. The contribution of the generalized-mean model is that it allows more points to contribute to the predicted class label, as compared to the min-max model which relies on individual instances. A term in the exponent of the generalized mean equation allows the term to act as varying aggregation operators. Optimization encourages the ChI of all instances in negative bags to zero and the ChI of at least one instance in each positive bag to one All three MICI variations were optimized using Evolutionary Algorithms and applied to applications in multi-sensor classifier fusion for target detection.

Table 2-1. Summary of multiple instance dimensionality reduction approaches.

| Method | Summary | Reduction method | Classification level |
|---|---|---|---|
| MIDR (2010) | Finds a sparse, orthogonal linear projection matrix optimized for bag-level logistic regression. Section 2.2.1. | Linear, orthogonal projection | bag-level |
| MidLABS (2010) | Finds linear projection vector using LDA defined from bag-similarity kernel. Section 2.2.2. | Linear, LDA | bag-level |
| MIDA (2014) | Learns linear projection vector using LDA defined from bag representative vectors. Section 2.2.3. | Linear, LDA | instance-level |
| CLFDA (2010) | Learns linear projection matrix using local discriminant analysis defined from instance scatter. Instance labels are provided as bag labels and refined. Section 2.2.4. | Linear, LFDA | instance-level |
| MI-FEAR (2015) | Incrementally leaves out feature and evaluates performance loss to provide feature score. Section 2.2.5. | Linear, feature selection | instance-level |

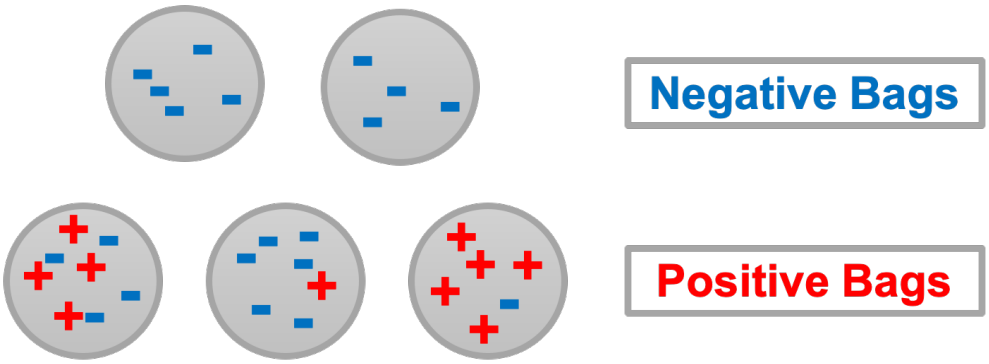



Figure 2-1. Illustration of example bags under the multiple instance learning framework. Red “plus signs” denote positive instances and blue “negative signs” represent negative instances. The two bags on the top row are labeled “negative” because they only contain negative instances. The three bags on the bottom row are “positive” because they each contain at least one positive instance.

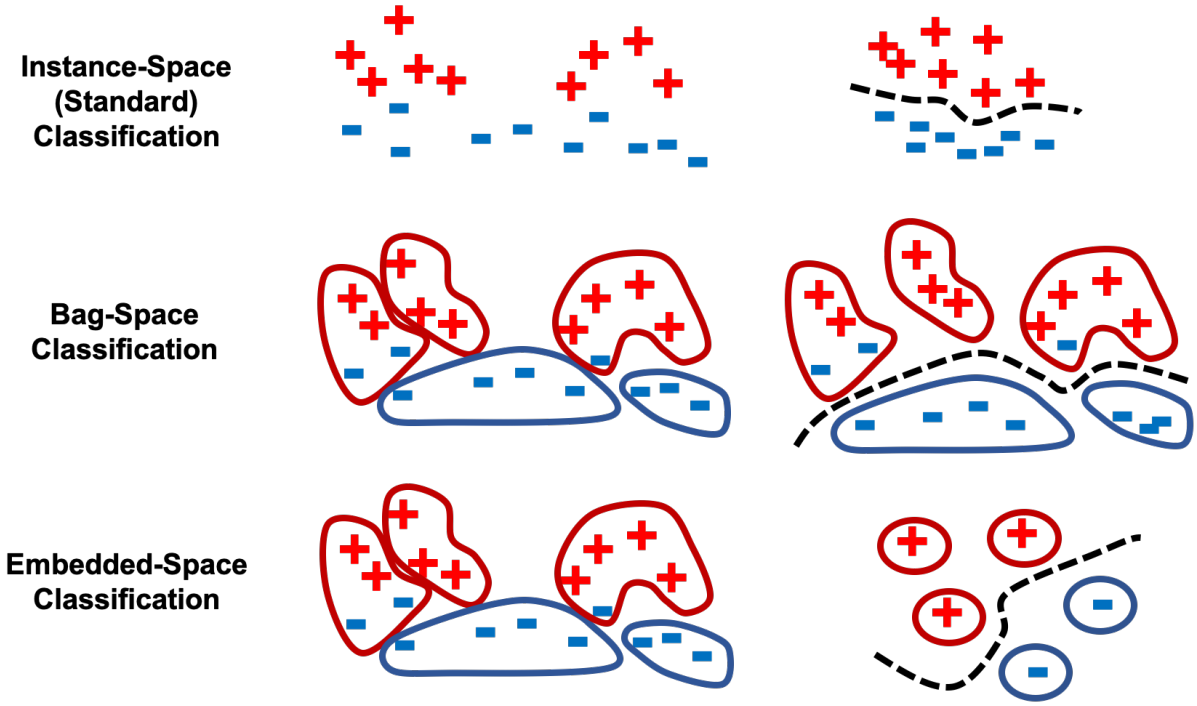



Figure 2-2. Illustration of MIL classification under the instance, bag and embedded-space paradigms. The figure depicts data in both the input feature space and classifier space along with learned hyperplanes. Red denotes positive bags/instances while blue represents negative bags/instances. Single instances in circles denote single-instance representations of bags after being embedded.

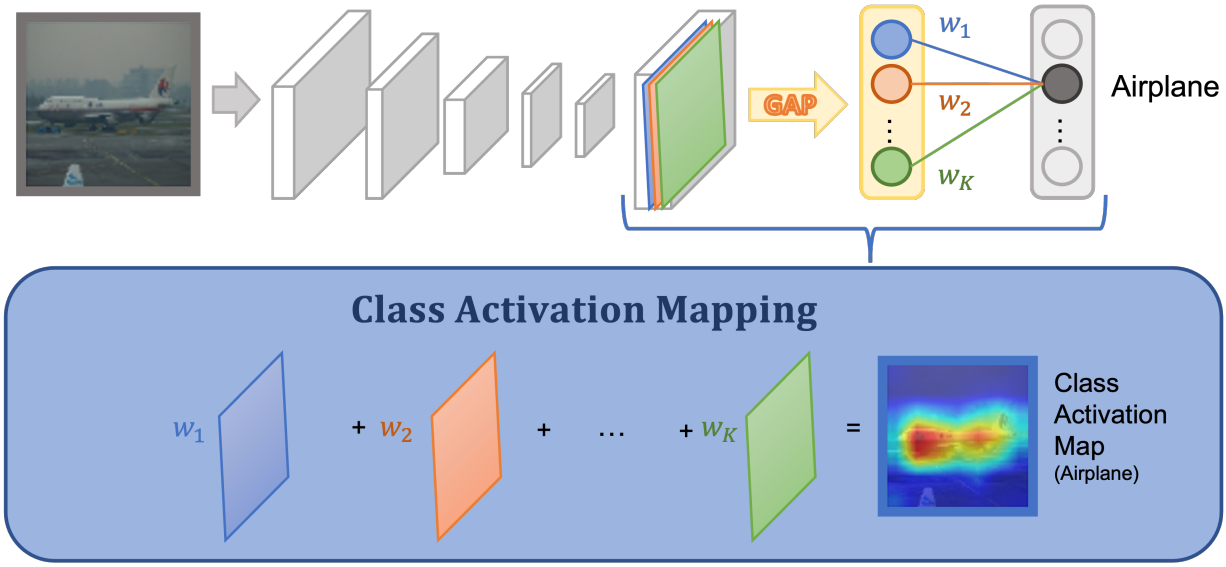



Figure 2-3. Class activation map computation. In the standard CAM, importance values are given as the weights between global average pooled feature maps and their weights toward a particular class in the fully-connected layer.

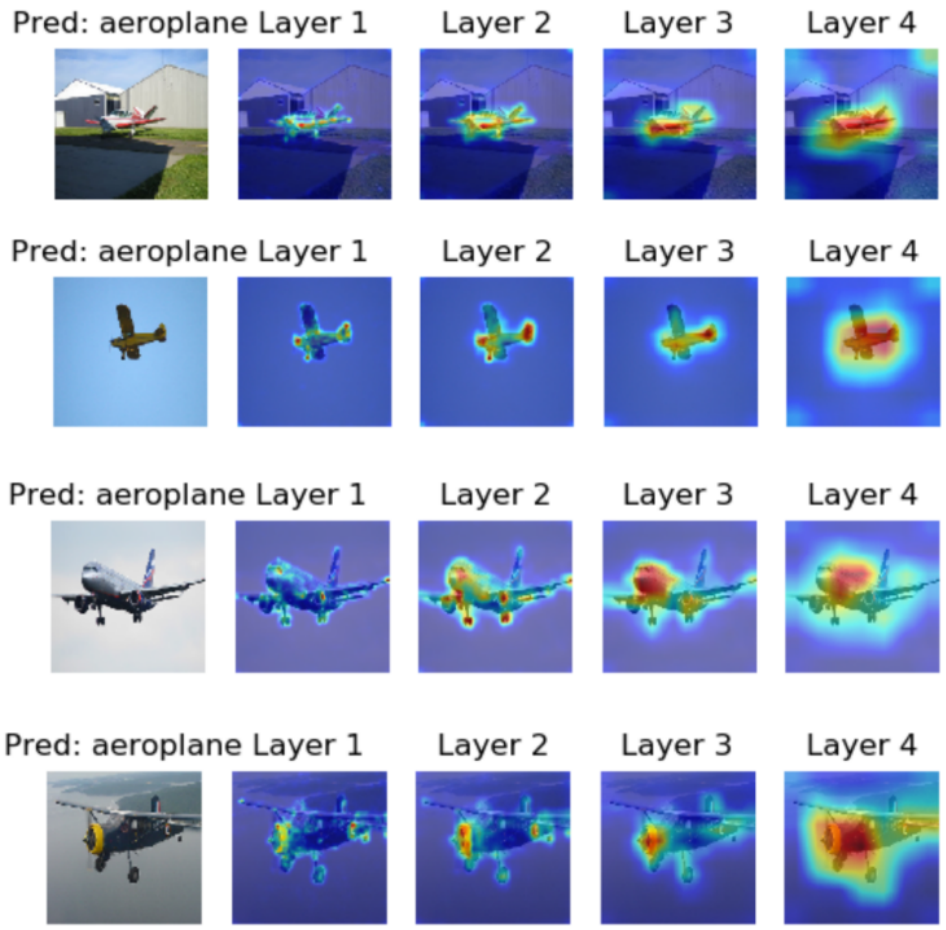



Figure 2-4. Examples of LayerCAM at various stages of a trained VGG16 network for the "aeroplane" class. Layer 1 corresponds to CAMs computed from the initial convolutional block while Layer 4 represents CAMs from the deepest convolutional layers. Earlier layers seemingly capture fine-grained information while later layers capture localization information.

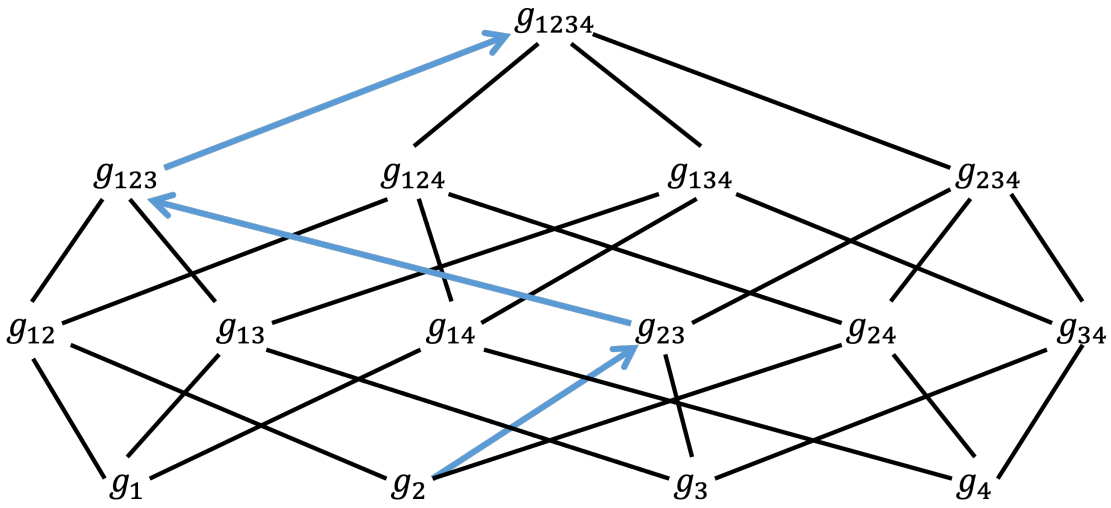



Figure 2-5. Illustration for the subset and superset relationships between fuzzy measure elements given four sources, as shown by Du [5]. The blue arrows describe one path for “climbing the trellis” or “walking” through the Hasse diagram.

# CHAPTER 3
# TECHNICAL APPROACH

This chapter provides a description of work explored in performing instance-level classification from bag-level labels. The chapter is primarily divided into two sections. In the first, the algorithmic pipeline is described and key insights from experimentation are discussed. In the second portion, a detailed description of the proposed method for improving feature selection and discriminative region estimation is given. The proposed method, named Fusion-CAM, uses a discrete Choquet Integral to fuse activation feature maps from a CNN to capture the regions of an input image which are important for the bag-level inference decision. This chapter outlines the conceptual and mathematical background. Methods for source/feature selection for the ChI are proposed. Objectives for learning the measure elements of the ChI within the framework of multiple instance learning and corresponding optimization schemes are given.

## 3.1 Overview of the WSSS Pipeline

Approaches for inferring instance-level detection/segmentation from imprecise groundtruth typically operate within a few stages. An overview of the weakly-supervised instance-level classification pipeline is shown in Figure 3-1. Each stage is briefly described in the following.

**Image-level classification** The first step of the WSSS pipeline using image-level labels typically involves training a model to predict the prominent class(es) contained within the image [194, 195, 196, 197, 198, 199]. There is in inherent assumption that if if a model can determine if an image contains an object, that it also understands where the object is in the image. Subsequent steps attempt to extract the object-level information from the model.

**Pseudo-label generation** Initial pseudo-labels are generated by extracting class-specific information from the trained image-level classification model. A popular approach for pseudo-label generation is to utilized class salience (Section 2.3.4). Many methods produce class activation maps to localize objects of interest

[102, 103, 104, 105, 200, 110]. The heatmaps are then binarized to provide pseudo-segmentation masks for the target [118, 121].

**Pseudo-label refinement** Pseudo-labels generated by class saliency methods are initially coarse. In order to utilize the labels in a semantic classification model, the pseudo-labels are often refined [120, 201, 117, 121]. Approaches in the literature for pseudo-label refinement typically propagate label information in the spatial dimension or utilize class-level information to improve pseudo-label boundaries.

**Semantic classification** In the final step of the WSSS pipeline, a semantic classification model is trained directly from the pseudo-labels [135, 120, 202]. While the pseudo-labels are expected to contain errors at the pixel-level, a model trained on a large enough dataset is expected to generalize despite the discrepancies.

While each stage of the pipeline is vital to instance-level classification performance, this dissertation focuses primarily on improving pseudo-label generation (Step 2 in Figure 3-1). We describe our initial investigation in each stage of the pipeline to ultimately perform weakly-supervised semantic segmentation and elaborate on key results. We then follow with our approaches for feature selection and discriminative region estimation. Every method explored in this work learns solely form imprecise label information to perform instance-level classification.

## 3.2 Pipeline Component Investigation and Key Insights

To provide insight into future directions for WSSS, this work utilized post-hoc attention under the paradigm of multiple instance learning to explore WSSS. Specifically, this work shows that a DCNN is capable of distinguishing between positive and negative bags, leading to the presumption that features which can discriminate instance (pixel) labels can be abstracted from the information contained in the model. Class activation maps were explored as a baseline for performing WSSS from the learned features of the trained bag-level classifiers. Finally, feature reduction/extraction from the trained models was explored as a way to abstract instance-level classification information from the

bag-level classification models. Each method was applied on hold-out test data to evaluate the effectiveness of the segmentation techniques. Quantitative results are given as overall classification accuracy for bag-level classification and mean intersection-over-union (mIoU) for semantic segmentation.

An overview of the investigated approach is shown in Figure 3-2. The training method is performed in three stages, each of which adheres to MIL constraints. Stage A) trains a bag-level classification network. Stage B) estimates a class activation map for the inferred bag-level label. Stage C) up-samples and concatenates the activation feature maps from the bag-level classification network, which provides a feature vector at every pixel/instance. The corresponding class activation map from Stage B) is used as a pseudo-label and feature ranking is performed. An instance-level classifier is trained using the ranked features and corresponding CAM as groundtruth. In this manner, an instance-level classifier is estimated from bag-level labels. Each stage of the approach is described in detail in the following.

### 3.2.1 Bag-Level Classification

In this work, a bag represents an image $\boldsymbol{B} \triangleq I$, where $I \in \mathbb{R}^{u \times v \times w}$. Thus, a bag is a collection of feature vectors (instances) at every pixel spatial location, $(u, v)$. Each training image $I_k$ is paired with a label $L_k \in \{0, 1\}$, where $L_n = 0$ is assigned if every pixel in the image belongs to the background class (i.e. $l_{kn} = 0 \, \forall \boldsymbol{x}_{kn} \in \boldsymbol{B}_k^-$), and $L_n = 1$ if at least one pixel in the image belongs to the target class ($\exists \boldsymbol{x}_{kn} \in \boldsymbol{B}_k^+ \ni l_{kn} = 1$). While each bag is given a label according to the standard MIL assumption, the labels of individual instances in positive training bags are unknown.

A binary classification network was trained on the images with image-level labels (Stage (a), Figure 3-2). Consider one-hot encoded labels $\boldsymbol{L}_k = [L_{k0}, L_{k1}]$ where $\boldsymbol{L}_k = [0, 1]$ for a positive bag and $\boldsymbol{L}_k = [1, 0]$ for a negative bag. The goal of the binary classification network is to estimate the probability $p(\boldsymbol{B}_k, \boldsymbol{\theta}) = [p_0(\boldsymbol{B}_k, \boldsymbol{\theta}), p_1(\boldsymbol{B}_k, \boldsymbol{\theta})]$ that a bag belongs to the target or background class. A softmax output is applied to the output

of the network such that $p_0(\boldsymbol{B}_k, \boldsymbol{\theta}), p_1(\boldsymbol{B}_k, \boldsymbol{\theta}) \geq 0$ and $p_0(\boldsymbol{B}_k, \boldsymbol{\theta}) + p_1(\boldsymbol{B}_k, \boldsymbol{\theta}) = 1$. Cross-entropy loss, $\mathcal{L}_{cls}(\boldsymbol{B}; \boldsymbol{\theta}, \boldsymbol{L})$, was used on the image classification network to minimize the error between each predicted image-level label and the true class:

$$\min_{\boldsymbol{\theta}} \mathcal{L}_{cls}(\boldsymbol{B}; \boldsymbol{\theta}, \boldsymbol{L}) = \frac{1}{K} \sum_{k=1}^{K} \sum_{i=1}^{2} L_{ki} \log p_{bag}^{i}(\boldsymbol{B}_k; \boldsymbol{\theta}) \tag{3-1}$$

where $p_{bag}^{i}(\boldsymbol{B}_k; \boldsymbol{\theta})$ is the $i^{th}$ element of the softmax output layer for the bag classification network parameterized by $\boldsymbol{\theta}$.

### 3.2.2 Pseudo-groundtruth Generation using Class Saliency

CAMs were computed from the trained bag-level classifier (Stage (b), (Figure 3-2)). The CAMs were binarized across a range of confidence thresholds to investigate the ability of the network to inherently infer pixel-level semantics from image-level labels. Additionally, CAMs were explored as pseudo-labels for feature selection. While all methods were trained using imprecise labels, they were tested on a small subset of test data with accompanying pixel-level groundtruth.

### 3.2.3 Feature Selection from CAM Pseudo-groundtruth

Feature selection is a popular and straightforward approach for dimensionality reduction. Feature selection techniques define a smaller feature set by selecting a subset of the original features. There are three primary categories of feature selection techniques: filtering acts as a preprocessing step to construct an independent feature set before a classifier is constructed, wrapper techniques use the performance of the classifier as a fitness function on the set of features, and embedded methods include feature selection as part of the classifier's optimization objective (i.e. sparsity constraints) [31]. Feature ranking is an appealing filtering approach for its speed and simplicity. Essentially, feature ranking approaches evaluate a fitness function on each feature, independently. The features are then sorted by their scores and the top $K'$ are selected. Algorithm 3-8 depicts the feature ranking scheme used in this work to select activation maps from the trained bag-level classifier for instance-level classification. All

activation maps are extracted from the trained classification network and up-sampled to the input dimensionality (Stage (c), Figure 3-2). The concatenation of all up-sampled activations provides a feature set which can be exploited for pixel-level classification. Let $\boldsymbol{A}$ be the set of up-sampled activations for an input image and let $\boldsymbol{L}_{\text{CAM}}$ be the set of corresponding class activation maps computed on the predicted image-level labels. The implemented feature ranking approach uses mIoU as the fitness function between a single activation map and the inferred class activation map. As commonly done in the literature, pixel-level pseudo-groundtruth are provided as binarized CAM images for the inferred class labels. Thus, weak learning is maintained since the pseudo-groundtruth information comes from the bag-level labels. There are implicit assumptions that the predicted bag labels are correct, and that the computed CAMs adequately define the targets at the pixel-level. Once features have been selected, a logistic regression classifier is trained to predict thresholded CAM pseudo-labels. The classification model with reduced feature set is then tested on the hold-out test data to evaluate the ability of the model trained on weak labels to compute accurate pixel-level segmentations.

A trade-off of using feature ranking is that, if multiple features are needed to discover a correlation, they will not be considered since each feature is evaluated independently. An alternative would be to use a method that selects features sequentially, such as forward or backward feature selection where features are added to the set one at a time depending on their fitness with the current set of features. Compared to feature ranking, which has a linear run time ($K'$), a forward feature selection tends to be polynomial in the number of features since each new feature added requires re-evaluation of the scoring function for every feature not already in the set.

### 3.2.4 Key Insights and Assumptions

Results from the pipeline investigation are provided in Section 4. In the following, a summary of results, key insights directing the proposed method, and assumptions are provided.

**Algorithm 3-8** Feature ranking

**Input:** Dataset $\boldsymbol{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_N\}, \boldsymbol{L}_{\text{CAM}} = \{\boldsymbol{L}_{\text{CAM}_1}, \ldots, \boldsymbol{L}_{\text{CAM}_K}\}$, scoring/fitness function $\mathcal{S}$, feature set $\boldsymbol{A} = \{\boldsymbol{A}_1, \ldots, \boldsymbol{A}_K\}$, new dimensionality $K'$

$V \leftarrow$ [ ]
**for** feature $\boldsymbol{A}_k$ in feature set $\boldsymbol{A}$ **do**
    $V[k] \leftarrow (\mathcal{S}(\boldsymbol{X}, \boldsymbol{L}_{\text{CAM}}, \boldsymbol{A}_k), \boldsymbol{A}_k)$
**end for**
$V \leftarrow$ SORTDECREASING$(V)$
**return** $[\boldsymbol{A}_k$ for $\boldsymbol{A}_k$ in $V[1, \ldots, K']]$

1. Experiments showed that the models were able to learn effective bag-level classifiers. Under the MIL assumption, if a model understands how to infer the bag-level label, then it understands how to abstract out the instance-level labels in positive bags (i.e. all instances are negative in a negative bag; at least one instance in the positive bag is positive).

2. Pseudo-label generation by directly binarizing the CAMs showed that the effectiveness of alternative methods varies within different stages. I.e. Methods using backward gradient flow showed better pseudo-label generation in later stages of the network (3-5). Alternatively, methods using forward information flow perform better, on average, in early stages of the network (1-2). These results beg the question, would combining early, coarse information with late, fine-grained activation information benefit class saliency estimation at the pixel-level?

3. Feature selection experiments showed, by properly estimating activation map importance, one could obtain similar pixel-level pseudo-labels with fewer activation maps than CAMs estimated in any single stage of the network. Moreover, this was only achieved when both early and late stage activations were selected.

4. Feature selection also showed that selecting activations by maximizing IoU, solely, was not as good as when contrasting target information was present. Thus, knowing where the target is not, can be as informative as knowing where the target is.

5. CAM methods using backward gradient flow assume a linear surrogate model in each layer. However, neural networks are inherently nonlinear. This would suggest that CAMs might be improved by estimating a nonlinear surrogate.

Considering these insights, this dissertation focuses on two portions of the WSSS pipeline. Specifically, the proposed methods look to improve feature selection and class-discriminative region estimation. The proposed Fusion-CAM produces

class-discriminative heatmaps by fusing a subset of activation feature maps. The instance-level feature information can come from a single stage of the network, as done in vanilla CAM approaches, or from any one of the network's stages to combine coarse and fine-grained information. A natural choice for the aggregation operation is the Choquet integral, which has the added bonus of providing nonlinear fusion. Since learning directly from all activations in the network is a high-dimensional problem, there was strong motivation for feature learning/dimensionality reduction. Additionally, too many sources in the Choquet integral would make learning the measure intractable. In the following, we describe our approaches for instance-level target detection and feature/source selection. Additionally, we introduce a multiple instance learning based training objective and corresponding optimization scheme for learning the ChI measure.

## 3.3 Fusion-CAM Overview

In this chapter, a detailed description of the proposed Fusion-CAM framework for activation map fusion is provided. A pixel-level detection map over the input image is given as the ChI output for each pixel. The approach takes advantage of imprecise labels under the multiple instance learning framework to estimate a monotonic normalized fuzzy measure. The following describes the proposed Fusion-CAM framework, along with methods for source selection, measure learning, and evolutionary algorithm based optimization.

## 3.4 Choquet Integral for Activation Map Fusion

Fusion-CAM defines a class-specific heatmap as the Choquet Integral aggregation of feature maps extracted from a convolutional neural network. The feature maps can come from a single stage of a CNN, which is common for CAM methods in the literature or, uniquely, the activation feature set can come from a combination of stages in the network. Feature selection for Fusion-CAM is described in Section 3.6. Each input activation map, or source, $\boldsymbol{A}$, is an activation feature map that has been up-sampled to

the size of the input image ($U \times V$) using bilinear interpolation and scaled using min-max normalization such that each pixel falls in $[0, 1]$.

For a subset of $M^c$ sources for the $c$-th class, $\boldsymbol{A}^{\phi^c} = \{\boldsymbol{A}^{\phi^c_1}, \boldsymbol{A}^{\phi^c_2}, \ldots, \boldsymbol{A}^{\phi^c_{M^c}}\}$, the class-specific importance value $\alpha^c$ at pixel $(i, j)$ is given as:

$$\alpha^c_{ij} = C_{\mu_c}\left(\boldsymbol{A}^{\phi^c}_{ij}\right) = \sum_{m=1}^{M^c} A^{\phi^c_m}_{ij}\left[\mu_c\left(A^{\phi^c_m}_{ij}\right) - \mu_c\left(A^{\phi^c_{m-1}}_{ij}\right)\right] \tag{3-2}$$

where $C_{\mu_c}(.)$ is a Choquet Integral defined by class-specific measure $\mu_c$, activation value set $\boldsymbol{A}^{\phi^c_m}_{ij} = \left\{A^{\phi^c_{\pi(1)}}_{ij}, A^{\phi^c_{\pi(2)}}_{ij}, \ldots, A^{\phi^c_{\pi(m)}}_{ij}\right\}$ and $A^{\phi^c_{(0)}}_{ij} = 0$. $\pi$ is a permutation of $\boldsymbol{A}^{\phi^c_{ij}}$ such that $A^{\phi^c_{\pi(1)}}_{ij} \geq A^{\phi^c_{\pi(2)}}_{ij} \geq \cdots \geq A^{\phi^c_{\pi(M^c)}}_{ij}$. Note that the notation is kept general to allow for class-specific source subsets and aggregation operators, which could have varying numbers of sources. If the set of sources and measure values are kept constant, the notation simplifies to $\boldsymbol{A}^{\phi}$ and $C_{\mu}(\boldsymbol{A}^{\phi}_{ij})$.

The Fusion-CAM heatmap, $L^c_{\text{Fusion-CAM}}$, for class $c$ is given as the matrix of Choquet Integral outputs for the sources at each pixel location in the input image

$$L^c_{\text{Fusion-CAM}} = \boldsymbol{\alpha}^c \circ \mathbf{1} \tag{3-3}$$

where $\mathbf{1}$ is a matrix of ones the same size as the input image.

### 3.5 Learning the Measure

A crucial aspect of using the ChI to perform fusion is learning the element values of the fuzzy measure $\boldsymbol{\mu}$ from training data. This work adopts use of the Multiple Instance Choquet Integral (MICI) mentioned in Section 2.4.4 to learn the measure elements from multiple instance learning annotations. Du and Zare [162] introduced three objectives to learn the fuzzy measure elements. The noisy-or model was reviewed in Section 2.4.4. In this work, sources for the ChI were selected using either the min-max or generalized-mean model (Section 3.5.1 and Section 3.5.2) with a binary fuzzy measure. The fusion model was then trained with a full measure on the selected sources. As done by Du and Zare [162], the fuzzy measure elements were estimated using evolutionary

algorithm-based optimization. Details on learning the measure elements for the MICI are detailed in the following.

### 3.5.1 Min-max Fitness

The min-max model substitutes the noisy-or for "min" and "max" operators. The MIL framework assumes that for negative bags, all instances are negative (label "0"):

$$J^- = \sum_{j=1}^{N_J^-} \max_{\boldsymbol{x}_{jn} \in B_j^-} (C_{\boldsymbol{\mu}}(\boldsymbol{x}_{jn}) - 0)^2. \tag{3-4}$$

For positive bags, at least one instance in each bag should be positive (label "1"):

$$J^+ = \sum_{k=1}^{N_K^+} \min_{\boldsymbol{x}_{kn} \in B_k^+} (C_{\boldsymbol{\mu}}(\boldsymbol{x}_{kn}) - 1)^2, \tag{3-5}$$

where $N^-$ and $N^+$ are the total numbers of negative and positive bags, respectively. $\boldsymbol{x}_{jn}$ is the $n^{th}$ instance in the $j^{th}$ negative bag $B_j^-$. $\boldsymbol{x}_{kn}$ is the $n^{th}$ instance in the $k^{th}$ positive bag $B_k^+$. $C_{\boldsymbol{\mu}}$ is the Choquet integral output given measure $\boldsymbol{\mu}$.

The objective function for the MICI Min-Max model is:

$$J = J^- + J^+. \tag{3-6}$$

Minimizing the objective function encourages the Choquet integral of all instances in a negative bag to zero ($J^-$ term) and encourages the Choquet integral of at least one point in each positive bag to one ($J^+$ term).

### 3.5.2 Generalized-mean Fitness

A generalized mean model replaces the hard min and max operators used in Section 3.5.1 for soft constraints.

**Definition 3.5.1.** If $p$ is a non-zero real number and $x_1, \ldots, x_N$ are positive real numbers, then the generalized mean of $x_1, \ldots, x_N$ with exponent $p$ is:

$$M_p(x_1, \ldots, x_N) = \left( \frac{1}{N} \sum_{n=1}^{N} x_n^p \right)^{\frac{1}{p}}. \tag{3-7}$$

The generalized mean has the following two properties:

$$M_{-\infty}(x_1, \ldots, x_N) = \lim_{p \to -\infty} M_p(x_1, \ldots, x_N) = \min(x_1, \ldots, x_N). \tag{3-8}$$

$$M_{\infty}(x_1, \ldots, x_N) = \lim_{p \to \infty} M_p(x_1, \ldots, x_N) = \max(x_1, \ldots, x_N). \tag{3-9}$$

For the MICI Generalized Mean objective, the negative term is defined as

$$J^- = \sum_{j=1}^{N^-} \left[ \frac{1}{N_j^-} \sum_{n=1}^{N_j^-} (C_{\boldsymbol{\mu}}(\boldsymbol{x}_{jn}) - 0)^{2p_1} \right]^{\frac{1}{p_1}}, \tag{3-10}$$

and the positive term is given as

$$J^+ = \sum_{k=1}^{N^+} \left[ \frac{1}{N_k^+} \sum_{n=1}^{N_k^+} (C_{\boldsymbol{\mu}}(\boldsymbol{x}_{kn}) - 1)^{2p_2} \right]^{\frac{1}{p_2}}, \tag{3-11}$$

where $N^-$ and $N^+$ are the total numbers of negative and positive bags, respectively. $N_j^-$ and $N_k^+$ are the total numbers of instances in the $j^{th}$ negative bag and $k^{th}$ positive bag, respectively. $\boldsymbol{x}_{jn}$ is the $n^{th}$ instance in the $j^{th}$ negative bag $B_j^-$. $\boldsymbol{x}_{kn}$ is the $n^{th}$ instance in the $k^{th}$ positive bag $B_k^+$. $C_{\boldsymbol{\mu}}$ is the Choquet integral output given measure $\boldsymbol{\mu}$. The terms $p_1$ and $p_2$ are the two exponent parameters of the generalized means and $p_1 \to \infty$ and $p_2 \to -\infty$ according to properties 3-9 and 3-8.

Similarly to the MICI Min-Max model, the MICI Generalized Mean is computed as:

$$J = J^- + J^+. \tag{3-12}$$

Minimizing the objective function encourages the Choquet integral of all instances in a negative bag to zero ($J^-$ term) and encourages the Choquet integral of at least one point in each positive bag to one ($J^+$ term). As compared to the min-max objective, the generalized mean objective allows more points within a bag to contribute to the objective value.

### 3.5.3 Evolutionary Algorithm Optimization

The optimization scheme proposed by [5] was used to find the measure values for the MICI models. Algorithm pseudo-code for estimating the parameters for MICI is given in Algorithm 3-9. Details of the optimization are described in the following.

**Algorithm 3-9** MICI training [162]

**Input:** Training Data $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, Labels $\{L_1, \ldots, L_K\}$, Parameters
Initialize a population of measures. Set $t = 0$ ▷ 3.5.3.1
$F^* = \max(\boldsymbol{F}_P^0)$, $\boldsymbol{\mu}^* = \arg\max_{\mathcal{M}} \boldsymbol{F}_P^0$
**for** $t = 1$ to $T$ **do**
    **for** $p = 1$ to $P$ **do**
        Evaluate valid intervals of $\mathcal{M}\{p\}$ ▷ 3.5.3.2
        Randomly sample $z \in [0, 1]$
        **if** $z < \eta$ **then** ▷ 3.5.3.3
            Update $\mathcal{M}\{p\}$ by small-scale mutation
        **else**
            Update $\mathcal{M}\{p\}$ by large-scale mutation
        **end if**
    **end for**
    Evaluate fitness of updated measures using Equation 3-12
    Select measures for next iteration ▷ 3.5.3.4
    **if** $\max(\boldsymbol{F}_P^t) > F^*$ **then**
        $F^* = \max(\boldsymbol{F}_P^t)$, $\boldsymbol{\mu}^* = \arg\max_{\mathcal{M}} \boldsymbol{F}_P^t$
    **end if**
**end for**
**return** $\boldsymbol{\mu}^*$

#### 3.5.3.1 Measure initialization

A population (size $P$) of Choquet integral measures is generated and each measure element is randomly set to values in $[0, 1]$ while satisfying monotonicity constraints.

Both "top-down" and "bottom-up" approaches were implemented. In "top-down" initialization, the values of the measure elements are sampled from the top of the Hasse diagram towards the bottom. Using the example from Du [5], assume a Choquet integral with four sources. The top element $\mu_{1234} = 1$ by definition. The $(m-1)$-tuple measure elements $(\mu_{123}, \mu_{124}, \mu_{134}, \mu_{234})$ are first sampled randomly between $0$ and $1$. Then the $(m-2)$-tuple measure elements $(\mu_{12}, \mu_{13}, \mu_{14}, \mu_{23}, \mu_{24}, \mu_{34})$ were sampled between $0$ and

its corresponding superset. For example, the value for $\mu_{12}$ is sampled between $0$ and $\min(\mu_{123}, \mu_{124})$ to maintain monotonicity. The process proceeds until the singletons $(\mu_1, \mu_2, \mu_3, \mu_4)$ are sampled between $0$ and their corresponding superset values.

Similarly, "bottom-up" sampling finds element values from the bottom of the Hasse diagram towards the top. Singletons are first sampled in $[0, 1]$, then the duples are sampled between the max value of their corresponding subsets and $1$. The process continues until the $(m-1)$-tuple measure elements are sampled.

In our experiments, the measures are initialized by randomly flipping a coin to pick either "top-down" or "bottom-up" initialization.

#### 3.5.3.2 Evaluation of valid intervals

When a measure element is updated, it must still satisfy the monotonicity constraint. The term "valid interval" was used by Du [5] to define how large an element value can deviate while still meeting monotonicity. The valid interval width for each measure element is defined as the difference between the upper and lower bounds for each element. The upper bound for a measure element is the smallest value of its superset while the lower bound is the largest value of its subset. As example, for a Choquet integral with thee sources, $\mu_1$ and $\mu_2$ correspond to the subset of measure $\mu_{12}$, while $\mu_{123}$ corresponds to the superset of $\mu_{12}$. The lower bound of $\mu_{12}$ is defined as $\max(\mu_1, \mu_2)$ and the upper bound is $\mu_{123} = 1$. The valid interval width for element $\mu_{12}$ is $1 - \max(\mu_1, \mu_2)$. In this step of the optimization procedure, the valid interval width for each measure element is computed.

#### 3.5.3.3 Mutation

The evolutionary algorithm search uses both large-scale and small-scale mutations to update the measure population. In the large-scale mutation, all measure elements are sampled. In the small-scale mutation, only one new element is generated. The valid intervals computed above are used to determine which measure element to update during both types of mutations.

In the large-scale mutation, all measure elements are arranged in order of descending valid interval widths. Each measure element is sampled according to its order in the sort. This means that the element with the largest valid interval width is sampled first, then the measure with the second most room to change, and so on. Each new measure element is sampled from a truncated Gaussian (TG) distribution where the upper and lower bounds of the truncated Gaussian are defined as the upper and lower bounds of the valid interval for the corresponding measure element. Details on sampling from the truncated Gaussian distribution can be found in the work of Du [5].

A single measure element is sampled during a small-scale mutation. The element chosen to be sampled is selected by randomly sampling from a multinomial distribution defined on the valid intervals of all the measure elements. The likelihood of sampling a particular measure element $\mu_k$ is given by

$$p(\mu_k) = \frac{\delta_k}{\sum_{l=1}^{2^C-1} \delta_l}, \tag{3-13}$$

where $\delta_k$ is the valid interval width for measure element $\mu_k$. The measure element with the largest valid interval has the highest probability of being sampled. Similarly to the large-scale mutation, the new measure element value is sampled from a truncated Gaussian distribution.

The rate of small-scale mutation $\eta \in [0, 1]$ is defined by the user. The rate of large-scale mutation is $1 - \eta$.

#### 3.5.3.4 Selection

The measures retained for the next generation are selected based on their fitness values, computed as the min-max objective (Section 3.5.1) for selecting sources and the generalized-mean (Section 3.5.2) for estimating the full measure. In each iteration, every measure in the population is sampled, yielding a child measure population of size $P$. The measure population before sampling is deemed the parent measure population (size P). Both the child and parent measure populations are pooled together (size $2P$) and their

fitness values are computed using either the the min-max or generalized mean objective. The $P/2$ measures with the top $25\%$ fitness values are carried over to the next iteration and the remaining $P/2$ to be carried over are sampled according to a multinomial distribution based on their fitness values from the remaining $75\%$ of the parent and child population pool. Within the new measure population, the measure with the best fitness value is kept as the current best measure $\boldsymbol{\mu}^*$. The optimization process continues until a stopping criteria is reached, i.e. the max number of iterations or the change in the objective function value from one iteration to the next is smaller than a fixed threshold.

Once the optimization process has finished, the measure $\boldsymbol{\mu}^*$ corresponding to the best seen fitness value is returned as the learned measure.

## 3.6 Source Selection Approaches

This section describes the proposed source selection approaches based on the Choquet integral with binary fuzzy measure. The following describes the binary fuzzy measure and corresponding learning scheme for the MICI [5, 203] with binary fuzzy measure. Three approaches for down-selecting discriminative sources from a larger set are described, specifically, using a single binary measure, performing "N choose K" selection, and seeding and selecting using Grad-CAM [103] importance.

### 3.6.1 Binary Fuzzy Measure

Compared to the "regular" fuzzy measure where each element $\mu \in [0, 1]$, a binary fuzzy measure (BFM) restricts each element to $\mu \in \{0, 1\}$ [164]. The binary fuzzy measure follows the same general properties outlined in Definition 2.4.1, with an additional constraint.

**Definition 3.6.1.** A binary fuzzy measure (BFM), $\mu$, on $\boldsymbol{X}$ is a real-valued function that maps $\mu : 2^{\boldsymbol{X}} \to \{0, 1\}$. It satisfies the following properties [5, 164]:

1. (boundary condition): $\mu(\emptyset) = 0$
2. (normalized): $\mu(X) = 1$

3. (monotonicity property): If $A, B \subseteq \boldsymbol{X}$ and $A \subseteq B$, then $\mu(A) \leq \mu(B)$. That is to say if $\mu(B) = 0$, $\mu(A) \equiv 0$. Otherwise, if $\mu(B) = 1$, $\mu(A) \equiv 0$ or $1$. Similarly, if $\mu(A) = 1$, $\mu(B) \equiv 1$. Otherwise, if $\mu(A) = 0$, $\mu(B) \equiv 0$ or $1$.

Compared to the "regular" or "full" measure, the BFM can significantly save on storage and computation since we only need to record single-valued variables. Optimization is also more efficient since the set of possible measures is exhaustive. In this work, the MICI with a binary fuzzy measure and min-max objective was used to efficiently select sources. After source selection, a full measure was trained with generalized mean fitness for use in testing.

While optimization of the BFM can be done exhaustively, it is often more efficient to use something like an evolutionary algorithm to find the measure elements for a large number of sources. Pseudo-code for the MICI-BFM optimization procedure is shown in Algorithm 3-10 and details are described in the following.

Compared to a regular measure, the binary fuzzy measure only needs to optimize the space of $\{0, 1\}^{2M}$, where $M$ is the number of sources for fusion. Since the BFM only consists of binary measure values, the search space is finite and the learning process can always converge if all possible permutations of the fuzzy measure are analyzed. The measure value can be found using the optimization scheme discussed in Section 3.5.3 with a few modifications. At each iteration, $t$, a new BFM is generated by either sampling an entire BFM or by changing a single element value. This is equivalent to aforementioned small-scale and large-scale mutations. A collection of unique BFMs, $P$, are kept as a "look-up table" which keeps track off all unique BFMs that have been seen during the search. After sampling a new measure, we perform a check to see if the updated BFM already exists in $P$. If the BFM is already in $P$, then a new measure is sampled. The process is repeated until a new BFM is generated which has not been seen in the search. The fitness of each new measure is computed. If the fitness is better than the current best, then the best fitness and best measure are updated. If a new BFM cannot be found after searching $U$ times, or if the fitness values have not updated after $Q$

iterations, the binary flag $U_f$ is set to $True$ and the search stops. After the search, the best BFM, $\mathcal{M}^* = \boldsymbol{\mu}^*$, is returned for testing.

**Algorithm 3-10** MICI-BFM optimization [203]

**Input:** Training Data $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, Labels $\{L_1, \ldots, L_K\}$, Parameters
Initialize a BMF $\mathcal{M}_0$. Compute fitness $J_0$ using Eq. 3-6.
$J^* \leftarrow J_0$; $t, p, q \leftarrow 0$; $U_f \leftarrow False$; $P_0 \leftarrow \mathcal{M}_0$; $\mathcal{M}^* \leftarrow \mathcal{M}_0$
**while** True **do**
    $t \leftarrow t + 1$
    $\mathcal{M}_t \leftarrow$ Sample a new BFM based on $\mathcal{M}_{t-1}$ using Algorithm 3-11
    $u \leftarrow 0$
    **while** $\mathcal{M}_t \in P$ **do**
        $\mathcal{M} \leftarrow$ Sample a new BFM based on $\mathcal{M}_t$ according to Algorithm 3-11
        $u \leftarrow u + 1$
        **if** $u > U$ **then**
            $U_f \leftarrow True$
            break
        **end if**
    **end while**
    **if** $U_f$ **then**
        break
    **else**
        $p \leftarrow p + 1$
        $P_p \leftarrow \mathcal{M}_t$
        Evaluate fitness $J_t$ of $\mathcal{M}_t$ using Eq. 3-6
        **if** $J_t > J^*$ **then**
            $J^* \leftarrow J_t$
            $\mathcal{M}^* \leftarrow \mathcal{M}_t$
        **else**
            $q \leftarrow q + 1$
        **end if**
        **if** $q > Q$ **then**
            break
        **end if**
    **end if**
**end while**
**return** $\mathcal{M}^*$

While the genetic algorithm optimization scheme is appropriate for measures with large numbers of elements, an exhaustive search is more reliable when working with fewer sources. An issue with an exhaustive search, however, is the growing number of

**Algorithm 3-11** Sampling a new BFM [203]

**Input:** BFM $\mathcal{M}$, rate of small-scale mutation $\eta$

Randomly generate $z \in [0, 1]$
**if** $z < \eta$ **then**
    $\mathcal{M}' \leftarrow$ Sample a measure element in $\mathcal{M}$ based on valid intervals and update value
**else**
    $\mathcal{M}' \leftarrow$ Randomly generate a new BFM
**end if**
**return** Updated measure $\mathcal{M}'$

possible measures when adding sources, which can make the search computationally intractable. In order to combat this limitation, we propose a data-supported exhaustive search to learn the BFM when fusing less than 10 sources. Data supported measures [142, 143] are those which are backed by the training set. Instead of searching through every possible monotonic measure, we only need to consider those which are data supported. First, the sorts and corresponding walks are found from the training data. Next, all possible monotonic combinations of the corresponding measure elements are generated. All other elements (non-supported) are set to zero since they are not considered during training. The fitness is computed for each of the measures according to Equation 3-6. The measure with the best fitness is returned as the optimal measure. In this way, the search space can be greatly reduced depending on the sorts seen in the training data.

### 3.6.2 Source Selection

Three approaches were explored for down-selecting discriminative sources for the Choquet integral. The selection approaches are:

1. Single measure
2. N choose K
3. Seed with Grad-CAM importance and perform subsequent selection

In the first case, a single binary fuzzy measure is learned on the set of all potential sources using Algorithm 3-10. If all measure elements $\mathcal{M}_d$ including a particular source,

$d$, are “0”, then the source is deemed unnecessary and can be removed from the set of potential sources, $V$. A single, full measure can then be learned on the down-selected sources to implement the proposed Fusion-CAM 3.4. Pseudo-code for selection with a single measure is shown in Algorithm 3-12.

Unlike filtering methods such as feature ranking (Algorithm 3-8) which independently evaluate the fitness of a feature before selection, wrapper feature selection approaches use the performance of a classifier trained with a single feature or set of features to evaluate utility. Thus, “N choose K” feature selection can be deemed as a wrapper method in which all combinations of $K$ sources in the feature set are used to train $\binom{N}{K}$ multiple instance Choquet integrals with binary fuzzy measures. In this approach, $\mathcal{A}$, is the set of all possible combinations. The third selection approach which was investigated explored seeding the down-sampled source set with one or more features by their Grad-CAM attribution. Essentially, the features were ranked in order of importance weight by either stage or within the entire network. Subsequent selection was performed as with the “N choose K” approach, but the search space was constrained to only consider combinations which contained the seeded features. The sources which provide the best MICI fitness are retained and a ChI with full measure are trained for the Fusion-CAM framework. Algorithm 3-13 defines pseudo-code for both the “N choose K” and Grad-CAM seeding selection approaches.

**Algorithm 3-12** Source selection using a single measure

**Input:** Training Data $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, Labels $\{L_1, \ldots, L_K\}$, feature set $\boldsymbol{A} = \{\boldsymbol{A}_1, \ldots, \boldsymbol{A}_D\}$, Parameters

$V \leftarrow$ [ ]
$\boldsymbol{\mu} \leftarrow$ BFM according to Algorithm 3-10
**for** source $\boldsymbol{A}_d$ in feature set $\boldsymbol{A}$ **do**
    **if** $\exists \mu \in \mathcal{M}_d$ s.t. $\mu = 1$ **then**
        $V \leftarrow d$
    **end if**
**end for**
**return** $\{\boldsymbol{A}_d$ for $d$ in $V\}$

Experimental investigation into the utility of the described MICI+BFM source selection and subsequent Fusion-CAM schemes is provided in Section 4.

