Scalable and Ensemble Learning for Big Data

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Panagiotis Traganitis

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Prof. Georgios B. Giannakis, Advisor

May, 2019

Acknowledgments

Firstly, I would like to extend my sincerest thanks and deepest gratitude to my advisor, Prof. Georgios B. Giannakis, for giving me the opportunity and welcoming me as his student in his prestigious research group and introducing me to the world of academic research. His vast knowledge, guidance and invaluable advice, have helped shape both this thesis and myself as an aspiring researcher.

I would also like to deeply thank my collaborators during my PhD years, Prof. Konstantinos Slavakis, and Prof. Alba Pagés-Zamora. Special thanks go to my thesis committee members, Prof. Mehmet Akcakaya, Prof. George Karypis, and Prof. Mostafa Kaveh for all their valuable comments and feedback on my research and thesis. I would also like to deeply thank Prof. Nikolaos Sidiropoulos. Our discussions with Prof. Sidiropoulos have always been fruitful and his advice has been pivotal.

In addition, I am grateful to all of my professors at the University of Minnesota, whose graduate courses were enlightening and gave me the necessary tools for my research. Furthermore, for the fruitful discussions we have had, as well as their company I would like to thank all the current and previous members of the SPiNCOM group: Dr. Brian Baingana, Prof. Juan-Andrés Bazerque, Dr. Dimitris Berberidis, Dr. Jia Chen, Prof. Tianyi Chen, Prof. Emiliano Dall'Anese, Vassilis Ioannidis, Georgios V. Karanikolas, Donghoon Lee, Prof. Geert Leus, Bingcong Li, Meng Ma, Dr. Morteza Mardani, Prof. Antonio G. Marqués, Prof. Gonzalo Mateos, Dr. Athanasios Nikolakopoulos, Prof. Daniel Romero, Alireza Sadeghi, Dr. Fatemeh Sheikholeslami, Prof. Yanning Shen, Dr. Gang Wang, Dr. Yunlong Wang, Liang Zhang, Prof. Yu Zhang, Dr. Qin Lu, Elena Ceci, Seth Barrash, Prof. Nasim Yahya Soltani; as well as my friends and colleagues from the Digital Technology Center and University of Minnesota: Charilaos Kanatsoulis, Agoritsa Polyzou, Ioanna Polyzou, Andreas Katis, Maria Kalantzi, Dr. Dimitrios Zermas, Dr. Panagiotis Stanitsas, Dr. Ioannis Nompelis, Dr. Michail Vlysidis, Dr.

Dionysios Angelidis, Dr. Athanasios Touris, Dr. Vasileios Kalantzis, Dr. Dimitrios Kalliontzis, Nikolaos Stefas, Vasileios Charitatos, Lambros Tassoulas, Stylianos Daoutidis, Nikos Kargas, Dr. Mohit Sharma, Dr. Spyros Charonis, Vasileios Lytsakis, Dr. Shaden Smith, Dr. Evangelia Christakopoulou, Dr. Konstantina Christakopoulou, Dr. Aritra Konar, Bo Yang, Ahmed Zamzam, and Faisal Almutairi.

I am very grateful to my parents, Apostolos Traganitis and Eleni-Alkmini Chatzigogou, and close friends, without whose support and understanding I would not have made it this far.

Panagiotis Traganitis, Minneapolis, Spring, 2019.

The work in this Thesis was supported by NSF grants 1343860, 1442686, 1500713, 1514056.

Dedication

This dissertation is dedicated to my family for all their love and support.

Abstract

The turn of the decade has trademarked society and computing research with a "data deluge." As the number of smart, highly accurate and Internet-capable devices increases, so does the amount of data that is generated and collected. While this sheer amount of data has the potential to enable high quality inference, and mining of information, it introduces numerous challenges in the processing and pattern analysis, since available statistical inference and machine learning approaches do not necessarily scale well with the number of data and their dimensionality. In addition to the challenges related to scalability, data gathered are often noisy, dynamic, contaminated by outliers or corrupted to specifically inhibit the inference task. Moreover, many machine learning approaches have been shown to be susceptible to adversarial attacks. At the same time, the cost of cloud and distributed computing is rapidly declining. Therefore, there is a pressing need for statistical inference and machine learning tools that are robust to attacks and scale with the volume and dimensionality of the data, by harnessing efficiently the available computational resources.