**Algorithm 3-13** Source selection using wrapper methods

**Input:** Training data $\{\boldsymbol{B}_1, \ldots, \boldsymbol{B}_K\}$, Labels $\{L_1, \ldots, L_K\}$, feature set $\boldsymbol{A} = \{\boldsymbol{A}_1, \ldots, \boldsymbol{A}_D\}$, scoring/fitness function $\mathcal{S}$, desired number of sources $K'$, Parameters

$V \leftarrow$ [ ]
$E \leftarrow$ [ ]
$idx \leftarrow$ [ ]
**if** Grad-CAM ranking **then**
    $\boldsymbol{\alpha}_{sorted} \leftarrow$ SORTDECREASING$(\boldsymbol{\alpha}_{Grad-CAM})$
    $idx \leftarrow$ original indices of the elements of $\boldsymbol{\alpha}_{sorted}$
    $E \leftarrow idx[0:k]$
    $\mathcal{A} \leftarrow$ COMBINATIONS$(\boldsymbol{A}, K') \cap \boldsymbol{A}_E$
**else**
    $\mathcal{A} \leftarrow$ COMBINATIONS$(\boldsymbol{A}, K')$
**end if**
**for** source set $\mathcal{A}_d \in \mathcal{A}$ **do**
    Train:
    $\boldsymbol{\mu} \leftarrow$ BFM according to Algorithm 3-10 for sources $\mathcal{A}_d$
    Evaluate fitness:
    $V[d] \leftarrow \mathcal{S}(C_{\boldsymbol{\mu}}(\boldsymbol{B}))$
**end for**
$V \leftarrow$ SORTDECREASING$(V)$
$idx^* \leftarrow$ index in $\mathcal{A}$ corresponding to $V[0]$
**return** $\mathcal{A}_{idx^*}$

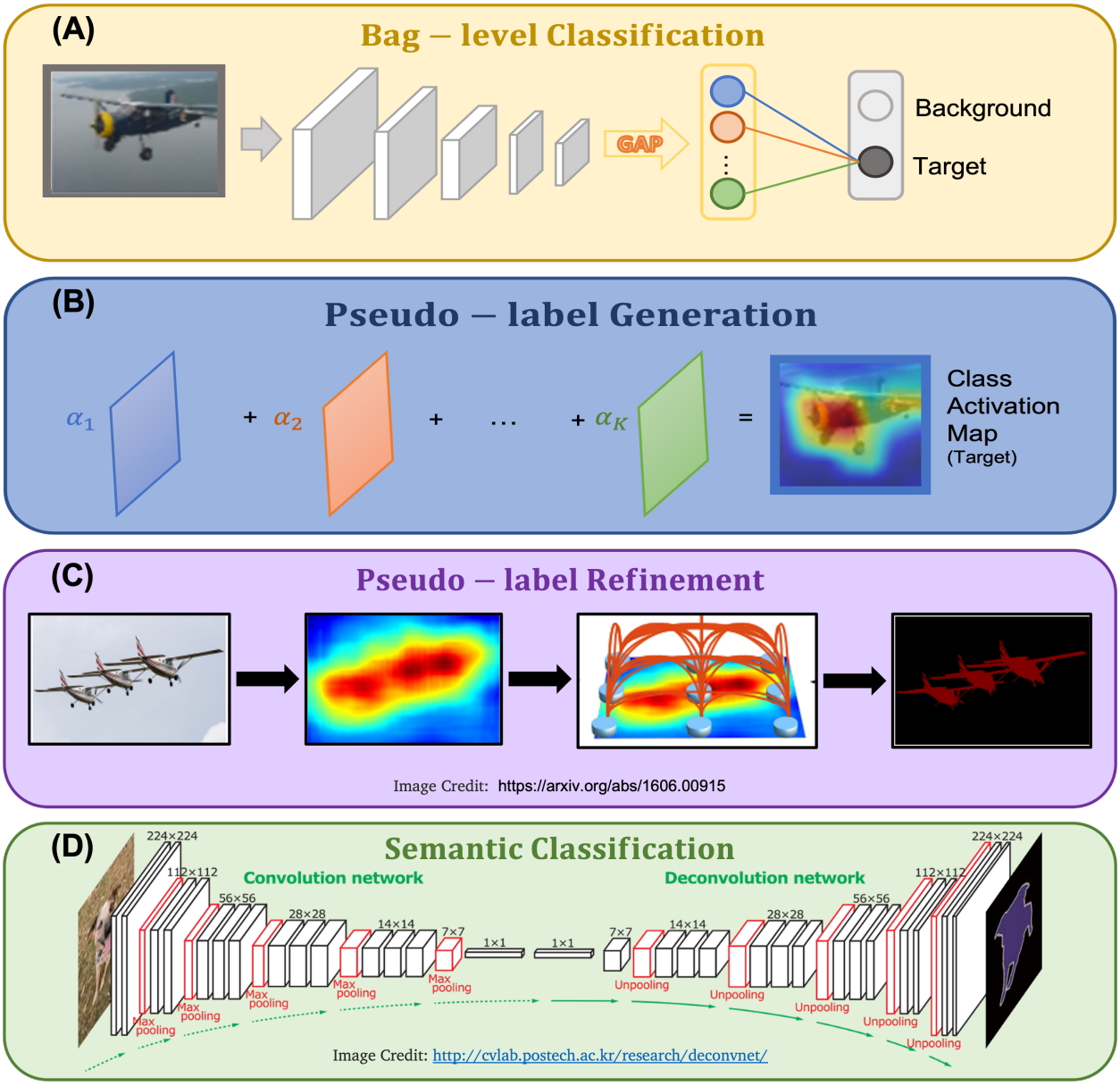



Figure 3-1. Overview of the WSSS pipeline. A) A model is trained to perform image/bag-level classification. B) Pseudo-labels are generated from the information contained within the classification model. C) Refinement is performed on the pseudo-labels to improve quality for semantic classification training. D) A semantic classifier is trained directly from the pseudo-labels.

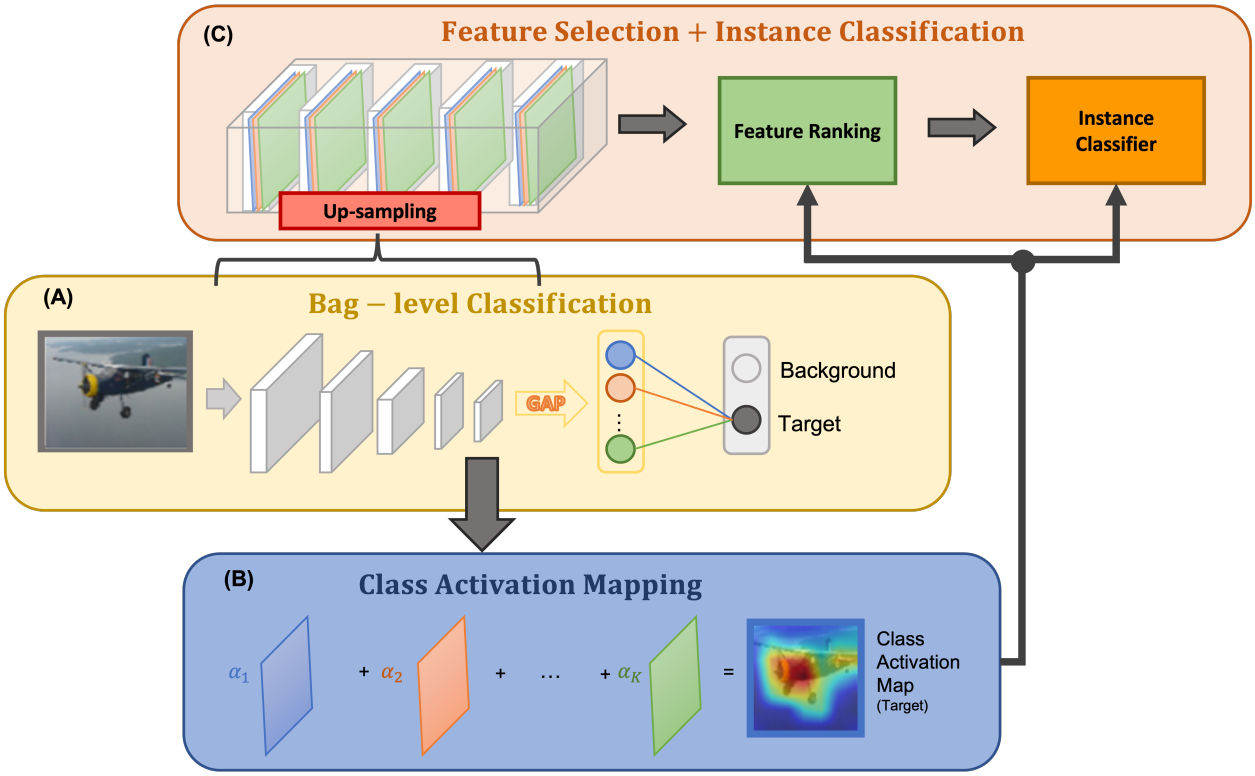
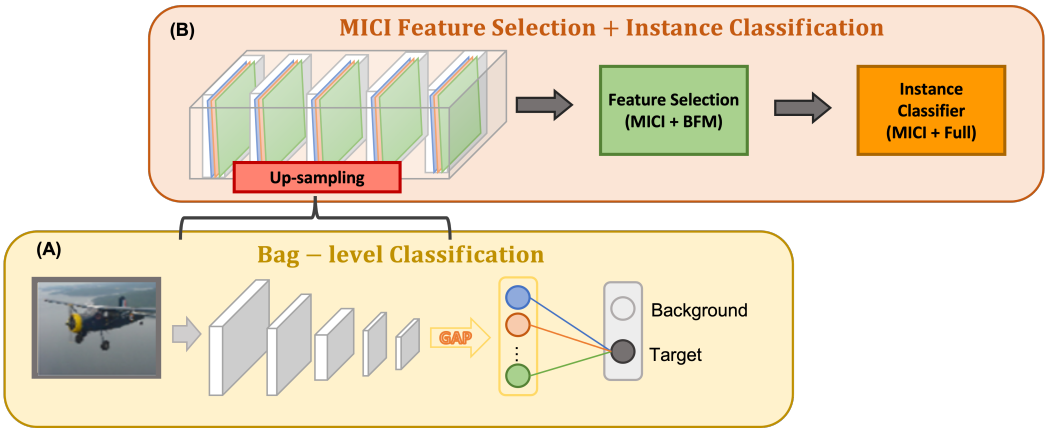



Figure 3-2. Overview of the investigated MIL instance-classification training approach. The method is performed in three stages. Stage A) trains a bag-level classification network. Stage B) estimates a class activation map for the inferred bag-level label for a training image. Stage C) up-samples and concatenates the activation feature maps from the bag-level classification network. This provides a feature vector at every pixel/instance. The corresponding class activation map from Stage B) is used as a pseudo-label to allow for feature ranking. Post-ranking, an instance-level classifier is trained using the CAM as groundtruth. All three training stages maintain MIL constraints.

**Training Stage:**




**Testing Stage:**

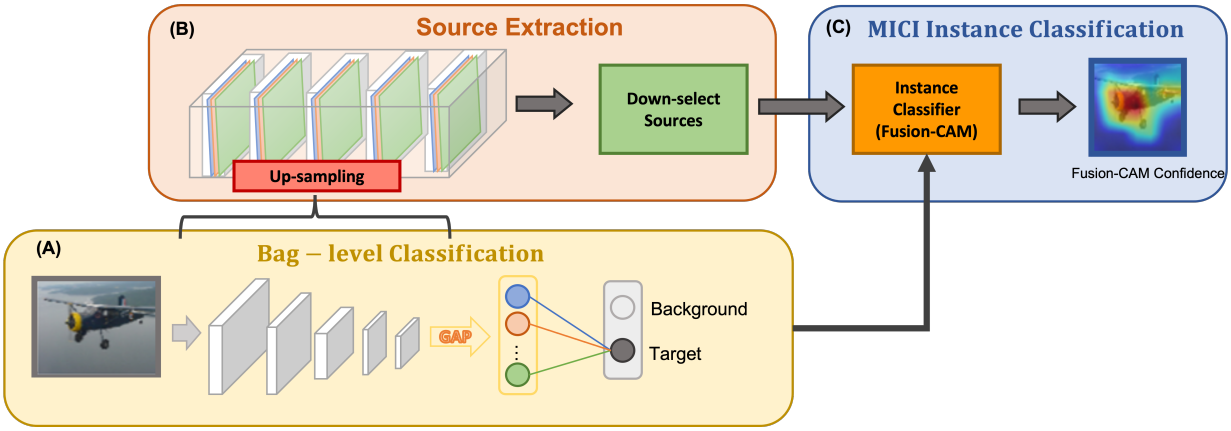



Figure 3-3. Overview of the proposed Fusion-CAM method for source selection and instance classification. In the training stage, A) a bag-level classifier is trained. In step B), the network features are up-sampled to the size of the input and then down-selected using the proposed MICI+BFM source selection. A MICI + full measure is then trained using multiple instance-learning labels to infer target instances. In the testing stage, A) bags are passed through the network to obtain activation map features and inferred bag-level classification labels. B) The activations maps are up-sampled and a set of sources is extracted from the global feature set. In C), the sources are fused according to a ChI with the previously learned measure to produce a confidence value for each instance in the bag. If the bag-classifier prediction indicated a negative bag then all instances are set to zero for the bag.

# CHAPTER 4
# EXPERIMENTAL RESULTS

## 4.1 Description of Data

This section describes the datasets used to evaluate the proposed methods. Specifically, MWIR data [14] was used in initial experimentation, a custom synthetic dataset [5] was used to show proof of concept, and MNIST [204] and PASCAL VOC [205] were used to evaluate pseudo-label generation.

### 4.1.1 Mid-wave Infrared

The data used in this work consists of mid-wave infrared (MWIR) video captures of both moving and non-moving civilian and military vehicles at various ranges and aspects. Each video can be considered as a collection of frames taken at the corresponding sensor's sampling rate. Datasets were each processed and broken into subsets amenable for learning with MIL. As shown in Figure 4-1, each gray-scale image frame, $I \in \mathbb{R}^{+(510\times720)}$, was clipped at $0.5\%$ and $98\%$. The images then underwent normalization by median absolute deviation (MAD), defined as

$$\text{MAD} = \text{median}(|I - \text{median}(I)|), \tag{4-1}$$

and were scaled to $[0, 255]$. Images in the datasets were originally annotated with bounding boxes. In order to analyze the data under MIL, sub-images were extracted to create sets of bags (see Section 2.1), where positive bags each contained at least one pixel on target and negative bags contained only background pixels. Following these constraints, sub-images were sampled such that negative bags had no overlap with target bounding boxes and positive bags had at least $25\%$ overlap with a target bounding box. Sub-samples were taken in scalar values of $(51 \times 121)$, which corresponds to the largest target present in the datasets. All sub-sampled images were up-sampled with bilinear interpolation to the original frame size of $(510 \times 720)$.

As can be seen in Figure 4-2, bags were constructed to represent various levels of groundtruth imprecision. This was done by changing the ratio of background to target pixels in the sub-sampled image chip. Essentially, canonical bags were constructed from

the provided bounding box annotations where the majority of pixels fall target. Ratios $\alpha =$ 1-3 increase the sample size as a scalar value of $(51 \times 121)$. As the scalar value increases, the ratio of background to target pixels in the image also increases. Training and testing on bags consisting of different ratios of target and non-target instances provides a way to capture the ability of a model to abstract pixel-level label information from the bags and corresponding bag-level labels.

Experiments were conducted on MWIR data taken from four distinct collection sites. The publicly-available DSIAC MS-003-DB Algorithm Development Database [14] is considered the easiest in the group, because targets in this set have very little occlusion and only collections from nighttime were used. It was anticipated that data from this site would provide the best bag-level performance and would be a good indicator of the possibility of learning pixel-level features from bag-level classification models. Sites “Y”, “H”, and “B” followed in assumed levels of difficulty. This was inferred from the targets included in the datasets, the levels of occlusion, and the various aspects to which the targets were visible. Also these three datasets included MWIR data collected at both day and night. The total numbers of image frames in each dataset are shown in Table 4-1. Only image-level labels were given to training and validation sets. Test sets were given hand-segmented pixel-level annotations for evaluating WSSS. Each split of the data contained equal numbers of positive/negative bags. Bag-level prediction was investigated for all four datasets, including all levels of bag imprecision (i.e. canonical/ratio 0 - ratio 3). To determine the role of positive bag construction (i.e. number of target versus background pixels) in inference, a bag-level classifier was also trained for each level of data imprecision for site “H”. Pixel-level segmentation was evaluated on the DSIAC dataset only.

### 4.1.2 Synthetic

A synthetic classification dataset was constructed as done by Du [5] which contains 1000 samples. Three individual sources, which can be viewed as class activation maps,

were provided such that each pixel had a value of “0” or “1”. Figure 4-9 shows scatter plots of the three sources for each data point. The data are randomly grouped into 50 bags with 20 instances per bag. Each positive bag contains $25\%$ positive instances and $75\%$ negative instances. This dataset was used to verify functionality of the MICI algorithm given the correct sources and source selection from a larger set of sources using a BFM.

### 4.1.3 MNIST

The MNIST [204] dataset is a collection of 60,000 grayscale images of size 28x28. Each image depicts a single handwritten digit between 0 and 9. The MNIST dataset is widely used and deeply understood, and for the most part is “solved” for image classification (i.e. classification accuracy above $99\%$). In this work, we consider two specific classes, specifically, “7” versus “1”. We considered each image as a bag where “7s” were positive bags and “1s” were negative bags. Ideal positive class CAMs should provide a “0” valued heatmap at every pixel in the “1” images, and a value of “1” should be given to every pixel in the “7” images which are unique from their alternative background classes. For example, the horizontal top or mid bars on a “7” should be activated by the CAM.

### 4.1.4 PASCAL VOC

Source selection and pseudo-label generation performance were evaluated on the PASCAL VOC 2012 [205] dataset. The training set contains 10,582 images from 20 unique classes, while the test set contains 1456 images with accompanying pixel-level annotations. We initially consider two classes for evaluation of the proposed source selection methods, specifically, “aeroplane” and “cat”.

## 4.2 Bag-level Classification

This section describes the training and evaluation of a bag-level classifier on the DSIAC data. Bag-level classification is the first step in the pseudo-label generation pipeline.

### 4.2.1 DSIAC

Bag-level classification was performed on all four datasets, individually. Each set contained data across all four investigated levels of bag imprecision. Additionally, each level of bag imprecision was investigated with it's own classifier for set "H". The model used was a ResNet18 [194] backbone (consisting of four convolutional blocks) with global average pooling (GAP) and a single fully-connected layer going to a softmax over two outputs. Following best-practices, models were initialized with parameters pre-trained on ImageNet [206]. Each model was trained to optimize Equation 3-1 for $1000$ epochs and parameters were updated with stochastic gradient descent (SGD). Table 4-2 reports the overall accuracy for the single best model for each dataset (selected from the validation data) on the hold-out test set, as well as the best epoch on the validation set. As can be observed, all models achieved over $93\%$ accuracy for bag-level classification, even on the "expert-designated difficult" datasets. Additionally, the model trained across all levels of bag imprecision for site "H" performed, on average, better than each individual model for a given level of bag imprecision. While counter-intuitive, an explanation for the combined model outperforming alternatives might be that the combination servers as a type of data augmentation, which has been shown to be beneficial to CNN training many times in the literature. Given that each CNN was able to effectively infer bag-level labels, (i.e. use the collection of instances to predict the label of the group), it was assumed that the models contained information about the labels of individual pixels which could be extracted from the networks.

## 4.3 Feature Ranking and Selection

This section describes the experiments and results conducted to investigate feature ranking and selection and pseudo-label generation for imprecisely annotated data.

### 4.3.1 Feature Ranking

Class activation map informed WSSS was further explored by investigating feature selection using CAM pseudo-labels. Following the previous CAM binarization

experiments, a LayerCAM model at Stage 4 of the trained VGG16 bag-level classifier was selected to generate pseudo-labels. Two methods for ranking features were explored. First, independent feature ranking was performed for each image by adaptively thresholding activation maps and corresponding LayerCAM pseudo-labels using Otsu's method [207]. The IoU between the binarized feature map and CAM pseudo-label was used as the scoring function. Features were sorted by average IoU performance across all training images. The second ranking was performed by simply sorting the importance values given to each map by Grad-CAM [103]. Figure 4-3 shows the training mIoU performance for each ordered feature index. As can be observed, the single top performing feature maps obtain, on average, IoU scores of $0.253$ (mIoU ranking) and $0.261$ (importance ranking). As expected, the feature ranking using mIoU on thresholded activations shows monotonic decreasing performance. Alternatively, the segmentation performance of independent features sorted by Grad-CAM importance shows much more variation. This might suggest that instead of weighting activation maps by their total abilities to cover the target, Grad-CAM may incrementally add activations with less IoU in order to "fill-in" missing areas on target and to reduce redundancy, similar to boosting approaches. Thus, non-linear feature combination may need to be considered to improve CAM computation for WSSS. Figure 4-4 shows examples of input bags with their top three features selected from ranking with mIoU between activations and CAM pseudo-labels. For the shown example, the first (importance) and second (mIoU) activation feature maps are the same. This is indicative that the feature is important for semantic segmentation. Qualitatively, this ranking is intuitive, as the feature map seems to queue on high response regions of the IR input.

Post feature ranking, the effect of feature set size was explored. A logistic regression classifier was trained $K$ times, where the feature set began with only the fittest feature, and subsequent iterations incrementally added the next fittest feature to the training set. In total, $K = 1472$ classifiers were trained to include every feature map

from the VGG16 backbone model. The activation maps in $[0, 1]$ were used as input features, while the binary classification labels were taken as the Otsu thresholded LayerCAM outputs for the training bags. Each classifier was tested on the hold-out test set, and the mIoU was computed between the predicted instance-level labels and pixel-level groundtruth. Figure 4-6 shows the results for each of the two fitness functions. Results are shown only for the first $k = 42$ feature sets, as performance only declined with additional features. It can be observed that the optimal number of features for the mIoU fitness is $k = 3$, which gave an average IoU of $0.369$. The CAM importance fitness gave an optimal mIoU of $0.352$ with $k = 42$ features. While the CAM importance fitness performance was on par with the CAM binarization approches for WSSS, the feature selection approach with binarized activations slightly outperformed alternative approaches.

From the results, it can be inferred that selecting feature maps from a trained bag-level classifier may be a viable option for learning instance-level classification features. The implemented feature selection approach benefits from combining both early and late network features. Additionally, feature selection and instance classification are limited by the quality of the CAM pseudo-labels. CAM computation for WSSS could likely be improved if early and late features were fused through intelligent, nonlinear combination.

## 4.4 Pseudo-groundtruth Estimation

This section explores pseudo-groundtruth generation by thresholding CAMs.

### 4.4.1 DSIAC Pseudo-labels

Following the bag-level classification experiments, an alternative bag-level classifier was trained for the DSIAC dataset across all levels of bag imprecision. A VGG16 backbone was trained in the same manner outlined in Section 4.2. As with the ResNet backbone, the VGG model achieved $100\%$ bag-level classification performance on the hold-out test set. The VGG model had five convolutional blocks which proceeded pooling

and ReLU activations. Each block of convolutional feature maps is referred to as a "stage", where Stage 1 represents early layers presumed to capture general feature information, and where Stage 5 represents late layers deemed to represent class-specific features and localization information. Six class activation map variations (GradCAM [103], GradCAM++ [104], LayerCAM [105], ScoreCAM [106], AblationCAM [107], and EigenCAM [108]) were computed in each of the five VGG16 stages for the DSIAC test data. Each CAM was binarized across a range of thresholds in $[0.001, 0.9]$ and compared to the pixel-level groundtruth. Semantic segmentation performance for each method can be observed in Figure 4-7 and performance in each stage is shown in Figure 4-8. Table 4-3 shows the mIoU at a fixed threshold of $\tau = 0.3$, which is consistent with the literature. While overall segmentation performance for each method was poor, there is a clear trend in the results. Specifically, the top scoring methods in late stages were GradCAM and GradCAM++. This might indicate that localization information in the late layers was very strong, as the model was relying on the large gradient magnitudes (i.e. strong localization) in those layers. In early stages, however, the effects of diminishing gradients can be seen, as GradCAM and GradCAM++ have a clear drop-off in performance. Alternatively, LayerCAM, ScoreCAM, and AblationCAM all showed improved performance in Stages 1-3. LayerCAM utilizes spatial information to improve early layer CAM computation, while ScoreCAM and AblationCAM substitute backward gradient flow for forward activation passes. These results might suggest a combination of methods taking advantage of both forward activation flow and backward gradient passing would benefit WSSS. EigenCAM was not computed in Stages 1 or 2 because of computational resource burden. Figure 4-5 shows examples of LayerCAM heatmaps for four DSIAC target bags.

### 4.5 Source Selection for the Choquet Integral by Measure Learning

In this section, we investigate the ability of the MICI to down-select instance-level discriminative features. Specifically, we investigate selection using a single measure (Algorithm 3-12) and wrapper selection (Algorithm 3-13).

First we investigate the utility of the BFM in learning an appropriate measure for source fusion. Table 4-4 shows the true measure and learned measures for 50 searches of a MICI+BFM and MICI+full measure for various generalized mean parameters. Results also show the average classification error, average fitness, and best fitness from the 50 runs. As can be seen, the binary measure was able to perfectly recover the true measure for all 50 searches. The MICI with generalized mean parameters $p1 = 500, p2 = -10$ performed the second best on average. It should be noted that the true measure demonstrates that a element value of one should have been given to the element for the combination of sources 1 and 2. This means that when the values of both sources were high, the ChI should have provided a high-valued output. Source 3 was irrelevant to learning the appropriate fusion. The binary measure was able to identify and enforce that result within the fusion operation.

Next, we investigated the ability of the MICI-BFM to learn an appropriate measure given valuable and invaluable sources. The true classification can be given by fusing only sources 1 and 2. Sources 3, 4, and 5 (Figure 4-10) do not add value to learning the correct inference, so they should be deemed irrelevant by the learned measure. Tables 4-5 and 4-6 show the MSE classification error and MSE error between the true and learned measures, respectively, for multiple scenarios: 3 sources + 1 measure, 4 sources + 1 measure, 5 sources + 1 measure, 4 choose 3 sources, 5 choose 3 sources, and 5 choose 4 sources. The measures were learned as a single binary measure, single full measure with random initialization, single full measure with BFM initialization, and single full measure initialized with $[0.1, 0.25, 0.5, 0.75, 0.9]$ percent random mutations of the true measures. The binary measures were learned with the proposed data-supported

exhaustive search and the full measures were estimated 20 times using a genetic optimization algorithm. The reported results are given for the measures with the best fitness over the 20 runs. As can be seen from the results, the BFM approach was able to recover the true measures in every scenario. This was possible since the data was binary and the correct fusion could be obtained from sources 1 and 2. The full measures learned with random initialization provided the worst performance on average, which is likely attributed to the fact that the algorithm timed out after 500 iterations. Alternatively, the full measure initialized with the learned BFM also recovered the true measures in every scenario. In every test using the BFM, the learned measure/ wrapper selection gave high value to sources 1 and 2 and minimal value to the other three sources. This result shows the potential benefit of using the BFM to pre-select sources/element values for the ChI.

Next, we look at source selection on a real image from a benchmark dataset. Specifically, we used a single aeroplane image from the PASCAL VOC dataset. Activation feature maps were extracted from various stages of a pre-trained VGG16 [110]. The input, groundtruth segmentation map, and considered sources are shown in Figure 4-11. Sources 1 and 2 are from stage 1, sources 3 and 4 are from stage 4, and the rest are from stage 5. Out of the selected sources, the best pseudo-segmentation label can be constructed from source 1, solely. Thus, the selection should always include source 1 if it is in the set of potential sources. Positive and negative bags were constructed where each bag contained 20 samples. Negative bags contained only negative samples, while positive bags contained $25\%$ of positive samples. Positive samples were estimated from the Grad-CAM heatmap for the target greater than $0.8$. The IoU means and standard deviations for the image are shown in Table 4-11 for various combinations of sources and selection approaches. As can be seen from the table, BFM selection in which source 1 was in the initial set consistently provided the best IoU. Inspection of the learned measures (Table 4-8) showed high worth on the

elements corresponding to source 1 and low worth for other sources. The BFM gave the worst performance when source 1 was not considered. This resuult is intuitive as a binary combination of the considered sources would not provided accurate pseudo-segmentation. While sources 4 and 6 might first appear to have high worth for target classification, they did not fall into the area denoted as “highly important” according to Grad-CAM. So while the BFM provided low IoU in this scenario, the full measures were able to consider various worths of the sources, which provided a boost in performance compared to the binary selection. Thus, when a high-worth source was included in the potential source set, the BFM selected it. Alternatively, when a high-worth source was not included, a full measure was able to learn higher-resolution combinations of less-valuable sources to improve performance.

Table 4-7 shows pseudo-segmentation performance on the single image for multiple CAM approaches in the literature for both stage 1 and stage 5 of the VGG network. As can be seen, our ChI fusion approach provided a significant performance boost compared to the alternative approaches. Two primary results can be gleaned from the table. First, fusion of activation maps from multiple stages of the network might be able to improve performance by combining both coarse and fine-grained class-level information. Additionally, selection and fusion of a few high-worth sources can provided competitive heatmap generation compared to traditional CAM approaches using every activation map in a single stage.

## 4.6 Bagging Approach

A crucial aspect to the success of multiple instance learning methods is the way in which data is bagged. Thus, performance of the MICI+BFM for source selection and fusion was tested on the MNIST [204] dataset for two bagging schemes. The first bagging approach considered “7” as the target class. Bags were constructed from each training image where likely positive samples were taken from the Grad-CAM heatmap estimates above Otsu’s threshold [207]. Negative instances were sampled from the pixel

values below Otsu's threshold. This approach considers a similar background across all "7" images. 20 bags were sampled from each training image, and each bag contained 20 instances. In positive bags, 75% of instances were estimated as negative and 5% were positive. We denote this bagging approach as "single class" in the results. In the second bagging approach, deemed "two class", negative bags were entire images from the "1" class and positive bags were entire images from the "7" class. This approach assumes that the background class is encapsulated by the negative bags. In the MNIST 7 vs. 1 example, the highlighted target points for the ChI output should be unique features distinguishing 7s from 1s, such as horizontal cross-bars on the tops and middles of the vertical line.

### 4.6.1 Single Class versus Two Class Bagging

For each bagging approach, binary fuzzy measures were learned for various combinations of up to 8 sources as described in Section 3.6. The sources were taken in both stage 1 and stage 5 of a trained VGG16 network. For every class in the training dataset, the Grad-CAM importance values were averaged for each neuron. The neurons were then ranked from greatest to least Grad-CAM importance in each stage. The top-8 ranked activation feature maps in stages 1 (early) and 5 (late) were considered for ChI measure learning and fusion. For each combination of sources, a single BFM was estimated. The ChI fusion was then tested on a hold-out test data consisting only of 7s.

Results comparing the two bagging approaches for various combinations of sources are shown in Tables 4-10 and 4-11. Quantitative evaluation is given as average classification error (MSE), mean intersection-over-union (mIoU), and min-max fitness. Values close to zero are best for MSE and fitness, while values close to 1 are best for IoU. From Tables 4-10 and 4-11, a few trends can be observed. First, both bagging approaches provided better segmentation/classification performance, on average, when fusing sources from stage 1 rather than stage 5. This is likely because the features in stage 1 have better pixel-level resolution than those in stage 5. This is an effect of the

networks' shrinking receptive field. This result is backed by Figure 4-13, which shows that the top ranked features for stage 1 have, on average, higher IoU than features from stage 5. The figure also shows that down-selection using Grad-CAM importance might be a valid method for narrowing the initial source set because the most important Grad-CAM features tend to have higher IoU than the sources corresponding to lower Grad-CAM importance. A qualitative trend from the results show that while stage 1 features typically provide higher resolution for segmentation, stage 5 feature tend to pickup on more nuanced class-level features, such as the horizontal bars on 7s. This result also aligns with previous intuition gleaned from the literature. The third result shows that single class bagging typically demonstrates better classification/segmentation performance while two class bagging provides better fitness. This result can be explained intuitively. Although the two class bagging showed better fitness, it does not necessarily mean that the fusion was better since fitness is a relative measure between the bags. I.e. fitness measures the difficulty between selecting a single point in a positive bag as positive while forcing all points in a negative bag to zero. While this metric might be better for the two class bagging, it is not representative of segmentation performance of how well the ChI picks up on discriminative class-level features. On the other hand, bagging with a single class will inherently force the ChI to maximize IoU. Since the negative bags are local to the target images, anything in the bag not representing background should denote target. While not apparent from the quantitative results, measures learned from single class bagging showed more diversity in the source combinations, whereas the measures learned for two class bagging often provided the highest worth when all sources were fused, regardless of source ordering. The prime takeaway from this experiment is that it might be appropriate to bag the data differently depending on the dataset. For example, if a second class encapsulates the expected background around the target class, two class bagging might be appropriate. Alternatively, if a second class is not representative of the local background to the target

class, then single class bagging might be a better choice. An example where single-class bagging might be better is for the PASCAL VOC benchmarking dataset. If trying to detect airplanes, for example, it isn't apparent that the "cat" class should be used for negative bags, since it is unlikely that an airplane and cat will exist in the same image. Alternatively, local background to the airplane (i.e. sky, airport, runway) should be extracted for negative bags.

### 4.6.2 Bag Size and Generalized-mean Parameters

The second critical parameter of bagging is the number of instances in each bag. To test this, we explored the number of instances in each bag using the aforementioned single class scheme on the PASCAL VOC Aeroplane class. We considered a 4 source ChI fusion, where the sources were taken as the top 4 Grad-CAM importance ranked features. The first and fifth VGG16 stages were considered separately. The number of points in each bag were varied as $[20, 40, 100, 50, 1000, 10000]$, and the fitness function was investigated as the min-max and the generalized-mean model with various $p$ parameters. 20 bags were taken from each of the 40 randomly sampled training bags. Figure 4-12 shows the mean intersection over union results, averaged over the 72 hold-out test images. It can be seen from the figure that the numbers of instances per bag had less of an effect for the fusion of stage 5 features. This is likely a result of the receptive field of the network, where negative bag features are more likely to be zero. Thus the difference between the min-max and generalized-mean fitness functions (for most parameter variations) are less prominent. On the other hand, for the fusion of stage 1, the performance increases as the number of instances per bag increases for the generalized-mean models with parameters $p_1 = 10, p_2 = -100$ and $p_1 = 100, p_2 = -10$. This result was expected, since the generalized-mean considers the inferred instance values over multiple points. The min-max is unlikely the optimal choice for the pseudo-label generation problem, since it only tries to maximize the value of a single point in each positive bag. Instead, multiple positive points should be pushed to one.

## 4.7 Source Down-selection

Initial down-selection approaches were explored by ranking Grad-CAM importance and objective fitness. ChI fusion of selected sources were compared to common pseudo-label generation approaches in the literature.

### 4.7.1 Grad-CAM Ranking

First, initial down-selection was performed by ranking Grad-CAM importance. Figures 4-13 and 4-14, show activation map ranking for both MNIST 7 and PASCAL VOC Aeroplane classes. From left to right, the columns show performance in earlier to later stages of the network. The top row shows the average Grad-CAM importance of each feature ordered from greatest to least. The middle row shows the mean intersection-over-union (IoU) of each feature ordered from greatest to least. The bottom row shows the average IoU ordered according to average Grad-CAM importance. As can be seen, the top 20 Grad-CAM ranked features for MNIST typically have high IoU values. This might indicate that the top 20 features are useful for semantic segmentation. This trend isn't apparent for the Aeroplane class. This could be a result of the variation in the class or network generalization, where the indices of activated neurons are more variable. Thus, for datasets where the background is uniform, initial down-selection using Grad-CAM ranking might be a good choice. Alternatively, when there is substantial variation in the dataset or when the information in the network is widely distributed, alternative down-selection approaches might be needed.

### 4.7.2 Fitness Ranking

The second down-selection approach investigated was fitness ranking. Figure 4-15 shows average IoU, precision, recall, and F1 score against fitness on the Aeroplane class for stages 1 and 5. The fitness comes from various combinations of 4 sources in the first and last stages of the neural network. It can be seen that as fitness improves, segmentation performance also improves for stage 5. This result is intuitive because the sources are more localized to the target objects in later stages than in early stages. This

might make it easier to meet the constraints of the fitness functions. Alternatively, this trend is not clear in the fitness values from stage 1 fusion. Thus, initial down-selection using fitness might not be useful for down-selecting sources in early stages.

### 4.7.3 Comparison to the Literature

Next, Fusion-CAM was compared to standard pseudo-label generation approaches in the literature. Features were initially down-selected to 10 or 20 sources using Grad-CAM importance. Four sources were then selected using N choose 4 selection. As can be seen from Tables 4-12 and 4-13, the best performing Fusion-CAM model performed second-best in terms of IoU and F1 score for stage 1. This implies that Fusion-CAM demonstrated improved coverage over the target class while striking a balance between precision and recall. For stage 5, Fusion-CAM actually performed worse than the alternatives in terms of IoU, but provided a substantial performance increase in precision. This means that Fusion-CAM detections did not cover as much of the target but were more likely to be true positives. Additionally, there were less false positives than the alternatives. Figures 4-16 and 4-17 show qualitative comparisons of Grad-CAM, Score-CAM, LIFT-CAM, and Fusion-CAM for stages 1 and 5, respectively. Visual results support Tables 4-12 and 4-13, where the IoU and F1 are improved for stage 1 pseudo-labels and the precision is improved for stage 5 pseudo-labels.

It should be noted that the threshold for converting detection maps to binary values was set at $\tau = 0.3$. This value was selected from validation testing on the compared approaches, excluding Fusion-CAM. For stage 1 pseudo-label generation, our method actually appears (visually) to perform better than the alternative approaches. Thus, alternative selection of the threshold might skew the quantitative evaluation in favor of our approach.

Table 4-1. Dataset breakdown for MWIR experiments.

| Dataset | Train | Valid | Test |
|---|---|---|---|
| D Total | 2800 | 548 | 1674 |
| D Ratio 0 | 92 | 16 | 54 |
| D Ratio 1 | 906 | 174 | 540 |
| D Ratio 2 | 920 | 160 | 540 |
| D Ratio 3 | 882 | 198 | 540 |
| Y Total | 359138 | 63017 | 147043 |
| Y Ratio 0 | 35914 | 6320 | 14686 |
| Y Ratio 1 | 107742 | 18945 | 44073 |
| Y Ratio 2 | 107742 | 18868 | 44149 |
| Y Ratio 3 | 107740 | 18883 | 44135 |
| H Total | 76660 | 29405 | 68614 |
| H Ratio 0 | 7666 | 2890 | 6912 |
| H Ratio 1 | 22998 | 8740 | 20666 |
| H Ratio 2 | 22998 | 8933 | 20473 |
| H Ratio 3 | 22998 | 8842 | 20563 |
| B Total | 143065 | 7589 | 17708 |
| B Ratio 0 | 35766 | 1873 | 4451 |
| B Ratio 1 | 35766 | 1882 | 4442 |
| B Ratio 2 | 35766 | 1930 | 4394 |
| B Ratio 3 | 35766 | 1904 | 4420 |

Table 4-2. Test accuracy and best validation epoch for bag-level classification models.

| Dataset | Accuracy | Best epoch |
|---|---|---|
| D Total | 1.000 | 90 |
| Y Total | 0.974 | 65 |
| H Total | 0.950 | 92 |
| B Total | 0.960 | 69 |
| H Ratio 0 | 0.943 | 70 |
| H Ratio 1 | 0.953 | 25 |
| H Ratio 2 | 0.948 | 40 |
| H Ratio 3 | 0.930 | 45 |

Table 4-3. DSIAC semantic segmentation performance using binarized VGG16 class activation maps at a fixed threshold of $\tau = 0.3$. Results are shown as the mean intersection-over-union (mIoU) and standard deviation for images predicted as "positive" bags.

| Method | Stage 1 | Stage 2 | Stage 3 | Stage 4 | Stage 5 |
|---|---|---|---|---|---|
| Grad-CAM [103] | 0.084(0.046) | 0.049(0.052) | 0.044(0.042) | 0.156(0.078) | 0.274(0.114) |
| Grad-CAM++ [104] | 0.114(0.103) | 0.114(0.106) | 0.214(0.138) | 0.376(0.129) | 0.242(0.135) |
| LayerCAM [105] | 0.078(0.069) | 0.194(0.140) | 0.322 (0.168) | 0.358(0.123) | 0.238(0.112) |
| Score-CAM [106] | 0.124(0.119) | 0.213(0.231) | 0.107 (0.094) | 0.113(0.094) | 0.169(0.120) |
| Ablation-CAM [107] | 0.115(0.107) | 0.088(0.083) | 0.098(0.081) | 0.168(0.116) | 0.195(0.093) |
| Eigen-CAM [108] | - | - | 0.048(0.042) | 0.065(0.078) | 0.045(0.106) |

Table 4-4. True and learned measures for synthetic 3-source dataset. The top row indicates the true measure for the three sources, where the binary fuzzy measure value should equal $1$ for $\mu_{12}$ and $\mu_{123}$ and $0$ for every other measure element. The second row from the top shows the learned measure for $50$ runs of the evolutionary optimization with min-max object and binary fuzzy measure. The remaining rows show the learned measures for $50$ runs of the evolutionary optimization for MICI with the generalized-mean objective and a full measure with varying parameters of the generalized mean, $p1$ and $p2$. Each column shows the mean value with standard deviation for each measure element. The rightmost columns of the table show the mean squared error, average fitness and best fitness from the $50$ optimization searches, where a value close to $0$ is best.

| $\mu_1$ | $\mu_2$ | $\mu_3$ | $\mu_{12}$ | $\mu_{13}$ | $\mu_{23}$ | MSE | Avg. fitness | Best fitness |
|---|---|---|---|---|---|---|---|---|
| True measure | | | | | | | | |
| 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 1.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000 |
| Binary fuzzy measure | | | | | | | | |
| 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 1.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000(0.0000) | 0.0000 |
| $p1 = 1, p2 = -10$ | | | | | | | | |
| 0.0070(0.0110) | 0.0060(0.0060) | 0.0090(0.0130) | 0.9830(0.0240) | 0.0210(0.0190) | 0.0210(0.0160) | 0.0002(0.0003) | -0.0120(0.0180) | -0.0530 |
| $p1 = 1, p2 = -500$ | | | | | | | | |
| 0.0100(0.0170) | 0.0050(0.0070) | 0.0047(0.0036) | 0.7250(0.1290) | 0.0150(0.0170) | 0.0110(0.0080) | 0.0031(0.0020) | -0.0072(0.0170) | -0.0780 |
| $p1 = 10, p2 = -1$ | | | | | | | | |
| 0.0178(0.0232) | 0.021(0.0187) | 0.0135(0.0139) | 0.9653(0.0552) | 0.0400(0.0338) | 0.0380(0.0365) | 0.0009(0.0014) | -0.1034(0.1556) | -0.5702 |
| $p1 = 10, p2 = -10$ | | | | | | | | |
| 0.0292(0.0482) | 0.0104(0.0151) | 0.0124(0.0128) | 0.9241(0.1319) | 0.0512(0.0519) | 0.0292(0.0246) | 0.0019(0.0031) | -0.1468(0.2470) | -0.8646 |
| $p1 = 10, p2 = -100$ | | | | | | | | |
| 0.0096(0.0123) | 0.0142(0.0098) | 0.0123(0.0123) | 0.9621(0.0338) | 0.0317(0.0332) | 0.0290(0.0158) | 0.0005(0.0006) | -0.0560(0.0926) | -0.4024 |
| $p1 = 10, p2 = -500$ | | | | | | | | |
| 0.0109(0.0087) | 0.0162(0.0161) | 0.0117(0.0129) | 0.7385(0.1266) | 0.0343(0.0362) | 0.0315(0.0271) | 0.0032(0.0022) | -0.0677(0.1338) | -0.5947 |
| $p1 = 10, p2 = -0.5$ | | | | | | | | |
| 0.0235(0.0255) | 0.0201(0.0164) | 0.0150(0.0140) | 0.9802(0.0317) | 0.0471(0.0406) | 0.0441(0.0364) | 0.0010(0.0013) | -0.1172(0.1582) | -0.4923 |
| $p1 = 100, p2 = -10$ | | | | | | | | |
| 0.0121(0.0090) | 0.0115(0.0138) | 0.0130(0.0180) | 0.9289(0.1516) | 0.0318(0.0321) | 0.0306(0.0319) | 0.0014(0.0031) | -0.0833(0.1884) | -0.6971 |
| $p1 = 500, p2 = -1$ | | | | | | | | |
| 0.1541(0.1131) | 0.1390(0.1305) | 0.0853(0.0934) | 0.9737(0.0181) | 0.3414(0.1032) | 0.3578(0.1071) | 0.0359(0.0151) | -0.0040(0.0046) | -0.0167 |
| $p1 = 500, p2 = -10$ | | | | | | | | |
| 0.0817(0.0640) | 0.0612(0.0527) | 0.0770(0.0525) | 0.9889(0.0343) | 0.2823(0.1115) | 0.2785(0.1192) | 0.0210(0.0090) | -0.0015(0.0064) | -0.0295 |
| $p1 = 0.5, p2 = -10$ | | | | | | | | |
| 0.0062(0.0052) | 0.0061(0.0098) | 0.0048(0.0067) | 0.9804(0.0366) | 0.0184(0.0235) | 0.0166(0.0258) | 0.0003(0.0006) | -0.0097(0.0221) | -0.0972 |

Table 4-5. Inference error for the synthetic 5-source dataset. Results show the mean squared error (MSE) between the true labels and ChI inference for various combinations of sources and selection approaches. Binary measures were only trained once, while the full measures were trained 20 times each.

| Sources | BFM | FFM random | FFM+BFM | FFM $0.1\%$ | FFM $0.25\%$ | FFM $0.5\%$ | FFM $0.75\%$ | FFM $0.9\%$ |
|---|---|---|---|---|---|---|---|---|
| | | | | 3 sources 1 measure | | | | |
| $\{1,2,3\}$ | 0.0(0.0) | 0.002(0.003) | 0.0(0.0) | 0.004(0.009) | 0.001(0.002) | 0.004(0.005) | 0.006(0.009) | 0.004(0.005) |
| | | | | 4 sources 1 measure | | | | |
| $\{1,2,3,3\}$ | 0.0(0.0) | 0.005(0.001) | 0.0(0.0) | 0.013(0.008) | 0.017(0.009) | 0.019(0.005) | 0.016(0.007) | 0.012(0.004) |
| $\{1,2,4,4\}$ | 0.0(0.0) | 0.003(0.001) | 0.0(0.0) | 0.005(0.001) | 0.009(0.002) | 0.009(0.003) | 0.009(0.005) | 0.009(0.004) |
| $\{1,2,5,5\}$ | 0.0(0.0) | 0.002(0.003) | 0.0(0.0) | 0.006(0.003) | 0.008(0.003) | 0.010(0.003) | 0.011(0.005) | 0.010(0.003) |
| $\{1,2,3,4\}$ | 0.0(0.0) | 0.007(0.009) | 0.0(0.0) | 0.013(0.007) | 0.023(0.016) | 0.016(0.009) | 0.017(0.012) | 0.018(0.014) |
| $\{1,2,3,5\}$ | 0.0(0.0) | 0.002(0.002) | 0.0(0.0) | 0.004(0.004) | 0.005(0.004) | 0.006(0.004) | 0.006(0.004) | 0.009(0.010) |
| $\{1,2,4,5\}$ | 0.0(0.0) | 0.004(0.006) | 0.0(0.0) | 0.007(0.001) | 0.004(0.002) | 0.010(0.010) | 0.007(0.004) | 0.006(0.002) |
| | | | | 5 sources 1 measure | | | | |
| $\{1,2,3,4,5\}$ | 0.0(0.0) | 0.018(0.007) | 0.0(0.0) | 0.012(0.012) | 0.017(0.006) | 0.022(0.005) | 0.020(0.008) | 0.033(0.017) |
| | | | | N sources choose K | | | | |
| 4 Choose 3 | 0.0(0.0) | 0.001(0.004) | 0.0(0.0) | 0.002(0.003) | 0.003(0.007) | 0.002(0.005) | 0.002(0.002) | 0.001(0.001) |
| 5 Choose 3 | 0.0(0.0) | 0.001(0.001) | 0.0(0.0) | 0.001(0.001) | 0.001(0.001) | 0.001(0.001) | 0.002(0.002) | 0.002(0.003) |
| 5 Choose 4 | 0.0(0.0) | 0.005(0.007) | 0.0(0.0) | 0.011(0.007) | 0.026(0.023) | 0.012(0.008) | 0.021(0.008) | 0.018(0.008) |

Table 4-6. Measure error for the synthetic 5-source dataset. Results show the mean squared error (MSE) between the true and learned measures for various combinations of sources and selection approaches. Binary measures were only trained once, while the full measures were trained 20 times each.

| Sources | BFM | FFM random | FFM+BFM | FFM $0.1\%$ | FFM $0.25\%$ | FFM $0.5\%$ | FFM $0.75\%$ | FFM $0.9\%$ |
|---|---|---|---|---|---|---|---|---|
| | | | | 3 sources 1 measure | | | | |
| $\{1,2,3\}$ | 0.0(0.0) | 0.001(0.009) | 0.0(0.0) | 0.005(0.019) | 0.000(0.006) | 0.002(0.019) | 0.007(0.023) | 0.005(0.011) |
| | | | | 4 sources 1 measure | | | | |
| $\{1,2,3,3\}$ | 0.0(0.0) | 0.012(0.032) | 0.0(0.0) | 0.004(0.071) | 0.007(0.105) | 0.008(0.080) | 0.009(0.072) | 0.009(0.052) |
| $\{1,2,4,4\}$ | 0.0(0.0) | 0.000(0.023) | 0.0(0.0) | 0.015(0.055) | 0.007(0.052) | 0.003(0.043) | 0.004(0.064) | 0.009(0.051) |
| $\{1,2,5,5\}$ | 0.0(0.0) | 2.750(0.014) | 0.0(0.0) | 0.033(0.042) | 0.020(0.036) | 0.017(0.042) | 0.012(0.076) | 0.023(0.059) |
| $\{1,2,3,4\}$ | 0.0(0.0) | 0.027(0.060) | 0.0(0.0) | 0.002(0.088) | 0.056(0.221) | 0.065(0.146) | 0.015(0.156) | 0.022(0.131) |
| $\{1,2,3,5\}$ | 0.0(0.0) | 0.041(0.085) | 0.0(0.0) | 0.147(0.145) | 0.234(0.125) | 0.268(0.114) | 0.211(0.109) | 0.174(0.155) |
| $\{1,2,4,5\}$ | 0.0(0.0) | 0.006(0.083) | 0.0(0.0) | 0.109(0.162) | 0.103(0.048) | 0.110(0.157) | 0.088(0.078) | 0.011(0.066) |
| | | | | 5 sources 1 measure | | | | |
| $\{1,2,3,4,5\}$ | 0.0(0.0) | 0.044(0.0287) | 0.0(0.0) | 0.390(0.584) | 0.688(0.358) | 0.744(0.235) | 0.680(0.325) | 0.971(0.372) |
| | | | | N sources choose K | | | | |
| 4 Choose 3 | 0.0(0.0) | 6.634(0.009) | 0.0(0.0) | 0.001(0.009) | 2.460(0.018) | 0.000(0.011) | 3.112(0.005) | 0.000(0.003) |
| 5 Choose 3 | 0.0(0.0) | 2.592(0.002) | 0.0(0.0) | 1.400(0.003) | 0.000(0.005) | 4.756(0.003) | 0.001(0.003) | 1.851(0.009) |
| 5 Choose 4 | 0.0(0.0) | 0.008(0.047) | 0.0(0.0) | 0.003(0.081) | 0.047(0.291) | 0.009(0.079) | 0.019(0.144) | 0.003(0.125) |

Table 4-7. Choquet integral measure learning and source selection for a single aeroplane image. Results show the intersection-over-union (IoU) between the binarized ChI output for the learned measure and the corresponding pixel-level segmentation map. Binary measures were only trained once, while the full measures were trained 20 times each.

| Sources | BFM | FFM random | FFM+BFM |
|---|---|---|---|
| | 3 sources 1 measure | | |
| $\{1, 2, 6\}$ | 0.579(0.0) | 0.412(0.044) | 0.415(0.048) |
| $\{1, 5, 6\}$ | 0.579(0.0) | 0.518(0.006) | 0.508(0.017) |
| $\{1, 6, 10\}$ | 0.579(0.0) | 0.446(0.021) | 0.457(0.013) |
| $\{2, 5, 6\}$ | 0.004(0.0) | 0.294(0.047) | 0.321(0.061) |
| $\{4, 5, 6\}$ | 0.343(0.0) | 0.456(0.003) | 0.455(0.003) |
| $\{4, 6, 10\}$ | 0.343(0.0) | 0.407(0.031) | 0.427(0.015) |
| | 4 sources 1 measure | | |
| $\{1, 2, 4, 6\}$ | 0.579(0.0) | 0.428(0.029) | 0.410(0.066) |
| $\{1, 2, 6, 10\}$ | 0.579(0.0) | 0.364(0.046) | 0.379(0.050) |
| $\{1, 4, 5, 6\}$ | 0.579(0.0) | 0.501(0.016) | 0.491(0.015) |
| $\{2, 4, 6, 10\}$ | 0.042(0.0) | 0.294(0.039) | 0.041(0.001) |
| | 5 sources 1 measure | | |
| $\{1, 2, 6, 8, 10\}$ | 0.579(0.0) | 0.311(0.032) | 0.579(0.001) |
| | $\{1, 2, 6, 10, 8\}$ choose K | | |
| 4 Choose 3 | 0.579(0.0) | 0.429(0.025) | 0.411(0.013) |
| 5 Choose 3 | 0.579(0.0) | 0.424(0.019) | 0.427(0.021) |
| 5 Choose 4 | 0.579(0.0) | 0.360(0.049) | 0.363(0.052) |

Table 4-8. Learned measures for the hand-picked aeroplane dataset. Results show the learned measures for combinations of 3 sources.

| Measure | $\mu_1$ | $\mu_2$ | $\mu_3$ | $\mu_{12}$ | $\mu_{13}$ | $\mu_{23}$ | IoU |
|---|---|---|---|---|---|---|---|
| | | | Sources $\{1, 2, 6\}$ | | | | |
| BFM | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 1.000(0.000) | 0.000(0.000) | 0.579(0.000) |
| FFM random | 0.983(0.013) | 0.381(0.164) | 0.762(0.030) | 0.990(0.007) | 0.994(0.006) | 0.774(0.031) | 0.412(0.044) |
| FFM BFM | 0.968(0.026) | 0.381(0.196) | 0.735(0.037) | 0.979(0.026) | 0.994(0.008) | 0.749(0.028) | 0.415(0.048) |
| | | | Sources $\{1, 5, 6\}$ | | | | |
| BFM | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 1.000(0.000) | 0.000(0.000) | 0.579(0.000) |
| FFM random | 0.960(0.029) | 0.587(0.014) | 0.230(0.049) | 0.988(0.010) | 0.992(0.009) | 0.591(0.014) | 0.518(0.006) |
| FFM BFM | 0.990(0.012) | 0.480(0.215) | 0.311(0.115) | 0.998(0.004) | 0.995(0.009) | 0.636(0.094) | 0.508(0.017) |
| | | | Sources $\{1, 6, 10\}$ | | | | |
| BFM | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 1.000(0.000) | 0.000(0.000) | 0.579(0.000) |
| FFM random | 0.973(0.040) | 0.763(0.025) | 0.156(0.125) | 0.993(0.007) | 0.993(0.008) | 0.876(0.067) | 0.446(0.021) |
| FFM BFM | 0.995(0.010) | 0.760(0.012) | 0.072(0.105) | 0.998(0.004) | 0.998(0.004) | 0.825(0.047) | 0.455(0.003) |
| | | | Sources $\{2, 5, 6\}$ | | | | |
| BFM | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 1.000(0.000) | 0.004(0.000) |
| FFM random | 0.681(0.077) | 0.011(0.008) | 0.734(0.033) | 0.766(0.092) | 0.752(0.038) | 0.994(0.006) | 0.294(0.047) |
| FFM BFM | 0.610(0.126) | 0.009(0.008) | 0.730(0.040) | 0.739(0.135) | 0.745(0.040) | 0.996(0.004) | 0.321(0.061) |
| | | | Sources $\{4, 5, 6\}$ | | | | |
| BFM | 0.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 0.343(0.000) |
| FFM random | 0.008(0.006) | 0.071(0.041) | 0.730(0.015) | 0.654(0.072) | 0.984(0.010) | 0.816(0.016 | 0.456(0.003) |
| FFM BFM | 0.016(0.013) | 0.067(0.037) | 0.739(0.021) | 0.690(0.093) | 0.976(0.016) | 0.821(0.012) | 0.455(0.003) |
| | | | Sources $\{4, 6, 10\}$ | | | | |
| BFM | 0.000(0.000) | 0.000(0.000) | 0.000(0.000) | 1.000(0.000) | 0.000(0.000) | 0.000(0.000) | 0.343(0.000) |
| FFM random | 0.044(0.067) | 0.811(0.022) | 0.236(0.169) | 0.981(0.011) | 0.628(0.225) | 0.860(0.052) | 0.407(0.031) |
| FFM BFM | 0.022(0.031) | 0.687(0.175) | 0.060(0.083) | 0.986(0.013) | 0.444(0.250) | 0.831(0.152) | 0.427(0.015) |

Table 4-9. CAM segmentation comparison on the example aeroplane image. Results are given as the IoU of the binarized CAM heatmap using Otsu's method to the true segmentation map.

| Method | Stage 1 | Stage 5 |
|---|---|---|
| Grad-CAM | 0.099 | 0.426 |
| Grad-CAM++ | 0.202 | 0.373 |
| Layer-CAM | 0.245 | 0.419 |
| Score-CAM | 0.203 | 0.385 |
| Ours (Source 1 Included) | 0.579 | 0.579 |

Table 4-10. Comparison of bagging approaches on MNIST classes “7” as positive and “1” as negative for VGG16 stage 1. Results show classification error (MSE), mean intersection-over-union (mIoU), and the best fitness over 20 optimization runs. The two bagging approaches compared were the single class - using only “7” images and estimating positive and negative bags from corresponding Grad-CAM heatmaps, and two class - where positive bags were entire “7” images and negative bags were “1” images. Values close to zero are best for MSE and fitness, while a value close to 1 is optimal for mIoU.

| Sources | MSE | mIoU | Fitness |
|---|---|---|---|
| | Single Class | | |
| $\{1, 2, 3\}$ | 0.047(0.006) | 0.518(0.049) | -9.403 |
| $\{1, 2, 3, 4\}$ | 0.047(0.006) | 0.518(0.049) | -9.403 |
| $\{1, 2, 3, 4, 5\}$ | 0.052(0.005) | 0.474(0.059) | -9.260 |
| 4 Choose 3 | 0.047(0.006) | 0.518(0.049) | -9.403 |
| 5 Choose 3 | 0.052(0.005) | 0.474(0.059) | -9.260 |
| 5 Choose 4 | 0.052(0.005) | 0.474(0.059) | -9.260 |
| 8 Choose 3 | 0.052(0.005) | 0.474(0.059) | -9.260 |
| 8 Choose 4 | 0.052(0.005) | 0.474(0.059) | -9.260 |
| 8 Choose 5 | 0.052(0.005) | 0.474(0.059) | -9.260 |
| | Two Class | | |
| $\{1, 2, 3\}$ | 0.061(0.010) | 0.456(0.089) | -8.074 |
| $\{1, 2, 3, 4\}$ | 0.062(0.011) | 0.448(0.082) | -7.780 |
| $\{1, 2, 3, 4, 5\}$ | 0.063(0.011) | 0.441(0.088) | -7.780 |
| 4 Choose 3 | 0.062(0.011) | 0.452(0.081) | -7.780 |
| 5 Choose 3 | 0.062(0.011) | 0.452(0.081) | -7.780 |
| 5 Choose 4 | 0.063(0.011) | 0.441(0.088) | -7.780 |
| 8 Choose 3 | 0.083(0.025) | 0.341(0.063) | -4.857 |
| 8 Choose 4 | 0.083(0.025) | 0.341(0.063) | -4.857 |
| 8 Choose 5 | 0.083(0.025) | 0.341(0.063) | -4.857 |

Table 4-11. Comparison of bagging approaches on MNIST classes “7” as positive and “1” as negative for VGG16 stage 5. Results show classification error (MSE), mean intersection-over-union (mIoU), and the best fitness over 20 optimization runs. The two bagging approaches compared were the single class - using only “7” images and estimating positive and negative bags from corresponding Grad-CAM heatmaps, and two class - where positive bags were entire “7” images and negative bags were “1” images. Values close to zero are best for MSE and fitness, while a value close to 1 is optimal for mIoU.

| Sources | MSE | mIoU | Fitness |
|---|---|---|---|
| | Single Class | | |
| $\{1, 2, 3\}$ | 0.057(0.009) | 0.400(0.067) | -21.875 |
| $\{1, 2, 3, 4\}$ | 0.064(0.009) | 0.378(0.067) | -19.705 |
| $\{1, 2, 3, 4, 5\}$ | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 4 Choose 3 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 5 Choose 3 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 5 Choose 4 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 8 Choose 3 | 0.072(0.011) | 0.308(0.056) | -19.460 |
| 8 Choose 4 | 0.072(0.011) | 0.308(0.056) | -19.460 |
| 8 Choose 5 | 0.072(0.011) | 0.308(0.056) | -19.460 |
| | Two Class | | |
| $\{1, 2, 3\}$ | 0.067(0.020) | 0.360(0.010) | -4.866 |
| $\{1, 2, 3, 4\}$ | 0.073(0.019) | 0.356(0.088) | -4.661 |
| $\{1, 2, 3, 4, 5\}$ | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 4 Choose 3 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 5 Choose 3 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 5 Choose 4 | 0.064(0.009) | 0.378(0.067) | -19.705 |
| 8 Choose 3 | 0.072(0.011) | 0.308(0.056) | -19.460 |
| 8 Choose 4 | 0.072(0.011) | 0.308(0.056) | -19.460 |
| 8 Choose 5 | 0.066(0.015) | 0.380(0.094) | -4.448 |

Table 4-12. Quantitative comparisons for stage 1. Results show the average metric value along with standard deviation.

| Method | mIoU | Precision | Recall | F1 |
|---|---|---|---|---|
| Grad-CAM | 0.001(0.020) | 0.195(0.339) | 0.014(0.038) | 0.024(0.060) |
| Grad-CAM++ | 0.188(0.123) | 0.414(0.201) | 0.307(0.215) | 0.300(0.168) |
| Layer-CAM | 0.065(0.091) | 0.421(0.350) | 0.085(0.134) | 0.110(0.141) |
| Score-CAM | 0.227(0.153) | 0.356(0.210) | 0.474(0.271) | 0.346(0.198) |
| LIFT-CAM | 0.139(0.109) | 0.140(0.111) | 0.975(0.067) | 0.228(0.159) |
| Ours (Top 10 Grad-CAM) BFM | 0.202(0.156) | 0.345(0.259) | 0.369(0.224) | 0.309(0.208) |
| Ours (Top 20 Grad-CAM) BFM | 0.202(0.192) | 0.344(0.261) | 0.377(0.238) | 0.306(0.239) |
| Ours (Top 20 Grad-CAM) FFM + BFM | 0.165(0.119) | 0.342(0.225) | 0.319(0.213) | 0.266(0.166) |
| Ours (Top 4 Grad-CAM w/ inverted) BFM | 0.161(0.119) | 0.345(0.228) | 0.305(0.208) | 0.260(0.166) |

Table 4-13. Quantitative comparisons for stage 5. Results show the average metric value along with standard deviation.

| Method | mIoU | Precision | Recall | F1 |
|---|---|---|---|---|
| Grad-CAM | 0.220(0.149) | 0.236(0.167) | 0.854(0.187) | 0.337(0.201) |
| Grad-CAM++ | 0.226(0.142) | 0.299(0.159) | 0.950(0.083) | 0.428(0.192) |
| Layer-CAM | 0.220(0.149) | 0.230(0.147) | 0.969(0.071) | 0.347(0.190) |
| Score-CAM | 0.259(0.142) | 0.265(0.147) | 0.932(0.178) | 0.391(0.185) |
| LIFT-CAM | 0.139(0.109) | 0.140(0.111) | 0.975(0.067) | 0.228(0.159) |
| Ours (Top 10 Grad-CAM) BFM | 0.116(0.100) | 0.230(0.222) | 0.270(0.237) | 0.194(0.153) |
| Ours (Top 20 Grad-CAM) BFM | 0.197(0.164) | 0.351(0.293) | 0.325(0.275) | 0.297(0.230) |
| Ours (Top 20 Grad-CAM) FFM + BFM | 0.193(0.134) | 0.382(0.254) | 0.319(0.246) | 0.302(0.193) |
| Ours (Top 4 Grad-CAM w/ inverted) BFM | 0.187(0.157) | 0.360(0.302) | 0.301(0.264) | 0.286(0.223) |

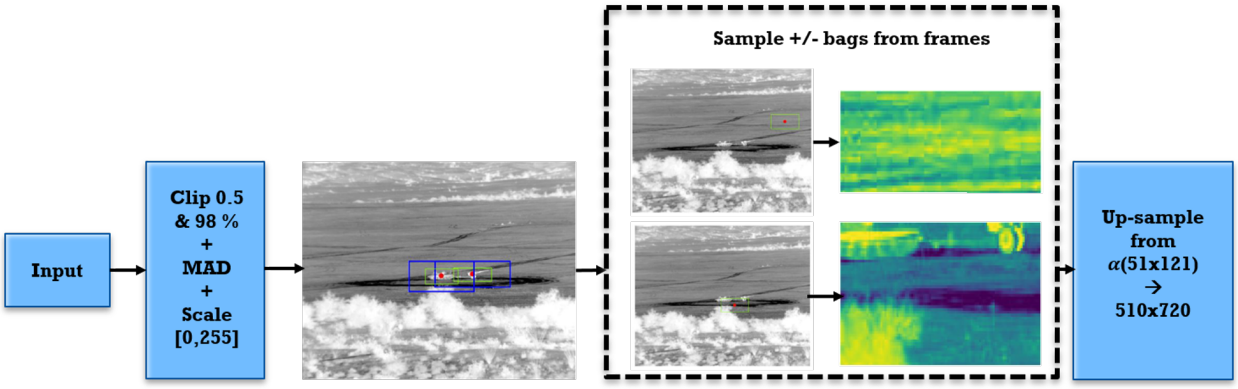



Figure 4-1. Frame pre-processing and bag sampling pipeline. First, input frames are clipped, normalized, and scaled. Next, positive and negative sub-samples are taken from the frames. Finally, sampled chips are scaled to the size of the input frame.

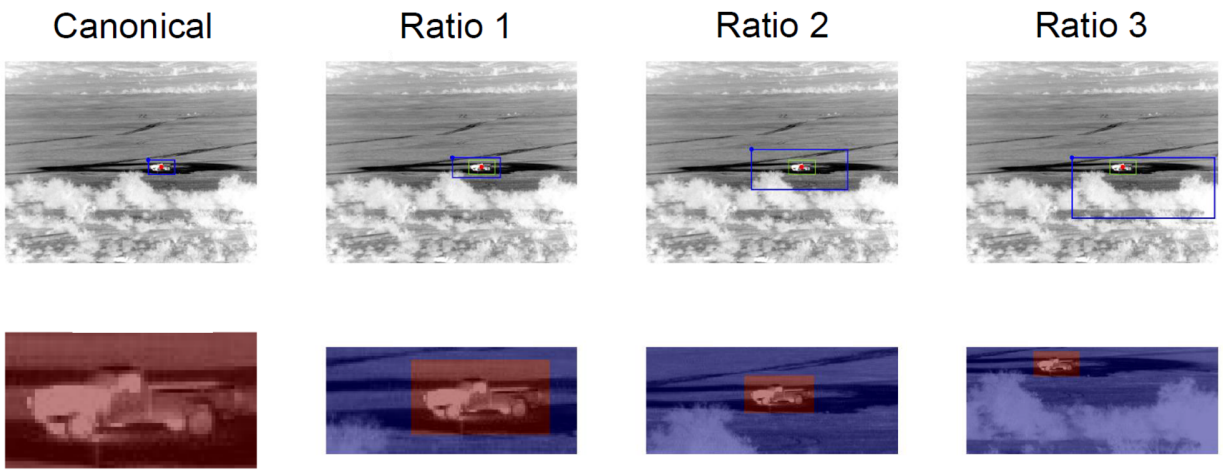
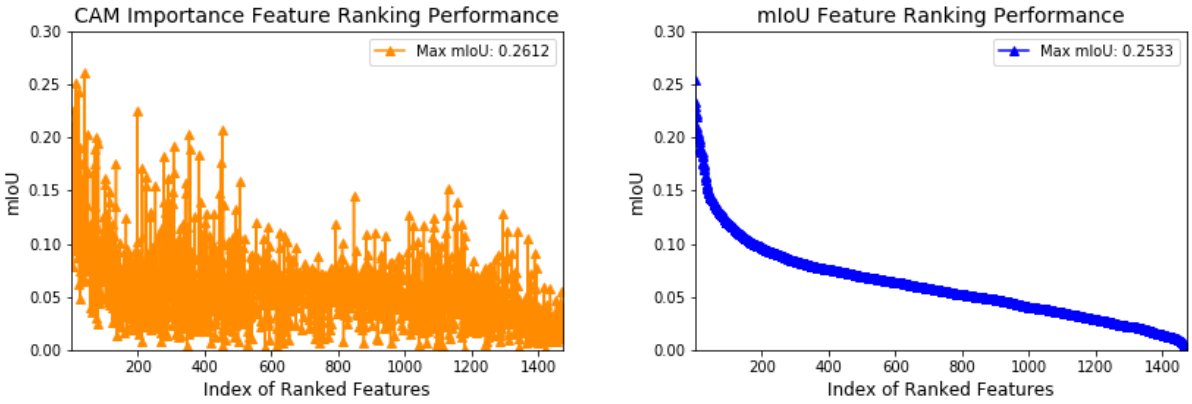



Figure 4-2. Demonstration of positive bag sampling for various levels of imprecision. In the infrared images, green boxes represent the bounding box annotation and blue boxes represent the frame sub-sample capture. Annotation bounding boxes are shown in red. Canonical bags are constructed from the provided bounding box annotations and the majority of pixels fall on target. Ratios 1-3 increase the sample size as a scalar value of $(51 \times 121)$. As the scalar value increases, the ratio of background to target pixels in the image also increases.




(A) (B)

Figure 4-3. Independent segmentation performance for A) Grad-CAM importance ranked features and B) mean intersection-over-union (mIoU) ranked features. The single top performing feature maps obtain, on average, IoU scores of $0.253$ (mIoU ranking) and $0.261$ (importance ranking). The mIoU ranked features show monotonic decreasing segmentation performance while the Grad-CAM ranked features show more variation.

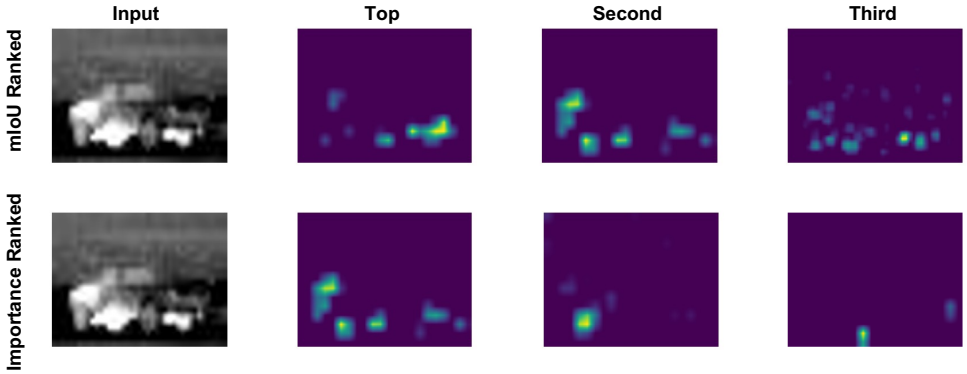



Figure 4-4. Input images with top three activation maps ranked by mIoU between activations and LayerCAM pseudo-labels and Grad-CAM importance weights.

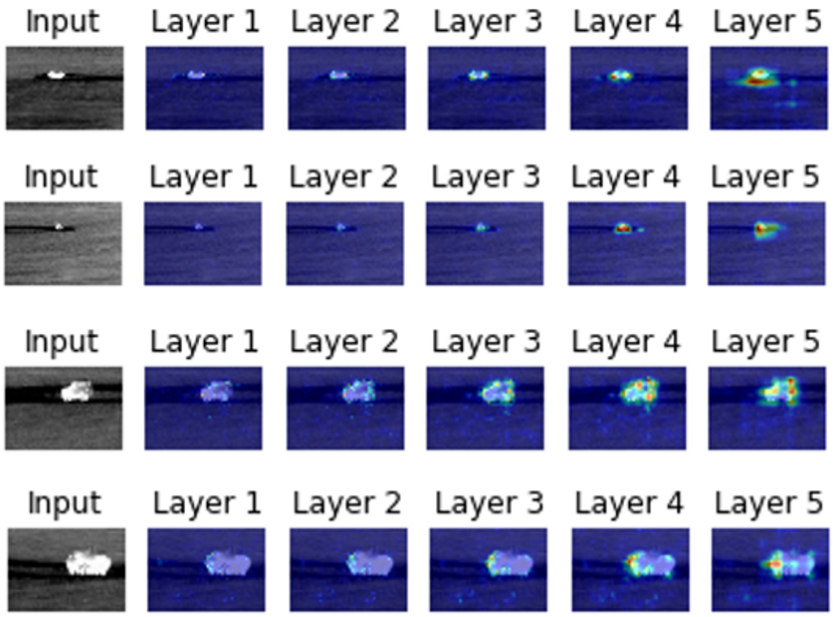
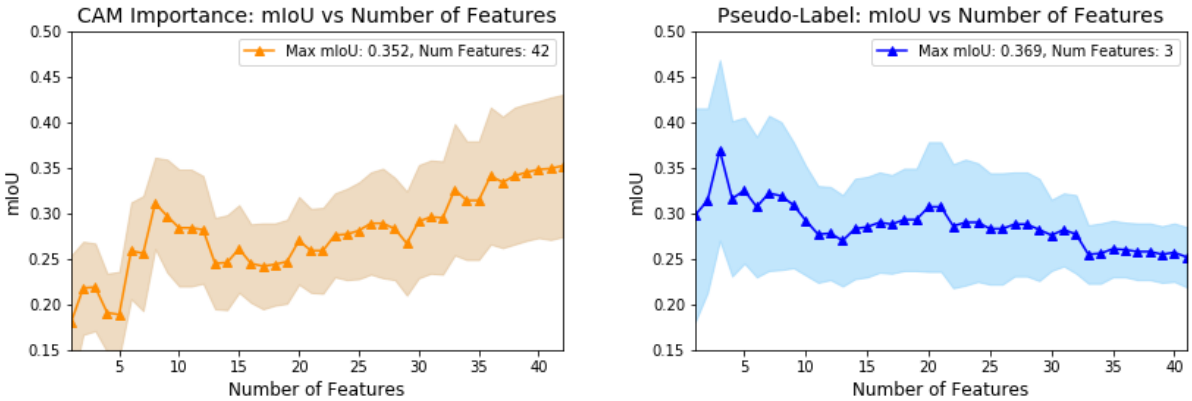



Figure 4-5. Examples of LayerCAM at various stages of a trained VGG16 network for the "target" class. Layer 1 corresponds to CAMs computed from the initial convolutional block while Layer 5 represents CAMs from the deepest convolutional layers. Earlier layers seemingly capture fine-grained information while later layers capture class-specific, localization information.




(A) (B)

Figure 4-6. Segmentation performance as a function of number of training features. The solid line shows the mean IoU on the test set and the shaded region shows one standard deviation. A) CAM importance fitness demonstrates a top mIoU of $0.352$ at $k = 42$ features. The optimal number of features for B) mIoU fitness is $k = 3$, which gave an average IoU of $0.369$.

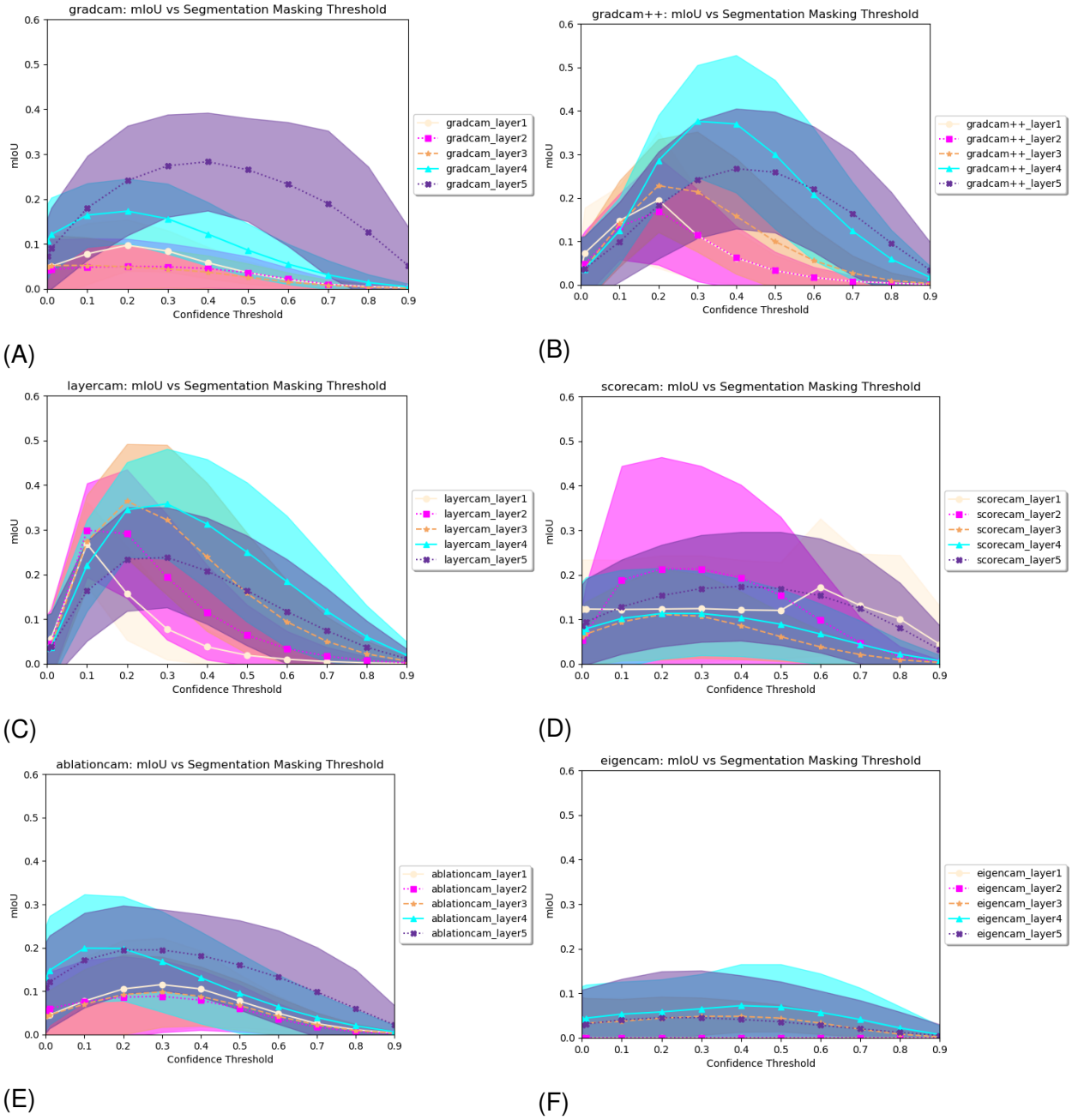



Figure 4-7. DSIAC CAM segmentation by method. Results show mIoU versus binarization threshold for A) Grad-CAM, B) Grad-CAM++, C) Layer-CAM, D) Score-CAM, E) Ablation-CAM, and F) Eigen-CAM.

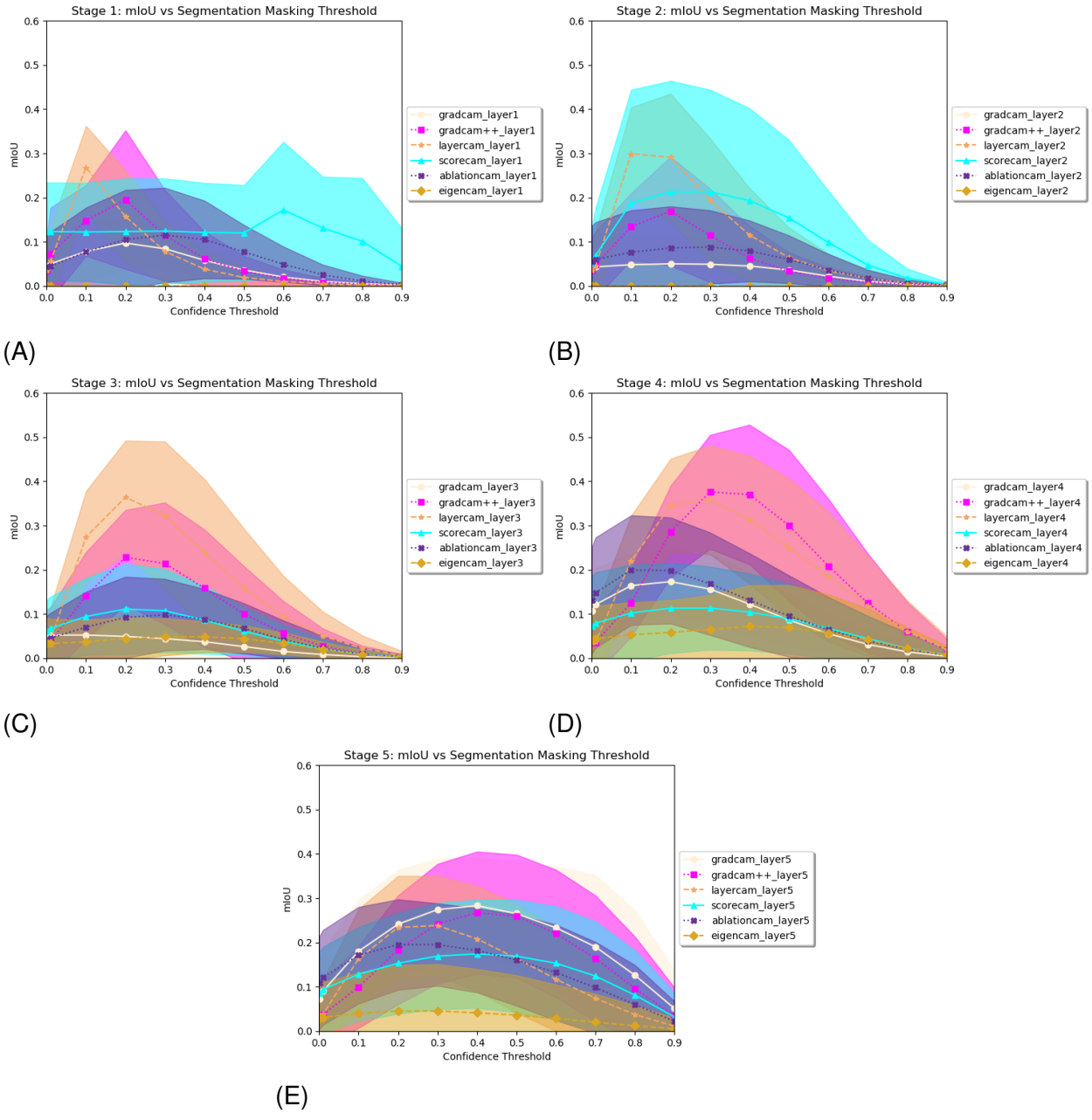



Figure 4-8. DSIAC CAM segmentation by stage. Results show mIoU versus binarization threshold for A) stage 1, B) stage 2, C) stage 3, D) stage 4, and E) stage 5.

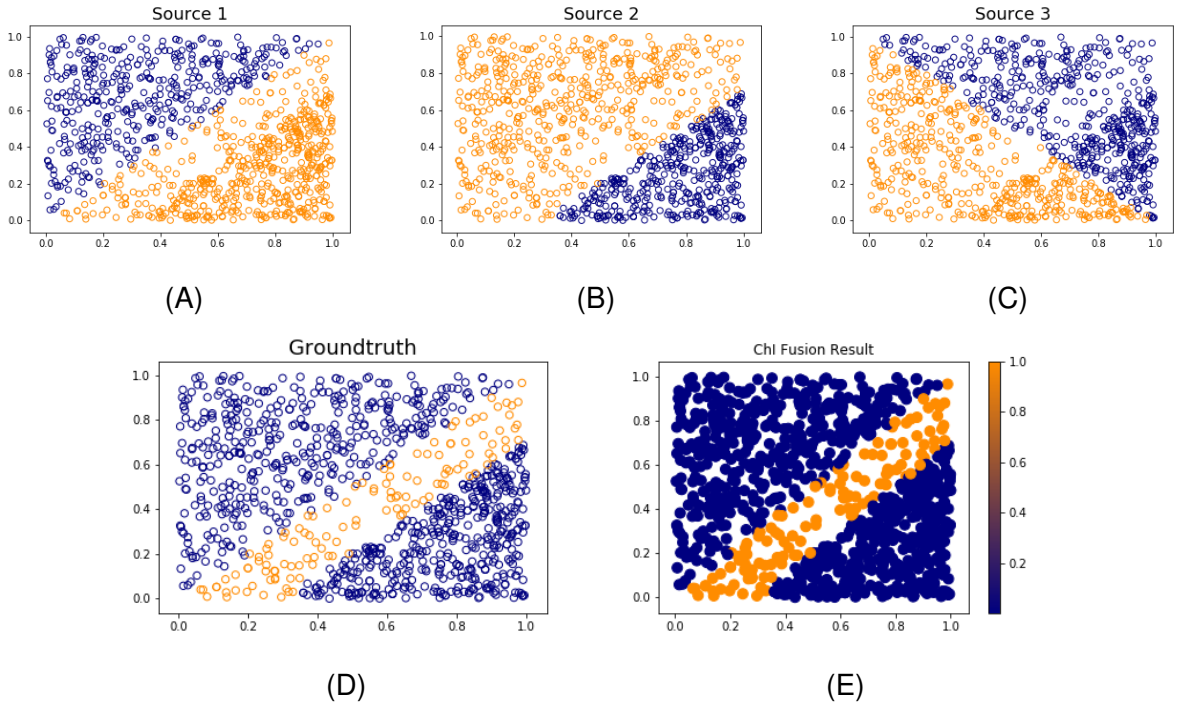



Figure 4-9. Synthetic 3-source dataset and results for the MICI classifier fusion model. Images represent A) source 1, B) source 2, C) source 3, D) true labels, and E) MICI fusion result. The 1000 data points are scatter plotted in 2-D for visualization. Orange represents a classification label of "1" while blue denotes a label of "0". The MICI classifier output provides a value in $[0, 1]$ which can be thought of as representing confidence over the positive class.

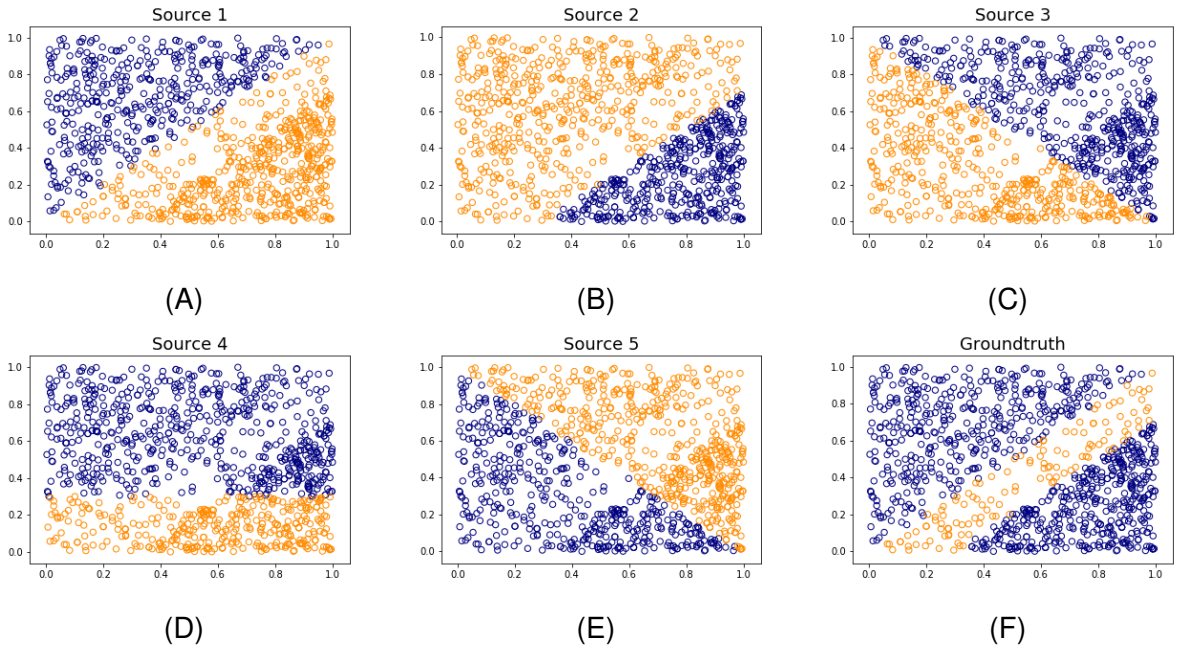



Figure 4-10. Synthetic 5-source dataset and corresponding groundtruth. The images represent A) source 1, B) source 2, C) source 3, D) source 4, E) source 5, and F) the true labels. The 1000 data points are scatter plotted in 2-D for visualization. Orange represents a classification label of “1” while blue denotes a label of “0”. Only sources 1 and 2 are needed to match the true labels.

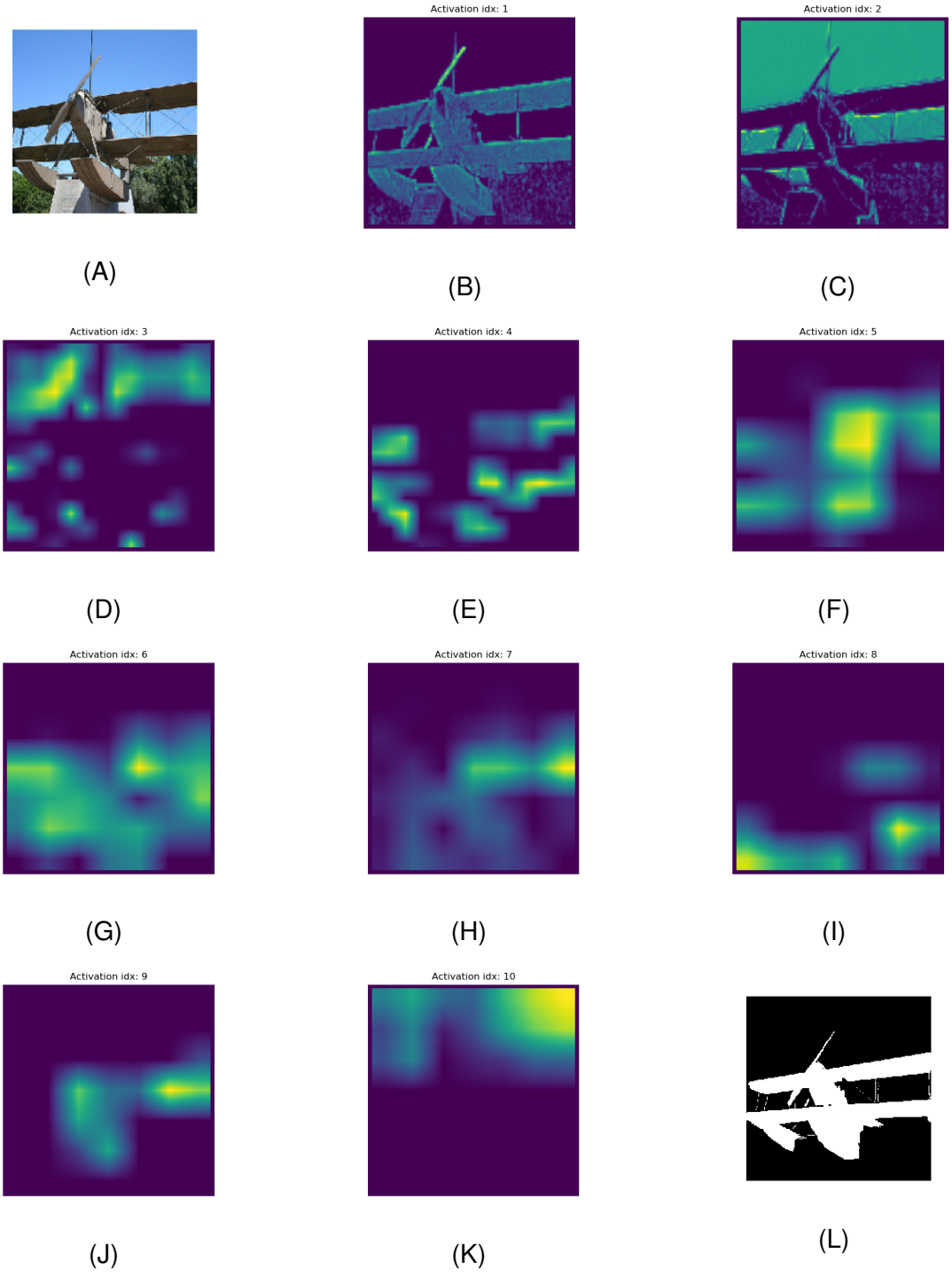
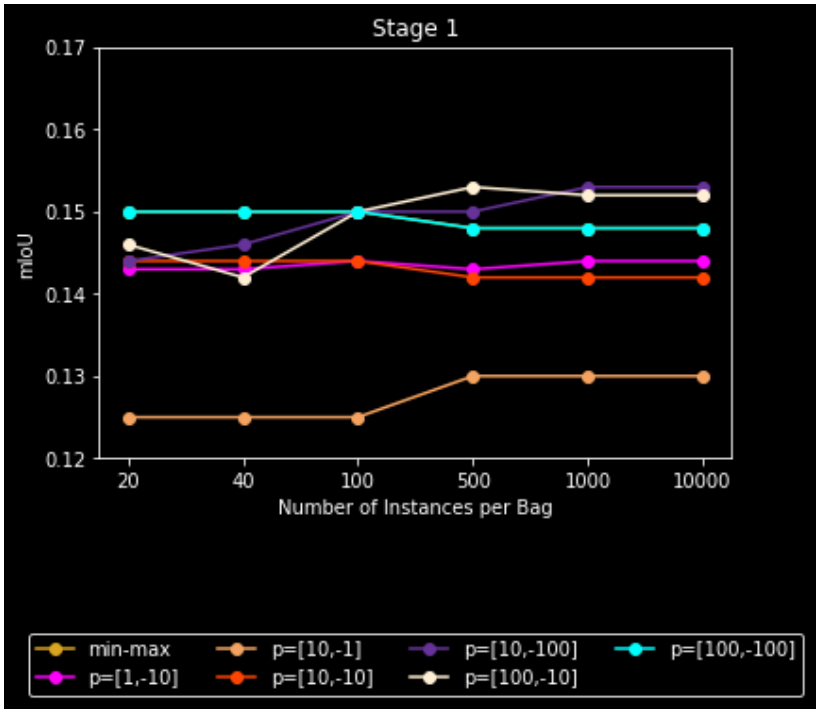
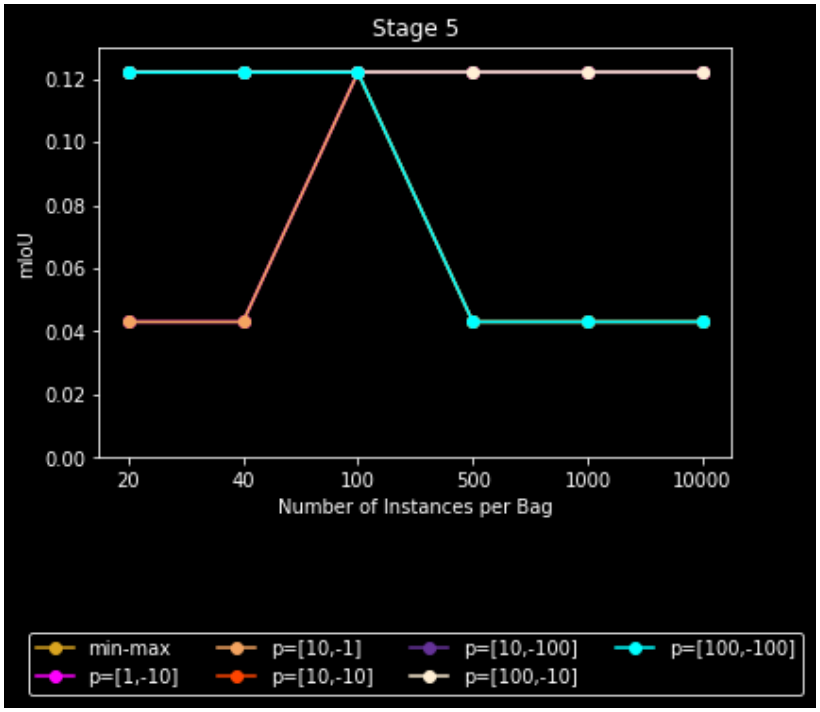



Figure 4-11. Hand-picked aeroplane activations and corresponding groundtruth. The images represent A) the input, B) source 1, C) source 2, D) source 3, E) source 4, F) source 5, G) source 6, H) source 7, I) source 8, J) source 9, K) source 10, and L) the true segmentation mask where white denotes target and black represents background. Sources 1 and 2 are from the first stage of the VGG16 model, sources 3 and 4 are from stage 4, and the rest are from stage 5.




(A)




(B)

Figure 4-12. Generalized-mean parameters versus number of instances per bag for A) stage 1 and B) stage 5. Performance is measured as mean intersection-over-union (mIoU).

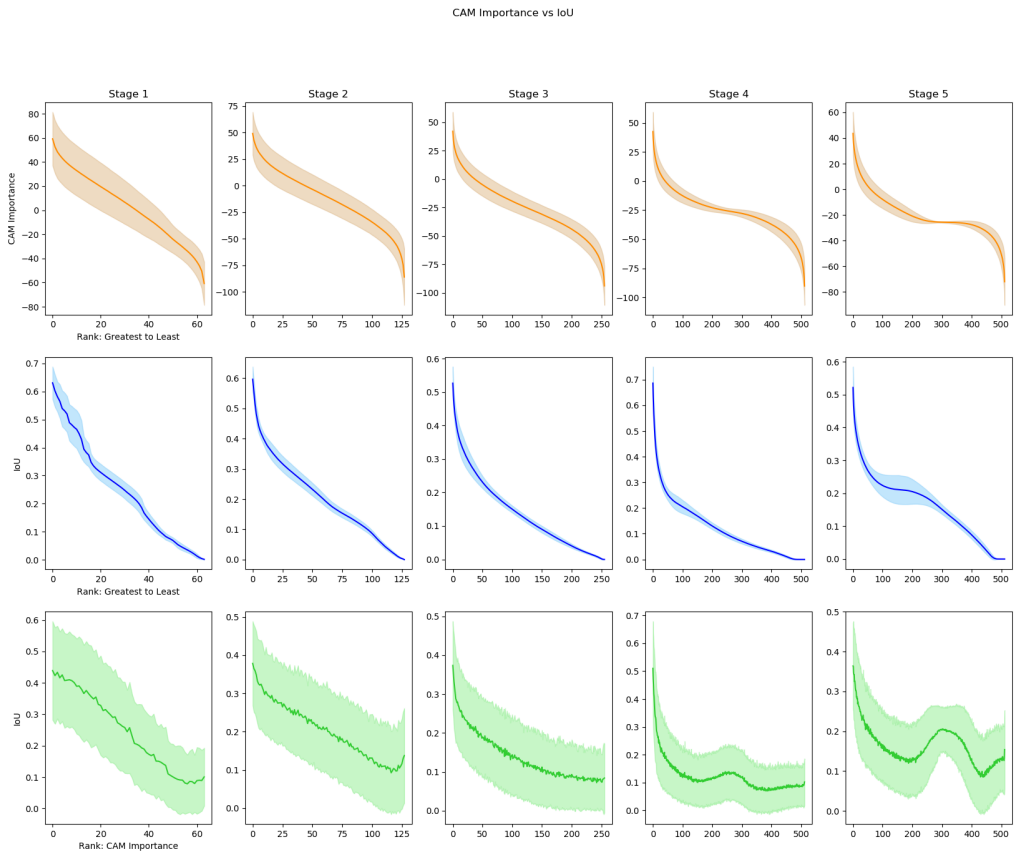



Figure 4-13. Average Grad-CAM importance versus IoU on the MNIST 7 class. Columns denote VGG16 stages 1-5. Orange plots show average Grad-CAM importance of each activation map on the vertical axis ordered from greatest to least. Blue plots show the average IoU of each feature map ordered from greatest to least. Green plots show the average IoU ordered according to descending Grad-CAM importance. Solid lines show the mean trend, while shaded regions denote standard deviation. Vertical axes are not on the same scales.