This thesis is centered on analytical and algorithmic foundations that aim to enable statistical inference and data analytics from large volumes of high-dimensional data. The vision is to establish a comprehensive framework based on state-of-the-art machine learning, optimization and statistical inference tools to enable truly large-scale inference, which can tap on the available (possibly distributed) computational resources, and be resilient to adversarial attacks. The ultimate goal is to both analytically and numerically demonstrate *how valuable insights from signal processing can lead to markedly improved and accelerated learning tools*.

To this end, the present thesis investigates two main research thrusts: i) Large-scale subspace clustering; and ii) unsupervised ensemble learning. The aforementioned research thrusts introduce novel algorithms that aim to tackle the issues of large-scale learning. The potential of the proposed algorithms is showcased by rigorous theoretical results and extensive numerical tests.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data analytics and machine learning already have a ubiquitous presence in our daily lives [29]. Social networking sites, such as Facebook and LinkedIn, analyze user activity and automatically adjust the content they offer. Services such as Amazon, Netflix or Spotify automatically recommend new movies and music to their clients based on their past activity and the activity of similar users. Banks and credit companies use machine learning to make credit decisions as well as detect fraudulent activities. E-mail services have developed intelligent algorithms to filter out unwanted emails (spam), but also automatically categorize incoming emails. In order to facilitate all of the aforementioned services, ever increasing amounts of data have to be processed. Aside from their social effect, machine learning and statistical inference have well documented impact in the sciences. As sensors and scientific instruments become increasingly accurate, the data produced by them increases in dimension, requiring increased computational resources for their processing. While the computational demands of modern machine learning and statistical inference tools increase dramatically, the cost of cloud computing is rapidly declining [101]. In addition, the emergent Internet-of-Things (IoT) [153], which consists of numerous connected devices, advocates data analytics methods that minimize the required communication. This prompts us to seek tools that perform efficiently in the highly distributed setups of IoT and cloud computing.

Adding to the challenges posed by the size and volume of the data, machine learning algorithms have been shown to be vulnerable to attacks from adversaries [89]. For example, spammers continuously try to outsmart the spam filtering algorithms deployed by email services. These attacks pose a serious threat, especially in systems where machine learning systems

perform critical tasks. Therefore, tools that are robust to and can detect adversarial attacks are of paramount importance.

This thesis will leverage contemporary science and engineering tools from disciplines as diverse as optimization, machine learning, signal processing, and big data processing, to put forth analytic and algorithmic foundations for learning efficiently from large volumes of high-dimensional data, possibly in the presence of adversaries.

The major challenge of large-scale learning is to design tools that are fast and efficient yet retain the accuracy of their batch counterparts. Existing approaches [17, 95] have relied on parallelization and stochastic optimization to develop efficient machine learning and data analytics schemes. However, parallelizing a large problem typically requires a lot of communication between computing nodes, and stochastic optimization suffers from slow convergence speeds, especially for non-convex problems. Other approaches [18, 151] tackle high-dimensional data by invoking the celebrated Johnson-Lindenstrauss (JL) lemma [67]. These methods reduce the dimensionality of the data by multiplying them with a data-agnostic random projection matrix, and then proceed with the learning task on the dimensionality reduced dataset. This approach however, is not tailored for large volumes of data.

Another approach for large-scale learning is ensemble learning. Ensemble learning is the task of creating a meta-learner, by combining the results of multiple individual learners [33]. Additionally, ensembling can significantly increase the performance of so-called "weak" learners, that is learners performing slightly better than random [44]. Thus, instead of training a highly complex algorithm, one can possibly use an array of simpler algorithms and combine their results in an appropriate manner. At the same time, as different machine learning and statistical inference algorithms operate under different assumptions, there is no "one-size-fits-all" algorithm. By combining the results from multiple algorithms, ensemble learning complements the strengths of each algorithm to produce a result that is better than the results provided from each individual algorithm.

Regarding adversarial machine learning, recent research has mainly focused on deep neural networks [52, 53], with only a few exploring how other machine learning algorithms are affected by adversarial examples [104, 139]. In addition, the effect of adversaries in an ensemble learning setup remains a basically uncharted territory.