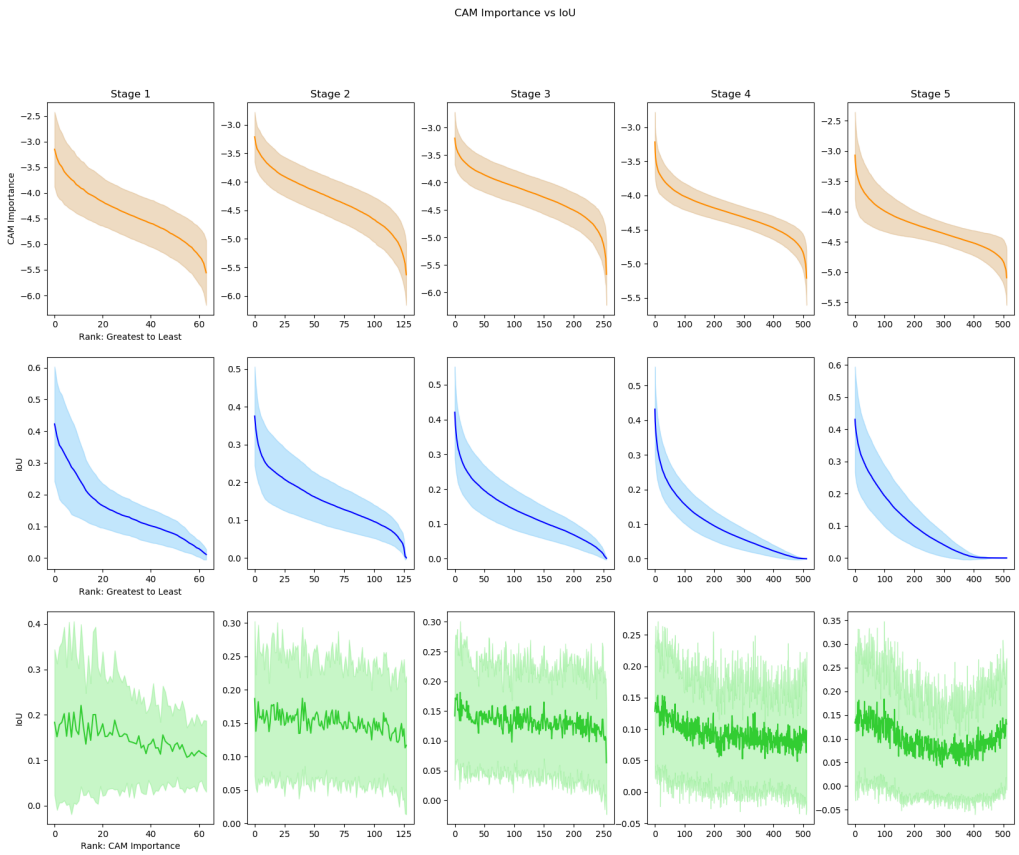
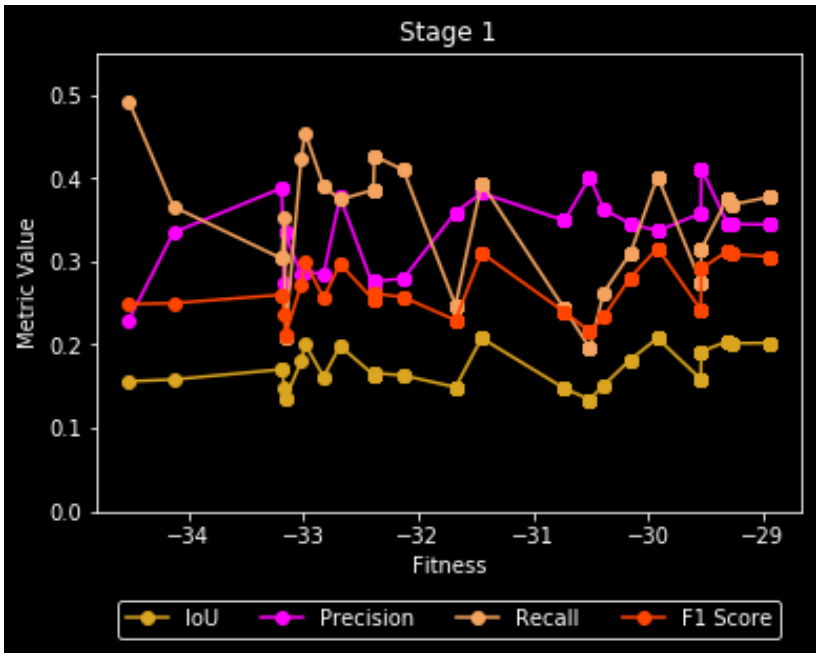
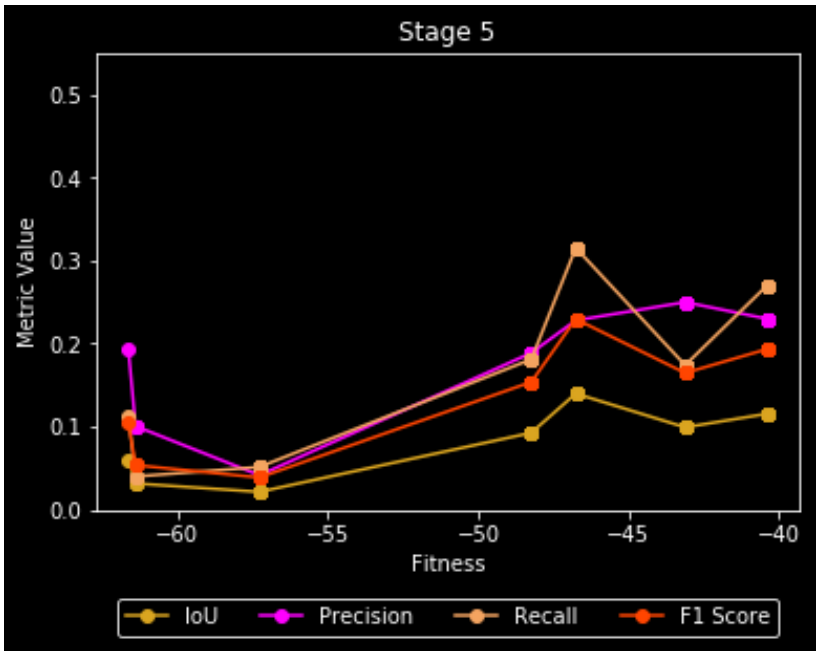



Figure 4-14. Average Grad-CAM importance versus IoU on the PASCAL VOC Aeroplane class. Columns denote VGG16 stages 1-5. Orange plots show average Grad-CAM importance of each activation map on the vertical axis ordered from greatest to least. Blue plots show the average IoU of each feature map ordered from greatest to least. Green plots show the average IoU ordered according to descending Grad-CAM importance. Solid lines show the mean trend, while shaded regions denote standard deviation. Vertical axes are not on the same scales.




(A)




(B)

Figure 4-15. IoU, precision, recall, and F1 score against fitness for A) stage 1 and B) stage 5. A value of 0 is the best for fitness and a value of 1 is best for the other metrics.

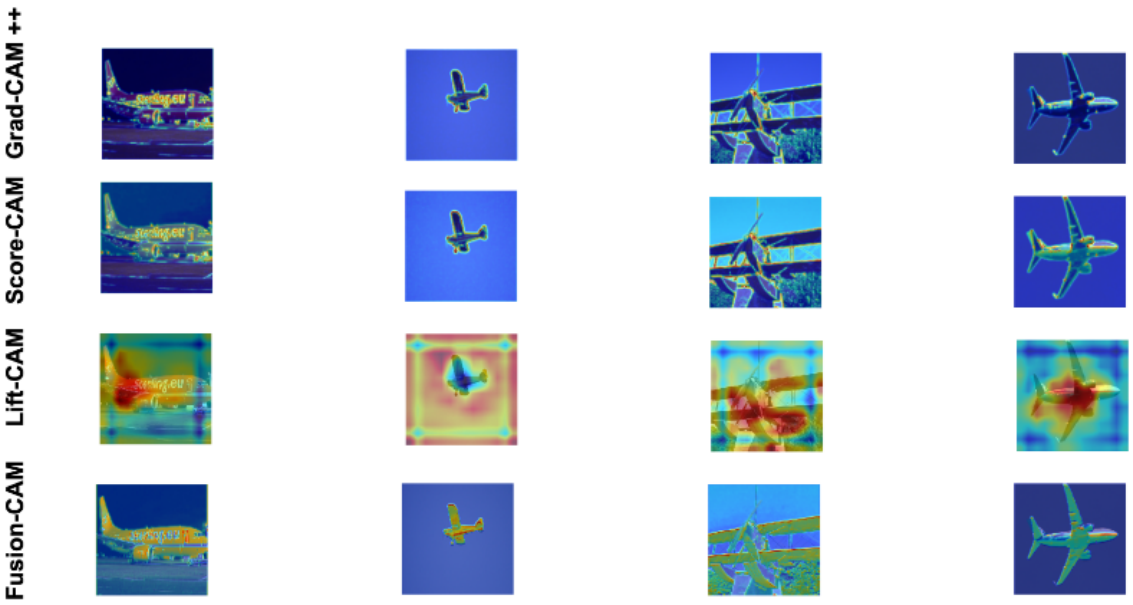



Figure 4-16. Qualitative comparisons for stage 1. Rows show example heatmaps for Grad-CAM++, Score-CAM, Lift-CAM, and Fusion-CAM (ours).

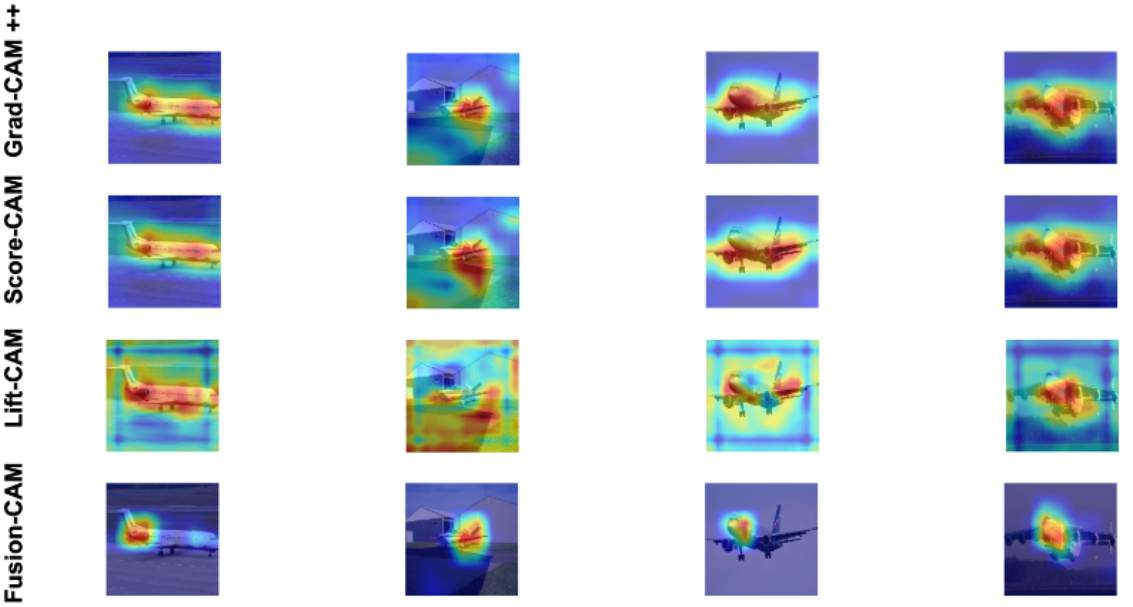



Figure 4-17. Qualitative comparisons for stage 5. Rows show example heatmaps for Grad-CAM++, Score-CAM, Lift-CAM, and Fusion-CAM (ours).

# CHAPTER 5
# CONCLUSION

## 5.1 Summary of Work

This dissertations explores the Multiple Instance Choquet Integral (MICI) [5] as a method for discriminative feature selection and fusion. Specifically, MICIs with binary and regular fuzzy measures were investigated as methods to select discriminative instance-level features. The explored approaches learn solely from imprecisely annotated data. Experimental results on both synthetic and real-world remote sensing datasets indicate the utility of the MICI for discriminative source down-selection. Target detection by MICI fusion demonstrated competitive performance compared to related methods in the literature. This research provided the following contributions:

1. Initial experimentation demonstrated the utility of CAM methods in producing pseudo-labels for learning instance-level classifiers from data with imprecise annotations. It was shown that features down-selected from a CNN must consider a variety of information to be informative in instance-level target detection, including but not limited to: coverage on the target, importance toward the inference decision, and information about the background.

2. The MICI was explored as a mechanism to down-select sources which are informative in instance-level discrimination tasks.

3. A new data-supported search for learning the binary fuzzy measure was proposed.

4. Experiments on synthetic and real-world datasets demonstrated the utility of the MICI in defining accurate confidence maps for target detection at the instance-level. Choquet integral fusion of down-selected sources provided competitive detection performance compared to current class activation mapping approaches in the literature.

## 5.2 Future Work

The biggest bottleneck in using the MICI for pseudo-label generation is the exponential growth of the measure with the addition of sources. In future work, our method might benefit from exploring alternative measures or integrals, such as linear order statistics [208] or the Sugeno lambda measure [166], which can scale better with additional sources. Additionally, performance might be improved by combining multiple approaches. For example, the ChI could be used to fuse multiple CAM outputs or to

combine CAM heatmaps with individual activation feature maps. Other insights might be provided by exploring alternative datasets or by adjusting the learning objective to explicitly consider spatial information.

# APPENDIX
# ADDITIONAL LITERATURE REVIEW

This appendix provides additional literature which is relevant to the technical approach of this work. Topics include manifold learning and metric embedding.

## A.1 Manifold Learning

Real-world remote sensing data such as hyperspectral imagery, ground-penetrating radar scans and sonar signals are naturally represented by high-dimensional feature vectors. However, in order to handle such real-world data adequately, its dimensionality usually needs to be reduced [1, 209]. The problem considered in this work is discovering feature representations that promote class discriminability for target or anomaly detection. This is typically achieved in a few different ways. First, features can be "engineered" by leveraging known qualities about the data statistics, sensors or collection environments. Alternatively, poor features can be projected into a high-dimensional space (such as a Kernel Hilbert Space) using a kernel function. A popular approach is to extract discriminative features from systems which learn data representations in an end-to-end fashion, such as artificial neural networks [210, 211, 212]. Finally, this work focuses on methods which transform the data into a new (often lower-dimensional) coordinate system which optimizes feature representations for discrimination [84].

The application of dimensionality reduction (DR) has proven useful in myriad applications in the literature, such as: visualization of high-dimensional data, classification, redundancy removal, compression and data management, improving computational tractability and efficiency, and reducing the effects of the Curse of Dimensionality [19, 20, 21, 22, 3, 23, 24, 25, 2]. In classification of object entities, it is often assumed that classes can be described by an intrinsic subset of representative features which describe the factors of variation in a set of data [213]. These structures are called intrinsic manifolds, and they represent the generating distributions of class objects exactly by the number of degrees of freedom in a dataset [4, 209]. Consider the example shown in Figure A-1. This figure shows samples from the LFW Faces in the Wild dataset [214]. While each individual image is represented by a vector of features

(pixel intensities in this case) in $\mathbb{R}^{1850}$, the dataset only exhibits a few degrees of freedom, such as: illumination, face direction, facial expression and whether or not the subject is wearing glasses. Thus, it is intuitive that the dataset lies on a smooth, intrinsic submanifold spanning approximately four dimensions which inherently capture the degrees of freedom in the data.

The goal of manifold learning is then to discover embedding functions which take data from the input feature space and transform it into a lower-dimensional coordinate system (also called a latent space in the literature) which captures the "useful" properties of the data, while enforcing constraints such as smoothness (the transformation function should not produce sporadic images), continuity (no discontinuous points on the hyper-surface), topological ordering (neighbors in the input space should also be neighbors in the embedded space ) or class separability (samples from the same class should fall metrically close to each other in the embedded space and disparate classes should be distinctly far) [84].

This dissertation focuses on investigating the use of manifold learning to increase instance discriminability in the latent space, where labels are solely provided at the bag-level. While there is an expansive literature in unsupervised manifold learning methods, this document will pay special attention to both strictly- and semi-supervised methods, since they are typically adaptations of unsupervised approaches, as well as manifold learning under the MIL framework.

### A.1.1 Definition and General Notation

Most studies perform classification or regression after applying unsupervised dimensionality reduction. However, it has been shown that there are advantages of learning the low-dimensional representations and classification/regression models simultaneously [83, 215]. Considering classification as the main goal of dimensionality reduction, this section provides a summary of the current literature in the area.

Given a data matrix $\boldsymbol{X} = [\boldsymbol{x}_1^T, \boldsymbol{x}_2^T, \ldots, \boldsymbol{x}_N^T] \in \mathbb{R}^{N \times D}$ where $N$ is the total number of samples and $D$ is the dimensionality of the input feature space, general dimensionality reduction seeks to find a representation $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ with $d \ll D$ that enhances the between-class separation while preserving the intrinsic geometric structure of the data[84]. In other words, it is assumed that the data lie on a smooth manifold $\mathcal{X}$, which is the image of some parameter domain $\mathcal{Z} \subset \mathbb{R}^d$ under a smooth mapping $\Psi : \mathcal{Z} \to \mathbb{R}^D$. The goal of manifold learning is to discover an inverse mapping to the low-dimensional pre-image coordinates $\boldsymbol{z}_n \in \boldsymbol{z}$ corresponding to points $\boldsymbol{x}_n \in \boldsymbol{X}$. The data matrices $\boldsymbol{X} = [\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_N^T]$ and $\boldsymbol{Z} = [\boldsymbol{z}_1^T, \ldots, \boldsymbol{z}_N^T]$ are of size $N \times D$ and $N \times d$, respectively. Since these low-dimensional data representations are unknown, they are often referred to as latent vectors and the span in $\mathbb{R}^d$ is sometimes called the latent feature space or latent space for brevity [47]. The primary difference between traditional, unsupervised manifold learning and supervised approaches is that, in supervised manifold learning, data matrix $\boldsymbol{X}$ is accompanied with a corresponding label vector $\boldsymbol{l} = [l_1, \ldots, l_N]$ indicating the corresponding class labels of each sample in $\boldsymbol{X}$.

Manifold learning methods can be subdivided into a wide taxonomy of approaches, with linear and nonlinear at the root. Nonlinear approaches can be further divided into purely global methods and approaches that capture global structure solely from local information. We begin with a review of popular linear manifold learning techniques before moving into the realm of nonlinear approaches. Base, unsupervised methods are reviewed along with corresponding supervised and semi-supervised adaptations.

### A.1.2 Linear Manifold Learning

A review of linear manifold learning approaches is provided. Linear approaches are advantageous over many nonlinear techniques because they inherently allow for out-of-sample extensions. In other words, linear transformation matrices are learned which can be easily applied on data not included in the training set. (Although there are nonlinear approaches for which this is true in locally linear neighborhoods [216].)

However, linear approaches are limited in their abilities to capture irregular data surfaces [25]. Principal Component Analysis (PCA), Multi-dimensional Scaling (MDS), Nonnegative Matrix Factorization (NMF) and Fisher's Linear Discriminant Analysis (LDA) are reviewed. General approaches are discussed and supervised as well as nonlinear extensions are elaborated. Special focus is given to (LDA), as it is the only inherently supervised technique out of the included approaches.

#### A.1.2.1 Principal Component Analysis (PCA)

This section describes Principal Component Analysis and related methods.

**Unsupervised PCA** Principal Component Analysis (PCA) is arguably the most popular (and best-studied) technique for dimensionality reduction and manifold learning. It attempts to learn an orthogonal projection of the input data into a lower-dimensional space, known as the principal subspace, such that the variance of the projected data is maximized [83]. In other words, each principal axis, or principal component, of the learned coordinate system is orthogonal to the other principal components. In summary, the problem of PCA is to discover basis vectors which linearly combine to reconstruct the data. In practice, data in the input feature space are projected into a new coordinate system of $d$ dimensions, such that the variance along each principal axis is maximized and the reconstruction errors of the data are minimized in the mean-square sense [4]. Let $V$ be a $d$-dimensional subspace of $\mathbb{R}^D$ and let $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_D$ be an orthonormal basis of $\mathbb{R}^D$ such that $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d$ is a basis of $V$. The goal of PCA is to find an orthogonal set of basis vectors $\boldsymbol{w}_n \in \mathbb{R}^D$ and corresponding latent coordinates $\boldsymbol{z}_n \in \mathbb{R}^d$ such that the average reconstruction error is minimized [47]

$$J(\boldsymbol{W}, \boldsymbol{Z}) = \frac{1}{N} \sum_{n=1}^{N} ||\boldsymbol{x}_n - \hat{\boldsymbol{x}}_n||^2 \tag{A-1}$$

where $\hat{\boldsymbol{x}}_n = \boldsymbol{W}\boldsymbol{z}_n$, subject to the constraint that $\boldsymbol{W}$ is orthonormal, or that

$\boldsymbol{w}_i^T \boldsymbol{w}_j = 0, \forall i \neq j$ and $\boldsymbol{w}_i^T \boldsymbol{w}_i = 1$. This is equivalently written as

$$J(\boldsymbol{W}, \boldsymbol{Z}) = ||\boldsymbol{X} - \boldsymbol{W}\boldsymbol{Z}||_F^2 \tag{A-2}$$

where $\boldsymbol{Z}$ is a $N \times d$ matrix with the $\boldsymbol{z}_n$ in its rows and $||\boldsymbol{A}||_F$ is the Frobenius norm of matrix $\boldsymbol{A}$, defined by

$$||\boldsymbol{A}||_F = \sqrt{\sum_{m=1}^{M}\sum_{n=1}^{N} a_{mn}^2} = \sqrt{tr(\boldsymbol{A}^T\boldsymbol{A})} = ||\boldsymbol{A}(:)||_2 \tag{A-3}$$

The optimal solution is obtained by setting $\hat{\boldsymbol{W}} = \boldsymbol{U}_d$, where $\boldsymbol{U}_d$ contains the eigenvectors corresponding to the $d$ largest eigenvalues of the mean-subtracted, empirical data covariance matrix, $\hat{\boldsymbol{S}} = \frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}})^T$, where $\hat{\boldsymbol{\mu}}$ is the empirical data mean. Therefore, the low-dimensional encoding of the data is given by $\boldsymbol{z}_n = \hat{\boldsymbol{W}}^T\boldsymbol{x}_n$, which is the orthogonal, linear projection of the data onto the column space spanned by the eigenvectors of the $d$ largest eigenvalues of the empirical data covariance.

The example shown in Figure A-2 demonstrates the projection of 2-dimensional data onto the first principal axis. As can be seen from the figure, the first principal axis corresponds to the direction of maximal variance of the data. PCA from the viewpoint of variance maximization is often called the analysis view of PCA [47]. PCA has been successfully applied to a large number of domains such as face recognition, coin classification and seismic series analysis [1]. It's success is partially because of its convenience of use. Embedding out-of-sample points with PCA is simple since the transformation is just a rotation and scaling. However PCA suffers from a few drawbacks. First, the dimensionality of the covariance matrix is proportional to the dimensionality of the data points. As a result, the computation of the eigenvectors may be infeasible or untrustworthy (singular) for high-dimensional data. Additionally, PCA focuses mainly on preserving large pairwise distances between data samples instead of retaining local relationships, which may be important in certain applications. PCA also assumes Gaussian distributed data, which is unlikely in real-world applications. Finally, PCA is

sensitive to feature magnitude. It is typical to standardized data before applying PCA as it can be misled by directions in which the variance is high simply because of the measurement scale.

Many extensions have been made to PCA, including the development of nonlinear versions (Sections A.1.3.2and A.1.4.1) [217, 218] and formulating it as a factor analysis problem (Section A.1.5.1) [219].

**Independent and Canoncial Component Analyses** It is also worth mentioning two popular approaches closely related to PCA, namely, Independent Component Analysis (ICA) and Canonical Component Analysis (CCA). ICA attempts to solve the blind-source separation problem, in which the goal is to deconvolve mixed signals into their constituent parts [47, 220, 221]. Instead of discovering the directions of maximum variance as done with PCA, ICA attempts to uncover the directions such that the data projected onto these directions have maximum statistical independence. On the other hand, CCA jointly considers multiple variables (multiple feature spaces) and tries to discover the correlations between them. CCA looks for directions in each feature space such that the data projected onto the direction found in each space has the maximum possible correlation [222]. Thus, CCA can be used to simultaneously reduce the dimensionality of data in multiple feature spaces [47].

### A.1.2.2 Multi-dimensional Scaling (MDS)

This section describes Multi-dimensional Scaling and related methods.

**Unsupervised MDS** Most modern manifold learners have theoretical and algorithmic roots in one of three basic dimensionality reduction techniques: PCA, K-means and Multidimensional Scaling (MDS) [223]. Whereas PCA looks for linear projection bases which are constructed from the eigenvectors of a data covariance or scatter matrix, MDS tries to find a linear projection that preserves pairwise distances as well as possible. This idea is demonstrated by Figure A-3, where the pairwise distances between samples in the 3-dimensional space are preserved in the 2-dimensional

embedding space. While MDS does not construct an embedded manifold explicitly, it holds the status of being one of the first "one-shot" (non-iterative) manifold learners, such as Isomap and Locally Linear Embedding (LLE), which are discussed later in this literature review [25]. The steps of MDS correspond exactly to those of PCA except that, instead of a scatter matrix $\boldsymbol{S} = \frac{1}{N}\boldsymbol{X}\boldsymbol{X}^T$, MDS operates with a positive semi-definite, dissimilarity matrix $\boldsymbol{D} \in \mathbb{R}^{N \times N}$, where $N$ is the number of data samples and a real, symmetric Gram matrix $\boldsymbol{K} = \boldsymbol{X}^T\boldsymbol{X}$ (inner-product matrix) where $\boldsymbol{K}_{mn}$ is the inner product between $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$. The problem of MDS is posed as finding $d$-dimensional Euclidean coordinates for each sample $\boldsymbol{x}_n$ in dataset $\boldsymbol{X}$ such that the Euclidean distances in the low-dimensional embedding space are proportional to the pairwise distances in the input space [4, 224]. While the literature poses several cost functions for this task, this review focuses on classical MDS, which is described as follows:

First, the pairwise distance matrix $\boldsymbol{D}$ is computed such that

$$\boldsymbol{D}_{mn} = \mathcal{D}_{\mathcal{X}}(\boldsymbol{x}_m, \boldsymbol{x}_n) = ||\boldsymbol{x}_m - \boldsymbol{x}_n||^2 = (\boldsymbol{x}_m - \boldsymbol{x}_n)^T(\boldsymbol{x}_m - \boldsymbol{x}_n) \tag{A-4}$$

where $\mathcal{D}_{\mathcal{X}}(\cdot, \cdot)$ is a chosen dissimilarity metric (Euclidean distance for classical MDS). Then, the double-centered Gram matrix $\boldsymbol{K}$ is computed by

$$\boldsymbol{K} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H} \tag{A-5}$$

where

$$\boldsymbol{H} = \mathbb{I}_N - \frac{1}{N}\boldsymbol{e}\boldsymbol{e}^T \tag{A-6}$$

with $\mathbb{I}_N$ denoting the $N \times N$ identity matrix and $\boldsymbol{e} = (1, \ldots, 1)^T$ the $N \times 1$ column vector of all ones. Multiplying $\boldsymbol{D}$ on both sides by $\boldsymbol{H}$ performs double centering, which subtracts the row and column means from $\boldsymbol{D}$ (and adds back the global mean which gets subtracted twice), so that both the row and column means of $\boldsymbol{K}$ are equal to zero. Double centering is necessary to remove the translational freedom from the

low-dimensional embedding coordinate representations. Without it, the embedding points would be arbitrary to a translational degree of freedom. Moreover, the mean removal ensures that the solution has a meaningful interpretation. With the pairwise distances centered around zero, the largest eigenvectors of the kernel matrix (Equation A-8) correspond to the directions of maximum variance between the pairwise similarities. This ensures that global distance relations are preserved through the projection to the low-dimensional space.

The goal of MDS is to find $\boldsymbol{Z} = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_N\} \in \mathbb{R}^d$ which minimizes the objective

$$J(\boldsymbol{D_X}, \boldsymbol{D_Z}) = ||\boldsymbol{K_X} - \boldsymbol{D_Z}||^2 = \left|\left| -\frac{1}{2}\boldsymbol{H}(\boldsymbol{D_X} - \boldsymbol{D_Z})\boldsymbol{H} \right|\right|^2 \tag{A-7}$$

Similarly to PCA, this can be solved by a generalized eigenvalue problem

$$\boldsymbol{K}v = \lambda v \tag{A-8}$$

such that

$$\boldsymbol{Z} = \boldsymbol{V}\boldsymbol{\Lambda}^{\frac{1}{2}} \tag{A-9}$$

with $\boldsymbol{\Lambda}^{\frac{1}{2}} = diag(\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_N})$ being a diagonal matrix with entries equal to the square roots of the eigenvalues of $\boldsymbol{K}$ sorted from largest to smallest ($\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$), and $\boldsymbol{V} = \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N\}$ the corresponding eigenvectors. The low-dimensional embedding coordinates $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ are obtained by $\boldsymbol{Z} = \{\sqrt{\lambda_1}\boldsymbol{v_1}, \ldots, \sqrt{\lambda_d}\boldsymbol{v_d}\}$ [83]. Pseudo-code for MDS is provided in Algorithm A-14.

**Algorithm A-14** MDS

**Input:** Distance matrix $\boldsymbol{D} \in \mathbb{R}^{N \times N}$, embedding space dimensionality $d$
**Output:** Low-dimensional data representations $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$

1: Calculate $\boldsymbol{K} = -\frac{1}{2}\boldsymbol{H}\boldsymbol{D}\boldsymbol{H}$, where $\boldsymbol{H} = \mathbb{I}_N - \frac{1}{N}\boldsymbol{e}\boldsymbol{e}^T$ is the centering matrix
2: Compute eigenvectors and eigenvalues $\boldsymbol{K}v = \lambda v$
3: $\boldsymbol{V}', \boldsymbol{\Lambda}' \leftarrow$ SORTDECREASING$(\boldsymbol{V}, \boldsymbol{\Lambda})$
4: $\boldsymbol{Z} \leftarrow \boldsymbol{V}'\boldsymbol{\Lambda}'^{\frac{1}{2}}$

It has been proven that the eigenvalues of Gram matrix $\boldsymbol{K}$ and covariance $\boldsymbol{S}$ are the same, and that the space spanned by MDS and PCA are the identical for any $d \leq rank(\boldsymbol{K}) = rank(\boldsymbol{S})$. This implies that a rotation matrix $\boldsymbol{A}$ could be found such that $\boldsymbol{A}^T \boldsymbol{Z}_{MDS} = \boldsymbol{Z}_{PCA}$ [224]. Additionally, MDS can be computed even if the data observation matrix $\boldsymbol{X}$ is unknown. All that is needed is the Gram matrix or a dissimilarity matrix. This feature potentially allows MDS to be applied in a variety of data-sensitive and privacy-concerned scenarios.

A pitfall of MDS is that it focuses on retaining global pairwise distances as opposed to local distances, which are typically much more important for capturing the geometry of the data [1]. Several MDS variants have been proposed to address this weakness. A popular variant is known as Sammon Mapping and is discussed in Section A.1.3.5.

**Supervised MDS** As with most manifold learning methods in the literature, MDS does not inherently consider class information when learning the embedding function. In attempt to promote class separability in the low-dimensional embedding space, Witten and Tibshirani [225] proposed Supervised Multidimensional Scaling (SMDS). this method follows the idea of traditional MDS where the goal is to find low-dimensional coordinate or configuration points $\boldsymbol{z}_n \in \mathbb{R}^d$, such that pairwise distances in the input feature space are preserved in the embedding space. Incorporating class label information, the goal of SMDS is to not only preserve distances, but ensure the coordinate values $z_{mk} > z_{nk}$ when $l_m > l_n, \quad \forall k = 1, \ldots, d$, where $l$ are the instance-level labels and $d$ is the dimensionality of the embedding space. Considering the binary target classification case, SMDS can be formulated as

$$\min_{\boldsymbol{Z}} \quad \frac{1}{2}(1-\alpha)\sum_{m=1}^{N}\sum_{n=1}^{N}(\boldsymbol{D}_{mn} - ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2) + \alpha \sum_{m:l_m=1}\sum_{n:l_n=2}\sum_{k=1}^{d}\left(\frac{\boldsymbol{D_{mn}}}{\sqrt{d}} - (z_{nk} - z_{mk})^2\right) \tag{A-10}$$

This objective has two terms. The first is the traditional metric MDS stress. This term attempts to ensure that the Euclidean distances of two points in the embedding space is

the same as the dissimilarity between the points in the input feature space. The second term is the supervised term which enforces that each dimension of the embedded configuration points be larger if belonging to the class with the larger label, and smaller if belonging to the class with a smaller-valued label. The term $\alpha \in [0, 1]$ is a tuning parameter. When $\alpha = 0$, the objective reduces to the MDS stress function. As $\alpha$ increases, however, the objective becomes increasingly more supervised, focused on ensuring class separation of the training data.

A least square regression was applied to estimate the embedding function for out-of-sample test points. SMDS was successfully applied to tasks in data visualization, bipartite ranking and classification of prostate data and USPS handwritten digits.

#### A.1.2.3 Non-negative Matrix Factorization (NMF)

Non-negative Matrix Factorization (NMF) is a tool for linear dimensionality reduction that, given a set of data $\boldsymbol{X} \in \mathbb{R}^{N \times D}$, aims to decompose the matrix into a coefficient matrix $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ and basis matrix $\boldsymbol{W} \in \mathbb{R}^{d \times D}$ [226]. In this case, the columns of matrix $\boldsymbol{W}$ are basis elements and the columns of matrix $\boldsymbol{Z}$ give the coordinates of data samples in the basis $\boldsymbol{W}$. Similar to PCA, the goal of NMF is to find a $\boldsymbol{Z}$ and $\boldsymbol{W}$ such that $\hat{\boldsymbol{X}} = \boldsymbol{ZW}$ approximates the data as close as possible, given that every element must be non-negative. This notion is formalized by

$$\min_{\boldsymbol{Z},\boldsymbol{W}} ||\boldsymbol{X} - \boldsymbol{ZW}||_F^2 \quad s.t. \quad \boldsymbol{Z} \geq 0, \boldsymbol{W} \geq 0 \tag{A-11}$$

Another popular variant of NMF is to substitute the Frobenius norm for the Kullback-Leibler (KL) divergence, where $D(\boldsymbol{X}||\hat{\boldsymbol{X}}) = \sum_{m,n}(\boldsymbol{X}_{mn} \log \frac{\boldsymbol{X}_{mn}}{\hat{\boldsymbol{X}}_{mn}} + \boldsymbol{X}_{mn} - \hat{\boldsymbol{X}}_{mn})$. Given that $d \ll D$, it is intuitive that $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ provides the desired low-dimensional representations of high-dimensional data $\boldsymbol{X}$. Many approaches have been used to solve for the non-negative matrices, including alternating least squares, projected gradient descent, coordinate descent and the Alternating Direction Method of Multipliers (ADMM) [83]. NMF has been successfully applied to applications in image processing, text

mining, hyperspectral imaging, air emission control, computational biology, blind source separation, single-channel source separation, clustering, music analysis and collaborative filtering [226].

As described by Chao et al. [83], two groups of supervised NMF have been proposed in the literature according to the way label information is utilized. In direct supervised NMF approaches, label information is incorporated directly into the loss function to promote learning of well-separated coordinate representations for samples in different classes. Approaches taken under this framework include incorporating simple indicator variables to denote the class of a sample, integrating NMF with a SVM and formulating the problem under task-driven dictionary learning. An alternative approach to the direct supervised NMF is discriminative NMF. Discriminative NMF approaches are based on Linear Discriminant Analysis (LDA). Essentially, the objective for this class of algorithm is to decompose the data matrix such that the between-class distances of the low-dimensional coordinates $\boldsymbol{Z}$ are maximized, while the within-class distances are minimized. Supervised NMF approaches have been applied successfully to problems in acoustic separation, brain tumor detection and emotion classification.

#### A.1.2.4 Fisher's Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA) is a popular method for supervised, linear dimensionality reduction. LDA currently forms the basis for Multiple Instance Learning dimensionality reduction methods exhibited in the literature [26, 30, 27, 227]. Whereas PCA tries to project data into a space which maximizes variance, LDA considers class label information and tries to find a transformation which both maximizes between-class (inter-class) dissimilarity and minimizes within-class (intra-class) scatter [228, 83, 26, 47]. This is done by maximizing the ratio between the inter-class $\boldsymbol{S}_b$ and intra-class $\boldsymbol{S}_w$ scatter matrices, defined as:

$$\boldsymbol{S}_w = \sum_{k=1}^{K} \boldsymbol{S}_k \tag{A-12}$$

$$\boldsymbol{S}_k = \sum_{n \in C_k} (\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_k)(\boldsymbol{x}_n - \hat{\boldsymbol{\mu}}_k)^T \tag{A-13}$$

$$\boldsymbol{S}_b = \sum_{k=1}^{K} N_k (\hat{\boldsymbol{\mu}}_k - \hat{\boldsymbol{\mu}})(\hat{\boldsymbol{\mu}}_k - \hat{\boldsymbol{\mu}})^T \tag{A-14}$$

Here, $\boldsymbol{S}_w$ is the global within-class scatter matrix which is defined as the sum over each individual class' scatter matrix $\boldsymbol{S}_k$, and $\boldsymbol{S}_k$ is essentially an outer product between all samples belonging to class $C_k$ after subtracting the respective empirical class mean $\hat{\boldsymbol{\mu}}_k$. This scatter matrix would be the class covariance if it was normalized by the number of samples $N_k$ in class $C_k$. However, this normalization constant does not affect the final solution and can thus be ignored. The between-class scatter $\boldsymbol{S}_b$ is defined by the sum of outer products of the differences between the empirical class means $\hat{\boldsymbol{\mu}}_k$ and the global data mean $\hat{\boldsymbol{\mu}}$, weighted by the number of samples in each class. The objective of LDA is then to solve for $\boldsymbol{W}^*$ which maximizes the ratio $J(\boldsymbol{W})$:

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} J(\boldsymbol{W}) = \arg\max_{\boldsymbol{W}} \frac{|\boldsymbol{W}^T \boldsymbol{S}_b \boldsymbol{W}|}{|\boldsymbol{W}^T \boldsymbol{S}_w \boldsymbol{W}|} \tag{A-15}$$

It has been shown that the optimal projection matrix $\boldsymbol{W}^*$ is the one whose columns are the eigenvectors corresponding to the largest eigenvalues of the generalized eigenvalue problem

$$\boldsymbol{S}_b \boldsymbol{w} = \lambda \boldsymbol{S}_w \boldsymbol{w} \Rightarrow \boldsymbol{S}_w^{-1} \boldsymbol{S}_b \boldsymbol{w} = \lambda \boldsymbol{w} \tag{A-16}$$

Since $\boldsymbol{S}_b$ is the sum of $K$ matrices of rank $\leq 1$, this implies that $\boldsymbol{S}_b$ will be of rank $(K-1)$ or less and only $(K-1)$ of the eigenvalues $\lambda$ will be non-zero. A low-dimensional coordinate representation $\boldsymbol{z}_n \in \mathbb{R}^{(K-1)}$ of sample $\boldsymbol{x}_n \in \mathbb{R}^D$ is given by the linear projection of $\boldsymbol{x}_n$ onto the hyper-plane parameterized by $\boldsymbol{W}^*$, $\boldsymbol{z}_n = \boldsymbol{W}^{*T} \boldsymbol{x}_n$ It should be noted that for LDA, the dimensionality of the latent space is not a free-parameter, but is always fixed at $d = (K-1)$, or one less than the number of classes present in the dataset.

Equivalently, LDA can be derived by maximum likelihood for normal class-conditional

densities where the covariances for each class are assumed to be equivalent [47]. For the special case of binary target classification, the LDA transformation will place every sample onto a single line in 1-dimension, and thus the LDA solution can be simplified:

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} \frac{\boldsymbol{w}^T \boldsymbol{S}_b \boldsymbol{w}}{\boldsymbol{w}^T \boldsymbol{S}_w \boldsymbol{w}} = \boldsymbol{S}_w^{-1}(\hat{\boldsymbol{\mu}}_2 - \hat{\boldsymbol{\mu}}_1) \tag{A-17}$$

where $\hat{\boldsymbol{\mu}}_1$ and $\hat{\boldsymbol{\mu}}_2$ are the empirical means for classes $1$ and $2$, respectively. Figure A-4 demonstrates the differences between PCA and LDA. While PCA projects data onto the axes exhibiting the maximal variation, LDA projects the data into a space which attempts to simultaneously enforce between-class separation and within-class compactness.

Although LDA is the basis for large number of discriminative dimensionality reduction approaches, it does not guarantee class separation in the embedding space. For example, LDA projects data into a space of at most $(K-1)$ dimensions, however, more features may be necessary for adequate class discrimination. Additionally, LDA is a parametric method which assumes unimodal Gaussian likelihoods. This implies that it may not be able to preserve complex data structure. Finally, LDA will fail if the discriminatory information is contained in the variance of the data instead of the mean. Despite these pitfalls, LDA has been successfully applied to object detection and recognition tasks [229] Many variations of LDA have been developed, such as Non-parametric LDA [230], Orthonormal LDA [229], Generalized LDA [231] and Multilayer Perceptrons [232]. Additionally, LDA serves as the foundation for many of the Multiple Instance Learning dimensionality reduction approaches in the current literature [26, 30, 27].

### A.1.3 Nonlinear Manifold Learning

Linear methods such as PCA and MDS are convenient for projecting out-of-sample test points into the embedding space. However, they are unable to capture the structure of data that are sampled from nonlinear manifolds [25]. This section will discuss a variety of nonlinear dimensionality reduction and manifold learning approaches. All methods

reviewed assume the data is distributed along a $d$-dimensional sub-manifold $\mathcal{X}$ embedded in $\mathbb{R}^D$.

#### A.1.3.1 Kernelization

Although each of the manifold learning techniques previously discussed are inherently linear, nonlinear adaptations have been made. One approach is to utilize kernel functions as means to provide nonlinearity in the embeddings.

**Kernels** A kernel function, $\kappa(\boldsymbol{x}, \boldsymbol{x}') \in \mathbb{R}$, is a real-valued function of two arguments, $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$, which maps vectors from the input feature space to a single value in $\mathbb{R}$. The function is typically symmetric (i.e. $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \kappa(\boldsymbol{x}', \boldsymbol{x})$) and non-negative (i.e. $\kappa(\boldsymbol{x}, \boldsymbol{x}') \geq 0$), which implies that it can be interpreted as a measure of similarity [47]. The notion of kernels is very useful in certain applications where data representation is not straightforward, such as representing text documents or molecular structures which can have variable length.

A popular choice of kernel in manifold learning is the radial basis function (RBF) kernel, defined as:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{||\boldsymbol{x} - \boldsymbol{x}'||^2}{\beta}\right) \tag{A-18}$$

where $\beta$ is the bandwidth of the isotropic function. Another popular kernel for text classification is cosine similarity, defined by:

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \frac{\boldsymbol{x}^T \boldsymbol{x}'}{||\boldsymbol{x}||_2 ||\boldsymbol{x}'||_2} \tag{A-19}$$

This kernel measures the cosine of the angle between vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ after scaling them onto the unit hyper-sphere. If $\boldsymbol{x}$ and $\boldsymbol{x}'$ are strictly positive vectors (counts in the bag-of-words model, for example), then the kernel provides values in $[0, 1]$, where a value of $0$ means the feature vectors are orthogonal and, therefore, have no features in common, and a value of $1$ means the vectors are the same.

Some of the nonlinear manifold learning methods in the literature require the kernel function to satisfy the requirement that the Gram matrix

$$\boldsymbol{K} = \begin{bmatrix} \kappa(\boldsymbol{x}_1, \boldsymbol{x}_1) & \dots & \kappa(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ & \vdots & \\ \kappa(\boldsymbol{x}_N, \boldsymbol{x}_1) & \dots & \kappa(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \tag{A-20}$$

be positive definite for any set of inputs $\{\boldsymbol{x}_n\}_{n=1}^N$. This type of kernel is called a Mercer kernel or positive definite kernel, and is required to induce a Reproducing Kernel Hilbert Space (RKHS). The importance of the Mercer Kernel is the following result, known as Mercer's theorem. This theorem states that if the Gram matrix is positive definite, its eigenvector decomposition can be written as

$$\boldsymbol{K} = \boldsymbol{U}^T \boldsymbol{\Lambda} \boldsymbol{U} \tag{A-21}$$

As derived by Murphy [47] and Liu et al. [233], it then follows that each entry of $\boldsymbol{K}$ can be computed as

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \phi(\boldsymbol{x})^T \phi(\boldsymbol{x}') =< \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') > \tag{A-22}$$

meaning that the entries of the kernel matrix can be defined by the inner product of some feature vectors that are implicitly defined by the eigenvectors $\boldsymbol{U}$. If the kernel is Mercer, then there exists a function $\phi$ which maps $\boldsymbol{x} \in \mathcal{X}$ to $\mathbb{R}^D$ such that Equation A-22 holds. Additionally, $\phi$ depends on the eigenfunctions of $\kappa$, meaning that $D$ is a potentially infinite dimensional space. Additionally, instead of representing feature vectors in terms of kernels $\phi(\boldsymbol{x}) = [\kappa(\boldsymbol{x}, \boldsymbol{x}_1), \dots, \kappa(\boldsymbol{x}, \boldsymbol{x}_N)]$, algorithms can instead work with the input feature vectors $\boldsymbol{x}$ by replacing all inner products $< \boldsymbol{x}, \boldsymbol{x}' >$ with a call to the kernel function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$. This is called the kernel trick, and it turns out that many algorithms can be kernelized in this way.

Kernel functions play an important role in the dimensionality reduction literature for both applying nonlinearity to inherently linear problems and in defining similarity

measures for graph-based manifold learning methods. Specifically, nonlinear adaptations of the inherently-linear PCA [217], MDS [234] and LDA [235] algorithms have been formulated. The kernelization of PCA is briefly described in the following.

#### A.1.3.2 Kernel PCA (KPCA)

Section A.1.2.1 showed how PCA could be used to compute linear low-dimensional embeddings of data. This process involved finding the eigenvectors of the empirical data covariance matrix $\hat{\boldsymbol{S}} = \frac{1}{N}\sum_{n=1}^{N} \hat{\boldsymbol{x}}_n \hat{\boldsymbol{x}}_n^T = \frac{1}{N}\hat{\boldsymbol{X}}^T\hat{\boldsymbol{X}}$, where $\hat{\boldsymbol{x}}_n = \boldsymbol{x}_n - \hat{\boldsymbol{\mu}}$ is the mean subtracted feature vector. However, PCA can also be computed by finding the eigenvectors of the inner product matrix $\boldsymbol{X}\boldsymbol{X}^T$ [47, 236]. This interpretation allows the production of nonlinear embeddings by taking advantage of the kernel trick. This approach is known as Kernel PCA (KPCA) [217]. Assuming the data is mapped to a new feature space in $\mathbb{R}^M$ by a nonlinear transformation $\phi(\boldsymbol{x})$, PCA could be performed in the new feature space. However, this computation can be extremely costly and inefficient. Instead, kernel methods can be used to simply the computation. Following the derivation defined by Wang [236], first assume that the data in the new feature space has zero mean:

$$\frac{1}{N}\sum_{n=1}^{N} \phi(\boldsymbol{x}_n) = 0 \tag{A-23}$$

The covariance matrix of the projected features is a $M \times M$ matrix

$$\boldsymbol{S}_\phi = \sum_{n=1}^{N} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^T \tag{A-24}$$

The eigenvectors $\boldsymbol{V}$ and eigenvalues $\lambda$ for $\boldsymbol{S}_\phi$ satisfy

$$\boldsymbol{S}_\phi \boldsymbol{v}_k = \lambda_k \boldsymbol{v}_k, \quad \forall k = 1, \ldots, M \tag{A-25}$$

From Equations A-24 and A-25, we have

$$\frac{1}{N}\sum_{n=1}^{N} \phi(\boldsymbol{x}_n)\phi(\boldsymbol{x}_n)^T \boldsymbol{v}_k = \lambda_k \boldsymbol{v}_k \tag{A-26}$$

which can be re-written as

$$\boldsymbol{v}_k = \sum_{n=1}^{N} \alpha_{kn} \phi(\boldsymbol{x}_n) \tag{A-27}$$

Substituting Equation A-27 into Equation A-26, gives

$$\frac{1}{N} \sum_{n=1}^{N} \phi(\boldsymbol{x}_n) \phi(\boldsymbol{x}_n)^T \sum_{n=1}^{N} \alpha_{kn} \phi(\boldsymbol{x}_n) = \lambda_k \sum_{n=1}^{N} \alpha_{kn} \phi(\boldsymbol{x}_n) \tag{A-28}$$

A $N \times N$ matrix $\boldsymbol{K}$ can be defined by

$$\boldsymbol{K}_{mn} = \kappa(\boldsymbol{x}_m, \boldsymbol{x}_n) = \phi(\boldsymbol{x}_m)^T \phi(\boldsymbol{x}_n) \tag{A-29}$$

which simplifies the problem to

$$\boldsymbol{K}^2 \boldsymbol{\alpha}_k = N \lambda_k \boldsymbol{K} \boldsymbol{\alpha}_k \tag{A-30}$$

where $\boldsymbol{\alpha}_k$ is a column vector with entries $\alpha_{k1}, \ldots, \alpha_{kN}$. Each $\boldsymbol{\alpha}_k$ can be found by solving the eigenvalue problem

$$\boldsymbol{K} \boldsymbol{\alpha}_k = N \lambda_k \boldsymbol{\alpha}_k \tag{A-31}$$

Then the projection of a test point $\boldsymbol{x}$ onto the $k^{th}$ principal component can be found by

$$z_k(\boldsymbol{x}) = \phi(\boldsymbol{x})^T \boldsymbol{v}_k = \sum_{n=1}^{N} \alpha_{kn} \kappa(\boldsymbol{x}_n, \boldsymbol{x}) \tag{A-32}$$

This formulation assumes that the projected data has zero mean. This is not generally the case and the mean cannot simply be subtracted in the projected space [47]. Therefore, KPCA can be performed on the centered Gram matrix, defined by

$$\tilde{\boldsymbol{K}} = \boldsymbol{H} \boldsymbol{K} \boldsymbol{H} \tag{A-33}$$

where

$$\boldsymbol{H} = \mathbb{I}_N - \frac{1}{N} \boldsymbol{e} \boldsymbol{e}^T \tag{A-34}$$

is the $N \times N$ centering matrix. Pseudo-code for KPCA is given in Algorithm A-15.

**Algorithm A-15** KPCA

**Input:** Gram matrix of training data $\boldsymbol{K} \in \mathbb{R}^{N \times N}$, Gram matrix augmented with test data $\boldsymbol{K}_* \in \mathbb{R}^{N_* \times N}$, dimensionality of latent space $d$
**Output:** Embedded data coordinates $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$

$\boldsymbol{H} \leftarrow \mathbb{I}_N - \frac{1}{N}\boldsymbol{e}\boldsymbol{e}^T$
$\tilde{\boldsymbol{K}} \leftarrow \boldsymbol{H}\boldsymbol{K}\boldsymbol{H}$
$[\boldsymbol{U}, \boldsymbol{\Lambda}] \leftarrow \text{EIG}(\tilde{\boldsymbol{K}})$
**for** $n \in N$ **do**
    $\boldsymbol{v}_n \leftarrow \boldsymbol{u}_n / \sqrt{\lambda_n}$
**end for**
$\boldsymbol{H}_* \leftarrow \frac{1}{N_*}\boldsymbol{e}_*\boldsymbol{e}^T$
$\tilde{\boldsymbol{K}}_* \leftarrow \boldsymbol{K}_* - \boldsymbol{H}_*\boldsymbol{K}_* - \boldsymbol{K}_*\boldsymbol{H}_* + \boldsymbol{H}_*\boldsymbol{K}_*\boldsymbol{H}_*$
$\boldsymbol{Z} \leftarrow \tilde{\boldsymbol{K}}_*\boldsymbol{V}_{1:d}$

Whereas linear PCA is limited to $d < D$ components, KPCA can use up to $N$ components. Using a nonlinear feature embedding function along with the kernel trick provides for an elegant solution for capturing global nonlinear data structure. As can be seen in Algorithm A-15, embedding out-of-sample test points is done by appending rows to the Gram matrix, where new entries are defined between the test points and training data points, but not between test points and other test points. Embedded data coordinates are computed by multiplying the augmented Gram matrix with the top $d$ scaled eigenvectors of the training Gram matrix.

#### A.1.3.3 Graph-based Methods

Nonlinear manifold learning methods typically rely on the use of computational graphs. These graphs represent data structure pooled from local neighborhoods of samples. Spectral graph theory focuses on constructing, analyzing and manipulating graphs. It has proved useful for object representation, graph visualization, spectral clustering, dimensionality reduction and numerous other applications in chemistry, physics, signal processing and computer science [237, 238]. An overview of computational graphs as well as prominent methods for graph construction in manifold learning are presented. Additionally, geodesic distance approximation from pairwise

distances is reviewed. It should be noted that the work by Yan et al. [228] shows how each of the undermentioned graph-based manifold learning approaches can be succinctly described under a general graph-based framework for dimensionality reduction. It is encouraged that readers turn to that work for additional information on the mathematical relationships between the algorithms, as well as how linearization, kernelization and tensorization are applied in the graph-based setting.

**Terminology** Many dimensionality reduction methods in the literature are interested in analyzing relationships between samples defined on an undirected, weighted graph $G = \{\mathcal{V}, \mathcal{E}, \boldsymbol{W}\}$, which consists of a finite set of vertices $\mathcal{V}$ (also called nodes or points) with cardinality $\mathcal{V} = N$, a set of edges $\mathcal{E} \subset \mathcal{V} \times \mathcal{V} = [\mathcal{V}]^2$ (also known as arcs or lines) and a weighted adjacency or affinity matrix $\boldsymbol{W}$ [237, 239, 238]. The the size or order of a graph is defined by the number of nodes $|\mathcal{V}|$ and edges $|\mathcal{E}|$. If two vertices in $G$, say $\boldsymbol{u}, \boldsymbol{v} \in \mathcal{V}$, are connected by an edge $e \in \mathcal{E}$, this is denoted by $e = (\boldsymbol{u}, \boldsymbol{v})$ and the two vertices are said to be adjacent or neighbors. When edges do not have a direction, they are coined as undirected. A graph solely containing this type of connection is termed as an undirected graph. When all edges have directions, meaning $(\boldsymbol{u}, \boldsymbol{v})$ and $(\boldsymbol{v}, \boldsymbol{u})$ are distinguishable, the graph is said to be directed. In the literature, the term arc is typically used to denote connections between nodes in directed graphs, while edge is used when they are undirected. The graph-based methods included in this literature review focus on analyzing affinities between data samples in undirected graphs. Moreover, a path between any two nodes in $\boldsymbol{u}, \boldsymbol{u'} \in \mathcal{V}$ is a non-empty sequence of $k$ different vertices $< \boldsymbol{v}_0, \boldsymbol{v}_1, \ldots, \boldsymbol{v}_k >$ where $\boldsymbol{u} = \boldsymbol{v}_0, \boldsymbol{u}' = \boldsymbol{v}_k$ and $(\boldsymbol{v}_{i-1}, \boldsymbol{v}_i) \in \mathcal{E}$, $i = 1, 2, \ldots, k$. Additionally, a graph is said to be acyclic if there are no cycles between its edges, regardless of whether it is directed or undirected.

When using graphs for dimensionality reduction, vertices usually represent features of individual samples, and edges express relationships between them. The most straight-forward way to construct a graph is to instantiate edges between every vertex in

the graph, where each edge is weighted by the distance between the vertices it connects according to a pre-defined metric. This type of graph is called full mesh. Weights on edges are captured in the graph adjacency matrix $\boldsymbol{W}$. When weights are not naturally defined by an application, a common way to the define the weight of an edge connecting vertices $\boldsymbol{u} \sim \boldsymbol{u}'$ is by a symmetric affinity function $W_{\boldsymbol{u},\boldsymbol{u}'} = K(\boldsymbol{u}; \boldsymbol{u}')$; typically a radial basis function (RBF) or heat kernel, defined as:

$$\boldsymbol{W}_{\boldsymbol{u},\boldsymbol{u}'} = w_{\boldsymbol{u},\boldsymbol{u}'} = \exp\left(-\frac{||\boldsymbol{u} - \boldsymbol{u}'||^2}{\beta}\right) \tag{A-35}$$

where $\beta$ is the non-negative bandwidth of the kernel. Vertices will have a nonzero weight only if they fall within the nonzero mapping domain of the kernel. Additionally, a threshold could be set to truncate the weights of neighbors far from individual samples.

**$\boldsymbol{K}$-nearest neighbor graph** In a $K$-nearest neighbor graph, every data point (vertex) $\boldsymbol{x}_n \in \boldsymbol{X}$ is connected by edges to its $K$-nearest neighbors, where $K \in \mathbb{Z}^+$ is fixed. An example of a $K$-nearest neighbor graph is depicted in a of Figure A-6. The downside of this graph is that it might impose edges between neighbors that should not actually be connected, as in the case where a sample is metrically distant from all of its nearest neighbors. Although, this feature may actually be useful in domains such as outlier detection, where low adjacency weights indicate that the sample is far form the sampling distribution. Two alternative $K$-nearest neighbor graphs, a symmetric and mutual neighbors, might instead by utilized. In the symmetric $K$-nearest neighbors graph, two vertices $\boldsymbol{u}$ and $\boldsymbol{u}'$ if $\boldsymbol{u}$ is among the $K$-nearest neighbors of $\boldsymbol{u}'$ or $\boldsymbol{u}'$ is among the neighbors of $\boldsymbol{u}$. The mutual $K$-nearest neighbors graph, however, only connects vertices $(\boldsymbol{u}, \boldsymbol{u}')$ if $\boldsymbol{u}$ is among the $K$-nearest neighbors of $\boldsymbol{u}'$ and $\boldsymbol{u}'$ is among the $K$-nearest neighbors of $\boldsymbol{u}$. The weights on each edge are provided as the similarity of the adjacent nodes.

**$\boldsymbol{\epsilon}$-neighborhood graph** Another method for graph construction is to use $\epsilon$-neighborhoods (or $\epsilon$-balls). In this graph, two vertices $(\boldsymbol{u}, \boldsymbol{u}')$ are connected by an edge

if and only if the distance between them is equal to or smaller than some value $\epsilon$, $\mathcal{D}_{\mathcal{U}}(\boldsymbol{u}, \boldsymbol{u}') \leq \epsilon$. This idea is represented in b of Figure A-6. In both the $K$-nearest and $\epsilon$-neighborhood graphs, a parameter controlling the number of edges in the graph, $K$ or $\epsilon$, must be chosen. These parameters are highly influential for graph construction and can thus greatly affect dimensionality reduction quality. Contrary to the $K$-nearest neighbor graph, an $\epsilon$-neighborhood will not create connections between distant vertices. However, when the data is sampled sparsely from a highly-curved manifold, the $\epsilon$-neighbor graph will not be able to appropriately capture the geometry [4].

**Geodesic distance approximation** The ultimate goal of manifold learning is to uncover an underlying low-dimensional sub-manifold which is embedded in $\mathbb{R}^D$. Many dimensionality reduction methods in the literature discover projections of data into a low-dimensional space which preserve topological ordering of the data [25]. These processes require a notion of distance between samples. Euclidean distance is a popular metric which captures the straight-line disparity between two points. As shown in Figure A-7, however, samples that are actually distant on the manifold may appear deceptively close in the high-dimensional input feature space, as measured by Euclidean distance [22]. Geodesic distance, also called curvilinear or shortest-path distance, Figure A-7, on the other hand, follows the curvature of a manifold and may provide a better measure of dissimilarity between data samples. Geodesic distance can be estimated by the shortest path through a graph constructed by assuming the distances between neighbors is locally Euclidean [224]. This can be conceptualized by a simple example. The Earth is a sphere and naturally has curvature. Two people standing in a room, however, would estimate the distance between themselves by a straight line. Thus, in a very local region on the Earth, the measure of curvature would be negligible and the true distances between objects could be estimated with Euclidean distance. The same concept is true for manifolds where, if data is sampled densely enough, geodesic distance can be approximated by the shortest-path through a neighborhood graph where

the dissimilarities between neighbors is assumed to be locally Euclidean. Geodesic distance can be estimated efficiently by methods such as Dijskstra's or Floyd's shortest-path algorithms [22].

#### A.1.3.4 Isomap

This section describes Isomap and related methods.

**Traditional Isomap** While MDS has proven to be successful in a variety of applications, it suffers from the fact that is solely aims to retain pairwise Euclidean distances and does not consider the distributions of neighboring samples. This implies that MDS is not able to capture the geometry of high-dimensional data which lies on or near to a curved manifold, such as the Swiss roll dataset [1, 83]. Isometric Feature Mapping (Isomap) [22] is a technique which resolves this problem by attempting to preserve pairwise geodesic distances between datapoints. Isomap can be considered as a generalization of classical MDS in which the pairwise distance matrix is replaced by a matrix of pairwise geodesic distances approximated by distances in the graph [4]. The classic, unsupervised algorithm consists of a few steps:

1. Given a set of input data $\boldsymbol{X} = \{\boldsymbol{x}_n\}_{n=1}^N \subset \mathbb{R}^D$, construct a sparse neighborhood graph (such as the $K$-nearest or $\epsilon$-ball graphs discussed previously) where each edge is weighted by the Euclidean distance between the neighbors it connects:

$$\boldsymbol{W}_{mn} = w_{mn} = ||\boldsymbol{x}_m - \boldsymbol{x}_n||^2 \tag{A-36}$$

   where $\boldsymbol{W}$ is the graph adjacency matrix.

2. Next, the geodesic distances between all pairs of samples is computed by finding the shortest paths between the points through the graph. This is commonly done with Dijkstra's or Floyd's shortest-path algorithms [22].

3. These geodesic distances form a pairwise distance matrix which is substituted into classical MDS as described in Section A.1.2.2. This provides the low-dimensional embedding coordinates $\boldsymbol{Z} = [\boldsymbol{z}_1^T, \ldots, \boldsymbol{z}_N^T] \in \mathbb{R}^{N \times d}$ of high-dimensional input data $\boldsymbol{X} = [\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_N^T] \in \mathbb{R}^{N \times D}$, where $d \ll D$.

While Isomap has been successfully applied in the areas of financial analysis [240], facial and object recognition [241], visualization and classification tasks [242], a few

important weaknesses are prevalent. First, Isomap may be topologically unstable. That is, it may construct erroneous connections in the neighborhood graph. This is known as short-circuiting, and it can severely impair the performance of Isomap. Several approaches have been proposed to nullify the short-circuiting problem, such as removing datapoints with large total flows or by removing nearest neighbors that violate local linearity of the neighborhood graph [1]. Another weakness of Isomap is that it may not perform correctly if there are holes in the manifold, as this causes the geodesic distances of some samples to appear further on the manifold than they truly are. A third weakness is that Isomap can fail if the manifold is non-convex. Therefore, we see that Isomap can perform very well due to theoretical guarantees on qualities such as convergence, as long as the manifold is isometric to a convex open set of $\mathbb{R}^d$, $\mathcal{D}_{\mathcal{X}}(\boldsymbol{u}, \boldsymbol{u}') = \mathcal{D}_{\mathcal{Y}}(f(\boldsymbol{u}), f(\boldsymbol{u}'))$, meaning that the geodesic distances in the graph are almost equal to the Euclidean distances in the embedding space $\mathbb{R}^d$. Continuing, an additional drawback of Isomap is the fact that it requires the decomposition of a large, dense Gram matrix which scales with the number of training data points. If the dataset grows too large, a solution will no longer be tractable. Furthermore, the constraint on $\mathcal{X}$ to be isometric to a convex open set of $\mathbb{R}^d$ is rarely met. As mentioned by Thorstensen [4], these problems may be circumvented by sparsifying large datasets using landmarks, as with Landmark Isomap [243] and looking at conformal maps, as is done in Conformal Isomap [244]. Finally, as with most nonlinear manifold learning techniques, it is nontrivial to embed out-of-sample data points into the lower dimensional feature space.

**Supervised Isomap approaches** As with most traditional manifold learning methods in the literature, Isomap is not inherently well-suited for classification tasks. However, supervised approaches which consider class label information have been adopted to increase class separability in the latent embedding space. The work by Vlachos et al. [242] was the first to investigate a supervised adaptation of Isomap. Two supervised Isomap procedures were proposed which combine Isomap with a nearest

neighbor classifier. These methods, Iso+Ada and WeightedIso take label information into consideration to scale the computed Euclidean distances utilized by Isomap by a constant factor according to class label. The idea is to make points closer in the embedding space if they have the same class label and farther if they have opposing class labels. Ribeiro et al. [240] proposed an enhanced supervised Isomap (ES-Isomap) in which the dissimilarity matrix is weighted according to rules which consider class label information. The dissimilarity matrix (considered as the adjacency matrix), $\boldsymbol{W}$, which was the same used in the Supervised Isomap method [3], is defined as:

$$\boldsymbol{W}(\boldsymbol{x}_m, \boldsymbol{x}_n) = \begin{cases} \sqrt{1 - \exp \frac{-\mathcal{D}^2(\boldsymbol{x}_m, \boldsymbol{x}_n)}{\beta}}, & l_m = l_n \\ \sqrt{\exp \frac{\mathcal{D}^2(\boldsymbol{x}_m, \boldsymbol{x}_n)}{\beta}} - \alpha, & l_m \neq l_n \end{cases} \tag{A-37}$$

where $\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)$ denotes the distance measure between samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$, $\beta$ is used to prevent $\boldsymbol{W}(\boldsymbol{x}_m, \boldsymbol{x}_n)$ from increasing too quickly when $\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)$ is large and is typically set according to the density of the data, $\alpha$ is a constant in $[0, 1]$ which controls the dissimilarity between points in different classes and keeps the graph from becoming disconnected, and $l_m$ and $l_n$ are the corresponding class labels of samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$, respectively. In Equation A-37, the dissimilarity between two points is greater than or equal to one if their class labels are different and less than one if the points have the same class label. Therefore, the between-class dissimilarity will always be larger than the within-class, which is an important property for classification tasks. Pseudo-code for SE-Isomap is given in Algorithm A-16.

Li et al. [245] proposed Supervised Isomap with Explicit mapping (SE-Isomap). SE-Isomap enforces discriminability on the matrix of geodesic distances, as compared to the Euclidean distance matrix used in the aforementioned approaches, to learn an explicit mapping to the low-dimensional embedding space. Finally, Zhang et al. [241] developed a semi-supervised Isomap to utilize both labeled and unlabeled data points in

**Algorithm A-16** SE-Isomap

**Input:** Dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \in \mathbb{R}^{N \times D}$, $\{l_1, \ldots, l_N\} \in \{-1, +1\}^N$, parameters $k, \beta, \alpha, d$

**Output:** Low-dimensional data representations $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$

**for** $m, n \in N$ **do**
  **if** $\boldsymbol{x}_n$ in $k$-NN of $\boldsymbol{x}_m$ and $L_m = L_n$ **then**
    $\boldsymbol{W}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leftarrow \sqrt{1 - \exp \frac{-||(\boldsymbol{x}_m, \boldsymbol{x}_n||^2}{\beta}}$
  **else if** $\boldsymbol{x}_n$ in $k$-NN of $\boldsymbol{x}_m$ and $L_m \neq L_n$ **then**
    $\boldsymbol{W}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leftarrow \sqrt{\exp \frac{||(\boldsymbol{x}_m, \boldsymbol{x}_n||^2}{\beta}} - \alpha$
  **else**
    $\boldsymbol{W}(\boldsymbol{x}_m, \boldsymbol{x}_n) \leftarrow 0$
  **end if**
**end for**
$\boldsymbol{D} \leftarrow$ squares of the shortest distances between all points using Dijkstra's or Floyd's algorithm on $\boldsymbol{W}$
$\boldsymbol{Z} \leftarrow \text{MDS}(\boldsymbol{D})$

training. This method aims at minimizing pairwise distances of within-class samples in the same manifold while maximizing the distances over different manifolds.

#### A.1.3.5 Sammon Mapping

Classical scaling, a convex technique for multidimensional scaling, was introduced in Section A.1.2.2. As discussed, a pitfall of MDS is that it focuses on retaining global pairwise distances as opposed to local distances, which are typically much more important for capturing the geometry of the data [1]. Several MDS variants have been proposed to address this weakness. A popular variant is Sammon Mapping [246].

The classical scaling cost function looks to minimize the difference between the pairwise distance matrices of input data and their corresponding low-dimensional representations. This cost puts emphasis on retaining the global data structure. Sammon mapping adapts the classical scaling cost function by weighting the

contribution of each sample pair $(\boldsymbol{x}_m, \boldsymbol{x}_n)$ by the inverse of their pairwise distances in the high-dimensional input space $\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)$. In this way, the objective becomes

$$J(\boldsymbol{X}, \boldsymbol{Z}) = \frac{1}{\sum_{m,n} \mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)} \sum_{m \neq n} \frac{(\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n) - \mathcal{D}(\boldsymbol{z}_m, \boldsymbol{z}_n))^2}{\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)} \tag{A-38}$$

where $\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)$ denotes the Euclidean distance between high-dimensional input samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$, $\mathcal{D}(\boldsymbol{z}_m, \boldsymbol{z}_n)$ is the Euclidean distance between low-dimensional data coordinates $\boldsymbol{z}_m$ and $\boldsymbol{z}_n$ and the constant in the front is added to simplify the gradient of the objective. This cost function gives more weight to preserving distances between samples that are close in the input space. Minimization of the Sammon objective function is typically performed with gradient descent or a pseudo-Newton method [224].

A weakness (and strength) of Sammon mapping is that it will give much more importance to retaining a very small distance, say $10^{-5}$, as compared to $10^{-4}$. Despite this, Sammon mapping has been successfully applied to visualization tasks and has reported on applications using gene data and geospatial information [1].

#### A.1.3.6 Maximum Variance Unfolding (MVU)

Section A.1.3.2 described how Kernel PCA [217, 236] allows PCA to be performed in a feature space defined by a kernel function $\kappa$. However, the choice of kernel function is arbitrary and may not be optimal for learning the intrinsic structure of a set of data. To resolve this issue, Maximum Variance Unfolding (MVU, formerly known as Semidefinite Embedding)[247] attempts to learn an appropriate kernel matrix to be used in conjunction with KPCA. This is done by defining a neighborhood graph over the data and retaining the pairwise distances through the embedding to a low-dimensional space, as done in Isomap (Section A.1.3.4). Unlike Isomap, however, MVU attempts to "unfold" the data data manifold by maximizing the Euclidean distances between data points under the constraint that the local geometry of the data manifold is unperturbed [1]. The optimization can be solved using semidefinite programming. MVU begins by forming a $k$-nearest neighbor graph $G$. It then tries to maximize the sum of Euclidean distances

between all samples such that the distances inside $G$ are unchanged. This equates to the to following maximization problem:

$$\max_{\boldsymbol{Z}} \sum_{m,n} ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2 \quad s.t. \quad ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2 = ||\boldsymbol{x}_m - \boldsymbol{x}_n||^2 \quad \forall (m,n) \in G \tag{A-39}$$

This objective can be solved through the optimization of a semidefinite programming problem (SDP) for the kernel matrix $\boldsymbol{K}$:

$$\begin{aligned} &\max_{\boldsymbol{K}} \quad Tr(\boldsymbol{K}) \quad s.t. \\ &1. \quad \boldsymbol{K} \succ 0 \\ &2. \quad \sum_{m,n} \boldsymbol{K}_{m,n} = 0 \\ &3. \quad \boldsymbol{K}_{mm} + \boldsymbol{K}_{nn} - 2\boldsymbol{K}_{mn} = ||\boldsymbol{x}_m - \boldsymbol{x}_n||^2 \quad \forall (m,n) \in G \end{aligned} \tag{A-40}$$

with $\boldsymbol{K}$ defined as the outer product matrix of the low-dimensional data coordinates $\boldsymbol{Z}$. The solution of the SDP provides a kernel matrix which is used in Kernel PCA. The low-dimensional embedding coordinates $\boldsymbol{Z}$ are found by eigendecomposition of the kernel, as described in Section A.1.3.2.

Similarly to Isomap, MVU may suffer from short-circuiting due to optimization constraints which impair successful manifold unfolding. Despite this weakness, MVU has been applied successfully to applications on microarray data and sensor localization [1].

#### A.1.3.7 Locally Linear Embedding (LLE)

Locally Linear Embedding (LLE) was first introduced by Roweis and Saul [216] and it is, along with Isomap, a foundational graph-based approach for nonlinear manifold learning. Whereas Isomap attempts to preserve global pairwise distances, LLE attempts to preserve solely local properties of the data [1]. Since LLE looks solely at local neighborhoods around samples, it is much less sensitive to short-circuiting than Isomap. Furthermore, the preservation of local properties allows the algorithm to effectively embed non-convex manifolds. Essentially, LLE looks to represent each data sample as a

linear combination of its $k$-nearest neighbors. This fits a hyperplane through every datapoint and it's nearest neighbors, thus assuming the intrinsic manifold is locally linear. The local linearity assumption implies that the reconstruction weights $\boldsymbol{w}_n$ of high-dimensional sample $\boldsymbol{x}_n$ are invariant to transformations such as translation, rotation and scaling. Therefore, it is assumed that the same reconstruction weights $\boldsymbol{w}_n$ that can be used to represent $\boldsymbol{x}_n$ in the high-dimensional space will also reconstruct it's low-dimensional representation $\boldsymbol{z}_n$ from its corresponding neighbors in the low-dimensional space [216, 224, 83]. Therefore, low-dimensional data representations $\boldsymbol{Z}$ are found by minimizing the objective

$$J(\boldsymbol{W}, \boldsymbol{Z}) = \sum_{n=1}^{N} \left|\left| \boldsymbol{z}_n - \sum_{k=1}^{K} w_{nk} \boldsymbol{z}_k \right|\right|^2 \quad s.t. \quad \frac{1}{N} \boldsymbol{Z}^T \boldsymbol{Z} = \mathbb{I}_d \tag{A-41}$$

The constraint on the covariance of the embedded data representations is included to avoid the trivial solution $\boldsymbol{Z} = \boldsymbol{0}$. Solving for the low-dimensional data representations is done in three steps [4]. First, a $k$-nearest neighborhood graph $G$ is constructed from the high-dimensional data $\boldsymbol{X}$. Next, the reconstruction weight vector $\boldsymbol{w}_n$ that best represents each point $\boldsymbol{x}_n$ as a linear combination of its $k$-nearest neighbors is found by optimizing the problem

$$\min_{\boldsymbol{W}} \sum_{n=1}^{N} \left|\left| \boldsymbol{x}_n - \sum_{k=1}^{K} w_{nk} \boldsymbol{x}_k \right|\right|^2 \quad s.t. \quad \sum_{k=1}^{K} w_{nk} = 1 \tag{A-42}$$

Optimization of Equation A-42 can be solved directly by using the method of Lagrange Multipliers. Finally, the low-dimensional embedding coordinates are found by minimizing the quadratic error function for $\boldsymbol{Z}$ according to Equation A-41. Roweis and Saul [216] showed that the coordinates that minimize the objective can be found as the eigenvectors corresponding to the $d$ smallest nonzero eigenvalues of the inner product $(\mathbb{I}_N - \boldsymbol{W})^T(\mathbb{I}_N - \boldsymbol{W})$ where $\boldsymbol{W}$ is a $N \times N$ sparse matrix whose entries are equal to the corresponding reconstruction weight if $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ are connected in the neighborhood graph and $0$ otherwise.

The popularity of LLE has led to the development of linear variants, namely Neighborhood Preserving Projections (NPP) [248] and Orthogonal Neighborhood Preserving Projections [249], and has been applied successfully to applications in super-resolution and sound localization [1]. However, LLE tends to collapse large portions of the data close to each other in the embedding space due to the constraint on the covariance matrix. This may also result in undesired scalings of the manifold. Despite these effects, supervised approaches based on LLE have also been developed [83, 250]. Existing approaches can be summarized by the LDA idea, that points in the same class should be embedded more closely to each other while points in opposing classes should be well-separated in the low-dimensional space. This notion is realized by altering the dissimilarity matrix used to construct the neighborhood graph.

#### A.1.3.8 Laplacian Eigenmaps (LE)

This section describes Laplacian Eigenmaps and related methods.

**Classical LE** Similar to LLE, Laplacian Eigenmaps [251], or Spectral Embedding, is a nonlinear dimensionality reduction technique which aims to preserve local structure of data [252, 1]. Using spectral graph theory, LE computes low-dimensional representations of data in which the dissimilarities between datapoints and their neighbors (according to an affinity measure) are minimized. The name Laplacian Eigenmaps is derived by the use of Laplacian regularization in the optimization procedure [4]. Given a set of $N$ samples $\boldsymbol{X} = \{\boldsymbol{x}_n\}_{n=1}^N \subset \mathbb{R}^D$, the first step of LE is to define a neighborhood graph on the samples. This graph, also called an affinity or adjacency matrix can be constructed in a variety of ways, such as $K$-nearest neighbor, $\epsilon$-ball, full mesh, or by weighting each edge $\boldsymbol{x}_m \sim \boldsymbol{x}_n$ by a symmetric affinity function $W_{mn} = K(\boldsymbol{x}_m; \boldsymbol{x}_n)$, typically a radial basis or heat kernel:

$$\boldsymbol{W}_{mn} = w_{mn} = \exp\left(-\frac{||\boldsymbol{x}_m - \boldsymbol{x}_n||^2}{\beta}\right) \tag{A-43}$$

where the kernel bandwidth $\beta$ is typically set as the variance of the dataset [252, 4].

The goal is to uncover the latent data representations $\{\boldsymbol{z}_n\}_{n=1}^N \subset \mathbb{R}^d$ where $d \ll D$ which minimizes the objective

$$J(\boldsymbol{W}, \boldsymbol{Z}) = \frac{1}{2} \sum_{m,n} ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2 w_{mn} = tr(\boldsymbol{Z}^T \boldsymbol{L} \boldsymbol{Z}) \tag{A-44}$$

with $\boldsymbol{W}$ denoting the symmetric affinity matrix, $\boldsymbol{D}$ the diagonal weight matrix whose entries are the sum of the rows (or columns since $\boldsymbol{W}$ is symmetric) of $\boldsymbol{W}$ (i.e. $d_{mm} = \sum_n w_{mn}$, and is $0$ otherwise). The graph Laplacian matrix is provided as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$. The matrix $\boldsymbol{Z} = [\boldsymbol{z}_1^T, \ldots, \boldsymbol{z}_N^T]$ is the $N \times d$ embedding matrix and $tr(.)$ denotes the trace of a matrix. The $n^{th}$ row of matrix $\boldsymbol{Z}$ provides the vector $\boldsymbol{z}_n$, which is the latent representation of sample $\boldsymbol{x}_n$. This objective discourages projecting similar points in the input feature space to disparate regions of the embedding space by enforcing heavy penalization.

The latent sample coordinates $\boldsymbol{Z}$ are found as the solution to the problem:

$$\min_{\boldsymbol{Z}} tr(\boldsymbol{Z}^T \boldsymbol{L} \boldsymbol{Z}) \quad s.t. \quad \boldsymbol{Z}^T \boldsymbol{D} \boldsymbol{Z} = \boldsymbol{I}, \boldsymbol{Z}^T \boldsymbol{L} \boldsymbol{e} = \boldsymbol{0} \tag{A-45}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{e} = (1, \ldots, 1)^T$. The first constraint eliminates the trivial solution $\boldsymbol{Z} = \boldsymbol{0}$ (scaling) and the second constraint avoids the trivial solution $\boldsymbol{Z} = \boldsymbol{e}$ (uniqueness). By applying the Lagrange multiplier method and using the fact that $\boldsymbol{L}\boldsymbol{e} = \boldsymbol{0}$, the low-dimensional data representations can be found by solving the generalized eigenvalue problem:

$$\boldsymbol{L}\boldsymbol{v} = \lambda \boldsymbol{D}\boldsymbol{v} \tag{A-46}$$

The column vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_N$ are the solutions of Equation A-46, ordered to the corresponding eigenvalues, in ascending order, $\lambda_1 = 0 \leq \lambda_2 \leq \cdots \leq \lambda_N$. The embedding of the input samples given by the matrix $\boldsymbol{Z}$, is obtained by concatenating the eigenvectors of the $d$ smallest non-zero eigenvalues. $\boldsymbol{Z}$ is a $N \times d$ matrix, where $d < N$

is the dimensionality of the embedded space. From observation, it is clear that the embedding dimensionality is limited by the number of samples $N$.

**Linear LE (LPP)** Because of the representation ability of LE, a linear approach called Locality Preserving Projections (LPP) was proposed by He and Niyogi [253]. Similarly to LE, LPP builds a neighborhood graph which incorporates local information of the data. LPP optimizes a similar objective to LE, however, it is assumed that the low-dimensional data representations come from a linear transformation of the high-dimensional input data. The primary benefit of LPP over LE is that, while it shares many of the representation properties of LE, it is defined everywhere in the latent space as opposed to just over the training data points. This allows for simple embeddings of out-of-sample test points. Moreover, one can perform LPP in the original space or in a reproducing kernel Hilbert space through the use of Mercer kernels as described in Section A.1.3.1.

**Supervised LE (S-LE)** In order to adopt LE for classification, Raducanu and Dornaika [252] proposed a supervised LE which minimizes the margin between samples with similar class labels and maximizes the margin between samples with opposing class labels. Supervised LE utilizes discriminative information contained in the class labels when finding the nonlinear embedding (spectral projection).

In order to discover both geometrical and discriminative manifold structure, supervised LE splits the global graph into two components: the within-class graph $G_w$ and the between-class graph $G_b$. To define the margin, they define two subsets, $N_w(\boldsymbol{x}_n)$ and $N_b(\boldsymbol{x}_n)$ for each sample $\boldsymbol{x}_n$. These two subsets contain the neighbors of $\boldsymbol{x}_n$ sharing the same label and having different labels, respectively, which have a similarity higher than the average.

$$N_w(\boldsymbol{x}_n) = \{\boldsymbol{x}_m | l_m = l_n, \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}_m||^2}{\beta}\right) > AS(\boldsymbol{x}_n)\} \tag{A-47}$$

$$N_b(\boldsymbol{x}_n) = \{\boldsymbol{x}_m | l_m \neq l_n, \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}_m||^2}{\beta}\right) > AS(\boldsymbol{x}_n)\} \tag{A-48}$$

where $AS(\boldsymbol{x}_n) = \frac{1}{N}\sum_{n=1}^{N} \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}_m||^2}{\beta}\right)$ denotes the average similarity of the sample $\boldsymbol{x}_n$ to the rest of the data. From Equations A-47 and A-48 it is clear that the neighborhoods for each data sample are not necessarily the same size. As a result, this function constructs the affinity graph according to both the local density and similarity between data samples in the input feature space.

With the two sets defined, the within-class and between-class weight matrices $\boldsymbol{W}_w$ and $\boldsymbol{W}_b$ are formed from the adjacency graphs $G_w$ and $G_b$, respectively. These weight matrices are defined as:

$$W_{w,mn} = \begin{cases} \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}_m||^2}{\beta}\right), \text{ if } \boldsymbol{x}_n \in N_w(\boldsymbol{x}_m) \text{ or } \boldsymbol{x}_m \in N_w(\boldsymbol{x}_n) \\ 0, \text{ otherwise} \end{cases} \tag{A-49}$$

$$W_{b,mn} = \begin{cases} 1, \text{ if } \boldsymbol{x}_n \in N_b(\boldsymbol{x}_m) \text{ or } \boldsymbol{x}_m \in N_b(\boldsymbol{x}_n) \\ 0, \text{ otherwise} \end{cases} \tag{A-50}$$

and the global affinity matrix, $\boldsymbol{W}$, can be written as:

$$\boldsymbol{W} = \boldsymbol{W}_w + \boldsymbol{W}_b \tag{A-51}$$

In order to obtain the low-dimensional representations $\boldsymbol{z}_n$ of the input data $\boldsymbol{x}_n$, the following objective functions can be optimized for $\boldsymbol{Z}$:

$$\min \frac{1}{2}\sum_{m,n} ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2 W_{w,mn} = tr(\boldsymbol{Z}^T \boldsymbol{L}_w \boldsymbol{Z}) \tag{A-52}$$

$$\max \frac{1}{2}\sum_{m,n} ||\boldsymbol{z}_m - \boldsymbol{z}_n||^2 W_{b,mn} = tr(\boldsymbol{Z}^T \boldsymbol{L}_b \boldsymbol{Z}) \tag{A-53}$$

where $\boldsymbol{L}_w = \boldsymbol{D}_w - \boldsymbol{W}_w$ and $\boldsymbol{L}_b = \boldsymbol{D}_b - \boldsymbol{W}_b$ indicate the corresponding graph Laplacians of the within-class and between-class affinity graphs, respectively. The matrix $\boldsymbol{Z} = [\boldsymbol{z}_1^T, \ldots, \boldsymbol{z}_N^T]$ contains the low-dimensional representations of the input in its rows.

By merging the two objective functions, the final optimization problem is given as:

$$\arg\max_{\boldsymbol{Z}} \left\{\gamma tr(\boldsymbol{Z}^T \boldsymbol{L}_b \boldsymbol{Z}) + (1-\gamma) tr(\boldsymbol{Z}^T \boldsymbol{W}_w \boldsymbol{Z})\right\} \quad s.t. \quad \boldsymbol{Z}^T \boldsymbol{D}_w \boldsymbol{Z} = \boldsymbol{I} \tag{A-54}$$

The term $\gamma$ is a scalar value in $[0, 1]$ which determines the trade-off between pulling similar samples toward each other in the latent space and pushing heterogeneous points away. A value of $\gamma = 1$ forces the objective to solely focus on maximizing the margin between dissimilar points. Alternatively, a value of $\gamma = 0$ priorities the objective on embedding homogeneous samples in close spatial proximity. By defining matrix $\boldsymbol{B} = \gamma \boldsymbol{L}_b + (1-\gamma)\boldsymbol{W}_w$, the problem becomes:

$$\arg\max_{\boldsymbol{Z}} \left(\boldsymbol{Z}^T \boldsymbol{B} \boldsymbol{Z}\right) \quad s.t. \quad \boldsymbol{Z}^T \boldsymbol{D}_w \boldsymbol{Z} = \boldsymbol{I} \tag{A-55}$$

The low-dimensional embedding matrix $\boldsymbol{Z}$ can be found by solving the generalized eigenvalue problem:

$$\boldsymbol{B}\boldsymbol{v} = \lambda \boldsymbol{D}_w \boldsymbol{v} \tag{A-56}$$

The column vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N$ are the generalized eigenvectors of Equation A-56 arranged by descending eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_d$. Then the $N \times d$ embedding matrix $\boldsymbol{Z} = [\boldsymbol{z}_1^T, \ldots, \boldsymbol{z}_N^T]$ is provided by concatenating the obtained eigenvectors $\boldsymbol{Z} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_d]$. Pseudo-code for S-LE is provided in Algorithm A-17.

The primary difference between the classic LE and S-LE is that traditional LE solely attempts to preserve the spatial relationships between samples, and thus, does not consider label information when learning the embeddings. Alternatively, S-LE aims at aiding discriminant analysis by collapsing the distance between samples with the same label that are in close spatial proximity and pushing away spatial neighbors with differing

**Algorithm A-17** S-LE

**Input:** Dataset $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\} \in \mathbb{R}^{N \times D}$, $\{l_1, \ldots, l_N\} \in \{-1, +1\}^N$, parameters $k, \beta, \gamma, d$
**Output:** Low-dimensional data representations $\boldsymbol{Z}$

**for** $m, n \in N$ **do**
  **if** $\boldsymbol{x}_n$ in $k$-NN of $\boldsymbol{x}_m$ and $L_m = L_n$ **then**
    $\boldsymbol{W}_{w,mn} \leftarrow \exp\left(-\frac{||\boldsymbol{x}_n - \boldsymbol{x}_m||^2}{\beta}\right)$
    $\boldsymbol{W}_{b,mn} \leftarrow 0$
  **else if** $\boldsymbol{x}_n$ in $k$-NN of $\boldsymbol{x}_m$ and $L_m \neq L_n$ **then**
    $\boldsymbol{W}_{w,mn} \leftarrow 0$
    $\boldsymbol{W}_{b,mn} \leftarrow 1$
  **else**
    $\boldsymbol{W}_{w,mn} \leftarrow 0$
    $\boldsymbol{W}_{b,mn} \leftarrow 0$
  **end if**
**end for**
Compute graph Laplacians $\boldsymbol{L}_w \leftarrow \boldsymbol{D}_w - \boldsymbol{W}_w$, $\boldsymbol{L}_b \leftarrow \boldsymbol{D}_b - \boldsymbol{W}_b$
Define $\boldsymbol{B} \leftarrow \gamma \boldsymbol{L}_b + (1 - \gamma)\boldsymbol{W}_w$
Solve the Eigenvector problem $\boldsymbol{B}\boldsymbol{v} = \lambda \boldsymbol{D}_w \boldsymbol{v}$
Sort Eigenvectors $\boldsymbol{v}$ and Eigenvalues $\lambda$ in decreasing order
$\boldsymbol{Z} \in \mathbb{R}^{N \times d} \leftarrow$ top $d$ Eigenvectors

class labels. This is done through the utilization of two affinity graphs: the within-class and between-class graphs. As with most graph-based methods, LE results vary highly according to the choice of neighborhood size. However, choosing the size of $K$ or $\epsilon$ in advance can be very difficult. S-LE does not require user-defined graph parameters, other than those associated with the chosen affinity measure. Instead, graph edges are chosen according to an adaptive neighborhood for each sample. Both methods, however, suffer from inherent difficulties associated with nonlinear manifold learning, namely, selecting the intrinsic embedding dimensionality and handling out-of-sample extensions.

Despite these nuances, LE (and its variants) have been successfully applied in nonlinear dimensionality reduction tasks for facial recognition, spectral clustering and object classification [1].

Apart from S-LE, other methods have been explored to integrate label information into Laplacian Eigenmaps. A review of supervised dimensionality reduction methods by Chao et al. [83] explains that author's have optimized the affinity matrix using label

information after constructing from spatial proximity, proposed deep learning-based approaches to achieve supervised LE and integrated label information into the affinity matrix construction process.

The special feature exhibited by all Laplacian Eigenmap methods is the use of laplacian regularization, which enforces properties such as smoothness and provides a level of resistance toward the influences of outliers. This useful feature has been applied in a variety of supervised and semi-supervised tasks, such as hyperspectral and synthetic aperture radar remote sensing classification [254, 255], classification of synthetic data [256], zero-shot learning [257] and reinforcement learning [258].

#### A.1.3.9 Hessian LLE

Hessian LLE (HLLE) [259] is a variant of LLE that minimizes the curvature of the high-dimensional manifold when embedding it into the low-dimensional space, under the constraint that the intrinsic manifold is locally isometric [1, 4]. This is achieved by the eigenanalysis of the local Hessian matrix $\mathcal{H}$, which measures the curvature of the manifold around every data point. HLLE begins by identifying the $k$-nearest neighbors for each sample $\boldsymbol{x}_n$ using Euclidean distance (local linearity). Because of the linearity assumption, a $d$-dimensional basis for the local tangent space of each sample can be found by PCA or singular value decomposition (SVD) on each sample's corresponding neighbors. Then, a local Hessian $\boldsymbol{H}^n$ of the manifold is estimated at each point $\boldsymbol{x}_n$ in the local tangent space. The data Hessian matrix is constructed from the individual estimators derived from the local tangent coordinates and is given as

$$\mathcal{H}_{mo} = \sum_n \sum_r ((\boldsymbol{H}^n)_{rm} (\boldsymbol{H}^n)_{ro}) \tag{A-57}$$

The symmetric matrix $\mathcal{H}$ gives an estimate of the curvature of the high-dimensional data manifold. The low-dimensional embedding coordinates $\boldsymbol{Z}$ of the input data are given by the eigenvectors of the corresponding $d$ smallest eigenvalues of $\mathcal{H}$.

HLLE shares many of the characteristics of Laplacian Eigenmaps, except that it replaces the manifold Laplacian with the manifold Hessian. Because of this, HLLE suffers from the same weaknesses as Laplacian Eigenmaps [1]. Despite its weaknesses, HLLE has been applied successfully to sensor localization tasks [260].

#### A.1.3.10 Local Tangent Space Alignment (LTSA)

Local Tangent Space Alignment (LTSA) [261] shares similarities with HLLE in that it describes local properties of high-dimensional data using the local tangent space of each data point [1]. The basic idea of LTSA is to use the tangent space in the neighborhood of a data point to represent the local geometry of the data, and then to align the local tangent spaces to construct a global coordinate system for the nonlinear manifold. In this way, the algorithm consists of two steps: 1.) constructing the principal manifold that is tangential to each data point and 2.) finding the global coordinate system that describes the set of data samples in a low-dimensional space.

Just as with Hessian LLE, LTSA begins by computing $d$-dimensional bases for the local tangent spaces of each of the data points $\boldsymbol{x}_n$. This can be done by applying PCA or SVD on the neighbors of each sample. This provides a linear transformation matrix $\boldsymbol{U}_n$ from the neighborhood of $\boldsymbol{x}_n$ to the local tangent space $\Theta_n$. It is assumed that there exists a linear mapping $\boldsymbol{M}_n$ from the local tangent space coordinates $\boldsymbol{\theta}_n$ to the low-dimensional representations $\boldsymbol{z}_n$. The linear transformation matrix and low-dimensional data representations are found by the following minimization problem [224]:

$$\min_{\boldsymbol{M}_n, \boldsymbol{Z}_n} \sum_{n=1}^{N} ||\boldsymbol{Z}_n \boldsymbol{H} - \boldsymbol{M}_n \Theta_n||_F^2 \tag{A-58}$$

where $\boldsymbol{H}$ is a centering matrix. Zhang and Zha [261] showed that the low-dimensional embedding coordinates $\boldsymbol{Z}$ can be constructed by the eigenvectors corresponding to the $d$ smallest nonzero eigenvalues of an alignment matrix $\boldsymbol{B}$.

Similar to other sparse spectral manifold learning techniques, the objective function of LTSA is vulnerable to the presence of the trivial solution [1]. However, LTSA has been successfully applied to applications in facial recognition [262] and microarray data [1]. Moreover, supervised approaches for LTSA were proposed by Li et al. [263] and Ma et al. [264].

#### A.1.3.11 Diffusion Maps

The success of "kernel eigenmap approaches" such as LLE, LE, HLLE and LTSA led to the development of a manifold learning method called Diffusion Maps (DM) [265]. Diffusion maps is a framework that originates from the field of dynamical systems and is based on defining a Markov random walk on the graph of the data. By performing a random walk for a number of time-steps, a measure for the proximity of the data points is obtained [1]. Essentially, a DM embeds data into a lower-dimensional space such that the Euclidean distance between points is approximated by the diffusion distance in the original feature space.

In the diffusion maps framework, a graph of the data is first constructed which measures the connectivity between all points in the graph. The connectivity of two samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ is defined as the probability of transitioning from $\boldsymbol{x}_m$ to $\boldsymbol{x}_n$ in one step of the random walk [266]. The weights and edges in the graph are computed by a heat-kernel (commonly called radial-basis function RBF), which provides a weight matrix $\boldsymbol{W}$ with entries

$$\boldsymbol{W}_{mn} = w_{mn} = \kappa(\boldsymbol{x}_m, \boldsymbol{x}_n) = \exp\left(-\frac{||\boldsymbol{x}_m - \boldsymbol{x}_n||^2}{\beta}\right) \tag{A-59}$$

where $\beta$ is the bandwidth of the isotropic function. By picking an appropriate kernel width, this constructs a matrix which measures the similarity within a certain neighborhood around each data point. Outside of the neighborhood, the function quickly goes to zero. Since this matrix should define the probabilities of jumping between data

points, a sum-to-one constraint is enforced such that the one-step transition probability matrix is defined as

$$\boldsymbol{P}_{mn}^{(1)} = p_{mn}^{(1)} = \frac{w_{mn}}{\sum_{o=1}^{N} w_{mo}} \tag{A-60}$$

This matrix represents the probability of a transition from one data sample to another in a single time-step. Relating DM to Laplacian Eigenmaps, this matrix can also be interpreted as the normalized graph Laplacian matrix [4]. Taking advantage of the Markov assumption, the forward transition probability matrix for $t$ time-steps $\boldsymbol{P}^{(t)}$ is given by $(\boldsymbol{P}^{(1)})^t$ [266] and a diffusion distance can be defined as

$$D^{(t)}(\boldsymbol{x}_m, \boldsymbol{x}_n) = \sum_{o=1}^{N} ||\boldsymbol{P}_{mo}^{(t)} - \boldsymbol{P}_{on}^{(t)}||^2 \tag{A-61}$$

From Equation A-61, it can be observed that the diffusion distance is small if there are many high-probability paths of length $t$ between two samples. By integrating over all paths through the graph, DM is more robust to short-circuiting and noise perturbation than other approaches, such as the geodesic distance utilized in the Isomap algorithm [1, 267, 22]. The objective of a diffusion map is then to embed the high-dimensional data coordinates into a lower-dimensional space such that the diffusion distance in the input space becomes Euclidean distance in the embedding space. It was shown by both Coifman and Lafon [265] and Delaporte et al. [267] that the low-dimensional data representations $\boldsymbol{Z}$ that retain the diffusion distances in the Euclidean embedding space can be formed by the eigenvectors of the $d$ largest, nontrivial eigenvalues of the eigenproblem

$$\boldsymbol{P}^{(t)}\boldsymbol{v} = \lambda\boldsymbol{v} \tag{A-62}$$

Since the graph is fully-connected, the largest eigenvalue is trivial (i.e. $\lambda_1 = 1$) and its corresponding eigenvector $\boldsymbol{v}_1$ is discarded. Thus, $\boldsymbol{Z} \in \mathbb{R}^{N \times d}$ is given by

$$\boldsymbol{Z} = \left[ \lambda_2 \boldsymbol{v}_2, \lambda_3 \boldsymbol{v}_3, \ldots, \lambda_{d+1} \boldsymbol{v}_{d+1} \right] \tag{A-63}$$

Diffusion maps have been successfully applied to tasks in shape matching and gene expression analysis [1].