The central goal of this thesis is to put forth algorithmic foundations and performance analyses for optimally handling large-scale high-dimensional data as well as dealing with

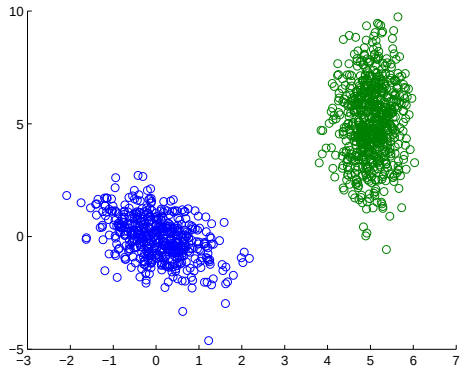


Figure 1.1: Example of a dataset with $K = 2$ clusters that are linearly separable.

adversarial agents that seek to undermine the machine learning task. The first line of research will focus on methods to accelerate clustering, and in particular the popular subspace clustering (SC) method, for general and tensor based data. The research in Chap. 2 is geared to address the following question:

- How can we accelerate existing SC schemes while maintaining high clustering accuracy?

The second line of research focuses on ensemble learning, and in particular blind (*unsupervised*) ensembles. Blind ensembles are well motivated when there is no knowledge of how different algorithms will perform on a particular dataset, and the meta-learner has no access to ground-truth data. In addition, the blind ensemble learning setup also emerges in fields such as crowdsourcing and distrubuted detection/estimation among others. The key contribution of this thrust will be to show that results from multiple heterogeneous learners can be judiciously combined, even without the presence of ground-truth data at the meta-learner. To this end, our research aims in Chapters 3 and 4 to answer the following question:

- How can we optimally and efficiently combine the answers of multiple algorithms/annotators in the absence of ground-truth data?
- How can we incorporate information about data dependencies in the blind ensemble learning task?
- How does the presence of adversaries affect the ensemble learning task?

The two main research thrusts of this thesis are briefly described in the following subchapters.

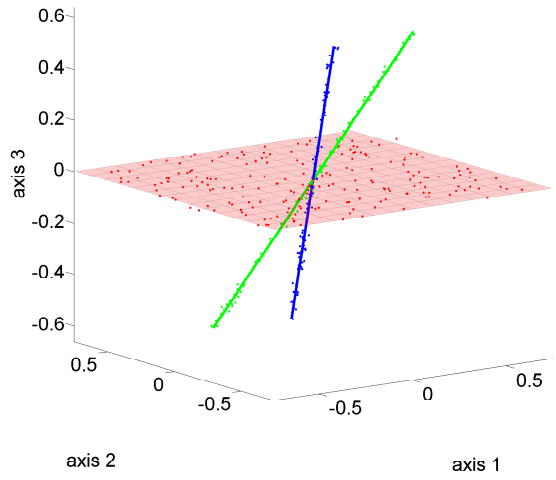


Figure 1.2: Data drawn from a union of subspaces model.

1.1 Large-scale subspace clustering

Clustering (a.k.a. unsupervised classification) is a method of grouping data, without having labels available. Also referred to as *graph partitioning* or *community identification*, it finds applications in data mining, signal processing, and machine learning. Arguably, the most popular clustering algorithm is K-means due to its simplicity [57]. However, K-means, as well as its kernel-based variants, provide meaningful clustering results only when data, after mapped to an appropriate feature space, form "tight" groups that can be separated by hyperplanes [57], see e.g. Fig. 1.1. Its scope is further broadened by the so-termed probabilistic and kernel K-means, with an instantiation of the latter being equivalent to *spectral clustering* – the popular tool for graph-partitioning that can cope even with nonlinearly separable data [32]

Subspace clustering (SC) on the other hand, is a popular method for clustering nonlinearly separable data which are generated by a union of (affine) subspaces in a high-dimensional Euclidean space [145], see e.g. Fig. 1.2. SC has well-documented impact in applications, as diverse as image and video segmentation, and identification of switching linear systems in controls [145]. The more recent SC methods can offer high levels of clustering performance, at the cost, however, of high computational complexity.