### A.1.4 Principal Curves, Surfaces and Manifolds

Principal Curves (PCs) were first proposed in a PhD thesis by Hastie in 1984 and are simply defined as lines or surfaces that pass through the "middle of a cloud representing a data set" [25, 268]. In terms of probability distributions, a principal curve satisfies the self-consistency property, which implies that any point on the curve is the average of all data points projected onto it. In fact, Principal Component Analysis (Section A.1.2.1) is a special case of PCs where the "middle structure" of data is assumed as a straight line (or hyper-plane) instead of a curve. Assuming data with zero mean, PCA looks for a hyper-plane passing through the origin with direction $\boldsymbol{w}_1$ with a linear mapping function $f(\boldsymbol{z}_n) = \hat{\boldsymbol{x}}_n = \boldsymbol{w}_1^T \boldsymbol{z}_n$ best approximates the data in the mean-square sense [224, 47]. The point $f(\boldsymbol{z}_n)$ is the orthogonal projection onto the line parameterized by $\boldsymbol{w}_1$. A new sample can be constructed as $\boldsymbol{x}_n^{'} = \boldsymbol{x}_n - f(\boldsymbol{z}_n)$. The previous procedure is repeated $d-1$ more times to discover the $d$ "most important" lines for representing the entirety of the dataset. Dimensionality reduction is achieved by replacing the data $\boldsymbol{x}_n$ with the corresponding collection of parameters $\boldsymbol{z}_n$ needed to project the sample onto its principal hyper-planes. As noted by Sorzano et al. [224], the objective is simply a linear regression on the input data. However, this process does not have to be restricted to lines (or hyper-planes). In general, it can be assumed that a given sample $\boldsymbol{x}_n$ may be approximated by $\boldsymbol{x}_n = f(\boldsymbol{z}_n) + \boldsymbol{\epsilon}$ where $f(\boldsymbol{z}_n)$ is an arbitrary (but sufficient) nonlinear mapping, and $\boldsymbol{\epsilon}$ is a term that captures the approximation error. One could substitute for a fixed family of curves (parabolic, hyperbolic, exponential,

polynomial, etc.). This is exactly a nonlinear regressor, and several methods based on artificial neural networks have been proposed: Nonlinear PCA [269], autoencoder networks [270] and self-organizing networks [24, 271]. This section will review popular methods for principal manifold estimation, including approaches based on deep-learning, self-organizing maps, neural gases and elastic maps.

#### A.1.4.1 Deep Learning

Many natural phenomena behave in a nonlinear way, meaning that the observed data define a curved subspace in the original feature space [218]. One method used to model this nonlinearity is to utilize a deep feedforward network, also often called feedforward neural network or multilayer perceptron (MLP) [270]. The goal of a feedforward network is to approximate a function $f$ which maps an input $\boldsymbol{x}$ to a desired output $\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{\theta})$. These models are called "networks" because they are typically represented as a composition of functions (usually perceptrons). In the standard perceptron model, the output is formed as a weighted combination of the inputs which is then passed through a nonlinear activation function. Weights on the input signals are adjusted based on an error signal back-propagated through the network. This model is similar to human neurons, where some input pathways (synapses) are strengthened for particular tasks. The multilayer perceptron has been proven as a universal approximator [272, 273]. This means that (under certain conditions), neural models have the ability to approximate any continuous function. Thus, feedforward neural networks have shown remarkable performance in countless applications
[274, 275, 276, 277, 278, 210, 279, 211], including manifold learning through use of a special type of neural architecture called an autoencoder.

**Autoencoders and Nonlinear PCA** Autoencoder neural networks (AE) fall into the category of neural architectures known as autoassociative networks [280]. The goal of an autoencoder is, given an input sample, to produce an exact copy at its output. Internally, the network has a hidden layer $\boldsymbol{h}$ that describes the latent code used to

represent the input [270]. Essentially, the network has two parts. The first part represents the encoder $\Phi_{encode} : \mathcal{X} \rightarrow \mathcal{Z}$, which maps the data sample coordinates $\boldsymbol{x} \in \mathbb{R}^D$ to the corresponding coordinates $\boldsymbol{z}$ in the $d$-dimensional subspace controlled by the architecture of the middle layer (or bottleneck). The second part of the network denotes the inverse function, or decoder $\Phi_{decode} : \mathcal{Z} \rightarrow \hat{\mathcal{X}}$. The job of the decoder is to learn an inverse mapping from $\boldsymbol{z}$ to the original data sample. Thus, $\Phi_{decode}$ approximates the assumed data generation process. In general, the AE is constrained either through its architecture or through a sparsity constraint. A loss function $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{x}})$ measures how closely the AE can reconstruct the output. A commonly used AE loss is MSE, which penalizes the estimated output from being different from the input, $\mathcal{L}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2$ [281]. Figure A-8 demonstrates an arbitrary feedforward autoencoder. The first half of the network projects the input data into a lower-dimensional space which is controlled by the size of the bottleneck layer. The decoder then attempts to reconstruct the input from the latent code.

Early investigation of modern autoencoders was performed by LeCun, Bourlard, Kamp, Hinton and Zemel beginning in 1986 [270, 2] and showed that having hidden layers in both the encoder and decoder enables the network to learn a Nonlinear PCA (NLPCA) [218]. Similarly, it has been shown that an autoencoder with only a single hidden layer will learn an equivalent transformation matrix to traditional PCA [2].

Autoencoders suffer from the same ailments as other neural networks, such as requiring large amounts of data to cope with extreme parameterization. Moreover, vanilla AEs are unsupervised models and thus do not consider class information when determining the underlying latent features. So while AEs are capable of learning powerful feature representations, they may not be optimal for discrimination tasks. Despite these drawbacks, autoencoders have been applied successfully in countless tasks [210, 279, 211, 212], including: feature representation learning, denoising, outlier detection, domain adaptation and anomaly detection in remote sensing [2]. Additionally,

alternative autoencoder architectures have been developed, such as the variational AE (VAE) [17, 282] which assume the bottleneck layer to be the parameters of the latent code generating distributions and graph autoencoders (GAE) [283] which are capable of learning latent embeddings of graph-structured data.

**Graph Convolutional Networks** Geometric deep learning, as described by [284], encapsulates emerging techniques which generalize deep neural models to non-Euclidean domains such as graphs and manifolds. This umbrella of models includes graph autoencoders (GAEs), which map graph nodes into a latent feature space and decode graph information from the latent node representations [283, 285, 286]. Similarly to the standard autoencoder, GAEs can be used to learn lower-dimensional representations of the input data (graphs in this case) or to generate new data from latent representations. The review by Wu et al. [283] provides a detailed summary of GAE models in the literature to date.

#### A.1.4.2 Self-Organizing Map (SOM)

Self-Organizing Maps (SOM) or Kohonen Networks [24] are among the most well-known methods for discrete principal manifold estimation [287, 224]. SOMs belong to the category of neural networks which use a technique of competitive learning called self-organization to learn spatially-organized intrinsic representations of features [288]. Self-organizing maps are initialized by a pre-defined set of neurons which are typically arranged in a 2 or 3-dimensional grid. Each neuron is associated with a set of weights which represent their corresponding locations in the input space. When presented an input pattern (or stimulus), neurons compete among themselves for the representation of the input. Winning neurons strengthen their weights or relationships with this input using and adaption of Hebb's Rule, which states that the change to a synaptic weight is proportional to the correlation between the input and its associated output [289]. As samples are introduced to the network during training, only a neighborhood of cells give an active response to the current input sample. These neurons are pulled toward the

location of the input pattern according to how well they represent the input. Given enough training iterations, the neurons in the lattice will disperse such that the spatial locations or coordinates of cells in the network correspond to different modes of the input distribution. In this manner, Kohonen networks are able to utilize simple heuristics to form discrete topological mappings of the input space which represent the principal data manifold [287].

The self-organizing map is also a form of vector quantization (VQ). The purpose of VQ is to approximate a continuous probability density function $p(\boldsymbol{x})$ of input vectors $\boldsymbol{x}$ using a finite number of codebook vectors, $\boldsymbol{w}_k$, $k = 1, 2, \ldots, K$. After the "codebook" is chosen, the approximation of $\boldsymbol{x}$ involves finding the reference vector, $\boldsymbol{w}_c$ closest to $\boldsymbol{x}$. The "winning" codebook vector for sample $\boldsymbol{x}$ satisfies the following:

$$||\boldsymbol{x} - \boldsymbol{w}_c|| = \min_k ||\boldsymbol{x} - \boldsymbol{w}_k|| \tag{A-64}$$

The algorithm operates by first initializing a spatial lattice of codebook elements (also called "units"), where each unit's representative is in $\boldsymbol{w}_k \in \mathbb{R}^D$ where $D$ is the dimensionality of the input samples $\boldsymbol{x}$. The training process proceeds as follows. A random sample is selected and presented to the network at iteration $t$ and each unit determines its activation by computing dissimilarity (typically Euclidean distance). The unit who's codebook vector provides the smallest dissimilarity is referred to as the winner.

$$c(t) = \arg\min_k \mathcal{D}(\boldsymbol{x}(t), \boldsymbol{w}_k(t)) \tag{A-65}$$

Both the winning vector and all vectors within a neighborhood of the winner are updated toward the sample by

$$\boldsymbol{w}_k(t+1) = \boldsymbol{w}_k(t) + \alpha(t) \cdot h_{ck}(t) \cdot [\boldsymbol{x}(t) - \boldsymbol{w}_k(t)] \tag{A-66}$$

where $\alpha(t)$ is a learning rate which decreases over time and $h_{ck}(t)$ is a neighborhood function which is typically unimodal and symmetric around the location of the winner

which monotonically decreases with increasing distance from the winner. A radial basis kernel is typically chosen for the neighborhood function as

$$h_{ck}(t) = \exp\left(-\frac{||\boldsymbol{w}_c - \boldsymbol{w}_k||^2}{\beta(t)}\right) \tag{A-67}$$

where the top expression represents the Euclidean distance between units $c$ and $k$ with $\boldsymbol{w}_k$ representing the 2-D or 3-D location of unit $k$ in the lattice. The neighborhood kernel's bandwidth is typically initialized to a value which covers a majority of the input space and decreases over time such that solely the winner is adapted toward the end of the training procedure.

The SOM essentially performs density estimation of high-dimensional data and represents it in a 2 or 3-D representation. At test time, the dissimilarity between each unit in the map and an input sample are computed. This dissimilarity can be used to effectively detect outliers, thus making the SOM a robust method which can provide confidence values for it's representation abilities of the principal manifold. A downside of the SOM, however, is its lack of proven convergence criteria.

Despite the lack of theoretical training guarantees, SOMs have been applied to thousands of applications [287], which are too numerous to list here. Examples, however, include tasks in: speech recognition, finance, computer network traffic analysis, manufacturing, image analysis [288, 23], robotics, control of diffusion processes, optimization problems, adaptive telecommunications, image compression, sentence understanding, radar classification of sea-ice [24], cross-modal information processing, text and document mining, gene expression data analysis and discovery, novelty detection, computer animation, principal curve and surface discovery, data visualization [287], hand-written word classification [290] and explosive hazard detection [291]. Moreover, numerous extensions to SOMs have been proposed, including adaptations which allow the lattice to grow and shrink to fit the input data space [288], an approach

for space visualization [287] and a supervised approach known as Learning Vector Quantization (LVQ) [292].

#### A.1.4.3 Growing Neural Gas (GNG)

Growing Neural Gas (GNG) Networks [271] are similar to the standard SOM, except that network topology is not fixed [224]. This approach is called a "neural gas" because the codebook vectors are allowed to move around the data space similar to the Brownian motion of gas molecules in a closed container [293]. The primary difference between GNG and SOM is that the network is allowed to shrink and grow to better fill the data space [23, 294]. The dynamic topology is typically controlled by an edge age-based strategy. Similarly to the SOM, growing neural gas networks train based on heuristics and have not been proven to converge to an optimal solution. Additionally, constraints must be put into place to avoid over-fitting the data space. Despite these nuances, GNGs have proven successful in a variety of tasks. This is likely because the automatic topology learning exhibited by these networks allows them to learn complex manifolds which may have locally-different intrinsic dimensionality [25]. Thus, GNG networks have been applied successfully to tasks in motion detection [295], visualization of high-dimensional data, web mining, classification, image deblocking [296], computer vision, robotics, color quantization, video clustering [23] and foreground detection [294].

#### A.1.4.4 Elastic Maps and Nets

Elastic Maps and Nets [268] (also called Principal Graphs) are a mid-way between self-organizing maps (Section A.1.4.2) and the generative topographic mapping (Section A.1.5.2). Essentially, elastic maps represent a mathematical analogy of a set of elastic springs embedded in the data space which is used to approximate a low-dimensional principal manifold. Just as with the SOM, elastic maps are represented by a set of

neurons (nodes), $\boldsymbol{w}_k \in \mathbb{R}^D$, $k = 1, 2, \ldots, K$. Each sample in the data set is assigned to a class based on its closest neuron

$$C_k = \{\boldsymbol{x} | \boldsymbol{w}_k \text{ is the closest codebook vector to } \boldsymbol{x}\} \tag{A-68}$$

The approximation energy for the entire map measures the representation ability of each codebook vector to approximate the data assigned to it

$$U_A = \frac{1}{2} \sum_{k=1}^{K} \sum_{\boldsymbol{x} \in C_k} ||\boldsymbol{x} - \boldsymbol{w}_k||^2 \tag{A-69}$$

This is analogous to the energy of the springs with unit elasticity which connect each data point to the codebook which is the most representative. Additional structure is also enforced on the map to simulate stretching and bending. Specifically, some pairs of nodes $(\boldsymbol{w}_m, \boldsymbol{w}_n)$ are connected by elastic edges and some triplets of neurons $(\boldsymbol{w}_m, \boldsymbol{w}_n, \boldsymbol{w}_o)$ form bending ribs. The stretching energy and bending energy on these sets $E$ and $R$ are then defined by

$$U_E = \lambda \frac{1}{2} \sum_{(\boldsymbol{w}_m, \boldsymbol{w}_n) \in E} ||\boldsymbol{w}_m - \boldsymbol{w}_n||^2 \tag{A-70}$$

$$U_R = \mu \frac{1}{2} \sum_{(\boldsymbol{w}_m, \boldsymbol{w}_n, \boldsymbol{w}_o) \in R} ||\boldsymbol{w}_m - 2\boldsymbol{w}_n + \boldsymbol{w}_o||^2 \tag{A-71}$$

where $\lambda$ and $\mu$ are the stretching and bending moduli, respectively. The total energy of the map is given as

$$U_T = U_A + U_E + U_R \tag{A-72}$$

Essentially, the first term accounts for the fidelity of the data representations, the second term reinforces smoothness of the manifold by favoring similar neighbors and the third term imposes smoothness by stating that a codebook must be similar to the average of its neighboring nodes [224]. The position of the nodes $\{\boldsymbol{w}_k\}$ is given by the mechanical

equilibrium of the elastic map, or the locations which minimize the total energy $U_T$. In practice, this is typically solved by the Expectation-Maximization algorithm, which guarantees a local minimum of $U_T$.

Unlike the SOM, but similarly to the GNG, elastic nets can add or delete neurons adaptively. This allows them to fit very complex manifolds. The elastic map approach has led to practical applications in data visualization, data recovery, visualization of genetic texts and recovery of geophysical time series [268].

### A.1.5 Probabilistic Latent Variable Models (LVM)

The probabilistic interpretation of latent feature learning can be formulated as attempting to discover the generating distribution for the low-dimensional factors of variation (latent random variables) that describe a distribution over the observed, high-dimensional data [2, 47]. While many manifold learning and dimensionality reduction models are regarded as defining a projection from a $D$-dimensional feature space into a lower $d$-dimensional space, a latent variable model is defined by a mapping from the latent space into the high-dimensional data space [19]. The goal of latent variable models in terms of low-dimensional feature learning is to discover the parameters of an invertible mapping from the latent generating distribution to the high-dimensional data distribution which maximize the likelihood of the observed data.

Probabilistic methods provide two possible modeling paradigms for inferring latent variables, directed and undirected graphical models, which differ in their parameterizations of the joint distribution $p(\boldsymbol{x}_n, \boldsymbol{z}_n)$. The remainder of this section focuses on directed graphical models which follow a particular formulation called Factor Analysis.

#### A.1.5.1 Factor Analysis (FA) and Probabilistic PCA (PPCA)

Factor Analysis (FA) is a special case of a directed latent variable model that assumes the distribution of high-dimensional random variables is induced by a generalized linear mapping of the latent random variables [47, 219, 215, 297].

A common assumption is that the high-dimensional data are Gaussian distributed and the the conditional likelihood is given as

$$p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{W}\boldsymbol{z}+\boldsymbol{\mu},\boldsymbol{\Psi}) \tag{A-73}$$

where $\boldsymbol{W}$ is a $D \times d$ matrix known as the factor loading matrix and $\boldsymbol{\Psi}$ is a $D \times D$ covariance matrix. $\boldsymbol{\Psi}$ is assumed to be diagonal since the goal of of the model to to "force" $\boldsymbol{z}$ to explain the correlation between the data instead of "baking it in" to the covariance. $\boldsymbol{\theta}$ denotes the parameters of the model. Typically a prior is defined which captures any presumed knowledge about the distribution of $\boldsymbol{z}$. The data likelihood conditioned solely on the parameters is obtained by integrating over the latent variables

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = \int p(\boldsymbol{x}|\boldsymbol{z},\boldsymbol{\theta})p(\boldsymbol{z})d\boldsymbol{z} \tag{A-74}$$

and the model parameters can be obtained by maximum likelihood.

By assuming $\boldsymbol{\Psi} = \beta\mathbb{I}_D$, the factor analysis model provides a probabilistic formulation of PCA (PPCA) [219]. A typical choice on this model is a Gaussian-Gaussian conjugate prior pair where the mean of the data likelihood is a linear function of the latent inputs. As an example, the prior over the hidden data representations can be expressed as a Gaussian distribution

$$p(\boldsymbol{z}_n) = \mathcal{N}(\boldsymbol{z}_n|\boldsymbol{\mu}_0,\boldsymbol{S}_0) \tag{A-75}$$

and the data likelihood is provided as a multivariate Gaussian (given in Equation A-73). Classical PCA assumes the data covariance to be $\boldsymbol{\Psi} = \sigma^2\boldsymbol{I}$ with $\beta \to 0$, thus the model is deterministic. Alternatively, when $\beta > 0$,the projection is no longer orthogonal since it is shrunk toward the prior mean. The trade-off is that the reconstructions of $\boldsymbol{x}_n$ will be closer to the data mean.

While the typical approach for fitting PCA is to use eigen-decomposition or Singular Value Decomposition (SVD), PPCA can be fit with the Expectation-Maximization (EM)

algorithm (which may be more computationally efficient for high-dimensional data [47]). Once the model parameters have been estimated, dimensionality reduction can be performed by applying the inverse mapping

$$\boldsymbol{z}_n = \boldsymbol{W}^T(\boldsymbol{x}_n - \boldsymbol{\mu}) \tag{A-76}$$

The projections onto the discovered principal axes, however, may not be optimal in the mean-squared sense because they will not be orthogonal.

Murphy [47] showed examples on how supervision has been applied to PCA. Supervised PCA or Bayesian factor regression is a model like PCA, except that the target variable (or label), $l_n$ is taken into account when learning the low-dimensional embedding. For the case of binary classification, the Bayesian model can be decomposed into the following elements:

$$p(\boldsymbol{z}_n) = \mathcal{N}(0, \mathbb{I}_d) \tag{A-77}$$

$$p(l_n|\boldsymbol{z}_n) = \text{Ber}(\text{sigm}(\boldsymbol{w}_l^T \boldsymbol{z}_n)) \tag{A-78}$$

$$p(\boldsymbol{x}_n|\boldsymbol{z}_n) = \mathcal{N}(\boldsymbol{W}_x \boldsymbol{z}_n + \boldsymbol{\mu}_x, \beta \mathbb{I}_D) \tag{A-79}$$

where the distribution of a label given a latent sample is defined by a Bernoulli distribution. By placing weighting terms on the components of the likelihood function, Supervised PCA can become discriminative.

Moreover, Gaussian Processes (GPs) have been used to learn nonlinear mappings between the input and latent feature spaces [1, 298], effectively creating nonlinear PPCA. A supervised GP latent factor model for dimensionality reduction was proposed by Gao et al. [299].

### A.1.5.2 Generative Topographic Mapping (GTM)

The Generative Topographic Mapping (GTM) [19] is a nonlinear latent variable model proposed as alternative to the self-organizing map. Specifically, the GTM provides solutions to known deficiencies exhibited by SOMs, such as the lack of an objective function, proofs of convergence, theoretical basis for choosing a learning rate parameter and guarantee of topological ordering. The GTM defines a nonlinear, parametric mapping from a $d$-dimensional latent space to a $D$-dimensional data space $\boldsymbol{x} \in \mathbb{R}^D$, where typically $d \ll D$ [293]. A continuous and differentiable function $f(\boldsymbol{x}; \boldsymbol{W})$ maps every point in the latent space to a point in the data space. This function can be any arbitrary, parametric mapping such as a feedforward neural network. To learn the parameters, GTM attempts to maximize the negative log-likelihood of the latent coordinates given the high-dimensional data and model parameters. Assuming a probability distribution $p(\boldsymbol{z})$, the GTM derivation begins by assuming each high-dimensional sample $\boldsymbol{x}$ is generated from a nonlinear mapping with additive Gaussian noise

$$p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{W}, \beta) = \left(\frac{1}{2\pi\beta}\right)^{D/2} \exp\left(-\frac{1}{2\beta}||f(\boldsymbol{z}; \boldsymbol{W}) - \boldsymbol{x}||^2\right) \tag{A-80}$$

where $\beta$ is the variance of the radially-symmetric Gaussian function centered on $f(\boldsymbol{z}; \boldsymbol{W})$. The density over the input data space is obtained by integrating over the latent space by

$$p(\boldsymbol{x}|\boldsymbol{W}, \beta) = \int p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{W}, \beta) p(\boldsymbol{z}) d\boldsymbol{z} \tag{A-81}$$

which is inherently intractable. By defining $p(\boldsymbol{z})$ as a set of $K$ equally weighted delta functions on a regular grid,

$$p(\boldsymbol{z}) = \frac{1}{K} \sum_{k=1}^{K} \delta(\boldsymbol{z} - \boldsymbol{z}_k) \tag{A-82}$$

the integral turns into a summation

$$p(\boldsymbol{x}|\boldsymbol{W},\beta) = \frac{1}{K}\sum_{k=1}^{K} p(\boldsymbol{x}|\boldsymbol{z}_k,\boldsymbol{W},\beta) \tag{A-83}$$

which represents the model as a constrained mixture of Gaussians. The data log-likelihood is given by

$$J(\boldsymbol{W},\beta) = \sum_{n=1}^{N} \ln\left(\frac{1}{K}\sum_{k=1}^{K} p(\boldsymbol{x}|\boldsymbol{z}_k,\boldsymbol{W},\beta)\right) \tag{A-84}$$

Given a set of data $\boldsymbol{X} \in \mathbb{R}^{N\times D}$, the log-likelihood function in Equation A-84 can be maximized by the Expectation-Maximization algorithm.

There are two obvious limitations of the GTM. First, the computational complexity grows exponentially with the assumed intrinsic dimensionality. As noted by Peña et al.[293], however, this is typically not an issue if the goal is visualization. Additionally, poor initialization can lead the algorithm into local optima. However, the GTM has been used extensively for visualization tasks, clustering and applications in molecular biology [19, 293, 224, 25]. Additionally, successful variations of the GTM have been developed, such as the Topographic Product of Experts, Harmonic Topographic Map, Topographic Neural Gas and Inverse-Weighted K-Means Topology-Preserving Map [293].

#### A.1.5.3 Manifold Charting

Charting is the problem of assigning a low-dimensional coordinate system to data points in a high-dimensional sample space. Manifold Charting, introduced by Brand [300], constructs low-dimensional data representations by aligning mixtures of factor analyzers (MoFA) or mixtures of probabilistic PCA (MoPPCA) models. Essentially, manifold charting minimizes a convex cost function measuring the amount of disagreement between the linear models on the global coordinates of the data. This is done in two steps: 1.) computing a mixture of locally linear models on the data and 2.) aligning the models in order to obtain the low-dimensional data representations. Manifold charting begins by performing EM to learn a mixture of factor analyzers in order to obtain

$M$ low-dimensional data representations $\boldsymbol{w}_{nm}$ and corresponding responsibilities $r_{nm}$ for each sample $\boldsymbol{x}_n$. The responsibilities $r_{nm}$ describe how much sample $\boldsymbol{x}_n$ belongs to model $m$, and satisfy $\sum_{m=1}^{M} r_{nm} = 1$. A linear mapping $\boldsymbol{M}$ is found from the local data representations $\boldsymbol{w}_{nm}$ to the global coordinates $\boldsymbol{z}_n$ that minimize the objective

$$J(\boldsymbol{Z}) = \sum_{n=1}^{N} \sum_{m=1}^{M} r_{nm} ||\boldsymbol{z}_n - \boldsymbol{z}_{nm}||^2 \tag{A-85}$$

where $\boldsymbol{z}_n = \sum_{o=1}^{M} r_{no} \boldsymbol{z}_{nm}$ and $\boldsymbol{z}_{nm} = \boldsymbol{w}_{nm} \boldsymbol{M}$. This objective function implies that all models for which a datapoint has a high responsibility should agree on the final coordinate for that datapoint. As shown by van der Maaten [1], the cost function can be reformulated as a generalized eigenvalue problem in which the $d$ smallest nonzero eigenvectors form the linear transformation matrix from the local data representations $\boldsymbol{W}$ to the globally-aligned, low-dimensional data representations $\boldsymbol{Z}$.

Manifold charting has been shown to be more robust to noise than alternative approaches such as Isomap and LLE [300], however, the main weakness of manifold charting is that fitting the MoFA is susceptible to the presence of local maxima in the log-likelihood function.

### A.1.6 High-Dimensional Data Visualization

While not directly related to classification, it is important for the reader to be aware of the current SOA in manifold learning. At the time this document was written, much work was being performed in dimensionality reduction for the visualization of extremely large datasets consisting of hundreds (or thousands) of features. Specifically, the remainder of this section discusses three related, SOA approaches for data visualization. t-SNE, LargeVis and UMAP are compared in terms of their differences in use of manifold learning for the visualization of large, high-dimensional datasets.

#### A.1.6.1 Stochastic Neighbor Embedding (SNE and t-SNE)

t-Distributed Stochastic Neighbor Embedding [301] was proposed as an extension to Stochastic Neighbor Embedding (SNE) [302]. SNE is intended as a visualization tool

for high dimensional datasets. Essentially, SNE finds low-dimensional data coordinates by defining pairwise probabilities between all samples in the input data space. It then assumes that the low-dimensional embedding coordinates should be distributed according to the same conditional probabilities. SNE begins by defining conditional probabilities between all points in the training set which measure their likelihood of being spatial neighbors. Given a datapoint $\boldsymbol{x}_n$, the probability of selecting another point $\boldsymbol{x}_m$ as its neighbor is given as

$$p_{m|n} = \frac{\exp(-||\boldsymbol{x}_n - \boldsymbol{x}_m||^2/2\beta_n)}{\sum_{o \neq p} \exp(-||\boldsymbol{x}_p - \boldsymbol{x}_o||^2/2\beta_n)} \tag{A-86}$$

where $\beta_n$ is the variance of the RBF kernel centered on sample $\boldsymbol{x}_n$. Therefore, points in a local neighborhood to sample $\boldsymbol{x}_n$ will have a higher probability of being selected as a neighbor. Similarly, the pairwise conditional probabilities between all low-dimensional samples can be defined by

$$q_{m|n} = \frac{\exp(-||\boldsymbol{z}_n - \boldsymbol{z}_m||^2)}{\sum_{o \neq p} \exp(-||\boldsymbol{z}_p - \boldsymbol{z}_o||^2)} \tag{A-87}$$

Since the similarity of a sample with itself is not important, $p_{n|n}$ and $q_{n|n}$ are set to zero. The intuition behind this formulation is that, if the low-dimensional coordinates $\boldsymbol{z}_n$ and $\boldsymbol{z}_m$ correctly model the similarity between the high-dimensional data points $\boldsymbol{x}_n$ and $\boldsymbol{x}_m$, the conditional probabilities $p_{m|n}$ and $q_{m|n}$ will be equal [301]. Thus, it is intuitive that the Kullback-Leibler (KL) divergence (which measures the cross-entropy between probability densities) would make an appropriate cost function between the high and low-dimensional probabilities. The objective function of SNE is given by

$$J = \sum_{n=1}^{N} KL(P_n||Q_n)) = \sum_{n=1}^{N} \sum_{m=1}^{N} p_{m|n} \log \frac{p_{m|n}}{q_{m|n}} \tag{A-88}$$

where $P_n$ denotes the conditional probability distribution over all other high-dimensional datapoints given $\boldsymbol{x}_n$, and $Q_n$ represents the conditional probability distribution over all other low-dimensional embedding coordinates given $\boldsymbol{z}_n$. Given that the KL divergence is

asymmetric, it can be observed that using widely separated points in the embedding space to model points close in the input space accrues a large error. However, the cost of using nearby points in the embedding space to represent points far in the data space is small. Therefore, the SNE cost focuses on retaining the local structure of data in the embedding space. This problem can be optimized using gradient descent.

A problem with is that, due to the curse of dimensionality, points in "middle distances" get squashed closely together in the low-dimensional embedding space. This is known as the crowding problem, and stems from the fact that in high-dimensional spaces, the surface of the a hypersphere (Guassian distribution) grows much more quickly with its radius as compared to a hypersphere in a low-dimensional space. t-SNE handles this problem by forcing the optimization to "spread-out" the medium distance points. This is done by utilizing a symmetric KL-divergence and by giving the low-dimensional density a longer tail. Using a Student t-distribution, the conditional neighbor probabilities of the low-dimensional data coordinates become

$$q_{m|n} = \frac{(1 + ||\boldsymbol{x}_n - \boldsymbol{x}_m||^2)^{-1}}{\sum_{o \neq p} 1 + ||\boldsymbol{x}_p - \boldsymbol{x}_o||^2)^{-1}} \tag{A-89}$$

and optimization of the KL-divergence is performed with gradient descent.

Whereas many manifold learning and dimensionality reduction methods in the literature can only capture global or local data structure, t-SNE is effectively able to capture both. Thus, t-SNE has been applied successfully to 2D and 3D visualization of datasets consisting of millions of samples with hundreds of features [303, 304]. However, t-SNE suffers from a few drawbacks. t-SNE is bottlenecked by the size of the dataset due to the construction of a $k$-NN graph. Moreover, the efficiency of the graph visualization step deteriorates significantly as the data becomes large and the performance of t-SNE is highly dependent on its hyperparameters. Also, since t-SNE directly optimizes the embeddings, it cannot be applied directly to new data. t-SNE lead as the SOA data

visualization tool for approximately ten years after its inception, but because of its deficiencies, LargeVis [303] and UMAP [304] were eventually proposed.

#### A.1.6.2 LargeVis

LargeVis was proposed by Tang et al. [303] for visualizing large-scale and high-dimensional data in a low-dimensional space (typically 2D or 3D). Specifically, LargeVis addresses the scaling inefficiencies exhibited by t-SNE by adopting the "a neighbor of my neighbor is my neighbor" approach for $k$-NN graph construction. Essentially, the algorithm builds a few random projection trees to quickly construct a nearest neighbor graph. Since the resulting graph is likely not very accurate, a search is performed for each node and neighbors of neighbors are also given connectivity. This method is used to quickly build an approximate neighbor graph and is much more computationally efficient than constructing a traditional $k$-NN graph. Weights and edges are assigned in the same manner as SNE (Equation A-86). Once an approximate $k$-NN graph is constructed, a probabilistic model is used to discover the low-dimensional embedding coordinates which preserve the similarities of samples in the high-dimensional space. The likelihood of the graph is given by

$$\begin{aligned} J &= \prod_{(m,n)\in E} p(e_{mn}=1)^{w_{mn}} \prod_{(m,n)\in E} (1-p(e_{mn}=1))^{\gamma} \\ &\propto \sum_{(m,n)\in E} w_{mn} \log p(e_{mn}=1) + \sum_{(m,n)\in \bar{E}} \gamma \log\left(1-p(e_{mn}=1)\right) \end{aligned} \tag{A-90}$$

where $p(e_{mn}=w_{mn}) = p(e_{mn}=1)^{w_{mn}}$ is the likelihood of observing a weighted edge, $\bar{E}$ is the set of vertices that are not observed and $\gamma$ is a weight assigned to the pairs of vertices without edges between them. Maximizing the first part of Equation A-90 ensures that similar datapoints in the input data space will be close in the embedding space. The second part of the equation models the likelihood of all vertex pairs without edges. Maximizing this part ensures that dissimilar points are embedded far from each other. The graph likelihood is maximized efficiently by asynchronous gradient descent.

LargeVis was able to provide 2D and 3D visualizations of millions of data points consisting of hundreds of features from datasets representing hand-written digits, social networks, text documents and images [303]. While LargeVis provides a significant increase in computational efficiency as compared to t-SNE, it still places more importance on preserving local distances, as compared to a combination of local and global.

#### A.1.6.3 Uniform Manifold Approximation and Projection (UMAP)

Similarly to LargeVis, Uniform Manifold Approximation and Projection (UMAP) [304] attempts to address the deficiencies exhibit by t-SNE. Specifically, t-SNE has difficulties working with high dimensional data diectly due to memory limitations and can only practically embed data into two or three dimensions [303]. UMAP uses manifold approximation and local fuzzy simplicial set representations to construct a topological representation of the high dimensional data. The layout of data in the low dimensional space is then optimized to minimize the error between the two topological representations. While UMAP may appear wildly different from t-SNE at first glance, there are actually only a few key differences which make it more computationally efficient and scalable.

As with t-SNE (Section A.1.6.1) and LargeVis (Section A.1.6.2), UMAP can be described in two phases. A weighted nearest neighbor graph is constructed in the first phase and a mimetic, low-dimensional representation is computed in the second. The differences between t-SNE, LargeVis and UMAP amount to the specific details on how the neighborhood graphs are constructed and how the low-dimensional embeddings are computed.

t-SNE defines the high-dimensional input probabilities by normalizing and symmetrizing RBF kernel similarities, which utilizes Euclidean distance (Equation A-86).

The symmetrized neighbor probabilities are defined by

$$p_{mn} = \frac{p_{m|n} + p_{n|m}}{2N} \tag{A-91}$$

The similarities in the low-dimensional embedding space are then defined by a Student t-distribution with a single degree of freedom on the Euclidean distance (Equation A-89). The objective function of t-SNE is given as the KL-divergence between the high and low-dimensional probability distributions

$$J_{t-SNE} = \sum_{m \neq n} p_{mn} \log \frac{p_{mn}}{q_{mn}} = \sum_{m \neq n} p_{mn} \log p_{mn} - p_{mn} \log q_{mn} \tag{A-92}$$

Because the computation of both $p_{mn}$ and $q_{mn}$ requires calculation over all pairs of training points, different approaches have been suggested to improve efficiency. LargeVis, for example, considers only the approximate nearest neighbors for each sample and abandons the graph weight normalization. Instead of KL-divergence, LargeVis maximizes the graph likelihood

$$J_{LV} = \sum_{m \neq n} p_{mn} \log w_{mn} + \gamma \sum_{m \neq n} \log \left(1 - w_{mn}\right) \tag{A-93}$$

where $p_{mn}$ and $w_{mn}$ are defined as in Barnes-Hut t-SNE. $\gamma$ is a parameter which controls the repulsion force (second part of the objective) relative to the attractive force (first part). The first part of this objective resembles the optimizable portion of the KL-divergence used by t-SNE, except with the substitution of $w_{mn}$ for $q_{mn}$.

UMAP's cost function can be described by the cross-entropy between $v$ and $w$

$$J_{UMAP} = \sum_{m \neq n} v_{mn} \log \left(\frac{v_{mn}}{w_{mn}}\right) + \left(1 - v_{mn}\right) \log \left(\frac{1 - v_{mn}}{1 - w_{mn}}\right) \tag{A-94}$$

which can be rearranged into two constant terms and two optimizable terms

$$J_{UMAP} = \sum_{m \neq n} v_{mn} \log v_{mn} + \left(1 - v_{mn}\right) \log \left(1 - v_{mn}\right) - v_{mn} \log w_{mn} - \left(1 - v_{mn}\right) \log \left(1 - w_{mn}\right) \tag{A-95}$$

and has a similar form to LargeVis without the $\gamma$ term and the requirement of matrix-wise normalization in the high-dimensional space. The high-dimensional data similarities $v_{m|n}$ in Equation A-95 are the local fuzzy simplicial set memberships based on smoothed nearest neighbor distances

$$v_{m|n} = \exp\left(\frac{-\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n) - \rho_n}{\beta_n}\right) \tag{A-96}$$

Just as with LargeVis, $v_{m|n}$ is calculated from only the $k$ approximate nearest neighbors and $v_{m|n} = 0$ for all other samples. $\mathcal{D}(\boldsymbol{x}_m, \boldsymbol{x}_n)$ is an arbitrary distance between high-dimensional samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ which is not required to be Euclidean distance. $\rho_n$ is the distance to the nearest neighbor of sample $n$ and $\beta_n$ controls the bandwidth of the exponential distribution. UMAP leverages ideas of earlier-developed approaches such as Laplacian Eigenmaps (Section A.1.3.8) which assume that data on the manifold are uniformly distributed. While this constraint can easily be enforced in the embedding space, the training sets of remotely sensed data typically do not exhibit uniform distributions. Because of this, using a fixed metric may leave samples unconnected in the graph. $\rho_n$ plays an important role in that it ensures local connectivity by defining a locally adaptive kernel for each datapoint which allows the metric to vary between each point.

Symmetrization is performed by the fuzzy set union using the probabilistic t-conorm

$$v_{mn} = (v_{m|n} + v_{n|m}) - v_{m|n}v_{n|m} \tag{A-97}$$

and the low dimensional data similarities are given by

$$w_{mn} = (1 + a||\boldsymbol{z}_m - \boldsymbol{z}_n||_2^{2b})^{-1} \tag{A-98}$$

where $a$ and $b$ are user-defined hyper-parameters. Setting $a = b = 1$ results in the Student t-distribution used by t-SNE. With all terms defined, the objective function in Equation A-95 can be optimized efficiently with stochastic gradient descent.

Experiments by McInnes et al. [304] showed that UMAP was both faster than t-SNE (due to efficient neighbor search and lack of normalization) and was capable of preserving more global data structure. While the KL-divergence cost used in t-SNE heavily penalizes mapping close points in the high-dimensional space with far points in the low-dimensional space, it is less concerned with mapping points far in the high-dimensional space with samples close in the embedding space. This reveals that t-SNE places emphasis on locality preservation but does not guarantee that global structure will be retained through the embedding. UMAP, on the other hand, penalizes for both cases through the use of a cross-entropy cost function. This effectively allows UMAP to retain (with trade-offs) both local and global data structure. Additionally, whereas t-SNE which uses random initialization, UMAP initializes the low-dimensional data coordinates through the Laplacian Eigenmaps algorithm which assumes uniformity in the embedding space. Removal of initialization randomness tends to make UMAP embeddings more repeatable. All in all, UMAP was shown to be competitive with the SOA t-SNE and LargeVis algorithms for data visualization tasks while providing improved speed and flexibility. UMAP has been used for visualization of data in bioninformatics, materials science and machine learning [304].

### A.1.7 Summary of Manifold Learning Algorithms

This literature review discusses a myriad of manifold learning techniques rooted in a variety of fundamental approaches. However, this review barely scratches the surface of proposed techniques. With such an expansive corpus of methods, it is often difficult to distinguish which approaches are “best” for any given application. This is, of course, application dependent and the optimal features for data visualization are likely very different from those for classification. However, each method reviewed demonstrates a unique approach toward learning which brings its own strengths and weaknesses. Table A-1 provides the fundamental strategies used by the manifold learning algorithms reviewed in this chapter, ranging from classic to SOA approaches. While manifold

learning techniques are inherently unsupervised, this review showed that many supervised and semi-supervised methods have been developed from base algorithms. However, the current state of the literature is greatly lacking in manifold learning and dimensionality reduction approaches designed for classification which can effectively address label uncertainty. In fact, none of the methods reviewed in Section A.1 are inherently capable of reliably learning useful, discriminative low-dimensional features from weakly-supervised data. Section 2.2 discusses dimensionality reduction and manifold learning methods designed specifically for discrimination under the weak learning paradigm.

## A.2 Metric Embedding

The concepts of "near" and "far" are very powerful and useful utilities in everyday life. They classify the relationship between two "primitives" as being similar or dissimilar, as well as the degree of compatibility [4]. As an example, a medical doctor might consider a machine learning researcher and a software engineer as being similar (near), because they both perform research for computer applications. However, the same researcher and engineer would likely consider their jobs as being very disparate (far) based on the details of their work. In order to capture this abstraction of distance in a mathematical construct, a metric space is defined.

**Definition A.2.1.** Metric Space A metric space is an ordered pair $(\mathcal{X}, \mathcal{D})$ where $\mathcal{X}$ is a set and $\mathcal{D}$ is a metric on $\mathcal{X}$, or $\mathcal{D} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ such that $\forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \mathcal{X}$, the following holds:

1. Non-negativity: $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) \geq 0$
2. Identity: $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) = 0 \iff \boldsymbol{x} = \boldsymbol{y}$
3. Symmetry: $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) = \mathcal{D}(\boldsymbol{y}, \boldsymbol{x})$
4. Triangle Inequality: $\mathcal{D}(\boldsymbol{x}, \boldsymbol{z}) \leq \mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) + \mathcal{D}(\boldsymbol{y}, \boldsymbol{z})$

The non-negativity rule states that the metric evaluated on two instances must have a positive value or be equal to zero. From the Identity rule, we can see that the metric

may only be defined as zero if the two instances being evaluated are exactly the same (thus the dissimilarity is zero). The Symmetry rule states that a metric evaluated between two instances must be the same regardless of ordering. Finally, the Triangle Inequality says, intuitively, that the direct distance between two instances $\boldsymbol{x}$ and $\boldsymbol{z}$ is smaller than the distance between $\boldsymbol{x}$ and $\boldsymbol{y}$ plus the distance between $\boldsymbol{y}$ and $\boldsymbol{z}$. Both sides of the inequality will be equal if and only if $\boldsymbol{y}$ lies on the path between $\boldsymbol{x}$ and $\boldsymbol{z}$ on which the metric is defined.

The goal of metric embedding learning it to learn a function $f_\theta(\boldsymbol{x}) : \mathbb{R}^D \rightarrow \mathbb{R}^d$ which maps semantically similar points from the data input feature space of $\mathbb{R}^D$ onto metrically close points in $\mathbb{R}^d$. Similarly, $f_\theta$ should map semantically different points in $\mathbb{R}^D$ onto metrically distant points in $\mathbb{R}^d$. The function $f_\theta$ is parameterized by $\theta$ and can be anything ranging from a linear transformation to a complex non-linear mapping as in the case of deep artificial neural networks [305]. Let $\mathcal{D}(\boldsymbol{x}, \boldsymbol{y}) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ be a metric function measuring similarity or dissimilarity in the embedded space. For succinctness, $\mathcal{D}_{m,n} = \mathcal{D}(f_\theta(\boldsymbol{x_m}), f_\theta(\boldsymbol{x_n}))$ defines the dissimilarity between samples $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$, after being embedded. It should be noted that, unlike metric learning where the objective is to learn an appropriate metric to measure dissimilarity between samples in opposing classes, metric embedding attempts to learn a transformation function such that samples in the embedding space adhere to a pre-defined measure of similarity [305]. Metric embedding is often realized through weakly-supervised learning, where instead of labels, the data is accompanied with sets of preferences.

### A.2.1 Ranking Loss

Unlike other loss function such as mean-squared error or cross-entropy whose objective is to directly compare labels, values or sets of values assigned to a given input, the objective of ranking loss (also called margin loss) functions is to measure relative distances between sets of inputs. To use a ranking loss in a learning scenario, sets of inputs (usually two or three) are embedded through a transformation function into a

defined metric space. A metric is used to measure similarity (often Euclidean distance), and the transformation function is updated such that points which have a higher semantic similarity label are embedded more closely, and points which are dissimilar are pushed further apart, according to the metric. In this framework, the actual values of the embedded features are ignored. Instead, only the distances between them matter. While ranking losses have been developed which consider many datapoints at a time [306], the most popular ranking loss functions in the literature are based on contrastive [307] and triplet [308] losses.

#### A.2.1.1 Contrastive loss

Let $\boldsymbol{x} \in \mathcal{X}$ be data with corresponding instance-level labels $l \in \{l_1, \ldots, l_N\}$. The superscripts $\boldsymbol{x}^a, \boldsymbol{x}^p, \boldsymbol{x}^n$ are used to denote anchor, positive and negative samples, respectively. The anchor point is the sample of interest, while the positive is a sample of the same class (or deemed to be similar to the anchor) and the negative sample is a point of a different class (or dissimilar to the anchor). Contrastive loss functions take pairs of examples as input and learns and embedding function to predict whether two inputs are from the same class or not. Specifically, contrastive loss can be written as:

$$\begin{aligned}\mathcal{L}^{\alpha}_{\text{cont}}(\boldsymbol{x}_m, \boldsymbol{x}_n; f_\theta) =& \mathbf{1}\{l_m = l_n\}\mathcal{D}(f_\theta(\boldsymbol{x}_m), f_\theta(\boldsymbol{x}_n)) \\ &+\mathbf{1}\{l_m \neq l_n\}\max(0, \alpha - \mathcal{D}(f_\theta(\boldsymbol{x}_m), f_\theta(\boldsymbol{x}_n)))\end{aligned} \tag{A-99}$$

where $\alpha$ is a margin parameter imposing a distance between different classes to be larger than $\alpha$. Analyzing Equation A-99, it can be observed that this loss has two unique terms. If the label of $\boldsymbol{x}_m$ and $\boldsymbol{x}_n$ are the same (anchor and positive pair), only the first half of the loss is computed. This term is simply the dissimilarity between the samples in the embedding space. Ideally, this term should equate to zero, thus implying the terms are close to each other in the embedding space. If the labels of the samples are opposing (anchor and negative pair), the second term is computed. This equates to the maximum value between zero and the difference between the margin constraint and the

dissimilarity between the samples. Thus, if the distance between the anchor and negative is greater than the margin $\alpha$, the objective is met and the loss equates to zero. Otherwise, the loss is positive. Figure A-10 demonstrates the idea of contrastive loss. If given a positive anchor pair, the transformation function should embed the samples closer in the embedding space. If given a negative pair, the opposite should occur. It is obvious that with each sample pair the embedding function is only updated in one way (pushing or pulling) [306, 307].

#### A.2.1.2 Triplet loss

Whereas contrastive loss considers pairs of samples, triplet loss jointly optimizes between three samples $\{\boldsymbol{x}^a, \boldsymbol{x}^p, \boldsymbol{x}^n\}$, the anchor, a positive and a negative [305, 308, 309]. The fundamental idea behind triplet loss is that, for an input sample, we desire to shorten the distances between its embedded representation and those of similar examples, while simultaneously enlarging the distances between dissimilar examples [306]. Triplet loss can be written as:

$$\mathcal{L}^{\alpha}_{\text{tri}}(\boldsymbol{x}^a, \boldsymbol{x}^p, \boldsymbol{x}^n; f_\theta) = \max(0, \mathcal{D}(f_\theta(\boldsymbol{x}^a), f_\theta(\boldsymbol{x}^p)) - \mathcal{D}(f_\theta(\boldsymbol{x}^a), f_\theta(\boldsymbol{x}^n)) + \alpha) \tag{A-100}$$

From Equation A-100, it can be observed that the triplet loss is satisfied whenever the distance between the negative and anchor is greater than the distance between the corresponding positive sample and the anchor by a margin $\alpha$. Whereas contrastive loss simply pushes or pulls, triplet loss does both, simultaneously. This is depicted visually by Figure A-11.

The advantage of this formulation is that, while all points of the same class will eventually form a cluster, they are not required to collapse to a single point; they just need to be close to each other than to any point from a different class [305]. A major drawback of triplet loss, however, is that as datasets grow larger, the possible number of triplets grows cubically. This can cause practical issues as many of the triplets will likely already satisfy the margin constraints and will not contribute toward learning. Therefore,

Schroff et al. [308] argue that hard-mining is often imperative for the success of triplet-based methods. Essentially, three scenarios exists. Easy triplets are triplet sets which already satisfy the margin constraint, thus inducing zero loss. Hard triplets are sets where the negative sample is closer to the anchor than the positive. The loss is positive (and greater than $\alpha$). Semi-hard triplets are sets where the negative point is more distant to the anchor than the positive, but it is not greater than the margin $\alpha$, so the loss is still positive (but smaller than $\alpha$). Therefore, careful consideration should be taken in triplet construction to maximize the number of hard and semi-hard triplets shown to the learning algorithm.

Alternative approaches such as the ones by Sohn [306], Deng et al. [309] and Xu et al. [9] use modifications of contrastive or triplet loss to obtain feature representations which adhere to a desired metric embedding.

### A.2.2 Large-Margin K-Nearest Neighbors (LMNN)

Weinberger and Saul [310] explored the topic of ranking loss with the explicit goal of performing $k$-nearest neighbor classification in the learned embedding space [305]. This approach is based on the Large Margin Nearest Neighbor loss for optimizing $f_\theta$, defined as:

$$\mathcal{L}_{\text{LMNN}}(\theta) = (1-\mu)\mathcal{L}_{\text{pull}}(\theta) + \mu\mathcal{L}_{\text{push}}(\theta) \tag{A-101}$$

which is comprised of a pull-term that pulls data points toward its target neighbors, or its nearest neighbors from the same class, and a push-term that pushes data points from a different class. The term $\mu$ is a trade-off parameter used to control the priority of pushing or pulling. The loss terms are defined as:

$$\mathcal{L}_{\text{pull}}(\theta) = \sum_{m,n} \eta_{mn}\mathcal{D}(f_\theta(\boldsymbol{x}_m), f_\theta(\boldsymbol{x}_n)) \tag{A-102}$$

$$\mathcal{L}_{\text{push}}(\theta) = \sum_{m,n,o} \eta_{mn}(1 - l_{m,o}) \max(0, \mathcal{D}(f_\theta(\boldsymbol{x}_m), f_\theta(\boldsymbol{x}_n)) - \mathcal{D}(f_\theta(\boldsymbol{x}_m), f_\theta(\boldsymbol{x}_o)) + \alpha) \quad \text{(A-103)}$$

where $\eta_{mn} \in \{0, 1\}$ is a binary indicator variable used to express whether input $\boldsymbol{x}_n$ is a target neighbor of input $\boldsymbol{x}_m$ and $\alpha$ is a margin enforcing distance between the classes. Analyzing Equations A-102 and A-103, it can be observed that the pull term only penalizes large distances between inputs and target neighbors ($k$-nearest neighbors with the same class label). The push term, however, is comparable to the triplet loss, which enforces that the dissimilarities between the anchor sample $\boldsymbol{x}_m$ and all negative samples is greater than the dissimilarities between the anchor and its corresponding target neighbors by at least a margin $\alpha$. Figure A-12 demonstrates the pushing and pulling idea captured by LMNN. In the rest of the algorithm, a matrix $\boldsymbol{z}$ is learned which captures an appropriate Mahalanobis metric to measure the dissimilarity between samples, optimized for the LMNN loss.

### A.2.3 Siamese Neural Networks

Siamese Neural Networks are a unique neural architecture proposed by Bromley et al. [311] used to rank similarity between inputs. Essentially, Siamese networks use a base architecture to perform feature encoding. This base network is replicated to form twin networks which share parameters [307, 312]. The two networks are tied at the output, where a distance metric is used to compare the embeddings of the inputs to each twin. Figure A-13 demonstrates this idea. The base network can be any architecture (i.e. fully-connected, convolutional, recurrent) so long as it produces a fixed-sized embedding vector at its output. A common implementation is to embed sample pairs using a contrastive loss function [307]. In this scenario, pairs of samples (anchor and positive or anchor and negative) are passed into the twin networks. An embedded representation is produced for each sample, and a dissimilarity metric (typically $L_2$ or $L_1$ norm) is applied to compare the samples. Thus, the actual values of the features do not matter, and only the relative distance between them in the embedding space is compared. Following the

idea of contrastive loss, pairs with the same label should be embedded close to each other, while contrasting pairs should be mapped further apart in the embedding space. While Siamese networks are traditionally applied as twins, they can be extended to take any number of inputs. Another common implementation is a Siamese Triplet network [313]. As the name suggests, the base neural architecture is duplicated not once, but twice to form three networks with shared weights. Triplet loss is used to update the network parameters. As demonstrated by Figure A-14, the network takes a triplet set of samples as input and an embedded representation is produced for each sample (through the same transformation because the weights are shared). A metric function then compares the dissimilarities between the anchor and positive samples, as well as the anchor and negative. A comparator is applied at the output to compute the triplet loss between the embedded representations. The idea is that samples from the same class or with high semantic similarity should have a lower dissimilarity in the embedding space than samples from opposing classes.

The embeddings produced by Siamese networks have been shown to be powerful representations for discrimination tasks [314, 308, 307]. In test, pairs or sets of images are passed through the contrastive network to simultaneously produce embeddings. The pairing with the lowest dissimilarity would assign the class label to the test point. For example, if classifying a test point into one of ten classes, ten pairs could be passed through the network (the test sample and a representative from each class), and the label of the test point would be assigned as the label of the representative from the best-ranking pair. Alternatively, after all training samples are embedded through the triplet network, an alternative classification scheme such as $k$-NN or a SVM could be trained on the embedded representations. Testing would simply consist of embedding a test sample through the network and using the test procedure of the external learner.

Chen et al. [314] noted a few interesting observations regarding Siamese networks. First, data augmentation is often necessary when training Siamese networks to avoid

overfitting. Second, representations often benefit from normalization. Finally, contrastive learning benefits from larger batch sizes and longer training times than its supervised counterpart.

Despite the nuances mentioned, Siamese networks have been applied successfully to applications in object recognition [313], one-shot learning [307], visual representation learning [314] and image retrieval [313].

### A.2.4 FaceNet

FaceNet [308] is a convolutional neural network which learns a mapping from face images to a compact Euclidean space where distances directly correspond to a measure of face similarity. The method is based on learning a Euclidean embedding per image using a deep convolutional neural network. Once each sample has been embedded through the network, facial recognition can be achieved through the use of a simple $k$-NN classifier. FaceNet directly trains its outputs to be a compact 128-dimensional embedding using a triplet-based loss function based on LMNN. FaceNet is based on a Inception network architecture and optimizes a triplet loss formulated as:

$$\mathcal{L} = ||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha, \quad \forall (f(x_i^a), f(x_i^p), f(x_i^n)) \in \mathcal{T} \tag{A-104}$$

where $\alpha$ is the margin enforced between positive and negative pairs and $\mathcal{T}$ is the set of all possible triplets in the training set. The idea is that the triplet loss does not only promote similar faces to be close to each other in the embedding space, but also enforces a margin between every other face in the dataset. To address the hard-mining problem, triplets can be mined online out of mini-batches. This provides a trade-off between speed and utility toward training. FaceNet is currently SOA for the facial recognition problem, and in response, similar networks have been explored in a variety of applications, including: metric learning, image classification and image retrieval [313].

Table A-1. Overview of manifold learning methods and their properties.

| Method | Strategy | Intro. |
|---|---|---|
| | Linear | |
| PCA | Preserves variance of data, global structure. Section A.1.2.1 | 1901 |
| MDS | Preserves pairwise distances, global structure. Section A.1.2.2 | 1952 |
| LDA | Minimizes within-class distance and maximizes between-class distance, global structure. Section A.1.2.4 | 1936 |
| NMF | Decomposes data into low-dimensional coordinates and basis vectors, minimizes approximation error. Section A.1.2.3 | 1994 |
| NPP | Linear variant of LLE, preserves local neighborhoods, local structure. Section A.1.3.7 | 2005 |
| LPP | Linear variant of LE, preserves local neighborhoods, local structure. Section A.1.3.8 | 2003 |
| | Nonlinear | |
| KPCA | Performs PCA in a feature space defined by a kernel function, preserves variance of data, global structure. Section A.1.3.2 | 1999 |
| MVU | Learns a kernel function to be used with KPCA that "unfolds" the manifold, preserves variance of data, global structure. Section A.1.3.6 | 2004 |
| Isomap | Preserves geodesic distances, global structure. Section A.1.3.4 | 2000 |
| Sammon Mapping | Preserves pairwise distances. More emphasis is placed on samples that are close in the input space. Section A.1.3.5 | 1969 |
| LLE | Preserves local neighborhoods, local structure. Section A.1.3.7 | 2000 |
| Laplacian Eigenmaps | Preserves local neighborhoods, local structure. Section A.1.3.8 | 2003 |
| Hessian LLE | Minimizes curvature of high-dimensional manifold estimated from local neighborhoods. Section A.1.3.9 | 2003 |
| LTSA | Learns manifold in local tangent spaces then aligns to form global coordinate system. Section A.1.3.10 | 2002 |
| Diffusion Maps | Preserves diffusion distance. Section A.1.3.11 | 2006 |
| | Neural networks | |
| Autoencoders | Neural network where desired output is the input. Network decreases in dimensionality then increases to input size. Effectively performs nonlinear PCA. Section A.1.4.1 | 1986 |

Table A-1. Continued.

| Method | Strategy | Intro. |
|---|---|---|
| | Neural networks (continued) | |
| SOM | Distributes fixed lattice of neurons to cover input feature space using competitive learning. Section A.1.4.2 | 1990 |
| GNG | Distributes lattice of neurons to cover input feature space using competitive learning. The network is able to add and delete neurons to fill the data space. Section A.1.4.3 | 1994 |
| Elastic Maps and Nets | Uses map of neurons to cover the input feature space. Simulates network of springs and is optimized to the mechanical equilibrium of the system. Section A.1.4.4 | 2008 |
| | Probabilistic | |
| PPCA | Learns parameters of linear mapping from latent space to input space by maximizing data likelihood of a factor analysis model. Section A.1.5.1 | 1999 |
| GTM | Learns a mapping from the latent space to the data space by assuming the data distribution as a constrained mixture of Gaussians. Probabilistic alternative to the SOM. Section A.1.5.2 | 1998 |
| Manifold Charting | Computes mixture of locally linear models on the data then aligns local low-dimensional models to find global coordinates. Section A.1.5.3 | 2003 |
| | High-dimensional data visualization | |
| t-SNE | Defines pairwise probabilities of samples being neighbors. Minimizes the KL-divergence between high and low-dimensional data densities. Retains global and local data structure. Section A.1.6.1 | 2008 |
| LargeVis | Efficiently constructs neighborhood graph weighted by conditional probabilities of being neighbors. Maximizes the graph likelihood in the embedding space. Preserves local distances. Section A.1.6.2 | 2016 |
| UMAP | Uses fuzzy simplicial sets to define locally adaptive neighborhood graph. Minimizes the cross-entropy between the high and low-dimensional similarity distributions. Preserves global and local structure. Section A.1.6.3 | 2018 |

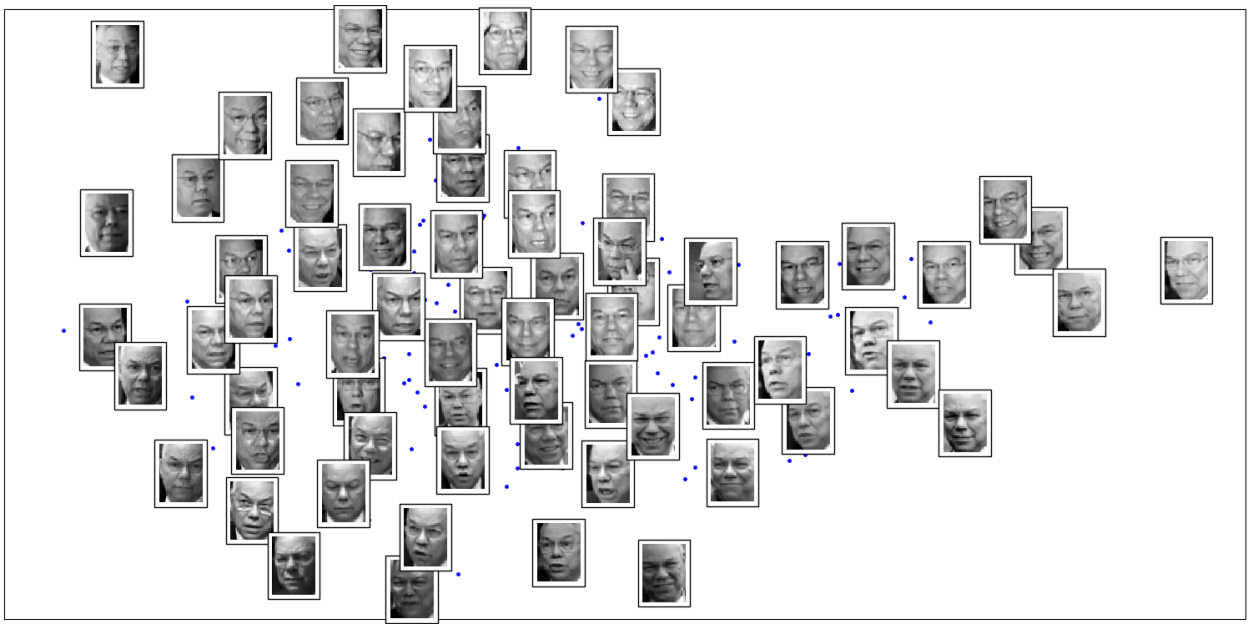
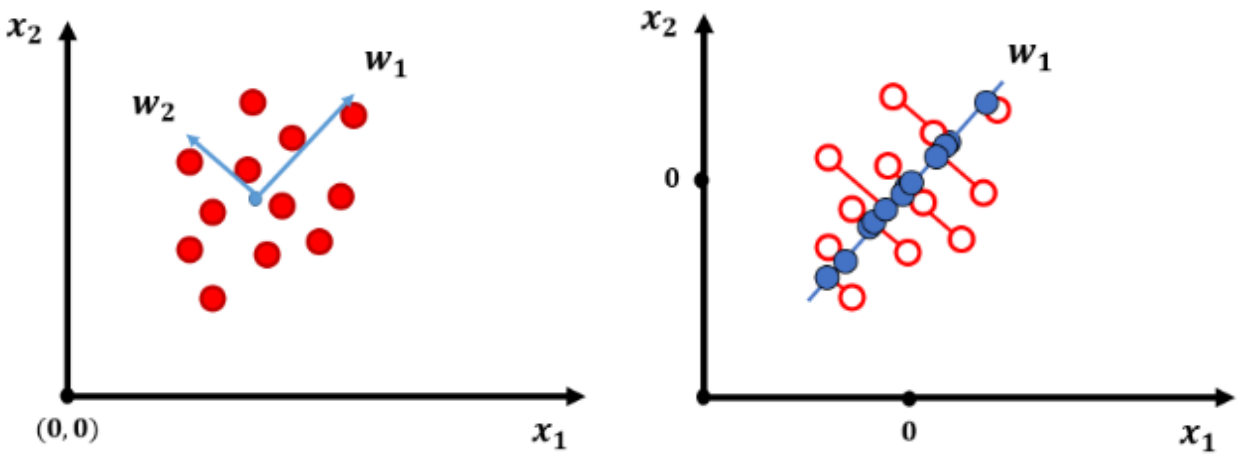


Figure A-1. Manifold learning example on a single class of the LFW Faces in the Wild dataset. The image shows the embedding of 1850-dimensional features into a 2-dimensional space. The samples are distributed according to the two greatest factors of variation in the dataset, seemingly, illumination and the direction the subject is facing.




(A) (B)

Figure A-2. Example of dimensionality reduction on a synthetic dataset using principal component analysis. A) Two-dimensional data, where $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ correspond to the first and second principal axes of the dataset, respectively. B) Orthogonal projection of the data onto the first principal axis. It can be observed that $\boldsymbol{w}_1$ corresponds to the direction of maximum variance for the data.

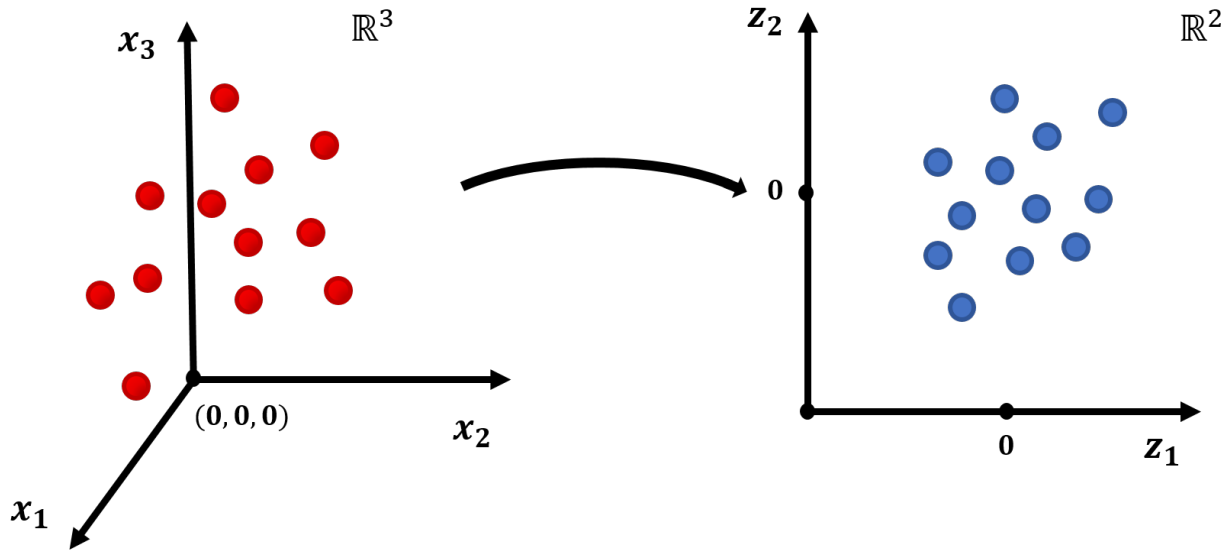
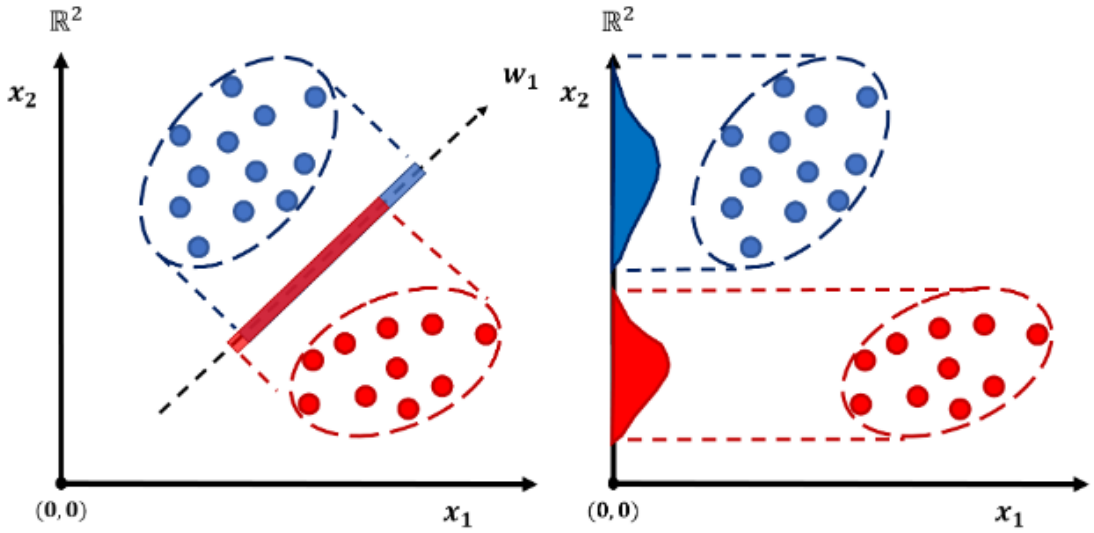



Figure A-3. Example of multidimensional scaling on a synthetic dataset. In this example, MDS transforms the data into a two-dimensional coordinate space which approximately preserves the pairwise distances between the three-dimensional samples.




(A) (B)

Figure A-4. Example of dimensionality reduction on a synthetic dataset using PCA and LDA. The dataset shown is comprised of two classes, red and blue, respectively. A) PCA projects the data onto the principal axis corresponding to the direction of maximum variance for the dataset. The classes would not be separable in the one-dimensional projection space found by PCA. B) LDA projects the data onto the hyperplane which both maximizes between-class dissimilarity while minimizing within-class dissimilarity. For this dataset, the data in the embedding space found by LDA would be linearly separable.

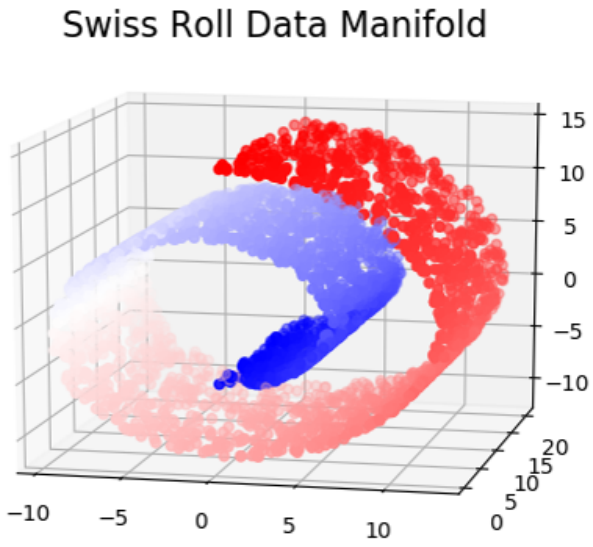



Figure A-5. Example of a nonlinear manifold. This particular dataset is known as the "Swiss Roll", and it represents a two-dimensional manifold embedded in three dimensions. Global, linear manifold learning approaches cannot appropriately unroll the manifold.

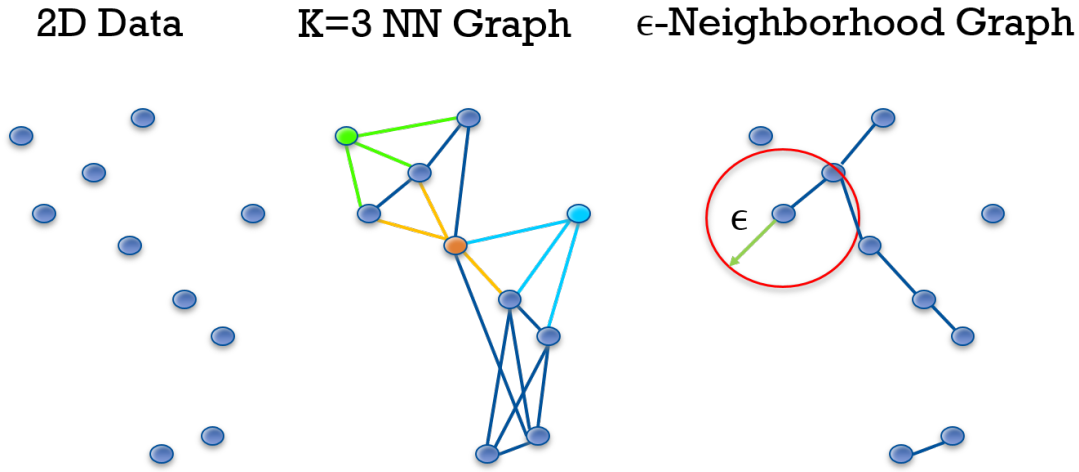



Figure A-6. Examples of $k$-nearest neighbor and $\epsilon$ graphs on two-dimensional, synthetic data. In the $k = 3$, $k$-nearest neighbor graph, the instances colored green, orange, and blue, are each connected only to their three closest neighbors (as denoted by corresponding colored edges) in the data space. For the $\epsilon$-ball or $\epsilon$-neighborhood graph, edges only exist between samples if they fall within an $\epsilon$ radius (denoted by the red circle) from the query instance.

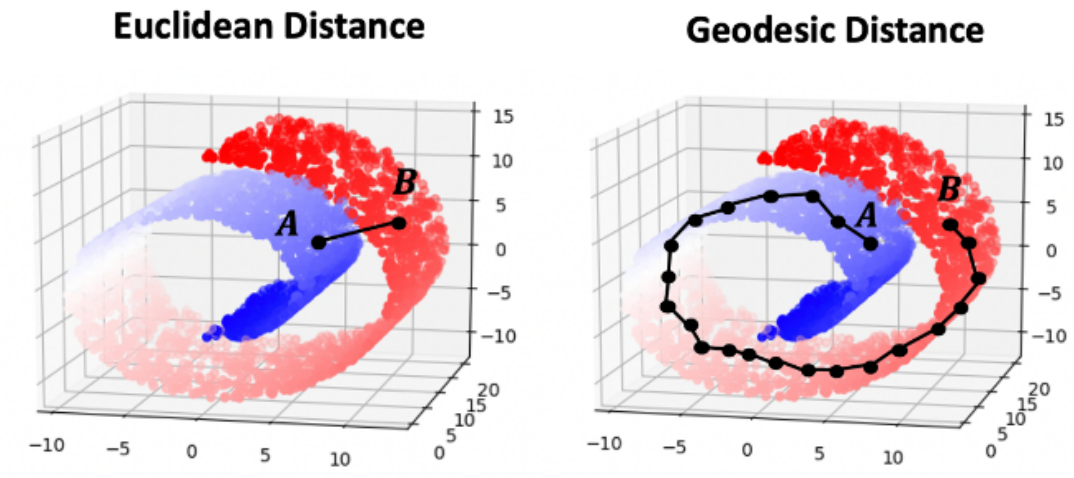



Figure A-7. Demonstration of geodesic distance. A) shows the Euclidean distance between two points, A and B, from the blue and red classes, respectively. A classifier using Euclidean distance would assume the two points should be assigned the same class label because they are (Euclidean-wise) close in the feature space. However, the two points actually belong to two disparate classes, each at an opposite end of the manifold. Instead, B) shows an approximation of the geodesic distance between the samples. The geodesic distance, which measures the distance through the manifold, would be a more accurate representation of dissimilarity for the two instances.

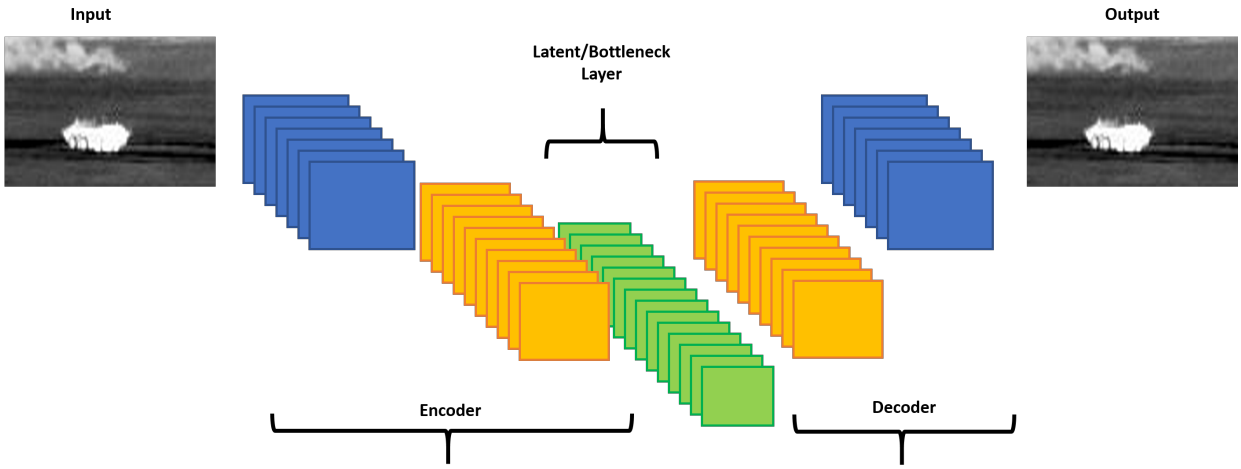



Figure A-8. Depiction of an autoencoder neural network. The architecture consists of an encoder, a latent or bottleneck layer, and a decoder, which is the inverse transformation of the encoder. The output of an autoencoder should be, as close as possible, a reproduction of the input.

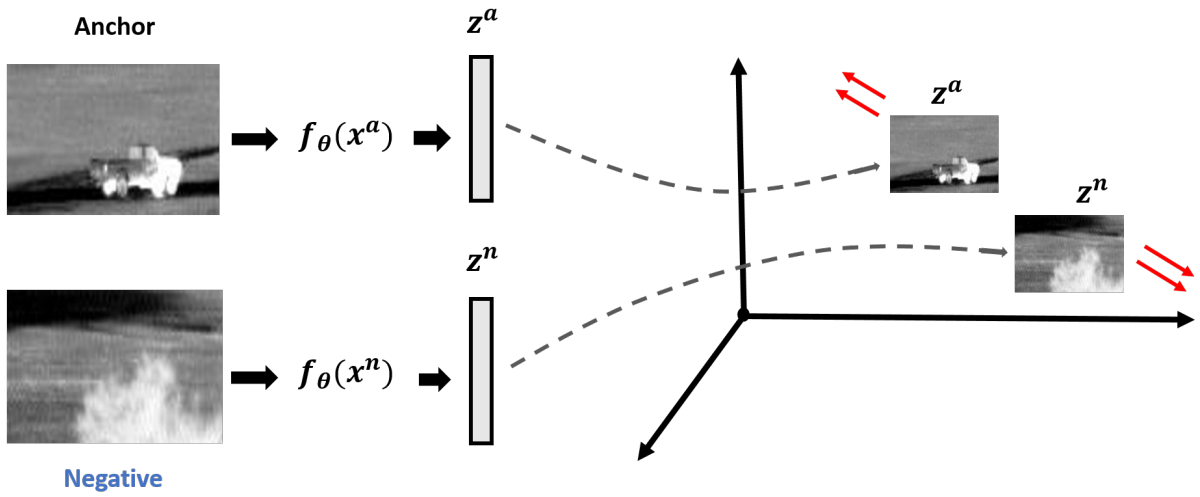



Figure A-9. Contrastive loss with an anchor/negative pair. After embedding, the samples should be metrically farther away than in the input feature space.

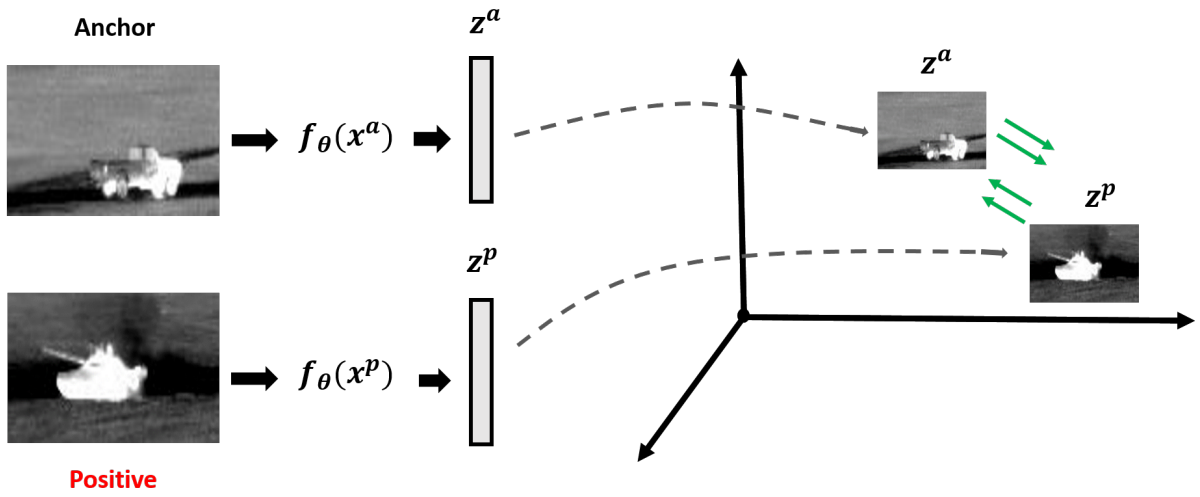



Figure A-10. Contrastive loss with an anchor/positive pair. After embedding, the samples should be metrically closer than they were in the input feature space.

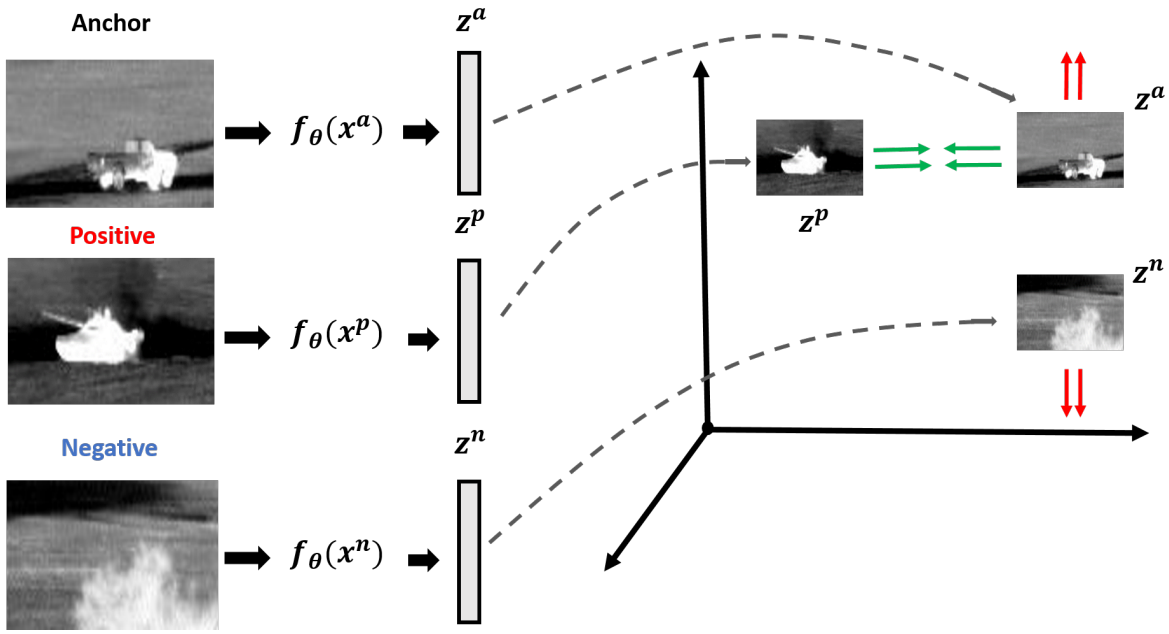



Figure A-11. Embedding according to triplet loss. After embedding, the positive example should be closer to the anchor point than the negative example by a margin.

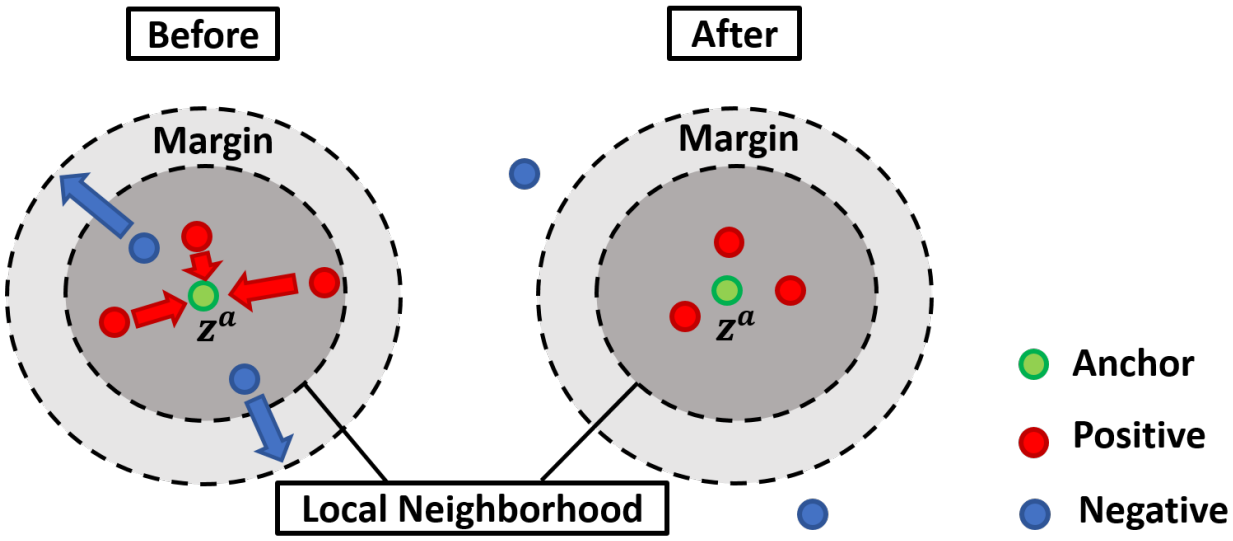



Figure A-12. Large Margin $k$-NN. The goal of LMNN is to discover a metric which represents neighbors of the same class as being closer to the anchor points than neighbors with differing class labels, plus a margin.

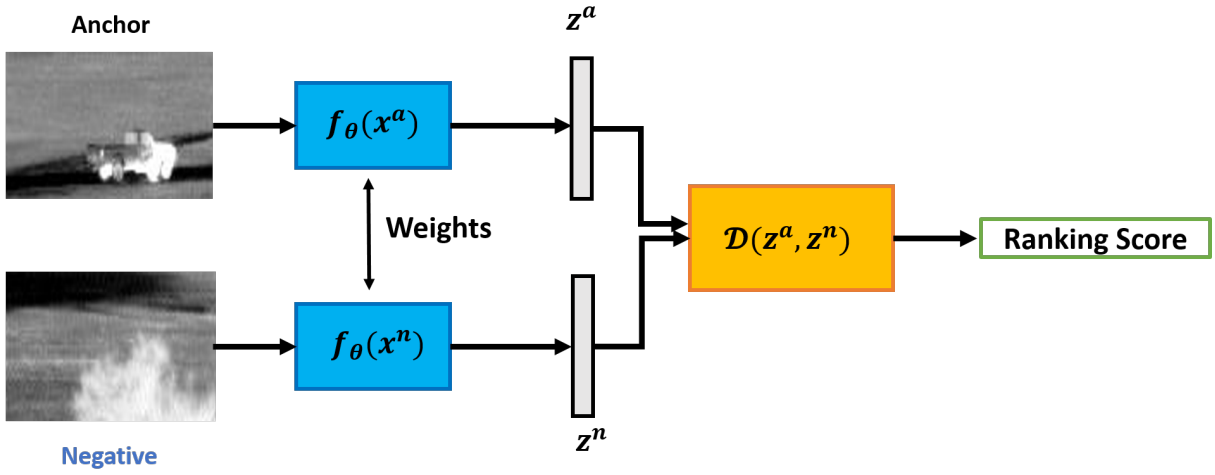



Figure A-13. Siamese neural network structured from contrastive loss. Two identical neural networks simultaneously embed an anchor and positive/negative sample. After the shared network weights are learned, anchor/positive pairs should have a lower dissimilarity score than anchor/negative sample pairs.

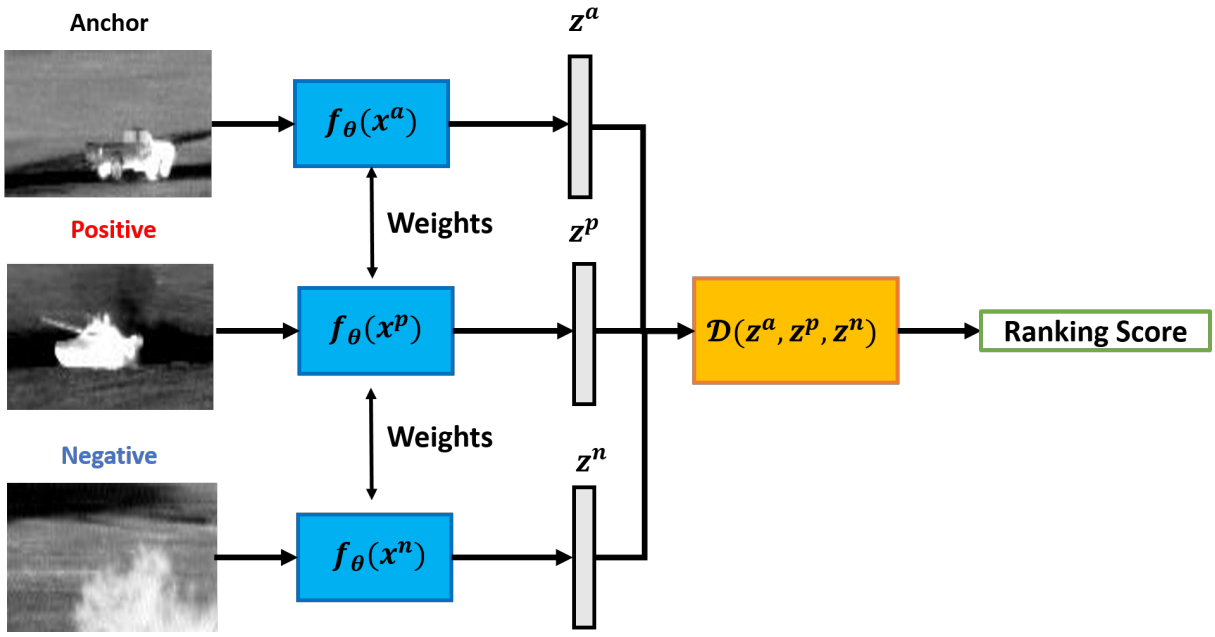



Figure A-14. Siamese neural network structured from triplet loss. Three identical neural networks simultaneously embed triplets of anchor, positive and negative samples. After the shared network weights are learned, anchor/positive pairs should have a lower dissimilarity score than anchor/negative pairs.

# REFERENCES


[1] L. van der Maaten, E. Postma, and H. Herik, "Dimensionality reduction: A comparative review," *Journal of Machine Learning Research (JMLR)*, vol. 10, 01 2007.

[2] Y. Bengio, A. C. Courville, and P. Vincent, "Unsupervised feature learning and deep learning: A review and new perspectives," *CoRR*, vol. abs/1206.5538, 2012. [Online]. Available: http://arxiv.org/abs/1206.5538

[3] X. Geng, D. Zhan, and Z. Zhou, "Supervised nonlinear dimensionality reduction for visualization and classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 35, no. 6, pp. 1098–1107, Dec 2005.

[4] N. Thorstensen, "Manifold learning and applications to shape and image processing," Ph.D. dissertation, Ecole Nationale des Ponts et Chaussees, Paris, France, Nov. 2009.

[5] X. Du, "Multiple instance choquet integral for multiresolution sensor fusion," Ph.D. dissertation, Univ. of Missouri, Columbia, MO, Dec. 2017.

[6] X. Geng, L. Ji, and Y. Zhao, "The basic equation for target detection in remote sensing," 2017.

[7] B. Chaudhuri and S. Parui, "Target detection: Remote sensing techniques for defence applications," *Defence Science Journal*, vol. 45, pp. 285–291, 04 1995.

[8] A. Zare, C. Jiao, and T. Glenn, "Discriminative multiple instance hyperspectral target characterization," *IEEE Trans. Pattern Anal. Mach. Inteli.*, vol. 40, no. 10, pp. 2342–2354, Oct. 2018.

[9] C. Xu, D. Tao, and Y. Rui, "Large-margin weakly supervised dimensionality reduction," *31st International Conference on Machine Learning, ICML 2014*, vol. 3, pp. 2472–2482, 01 2014.

[10] J. Bocinsky, "Learning multiple target concepts from uncertain, ambiguous data using the adaptive cosine estimator and spectral match filter," Master's thesis, Univ. of Florida, Gainesville, FL, May 2019.

[11] A. Zare, "Hyperspectral endmember detection and band selection using bayesian methods," Ph.D. dissertation, Univ. of Florida, Gainesville, FL, 2008.

[12] V. Cheplygina, M. Bruijne, and J. P. W. Pluim, "Not-so-supervised: A survey of semi-supervised, multi-instance, and transfer learning in medical image analysis," *Medical Image Analysis*, vol. 54, pp. 280 – 296, 2019.

[13] J. F. Ruiz-Muñoz, M. Orozco-Alzate, and G. Castellanos-Domínguez, "Multiple instance learning-based birdsong classification using unsupervised recording segmentation," in *IJCAI*, 2015.

[14] D. S. I. A. C. DSIAC, "DSIAC MS-003-DB Algorithm Development Database," https://www.dsiac.org/resources/research-materials/cds-dvds-databases-digital-files/atr-algorithm-development-image, 2014.

[15] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *CoRR*, vol. abs/1804.02767, 2018. [Online]. Available: http://arxiv.org/abs/1804.02767

[16] M. Cook, "Task driven extended functions of multiple instances (td-efumi)," Master's thesis, Univ. of Missouri, Columbia, MO, 2015.

[17] S. Ghaffarzadegan, "Deep multiple instance feature learning via variational autoencoder," in *AAAI Workshops*, 2018.

[18] M. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, "Multiple instance learning: A survey of problem characteristics and applications," *CoRR*, vol. abs/1612.03365, 2016. [Online]. Available: http://arxiv.org/abs/1612.03365

[19] C. Bishop, M. Svensén, and C. K. I. Williams, "GTM: The generative topographic mapping," pp. 215–234, January 1998. [Online]. Available: https://www.microsoft.com/en-us/research/publication/gtm-the-generative-topographic-mapping/

[20] M. Nickel and D. Kiela, "Poincaré embeddings for learning hierarchical representations," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 6338–6347.

[21] R. Talmon, S. Mallat, H. Zaveri, and R. R. Coifman, "Manifold learning for latent variable inference in dynamical systems," *IEEE Transactions on Signal Processing*, vol. 63, no. 15, pp. 3843–3856, Aug 2015.

[22] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000. [Online]. Available: https://science.sciencemag.org/content/290/5500/2319

[23] E. J. Palomo and E. Lopez-Rubio, "The growing hierarchical neural gas self-organizing neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 9, pp. 2000–2009, Sep. 2017.

[24] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[25] B. Kegl, D. Wunsch, and A. Zinovyev, "Principal manifolds for data visualisation and dimension reduction," 01 2008.

[26] Y.-Y. Sun, M. K. Ng, and Z.-H. Shou, "Multi-instance dimensionality reduction," in *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, ser. AAAI'10. AAAI Press, 2010, pp. 587–592. [Online]. Available: http://dl.acm.org/citation.cfm?id=2898607.2898702

[27] H. Zhu, L.-Z. liao, and M. K. Ng, "Multi-instance dimensionality reduction via sparsity and orthogonality," *Neural Comput.*, vol. 30, no. 12, pp. 3281–3308, dec 2018. [Online]. Available: https://doi.org/10.1162/neco_a_01140

[28] W. Ping, Y. Xu, R. Kexin, C.-H. Chi, and S. Furao, “Non-i.i.d. multi-instance dimensionality reduction by learning a maximum bag margin subspace.” *Proceedings of the National Conference on Artificial Intelligence*, vol. 1, 01 2010.

[29] Saehoon Kim and Seungjin Choi, “Local dimensionality reduction for multiple instance learning,” in *2010 IEEE International Workshop on Machine Learning for Signal Processing*, Aug 2010, pp. 13–18.

[30] J. Chai, X. Ding, H. Chen, and T. Li, “Multiple-instance discriminant analysis,” *Pattern Recognition*, vol. 47, no. 7, pp. 2517 – 2531, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320314000387

[31] A. C. Latham, “Multiple-instance feature ranking,” Master’s thesis, Case Western Reserve University, Cleveland, OH, August 2015.

[32] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, “Solving the multiple instance problem with axis-parallel rectangles,” *Artificial Intelligence*, vol. 89, no. 1, pp. 31 – 71, 1997.

[33] B. Babenko, N. Verma, P. Dollár, and S. Belongie, “Multiple instance learning with manifold bags.” *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, pp. 81–88, 01 2011.

[34] Yixin Chen, Jinbo Bi, and J. Z. Wang, “Miles: Multiple-instance learning via embedded instance selection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 1931–1947, Dec 2006.

[35] J. Amores, “Multiple instance classification: Review, taxonomy and comparative study,” *Artificial Intelligence*, vol. 201, pp. 81 – 105, 2013.

[36] J. Bocinsky, C. H. McCurley, D. Shats, and A. Zare, “Investigation of initialization strategies for the multiple instance adaptive cosine estimator,” in *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXIV, 110120N*, ser. Proc.SPIE, vol. 11012, May 2019.

[37] C. Jiao, “Target concept learning from ambiguously labeled data,” Ph.D. dissertation, Univ. of Missouri, Columbia, MO, Dec. 2017.

[38] O. Maron and T. Lozano-Pérez, “A framework for multiple-instance learning,” in *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*, ser. NIPS ’97. Cambridge, MA, USA: MIT Press, 1998, pp. 570–576.

[39] Q. Zhang and S. A. Goldman, “EM-DD: An improved multiple-instance learning technique,” in *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani, Eds. MIT Press, 2002, pp. 1073–1080.

[40] O. Maron and A. L. Ratan, “Multiple-instance learning for natural scene classification,” in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. ICML ’98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 341–349.

[41] A. Zare, M. Cook, B. Alvey, and D. K. Ho, “Multiple instance dictionary learning for subsurface object detection using handheld EMI,” in *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XX, 94540G*, ser. Proc. SPIE, May 2015. [Online]. Available: https://doi.org/10.1117/12.2179177

[42] C. H. McCurley, J. Bocinsky, and A. Zare, “Comparison of hand-held WEMI target detection algorithms,” in *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXIV, 110120U*, ser. Proc.SPIE, vol. 11012, May 2019.

[43] T. Gärtner, P. A. Flach, A. Kowalczyk, and A. J. Smola, “Multi-instance kernels,” in *ICML*, 2002.

[44] Z.-H. Zhou and M.-L. Zhang, “Ensembles of multi-instance learners,” in *Machine Learning: ECML 2003*, N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 492–502.

[45] Z.-H. Zhou, Y.-Y. Sun, and Y.-F. Li, “Multi-instance learning by treating instances as non-i.i.d. samples,” in *Proceedings of the 26th Annual International Conference on Machine Learning*, ser. ICML ’09. New York, NY, USA: Association for Computing Machinery, 2009, p. 1249–1256. [Online]. Available: https://doi.org/10.1145/1553374.1553534

[46] S. Andrews, I. Tsochantaridis, and T. Hofmann, “Support vector machines for multiple-instance learning,” *Advances in Neural Information Processing Systems*, vol. 15, pp. 561–568, 01 2002.

[47] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[48] L. Cao, F. Luo, L. Chen, S. Yihan, H. Wang, C. Wang, and R. Ji, “Weakly supervised vehicle detection in satellite images via multi-instance discriminative learning,” *Pattern Recognition*, vol. 64, 12 2016.