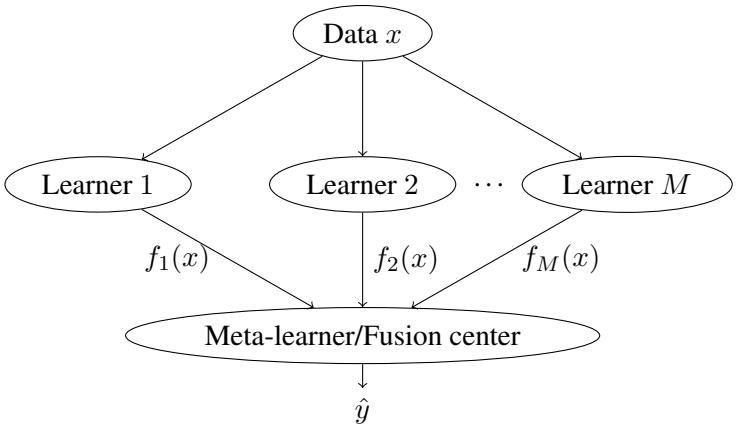


Figure 1.3: Unsupervised ensemble learning setup, where the outputs of learners are combined in parallel.

As the main issue with large-scale SC is the huge volume of data, to realize large-scale subspace clustering, this thesis will leverage results from the well-studied field of random projections. Specifically, random projections will be employed to reduce the number of data, while at the same time maintaining high clustering performance. The proposed approach is extended to distributed SC regimes.

1.2 Learning with blind (unsupervised) ensembles

Ensemble learning refers to the task of designing a meta-learner, by combining the results provided by multiple different learners or annotators[1]; see Fig. 3.1. This meta-learner should generally be able to outperform the individual learners. In particular, ensemble classification refers to fusing the results provided by different classifiers. Each classifier observes data, decides a class (out of K possible) each of these data belong to, and provides the meta-learner with those decisions. Such a setup emerges in diverse disciplines including medicine [152], biology [96], team decision making [102] and economics [130], and has recently gained attention with the advent of crowdsourcing [19, 61] as well as services such as Amazon's Mechanical Turk [75] and Clickworker, to name a few. In the crowdsourcing paradigm, multiple workers/annotators are asked to perform simple tasks and then the annotator answers are fused. Crowdsourcing has been successfully applied for mitosis detection in breast cancer images [80], MRI segmentation [10],

[1]The terms learner, annotator, and classifier will be used interchangeably.

topic modeling [118], and remote sensing [45] among others. A rrelated problem has been considered in the distributed detection or distributed estimation literature [141]. In this case, sensors are observing a phenomenon, decide which one out of K possible hypotheses is true, and transmit those decisions to a fusion center, which has to make a final decision. Additionally, a similar problem, termed the *CEO problem* or multiterminal source coding, has been considered in the information theory literature, albeit from a coding perspective [12].

When training data are available, a meta-learner can learn how to combine the results from individual classifiers, based on these ground-truth labels [33]. One such approach is boosting [42, 43], where multiple classifiers are combined according to their probability of error on the training set. In the boosting regime, each classifier is also using information from the rest. In many cases however, labeled data are not available to train the combining meta-classifier, or, the individual classifiers cannot be retrained, justifying the need for *unsupervised* (or *blind*) ensemble methods. One such paradigm is provided by crowdsourcing, where people are tasked with providing classification labels. Accordingly, in a distributed detection setup, the fusion center might not have access to the sensors, once they have been deployed. The task of blind ensemble classification is then to assess the reliability of each annotator while at the same time fusing their responses. Note that this setup is naturally more resilient to adversarial attacks than traditional machine learning approaches, as adversaries can be detected by the fusion center/meta learner. This thesis will use simple concepts from probability, optimization and detection theory to develop new algorithms that judiciously fuse annotator responses, by taking advantage of the special structure exhibited by annotator moments.

Furthermore, in many cases, there might be dependencies in the considered data. For example, the data might form a sequence. Such a setup arises in many natural language processing tasks such as part-of-speech tagging, where parts of a sentence have to be tagged as nouns, verbs, etc.; named-entity recognition, where the named entities, such as locations or people in a sentence have to be identified; and information extraction, to name a few [100]. In addition, more general dependencies can be captured by a graph, where pairwise relationships between data are encoded in the graph edges. Examples of such data include citation or brain networks to name a few. As will be shown in the subsequent chapters of this thesis, these dependencies can be accounted for in the blind ensemble classification task to enhance classification performance.