[49] Z.-H. Zhou and J.-M. Xu, “On the relation between multi-instance learning and semi-supervised learning,” in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML ’07. New York, NY, USA: Association for Computing Machinery, 2007, p. 1167–1174. [Online]. Available: https://doi.org/10.1145/1273496.1273643

[50] Y. Xiao, B. Liu, and Z. Hao, “A sphere-description-based approach for multiple-instance learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 2, pp. 242–257, Feb 2017.

[51] M. Gao, A. Li, R. Yu, V. I. Morariu, and L. S. Davis, "C-WSL: count-guided weakly supervised localization," *CoRR*, vol. abs/1711.05282, 2017. [Online]. Available: http://arxiv.org/abs/1711.05282

[52] M. Ilse, J. M. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," *CoRR*, vol. abs/1802.04712, 2018. [Online]. Available: http://arxiv.org/abs/1802.04712

[53] S. Li, Y. Liu, X. Sui, C. Chen, G. Tjio, D. S. W. Ting, and R. S. M. Goh, "Multi-instance multi-scale cnn for medical image classification," *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, p. 531–539, 2019. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-32251-9_58

[54] S. Wang, W. Chen, S. Xie, G. Azzari, and D. Lobell, "Weakly supervised deep learning for segmentation of remote sensing imagery," *Remote Sens.*, vol. 12, p. 207, 2020.

[55] M. Tu, J. Huang, X. He, and B. Zhou, "Multiple instance learning with graph neural networks," *CoRR*, vol. abs/1906.04881, 2019. [Online]. Available: http://arxiv.org/abs/1906.04881

[56] T. Deselaers and V. Ferrari, "A conditional random field for multiple-instance learning," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. Madison, WI, USA: Omnipress, 2010, p. 287–294.

[57] H. Hajimirsadeghi and G. Mori, "Multi-instance classification by max-margin training of cardinality-based markov networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 9, pp. 1839–1852, Sep. 2017.

[58] S. E. Yuksel, J. Bolton, and P. Gader, "Multiple-instance hidden markov models with applications to landmine detection," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 53, no. 12, pp. 6766–6775, Dec 2015.

[59] S. E. Yuksel, J. Bolton, and P. D. Gader, "Landmine detection with multiple instance hidden markov models," in *2012 IEEE International Workshop on Machine Learning for Signal Processing*, Sep. 2012, pp. 1–6.

[60] M. Cook, A. Zare, and D. K. C. Ho, "Buried object detection using handheld WEMI with task-driven extended functions of multiple instances," in *Proc. SPIE 9823, Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXI, 98230A*, ser. Proc. SPIE, vol. 9823, Apr. 2016, pp. 9823 – 9823 – 9.

[61] C. Jiao, C. Chen, R. G. McGarvey, S. Bohlman, L. Jiao, and A. Zare, "Multiple instance hybrid estimator for hyperspectral target characterization and sub-pixel target detection," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 146, pp. 235 – 250, 2018.

[62] C. Leistner, A. Saffari, and H. Bischof, “MIForests: Multiple-instance learning with randomized trees,” in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 29–42.

[63] C. Zhang, J. C. Platt, and P. A. Viola, “Multiple instance boosting for object detection,” in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. C. Platt, Eds. MIT Press, 2006, pp. 1417–1424.

[64] B. Babenko, Z. Tu, and S. J. Belongie, “Simultaneous learning and alignment: Multi-instance and multi-pose learning,” 2008.

[65] C. Bergeron, J. Zaretzki, C. Breneman, and K. P. Bennett, “Multiple instance ranking,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 48–55. [Online]. Available: https://doi.org/10.1145/1390156.1390163

[66] C. Bergeron, G. Moore, J. Zaretzki, C. M. Breneman, and K. P. Bennett, “Fast bundle algorithm for multiple-instance learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 6, pp. 1068–1079, June 2012.

[67] Y. Hu, M. Li, and N. Yu, “Multiple-instance ranking: Learning to rank images for image retrieval,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1–8.

[68] J. Fürnkranz and E. Hüllermeier, “Pairwise preference learning and ranking,” in *Machine Learning: ECML 2003*, N. Lavrač, D. Gamberger, H. Blockeel, and L. Todorovski, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, pp. 145–156.

[69] W. Chu and Z. Ghahramani, “Preference learning with Gaussian Processes,” in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML ’05. New York, NY, USA: Association for Computing Machinery, 2005, p. 137–144.

[70] O. Dekel, Y. Singer, and C. D. Manning, “Log-linear models for label ranking,” in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. K. Saul, and B. Schölkopf, Eds. MIT Press, 2004, pp. 497–504.

[71] F. Aiolli and A. Sperduti, “Learning preferences for multiclass problems,” in *Advances in Neural Information Processing Systems*, L. Saul, Y. Weiss, and L. Bottou, Eds., vol. 17. MIT Press, 2004. [Online]. Available: https://proceedings.neurips.cc/paper/2004/file/5b168fdba5ee5ea262cc2d4c0b457697-Paper.pdf

[72] A. Asif, W. A. Abbasi, F. Munir, A. Ben-Hur, and F. ul Amir Afsar Minhas, “pyLEMMINGS: Large margin multiple instance classification and ranking for bioinformatics applications,” *CoRR*, vol. abs/1711.04913, 2017. [Online]. Available: http://arxiv.org/abs/1711.04913

[73] H. Wu, “Weakly supervised learning on image manifolds,” Ph.D. dissertation, University of North Carolina at Charlotte, Charlotte, NC, 2015.

[74] Z. Zhang, H. Zha, and M. Zhang, “Spectral methods for semi-supervised manifold learning,” in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, June 2008, pp. 1–6.

[75] M. Chen, J. Wang, X. Li, and X. Sun, “Robust semi-supervised manifold learning algorithm for classification,” *Mathematical Problems in Engineering*, vol. 2018, pp. 1–8, 02 2018.

[76] Y. Zhang, X. Zheng, G. Liu, X. Sun, H. Wang, and K. Fu, “Semi-supervised manifold learning based multigraph fusion for high-resolution remote sensing image classification,” *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 2, pp. 464–468, Feb 2014.

[77] D. Hong, N. Yokoya, N. Ge, J. Chanussot, and X. X. Zhu, “Learnable manifold alignment (LeMA): A semi-supervised cross-modality learning framework for land cover and land use classification,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 147, pp. 193 – 205, 2019.

[78] R. Navaratnam, A. W. Fitzgibbon, and R. Cipolla, “The joint manifold model for semi-supervised multi-valued regression,” in *2007 IEEE 11th International Conference on Computer Vision*, Oct 2007, pp. 1–8.

[79] J. S. Stanley III, G. Wolf, and S. Krishnaswamy, “Manifold alignment with feature correspondence,” *CoRR*, vol. abs/1810.00386, 2018. [Online]. Available: http://arxiv.org/abs/1810.00386

[80] D. Tuia and G. Camps-Valls, “Kernel manifold alignment for domain adaptation,” *PLOS ONE*, vol. 11, no. 2, p. e0148655, 2 2016.

[81] C. Wang and S. Mahadevan, “Multiscale manifold alignment,” 09 2010.

[82] ——, “Heterogeneous domain adaptation using manifold alignment,” in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence*, ser. IJCAI’11. AAAI Press, 2011, vol. 2, pp. 1541–1546. [Online]. Available: http://dx.doi.org/10.5591/978-1-57735-516-8/IJCAI11-259

[83] G. Chao, Y. Luo, and W. Ding, “Recent advances in supervised dimension reduction: A survey,” *Machine Learning and Knowledge Extraction*, vol. 1, no. 1, pp. 341–358, 2019.

[84] E. Vural and C. Guillemot, “A study of the classification of low-dimensional data with supervised manifold learning,” *CoRR*, vol. abs/1507.05880, 2018. [Online]. Available: http://arxiv.org/abs/1507.05880

[85] U. Gaur and B. S. Manjunath, “Weakly supervised manifold learning for dense semantic object correspondence,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 1744–1752.

[86] S. Bell, P. Upchurch, N. Snavely, and K. Bala, “Material recognition in the wild with the materials in context database,” *CoRR*, vol. abs/1412.0623, 2014. [Online]. Available: http://arxiv.org/abs/1412.0623

[87] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei, “What’s the point: Semantic segmentation with point supervision,” in *Computer Vision (ECCV 2016)*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 549–565.

[88] Z. Li, Q. Chen, and V. Koltun, “Interactive image segmentation with latent diversity,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 577–585.

[89] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common objects in context,” in *Computer Vision (ECCV 2014)*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[90] Y. Boykov and M.-P. Jolly, “Interactive graph cuts for optimal boundary amp; region segmentation of objects in N-D images,” in *Proceedings IEEE International Conference on Computer Vision (ICCV)*, vol. 1, 2001, pp. 105–112 vol.1.

[91] C. Rother, V. Kolmogorov, and A. Blake, ““GrabCut”: Interactive foreground extraction using iterated graph cuts,” *ACM Trans. Graph.*, vol. 23, no. 3, p. 309–314, 8 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015720

[92] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum, “Lazy snapping,” *ACM Trans. Graph.*, vol. 23, no. 3, p. 303–308, aug 2004. [Online]. Available: https://doi.org/10.1145/1015706.1015719

[93] D. Batra, A. Kowdle, D. Parikh, J. Luo, and T. Chen, “iCoseg: Interactive co-segmentation with intelligent scribble guidance,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3169–3176.

[94] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, “ScribbleSup: Scribble-supervised convolutional networks for semantic segmentation,” *CoRR*, vol. abs/1604.05144, 2016. [Online]. Available: http://arxiv.org/abs/1604.05144

[95] J. Zhang, L. Zhang, Y. Teng, X. Zhang, S. Wang, and L. Ju, “Interactive binary image segmentation with edge preservation,” *CoRR*, vol. abs/1809.03334, 2018. [Online]. Available: http://arxiv.org/abs/1809.03334

[96] J. Dai, K. He, and J. Sun, “BoxSup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation,” *CoRR*, vol. abs/1503.01640, 2015. [Online]. Available: http://arxiv.org/abs/1503.01640

[97] G. Papandreou, L. Chen, K. Murphy, and A. L. Yuille, “Weakly- and semi-supervised learning of a DCNN for semantic image segmentation,” *CoRR*, vol. abs/1502.02734, 2015. [Online]. Available: http://arxiv.org/abs/1502.02734

[98] S. Jetley, N. A. Lord, N. Lee, and P. H. S. Torr, “Learn to pay attention,” *CoRR*, vol. abs/1804.02391, 2018. [Online]. Available: http://arxiv.org/abs/1804.02391

[99] K. Simonyan, A. Vedaldi, and A. Zisserman, “Deep inside convolutional networks: Visualising image classification models and saliency maps,” in *International Conference on Learning Representations (ICLR)*, 2014.

[100] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. A. Riedmiller, “Striving for simplicity: The all convolutional net,” *CoRR*, vol. abs/1412.6806, 2015.

[101] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://arxiv.org/abs/1311.2901

[102] B. Zhou, A. Khosla, À. Lapedriza, A. Oliva, and A. Torralba, “Learning deep features for discriminative localization,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929, 2016.

[103] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-CAM: Visual explanations from deep networks via gradient-based localization,” in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626.

[104] A. Chattopadhay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, “Grad-CAM++: Generalized gradient-based visual explanations for deep convolutional networks,” in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2018, pp. 839–847.

[105] P.-T. Jiang, C.-B. Zhang, Q. Hou, M.-M. Cheng, and Y. Wei, “LayerCAM: Exploring hierarchical class activation maps for localization,” *IEEE Transactions on Image Processing*, vol. 30, pp. 5875–5888, 2021.

[106] H. Wang, M. Du, F. Yang, and Z. Zhang, “Score-CAM: Improved visual explanations via score-weighted class activation mapping,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2921–2929, 2020.

[107] S. Desai and H. G. Ramaswamy, “Ablation-CAM: Visual explanations for deep convolutional network via gradient-free localization,” in *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 972–980.

[108] M. B. Muhammad and M. Yeasin, “Eigen-CAM: Class activation map using principal components,” *CoRR*, vol. abs/2008.00299, 2020. [Online]. Available: https://arxiv.org/abs/2008.00299

[109] M. A. A. K. Jalwana, N. Akhtar, M. Bennamoun, and A. Mian, “CAMERAS: Enhanced resolution and sanity preserving class activation mapping for image saliency,” *CoRR*, vol. abs/2106.10649, 2021. [Online]. Available: https://arxiv.org/abs/2106.10649

[110] H. Jung and Y. Oh, “LIFT-CAM: Towards better explanations for class activation mapping,” *CoRR*, vol. abs/2102.05228, 2021. [Online]. Available: https://arxiv.org/abs/2102.05228

[111] J. Kim, J. Choe, S. Yun, and N. Kwak, “Normalization matters in weakly supervised object localization,” *CoRR*, vol. abs/2107.13221, 2021. [Online]. Available: https://arxiv.org/abs/2107.13221

[112] K. H. Lee, C. Park, J. Oh, and N. Kwak, “LFI-CAM: Learning feature importance for better visual explanation,” *CoRR*, vol. abs/2105.00937, 2021. [Online]. Available: https://arxiv.org/abs/2105.00937

[113] A. Zhang, X. Wang, C. Fang, J. Shi, T. Chua, and Z. Chen, “A-FMI: Learning attributions from deep networks via feature map importance,” *CoRR*, vol. abs/2104.05527, 2021. [Online]. Available: https://arxiv.org/abs/2104.05527

[114] B. Zhou, Y. Sun, D. Bau, and A. Torralba, “Interpretable basis decomposition for visual explanation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[115] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” Red Hook, NY, USA, p. 4768–4777, 2017.

[116] A. Shrikumar, P. Greenside, and A. Kundaje, “Learning important features through propagating activation differences,” *CoRR*, vol. abs/1704.02685, 2017. [Online]. Available: http://arxiv.org/abs/1704.02685

[117] G. Yu, A. Zare, W. Xu, R. Matamala, J. Reyes-Cabrera, F. B. Fritschi, and T. E. Juenger, “Weakly supervised minirhizotron image segmentation with MIL-CAM,” *CoRR*, vol. abs/2007.15243, 2020. [Online]. Available: https://arxiv.org/abs/2007.15243

[118] Y. Wei, H. Xiao, H. Shi, Z. Jie, J. Feng, and T. S. Huang, “Revisiting dilated convolution: A simple approach for weakly- and semi- supervised semantic segmentation,” *CoRR*, vol. abs/1805.04574, 2018. [Online]. Available: http://arxiv.org/abs/1805.04574

[119] I. H. Laradji, D. Vázquez, and M. Schmidt, “Where are the masks: Instance segmentation with image-level supervision,” *CoRR*, vol. abs/1907.01430, 2019. [Online]. Available: http://arxiv.org/abs/1907.01430

[120] J. Ahn and S. Kwak, “Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[121] A. Kolesnikov and C. H. Lampert, “Seed, expand and constrain: Three principles for weakly-supervised image segmentation,” *CoRR*, vol. abs/1603.06098, 2016. [Online]. Available: http://arxiv.org/abs/1603.06098

[122] S. Hong, J. Oh, B. Han, and H. Lee, “Learning transferrable knowledge for semantic segmentation with deep convolutional neural network,” *CoRR*, vol. abs/1512.07928, 2015. [Online]. Available: http://arxiv.org/abs/1512.07928

[123] X. Li, T. Zhou, J. Li, Y. Zhou, and Z. Zhang, “Group-wise semantic mining for weakly supervised semantic segmentation,” *CoRR*, vol. abs/2012.05007, 2020. [Online]. Available: https://arxiv.org/abs/2012.05007

[124] J. Feng, X. Wang, and W. Liu, “Deep graph cut network for weakly-supervised semantic segmentation,” *Science China Information Sciences*, vol. 64, 2021.

[125] G. Sun, W. Wang, J. Dai, and L. V. Gool, “Mining cross-image semantics for weakly supervised semantic segmentation,” *CoRR*, vol. abs/2007.01947, 2020. [Online]. Available: https://arxiv.org/abs/2007.01947

[126] P.-T. Jiang, Q. Hou, Y. Cao, M.-M. Cheng, Y. Wei, and H. Xiong, “Integral object mining via online attention accumulation,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 2070–2079.

[127] Z. Shi, Y. Yang, T. M. Hospedales, and T. Xiang, “Weakly supervised learning of objects, attributes and their associations,” in *Computer Vision*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 472–487.

[128] K. Li, Z. Wu, K. Peng, J. Ernst, and Y. Fu, “Tell me where to look: Guided attention inference network,” *CoRR*, vol. abs/1802.10171, 2018. [Online]. Available: http://arxiv.org/abs/1802.10171

[129] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang, “Weakly-supervised semantic segmentation network with deep seeded region growing,” in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7014–7023.

[130] A. Roy and S. Todorovic, “Combining bottom-up, top-down, and smoothness cues for weakly supervised image segmentation,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7282–7291.

[131] L. Jing, Y. Chen, and Y. Tian, “Coarse-to-fine semantic segmentation from image-level labels,” *CoRR*, vol. abs/1812.10885, 2018. [Online]. Available: http://arxiv.org/abs/1812.10885

[132] Y. Zhu, Y. Zhou, H. Xu, Q. Ye, D. Doermann, and J. Jiao, "Learning instance activation maps for weakly supervised instance segmentation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 3111–3120.

[133] Q. Hou, P. Jiang, Y. Wei, and M. Cheng, "Self-erasing network for integral object attention," *CoRR*, vol. abs/1810.09821, 2018. [Online]. Available: http://arxiv.org/abs/1810.09821

[134] X. Li, H. Ma, and X. Luo, "Weaklier supervised semantic segmentation with only one image level annotation per category," *IEEE Transactions on Image Processing*, vol. 29, pp. 128–141, 2020.

[135] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[136] C. Labreuche and M. Grabisch, "The choquet integral for the aggregation of interval scales in multicriteria decision making," *CoRR*, vol. abs/0804.1762, 2008. [Online]. Available: http://arxiv.org/abs/0804.1762

[137] A. Mendez-Vazquez, P. Gader, J. M. Keller, and K. Chamberlin, "Minimum classification error training for choquet integrals with applications to landmine detection," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 1, pp. 225–238, 2008.

[138] M. Grabisch, "The application of fuzzy integrals in multicriteria decision making," *European Journal of Operational Research*, vol. 89, pp. 445–456, 1996.

[139] ——, "A new algorithm for identifying fuzzy measures and its application to pattern recognition," in *Proceedings of 1995 IEEE International Conference on Fuzzy Systems*, vol. 1, 1995, pp. 145–150.

[140] A. Mendez-Vazquez and P. Gader, "Learning fuzzy measure parameters by logistic LASSO," in *2008 Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, 2008, pp. 1–7.

[141] J. M. Keller and J. K. Osborn, "Training the fuzzy integral," *Int. J. Approx. Reason.*, vol. 15, pp. 1–24, 1996.

[142] B. Murray, D. T. Anderson, and T. C. Havens, "Actionable XAI for the fuzzy integral," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021, pp. 1–8.

[143] B. J. Murray, M. A. Islam, A. J. Pinar, D. T. Anderson, G. J. Scott, T. C. Havens, and J. M. Keller, "Explainable AI for the choquet integral," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 5, no. 4, pp. 520–529, 2021.

[144] D. T. Anderson, J. M. Keller, and T. C. Havens, “Learning fuzzy-valued fuzzy measures for the fuzzy-valued sugeno fuzzy integral,” in *IPMU*, 2010.

[145] D. T. Anderson, S. R. Price, and T. C. Havens, “Regularization-based learning of the choquet integral,” in *2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2014, pp. 2519–2526.

[146] M. Sugeno, “Theory of fuzzy integrals and its applications,” 1975.

[147] M. Fitting and E. Orłowska, “Beyond two: Theory and applications of multiple-valued logic,” 01 2003.

[148] X. Du and A. Zare, “Multi-resolution multi-modal sensor fusion for remote sensing data with label uncertainty,” *CoRR*, vol. abs/1805.00930, 2018. [Online]. Available: http://arxiv.org/abs/1805.00930

[149] T. Murofushi and M. Sugeno, “A theory of fuzzy measures: Representations, the choquet integral, and null sets,” *Journal of Mathematical Analysis and Applications*, vol. 159, no. 2, pp. 532–549, 1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022247X9190213J

[150] T. Murofushi, M. Sugeno, and M. Machida, “Non-monotonic fuzzy measures and the choquet integral,” *Fuzzy Sets and Systems*, vol. 64, pp. 73–86, 1994.

[151] Y. Narukawa and V. Torra, “Non-monotonic fuzzy measures and intuitionistic fuzzy sets,” in *Proceedings of the Third International Conference on Modeling Decisions for Artificial Intelligence*, ser. MDAI’06. Berlin, Heidelberg: Springer-Verlag, 2006, p. 150–160. [Online]. Available: https://doi.org/10.1007/11681960_16

[152] T. Chaira, “Fuzzy measures in image processing,” *Studies in Fuzziness and Soft Computing*, vol. 220, 01 2008.

[153] J. M. Keller and C. L. Carpenter, “Image segmentation in the presence of uncertainty,” *International Journal of Intelligent Systems*, vol. 5, 1990.

[154] J. Keller, P. Gader, and A. Hocaoglu, “Fuzzy integrals in image processing and recognition,” *Fuzzy Measures and Integrals: Theory and Applications*, 01 2000.

[155] R. Yang, Z. Wang, P.-A. Heng, and K. Leung, “Fuzzified choquet integral with a fuzzy-valued integrand and its application on temperature prediction,” *IEEE transactions on systems, man, and cybernetics*, vol. 38, pp. 367–80, 05 2008.

[156] M. Popescu, J. Keller, and J. Mitchell, “Gene ontology automatic annotation using a domain based gene product similarity measure,” in *The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ ’05.*, 2005, pp. 108–113.

[157] ——, “Fuzzy measures on the gene ontology for gene product similarity,” *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 3, pp. 263–74, 07 2006.

[158] E. Szmidt and J. Kacprzyk, “A similarity measure for intuitionistic fuzzy sets and its application in supporting medical diagnostic reasoning,” in *Artificial Intelligence and Soft Computing (ICAISC)*, L. Rutkowski, J. H. Siekmann, R. Tadeusiewicz, and L. A. Zadeh, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 388–393.

[159] A. Wilbik and J. M. Keller, “A fuzzy measure similarity between sets of linguistic summaries,” *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 1, pp. 183–189, 2013.

[160] H. Jiang and R. Eastman, “Application of fuzzy measures in multi-criteria evaluation in gis,” *International Journal of Geographical Information Science*, vol. 14, pp. 173–184, 03 2000.

[161] X. Du, A. Zare, J. M. Keller, and D. T. Anderson, “Multiple instance choquet integral for classifier fusion,” in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 1054–1061.

[162] X. Du and A. Zare, “Multiple instance choquet integral classifier fusion and regression for remote sensing applications,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 5, pp. 2741–2753, May 2019.

[163] D. Liginlal and T. T. Ow, “Modeling attitude to risk in human decision processes: An application of fuzzy measures,” *Fuzzy Sets Syst.*, vol. 157, pp. 3040–3054, 2006.

[164] D. T. Anderson, M. A. Islam, R. King, N. H. Younan, J. R. Fairley, S. Howington, F. Petry, P. Elmore, and A. Zare, “Binary fuzzy measures and choquet integration for multi-source fusion,” in *2017 International Conference on Military Technologies (ICMT)*, May 2017, pp. 676–681.

[165] M. Grabisch, “Fuzzy measures and integrals: Recent developments,” in *Fifty Years of Fuzzy Logic and its Applications*, 2015.

[166] P. Gader, A. Mendez-Vasquez, K. Chamberlin, J. Bolton, and A. Zare, “Multi-sensor and algorithm fusion with the choquet integral: applications to landmine detection,” in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, vol. 3, 2004, pp. 1605–1608.

[167] P. Gader, W.-H. Lee, and A. Mendez-Vasquez, “Continuous choquet integrals with respect to random sets with applications to landmine detection,” in *2004 IEEE International Conference on Fuzzy Systems*, vol. 1, 2004, pp. 523–528.

[168] Q. Wang, C. Zheng, H. Yu, and D. Deng, “Integration of heterogeneous classifiers based on choquet fuzzy integral,” in *7th International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 1, 2015, pp. 543–547.

[169] W.-t. Chen and P. Gader, “Word level discriminative training for handwritten word recognition,” 11 2000.

[170] P. D. Gader, M. A. Mohamed, and J. M. Keller, “Fusion of handwritten word classifiers,” *Pattern Recognition Letters*, vol. 17, no. 6, pp. 577 – 584, 1996.

[171] S. Feng, W. Shang, and Y. Wang, “A k-highest expert text classification algorithm based on choquet integral,” in *3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence*, 2015, pp. 499–503.

[172] R. Stanley, J. Keller, C. Caldwell, and P. Gader, “Abnormal cell detection using the choquet integral,” in *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference*, vol. 2, 2001, pp. 1134–1139 vol.2.

[173] Y.-L. Hsiao, P.-F. Chen, C.-F. Tsai, H.-C. Liu, and P.-C. Chang, “Choquet integral algorithm for t-cell epitope prediction based on fuzzy measure,” in *Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, vol. 3, 2011, pp. 1588–1591.

[174] K. Hirota, H. A. Vu, P. Q. Le, C. Fatichah, Z. Liu, Y. Tang, M. L. Tangel, Z. Mu, B. Sun, F. Yan, D. Masano, O. Thet, M. Yamaguchi, F. Dong, and Y. Yamazaki, “Multimodal gesture recognition based on choquet integral,” in *2011 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2011)*, 2011, pp. 772–776.

[175] T. C. Havens, D. T. Anderson, C. Wagner, H. Deilamsalehy, and D. Wonnacott, “Fuzzy integrals of crowd-sourced intervals using a measure of generalized accord,” in *2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2013, pp. 1–8.

[176] J. Keller, P. Gader, R. Krishnapuram, X. Wang, A. Koksal Hocaoglu, H. Frigui, and J. Moore, “A fuzzy logic automatic target detection system for LADAR range images,” in *1998 IEEE International Conference on Fuzzy Systems Proceedings*, vol. 1, 1998, pp. 71–76 vol.1.

[177] A. Hocaoglu and P. Gader, “An interpretation of discrete choquet integrals in morphological image processing,” in *The 12th IEEE International Conference on Fuzzy Systems (FUZZ ’03)*, vol. 2, 2003, pp. 1291–1295 vol.2.

[178] J. Bolton, P. Gader, and J. N. Wilson, “Discrete choquet integral as a distance metric,” *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 4, pp. 1107–1110, 2008.

[179] T. Back, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. USA: Oxford University Press, Inc., 1996.

[180] J. Wang and Z. Wang, “Using neural networks to determine sugeno measures by statistics,” *Neural Networks*, vol. 10, pp. 183–195, 1997.

[181] W. Wang, Z. Wang, and G. J. Klir, “Genetic algorithms for determining fuzzy measures from data,” *J. Intell. Fuzzy Syst.*, vol. 6, pp. 171–183, 1998.

[182] Z. Wang and H.-F. Guo, “A new genetic algorithm for nonlinear multiregressions based on generalized choquet integrals,” in *The 12th IEEE International Conference on Fuzzy Systems (FUZZ ’03)*, vol. 2, 2003, pp. 819–821.

[183] J.-L. Marichal and M. Roubens, “Determination of weights of interacting criteria from a reference set,” *European Journal of Operational Research*, vol. 124, no. 3, pp. 641–650, 2000. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221799001824

[184] I. Kojadinovic, J.-L. Marichal, and M. Roubens, “An axiomatic approach to the definition of the entropy of a discrete choquet capacity,” *Information Sciences*, vol. 172, no. 1, pp. 131–153, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025504001811

[185] I. Kojadinovic, “Minimum variance capacity identification,” *European Journal of Operational Research*, 2006.

[186] P. Meyer and M. Roubens, *Choice, Ranking and Sorting in Fuzzy Multiple Criteria Decision Aid*. New York, NY: Springer New York, 2005, pp. 471–503. [Online]. Available: https://doi.org/10.1007/0-387-23081-5_12

[187] M. Grabisch, I. Kojadinovic, and P. Meyer, “A review of methods for capacity identification in choquet integral based multi-attribute utility theory: Applications of the kappalab r package,” *European Journal of Operational Research*, vol. 186, no. 2, pp. 766–785, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0377221707002330

[188] H. Nemmour and Y. Chibani, “Fuzzy neural network architecture for change detection in remotely sensed imagery,” *International Journal of Remote Sensing*, vol. 27, pp. 705–717, 02 2006.

[189] X. Du, A. Zare, J. M. Keller, and D. T. Anderson, “Multiple instance choquet integral for classifier fusion,” in *2016 IEEE Congress on Evolutionary Computation (CEC)*, July 2016, pp. 1054–1061.

[190] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95 - International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948 vol.4.

[191] Z. Wang, K. sak Leung, and J. Wang, “A genetic algorithm for determining nonadditive set functions in information fusion,” *Fuzzy Sets and Systems*, vol. 102, no. 3, pp. 463–469, 1999, fuzzy Measures and Integrals. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0165011498002206

[192] T.-Y. Chen, J.-C. Wang, and G.-H. Tzeng, “Identification of general fuzzy measures by genetic algorithms based on partial information,” *Systems, Man, and Cybernetics*, vol. 30, pp. 517 – 528, 09 2000.

[193] E. Combarro and P. Miranda, “Identification of fuzzy measures from sample data with genetic algorithms,” *Computers*, vol. 33, pp. 3046–3066, 10 2006.

[194] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[195] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2015.

[196] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818–2826.

[197] M. Tan and Q. V. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” *CoRR*, vol. abs/1905.11946, 2019. [Online]. Available: http://arxiv.org/abs/1905.11946

[198] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” *CoRR*, vol. abs/1409.4842, 2014. [Online]. Available: http://arxiv.org/abs/1409.4842

[199] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” *CoRR*, vol. abs/1611.05431, 2016. [Online]. Available: http://arxiv.org/abs/1611.05431

[200] Q. Zheng, Z. Wang, J. Lu, and J. Zhou, “Shap-CAM: Visual explanations for convolutional neural networks based on shapley value,” 2022. [Online]. Available: https://openreview.net/forum?id=C1lIXY_T1LTs

[201] G. Papandreou, L. Chen, K. Murphy, and A. L. Yuille, “Weakly- and semi-supervised learning of a DCNN for semantic image segmentation,” *CoRR*, vol. abs/1502.02734, 2015. [Online]. Available: http://arxiv.org/abs/1502.02734

[202] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, “DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[203] X. Du, A. Zare, and D. T. Anderson, “Multiple instance choquet integral with binary fuzzy measures for remote sensing classifier fusion with imprecise labels,” in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2019, pp. 1154–1162.

[204] Y. LeCun and C. Cortes, “MNIST handwritten digit database,” http://yann.lecun.com/exdb/mnist/, 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[205] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[206] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 248–255.

[207] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[208] W.-T. Chen, P. Gader, and H. Shi, “Lexicon-driven handwritten word recognition using optimal linear combinations of order statistics,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 1, pp. 77–82, 1999.

[209] M. Belkin and P. Niyogi, “Semi-supervised learning on riemannian manifolds,” *Machine Learning*, vol. 56, no. 1, pp. 209–239, Jul 2004.

[210] M. Tschannen, O. Bachem, and M. Lucic, “Recent advances in autoencoder-based representation learning,” *CoRR*, vol. abs/1812.05069, 2018. [Online]. Available: http://arxiv.org/abs/1812.05069

[211] J. Chen, B. Xie, H. Zhang, and J. Zhai, *Deep Autoencoders in Pattern Recognition: A Survey*, 2019, ch. 9, pp. 229–255.

[212] A. Kosiorek, S. Sabour, Y. W. Teh, and G. E. Hinton, “Stacked capsule autoencoders,” in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/2e0d41e02c5be4668ec1b0730b3346a8-Paper.pdf

[213] M. Belkin, P. Niyogi, and V. Sindhwani, “Manifold regularization: A geometric framework for learning from labeled and unlabeled examples,” *J. Mach. Learn. Res.*, vol. 7, pp. 2399–2434, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248632

[214] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled faces in the wild: A database for studying face recognition in unconstrained environments,” University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[215] I. Rish, G. Grabarnik, G. Cecchi, F. Pereira, and G. J. Gordon, “Closed-form supervised dimensionality reduction with generalized linear models,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML ’08. New York, NY, USA: Association for Computing Machinery, 2008, p. 832–839. [Online]. Available: https://doi.org/10.1145/1390156.1390261

[216] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000. [Online]. Available: https://science.sciencemag.org/content/290/5500/2323

[217] B. Schölkopf, A. J. Smola, and K.-R. Müller, *Kernel Principal Component Analysis*. Cambridge, MA, USA: MIT Press, 1999, p. 327–352.

[218] M. Scholz, M. Fraunholz, and J. Selbig, “Nonlinear principal component analysis: Neural network models and applications,” in *Principal Manifolds for Data Visualization and Dimension Reduction*, A. N. Gorban, B. Kégl, D. C. Wunsch, and A. Y. Zinovyev, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 44–67.

[219] M. E. Tipping and C. Bishop, “Probabilistic principal component analysis,” *Journal of the Royal Statistical Society, Series B*, vol. 21, no. 3, pp. 611–622, January 1999. [Online]. Available: https://www.microsoft.com/en-us/research/publication/probabilistic-principal-component-analysis/

[220] A. Hyvarinen and E. Oja, “Independent component analysis: algorithms and applications,” *Neural Networks*, vol. 13, no. 4, pp. 411 – 430, 2000.

[221] A. Tharwat, “Independent component analysis: An introduction,” *Applied Computing and Informatics*, 2018.

[222] F. R. Bach and M. I. Jordan, “A probabilistic interpretation of canonical correlation analysis,” 2005.

[223] W. S. Torgerson, “Multidimensional scaling: I. theory and method,” *Psychometrika*, vol. 17, pp. 401 – 419, 1952.

[224] C. O. S. Sorzano, J. Vargas, and A. P. Montano, “A survey of dimensionality reduction techniques,” 2014.

[225] D. M. Witten and R. Tibshirani, “Supervised multidimensional scaling for visualization, classification, and bipartite ranking,” *Computational Statistics and Data Analysis*, vol. 55, no. 1, pp. 789 – 801, 2011.

[226] N. Gillis, “The why and how of nonnegative matrix factorization,” 2014.

[227] Y. Xu, W. Ping, and A. T. Campbell, “Multi-instance metric learning,” in *2011 IEEE 11th International Conference on Data Mining*, Dec 2011, pp. 874–883.

[228] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin, “Graph embedding and extensions: A general framework for dimensionality reduction,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 1, pp. 40–51, Jan 2007.

[229] Y. Wang, J.-B. Xie, and Y. Wu, “Orthogonal discriminant analysis revisited,” *Pattern Recognition Letters*, vol. 84, pp. 149 – 155, 2016.

[230] K. Fukunaga and J. M. Mantock, “Nonparametric discriminant analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 5, no. 6, pp. 671–678, Nov 1983.

[231] G. Baudat and F. Anouar, “Generalized discriminant analysis using a kernel approach,” *Neural Computation*, vol. 12, no. 10, pp. 2385–2404, 2000.

[232] A. R. Webb and D. Lowe, “The optimised internal representation of multilayer classifier networks performs nonlinear discriminant analysis,” *Neural Networks*, vol. 3, no. 4, pp. 367 – 375, 1990.

[233] W. Liu, J. C. Principe, and S. Haykin, *Kernel Adaptive Filtering: A Comprehensive Introduction*, 1st ed. Wiley Publishing, 2010.

[234] A. Webb, “A kernel approach to metric multidimensional scaling,” in *Structural, Syntactic, and Statistical Pattern Recognition*, T. Caelli, A. Amin, R. P. W. Duin, D. de Ridder, and M. Kamel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 452–460.

[235] B. Ghojogh, F. Karray, and M. Crowley, “Fisher and kernel fisher discriminant analysis: Tutorial,” *CoRR*, vol. abs/1906.09436, 2019. [Online]. Available: https://arxiv.org/abs/1906.09436

[236] Q. Wang, “Kernel principal component analysis and its applications in face recognition and active shape models,” *CoRR*, vol. abs/1207.3538, 2012. [Online]. Available: http://arxiv.org/abs/1207.3538

[237] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, May 2013.

[238] E. Bengoetxea, “Inexact graph matching using estimation of distribution algorithms,” Ph.D. dissertation, Ecole Nationale Supérieure des Télécommunications, Paris, France, Dec. 2002.

[239] L. Livi and A. Rizzi, “The graph matching problem,” *Pattern Anal. Appl.*, vol. 16, no. 3, pp. 253–283, Aug 2013.

[240] B. Ribeiro, A. Vieira, and J. Carvalho das Neves, “Supervised isomap with dissimilarity measures in embedding learning,” in *Progress in Pattern Recognition, Image Analysis and Applications*, J. Ruiz-Shulcloper and W. G. Kropatsch, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 389–396.

[241] Y. Zhang, Z. Zhang, J. Qin, L. Zhang, B. Li, and F. Li, “Semi-supervised local multi-manifold isomap by linear embedding for feature extraction,” *Pattern Recognition*, vol. 76, pp. 662 – 678, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320317303977

[242] M. Vlachos, C. Domeniconi, D. Gunopulos, G. Kollios, and N. Koudas, "Non-linear dimensionality reduction techniques for classification and visualization," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '02. New York, NY, USA: Association for Computing Machinery, 2002, p. 645–651. [Online]. Available: https://doi.org/10.1145/775047.775143

[243] V. D. Silva and J. B. Tenenbaum, "Global versus local methods in nonlinear dimensionality reduction," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds. MIT Press, 2003, pp. 721–728. [Online]. Available: http://papers.nips.cc/paper/2141-global-versus-local-methods-in-nonlinear-dimensionality-reduction.pdf

[244] V. de Silva and J. B. Tenenbaum, *Unsupervised Learning of Curved Manifolds*. New York, NY: Springer New York, 2003, pp. 453–465. [Online]. Available: https://doi.org/10.1007/978-0-387-21579-2_31

[245] Chun-Guang Li and Jun Guo, "Supervised isomap with explicit mapping," in *First International Conference on Innovative Computing, Information and Control (ICICIC)*, vol. 3, Aug 2006, pp. 345–348.

[246] J. W. Sammon, "A nonlinear mapping for data structure analysis," *IEEE Transactions on Computers*, vol. C-18, no. 5, pp. 401–409, 1969.

[247] K. Weinberger, F. Sha, and L. Saul, "Learning a kernel matrix for nonlinear dimensionality reduction," 07 2004.

[248] Y. Pang, L. Zhang, Z. Liu, N. Yu, and H. Li, "Neighborhood preserving projections (npp): A novel linear dimension reduction method," in *Advances in Intelligent Computing*, D.-S. Huang, X.-P. Zhang, and G.-B. Huang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 117–125.

[249] E. Kokiopoulou and Y. Saad, "Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 12, pp. 2143–2156, 2007.

[250] Z. Li, W. Shi, X. Shi, and Z. Zhong, "A supervised manifold learning method," *Comput. Sci. Inf. Syst.*, vol. 6, pp. 205–215, 12 2009.

[251] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Computation*, vol. 15, no. 6, pp. 1373–1396, 2003.

[252] B. Raducanu and F. Dornaika, "A supervised non-linear dimensionality reduction approach for manifold learning," *Pattern Recognition*, vol. 45, no. 6, pp. 2432 – 2444, 2012.

[253] X. He and P. Niyogi, "Locality preserving projections," in *Proceedings of the 16th International Conference on Neural Information Processing Systems*, ser. NIPS'03. Cambridge, MA, USA: MIT Press, 2003, p. 153–160.

[254] F. Ratle, G. Camps-Valls, and J. Weston, “Semisupervised neural networks for efficient hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 5, pp. 2271–2282, May 2010.

[255] B. Ren, B. Hou, J. Zhao, and L. Jiao, “Unsupervised classification of polarimetirc sar image via improved manifold regularized low-rank representation with multiple features,” *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 10, no. 2, pp. 580–595, Feb 2017.

[256] I. W. Tsang and J. T. Kwok, “Large-scale sparsified manifold regularization,” in *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman, Eds. MIT Press, 2007, pp. 1401–1408. [Online]. Available: http://papers.nips.cc/paper/3005-large-scale-sparsified-manifold-regularization.pdf

[257] M. Meng and X. Zhan, “Zero-shot learning via low-rank-representation based manifold regularization,” *IEEE Signal Processing Letters*, vol. 25, no. 9, pp. 1379–1383, Sep. 2018.

[258] H. Li, D. Liu, and D. Wang, “Approximate policy iteration with unsupervised feature learning based on manifold regularization,” in *2015 International Joint Conference on Neural Networks (IJCNN)*, July 2015, pp. 1–6.

[259] D. L. Donoho and C. Grimes, “Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data,” *Proceedings of the National Academy of Sciences*, vol. 100, no. 10, pp. 5591–5596, 2003.

[260] N. Patwari and A. O. Hero, “Manifold learning algorithms for localization in wireless sensor networks,” in *2004 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 3, May 2004.

[261] Z. Zhang and H. Zha, “Principal manifolds and nonlinear dimension reduction via local tangent space alignment,” *CoRR*, vol. cs.LG/0212008, 2002.

[262] T. Zhang, J. Yang, D. Zhao, and X. Ge, “Linear local tangent space alignment and application to face recognition,” *Neurocomputing*, vol. 70, no. 7, pp. 1547 – 1553, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231206004577

[263] H. Li, W. Chen, and I.-F. Shen, “Supervised local tangent space alignment for classification.” pp. 1620–1621, 01 2005.

[264] L. Ma, M. M. Crawford, and J. W. Tian, “Generalised supervised local tangent space alignment for hyperspectral image classification,” *Electronics Letters*, vol. 46, no. 7, pp. 497 –498, April 2010.

[265] R. R. Coifman and S. Lafon, “Diffusion maps,” *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5 – 30, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1063520306000546

[266] B. Hajek, *Random Processes for Engineers*. Cambridge University Press, 2015.

[267] J. Delaporte, B. M. Herbst, W. Hereman, and S. V. der Walt, "An introduction to diffusion maps," *Proceedings of the 19th Symposium of the Pattern Recognition Association of South Africa (PRASA)*, 2008.

[268] A. N. Gorban and A. Y. Zinovyev, "Elastic maps and nets for approximating principal manifolds and their application to microarray data visualization," in *Principal Manifolds for Data Visualization and Dimension Reduction*, A. N. Gorban, B. Kégl, D. C. Wunsch, and A. Y. Zinovyev, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 96–130.

[269] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE Journal*, vol. 37, no. 2, pp. 233–243, 1991. [Online]. Available: https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690370209

[270] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[271] B. Fritzke, "A growing neural gas network learns topologies," in *Proceedings of the 7th International Conference on Neural Information Processing Systems*, ser. NIPS'94. Cambridge, MA, USA: MIT Press, 1994, pp. 625–632. [Online]. Available: http://dl.acm.org/citation.cfm?id=2998687.2998765

[272] S. S. Haykin, *Neural networks and learning machines*, 3rd ed. Upper Saddle River, NJ: Pearson Education, 2009.

[273] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals through Simulations with CD-ROM*, 1st ed. USA: John Wiley and Sons, Inc., 1999.

[274] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11 – 26, 2017.

[275] A. Dhillon and G. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, 12 2019.

[276] O. Abiodun, A. Jantan, O. Omolara, K. Dada, N. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, p. e00938, 11 2018.

[277] N. Shahid, T. Rappon, and W. Berta, "Applications of artificial neural networks in health care organizational decision-making: A scoping review," *PLOS ONE*, vol. 14, no. 2, pp. 1–22, 02 2019. [Online]. Available: https://doi.org/10.1371/journal.pone.0212356

[278] M. Z. Alom, T. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. Nasrin, M. Hasan, B. Essen, A. Awwal, and V. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, p. 292, 03 2019.

[279] F.-N. Yuan, L. Zhang, J.-T. Shi, X. Xia, and G. Li, "Theories and applications of auto-encoder neural networks: A literature survey," *Jisuanji Xuebao/Chinese Journal of Computers*, vol. 42, pp. 203–230, 01 2019.

[280] R. Rojas, "Associative networks," in *Neural Networks - A Systematic Introduction*, 1st ed. Berlin, New-York: Springer-Verlag, 1996, ch. 12, pp. 311–336.

[281] J. E. Ball, D. T. Anderson, and C. S. C. Sr., "Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community," *Journal of Applied Remote Sensing*, vol. 11, no. 4, pp. 1 – 54, 2017.

[282] B. Dai, Y. Wang, J. Aston, G. Hua, and D. Wipf, "Hidden talents of the variational autoencoder," 2017.

[283] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *CoRR*, vol. abs/1901.00596, 2019. [Online]. Available: http://arxiv.org/abs/1901.00596

[284] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, July 2017.

[285] Q. Liu, M. Allamanis, M. Brockschmidt, and A. L. Gaunt, "Constrained graph variational autoencoders for molecule design," *CoRR*, vol. abs/1805.09076, 2018. [Online]. Available: http://arxiv.org/abs/1805.09076

[286] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder," *CoRR*, vol. abs/1802.04407, 2018. [Online]. Available: http://arxiv.org/abs/1802.04407

[287] H. Yin, "Learning nonlinear principal manifolds by self-organising maps," *Lecture Notes in Computational Science and Engineering*, vol. 60, pp. 68–95, 09 2007.

[288] A. Rauber, D. Merkl, and M. Dittenbach, "The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data," *IEEE Transactions on Neural Networks*, vol. 13, no. 6, pp. 1331–1341, Nov 2002.

[289] D. E. Rumelhart and D. Zipser, "Feature discovery by competitive learning," *Cognitive Science*, vol. 9, no. 1, pp. 75–112, 1985. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1207/s15516709cog0901_5

[290] J. Chiang and P. D. Gader, "Hybrid fuzzy-neural systems in handwritten word recognition," *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 4, pp. 497–510, Nov 1997.

[291] H. Frigui and P. Gader, "Detection and discrimination of land mines in ground-penetrating radar based on edge histogram descriptors and a possibilistic $k$-nearest neighbor classifier," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 1, Feb 2009.

[292] T. Kohonen, *Learning Vector Quantization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, pp. 175–189.

[293] M. Pena, W. Barbakh, and C. Fyfe, "Topology-preserving mappings for data visualisation," in *Principal Manifolds for Data Visualization and Dimension Reduction*, A. N. Gorban, B. Kégl, D. C. Wunsch, and A. Y. Zinovyev, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 131–150.

[294] E. J. Palomo and E. Lopez-Rubio, "Learning topologies with the growing neural forest," *International Journal of Neural Systems*, vol. 26, no. 04, p. 1650019, 2016. [Online]. Available: https://doi.org/10.1142/S0129065716500192

[295] Q. Sun, H. Liu, and T. Harada, "Online growing neural gas for anomaly detection in changing surveillance scenes," *Pattern Recognition*, vol. 64, pp. 187 – 201, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0031320316302771

[296] E. Lopez-Rubio and E. J. Palomo, "Growing hierarchical probabilistic self-organizing graphs," *IEEE Transactions on Neural Networks*, vol. 22, no. 7, pp. 997–1008, July 2011.

[297] M. Gönen, "Bayesian supervised dimensionality reduction," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2179–2189, Dec 2013.

[298] N. Lawrence, "Probabilistic non-linear principal component analysis with gaussian process latent variable models," *J. Mach. Learn. Res.*, vol. 6, pp. 1783–1816, dec 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1046920.1194904

[299] X. Gao, X. Wang, D. Tao, and X. Li, "Supervised gaussian process latent variable model for dimensionality reduction," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 41, no. 2, pp. 425–434, April 2011.

[300] M. Brand, "Charting a manifold," in *Advances in Neural Information Processing Systems 15*. MIT Press, 2003, pp. 961–968.

[301] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html

[302] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," in *Advances in Neural Information Processing Systems 15*, S. Becker, S. Thrun, and K. Obermayer, Eds. MIT Press, 2003, pp. 857–864.

[303] J. Tang, J. Liu, M. Zhang, and Q. Mei, “Visualization large-scale and high-dimensional data,” *CoRR*, vol. abs/1602.00370, 2016.

[304] L. McInnes, J. Healy, and J. Melville, “Umap: Uniform manifold approximation and projection for dimension reduction,” 2018.

[305] A. Hermans, L. Beyer, and B. Leibe, “In defense of the triplet loss for person re-identification,” *CoRR*, vol. abs/1703.07737, 2017. [Online]. Available: http://arxiv.org/abs/1703.07737

[306] K. Sohn, “Improved deep metric learning with multi-class n-pair loss objective,” in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 1857–1865.

[307] G. R. Koch, “Siamese neural networks for one-shot image recognition,” 2015.

[308] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832

[309] J. Deng, J. Guo, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” *CoRR*, vol. abs/1801.07698, 2019.

[310] K. Q. Weinberger and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” *J. Mach. Learn. Res.*, vol. 10, p. 207–244, Jun. 2009.

[311] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, “Signature verification using a “siamese” time delay neural network,” in *Proceedings of the 6th International Conference on Neural Information Processing Systems*, ser. NIPS’93. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993, p. 737–744.

[312] G. Koch, “Siamese neural networks for one-shot image recognition,” Ph.D. dissertation, University of Toronto, Toronto, Canada, 2015.

[313] E. Hoffer and N. Ailon, “Deep metric learning using triplet network,” *Lecture Notes in Computer Science*, p. 84–92, 2015. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-24261-3_7

[314] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, “A simple framework for contrastive learning of visual representations,” 2020.

## BIOGRAPHICAL SKETCH

Connor McCurley received the Doctor of Philosophy (Ph.D.) degree in electrical and computer engineering from the University of Florida, Gainesville, FL, USA, in 2022. He has been recognized with several awards including the University of Florida Graduate School Preeminence Award and the Wilson and Marie Collins Endowment for Graduate Fellowship. In addition to his own research, Connor has dedicated time to mentoring undergraduate researchers. His general research interests include target detection, information fusion, manifold learning/dimensionality reduction, and learning from imprecise, uncertain, and ambiguous annotations.