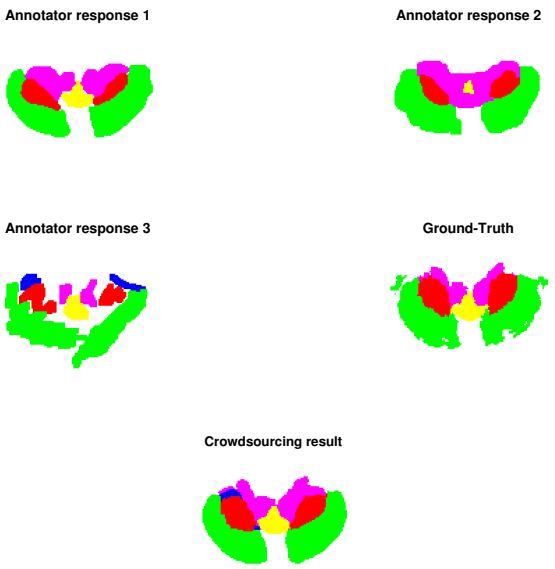


Figure 1.4: Example of crowdsourcing for cerebellum segmentation [10].

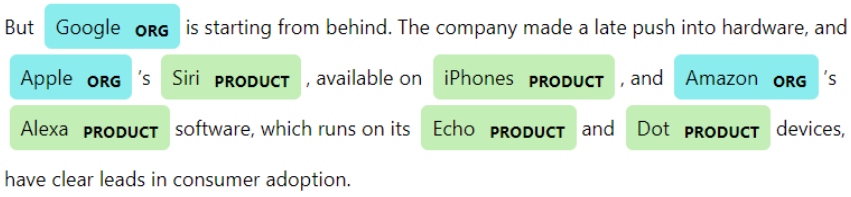


Figure 1.5: Named entity recognition example [60].

1.3 Thesis Outline

The remainder of the thesis is organized as follows.

Chapter 2 of the present thesis deals with large-scale subspace clustering. A random projections based approach, termed *Sketched SC*, is developed. The proposed approach "compresses" the data appropriately and can handle both high volumes, as well as high-dimensional data. Furthermore, a distributed version of the proposed algorithms is provided. The proposed algorithms are evaluated with a rigorous performance analysis and extended numerical tests on real datasets.

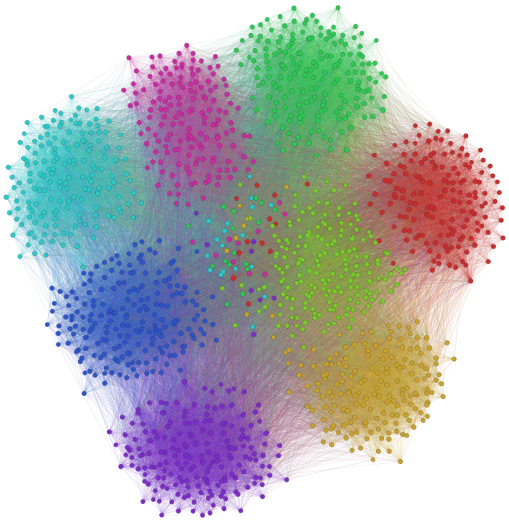


Figure 1.6: Example of graph data with 8 classes.

Chapter 3 introduces a novel approach to multiclass blind ensemble classification for independent and identically distributed (iid) data. The proposed approach is based on the PARAFAC structure of third-order annotator moments and can readily handle multiple imbalanced classes of data. A rigorous performance analysis is provided along with extended numerical simulations on synthetic and real datasets.

Chapter 4 builds upon the algorithms and results of Chapter 3 and introduces blind ensemble classification approaches for non-iid data. Two cases of dependent data are considered: sequential data, for which a moment-based algorithm and an expectation maximization (EM) algorithm are developed; and a generally dependent data case, where the dependencies are captured by a given graph. For the latter, the algorithm of 3 is combined with a novel EM-based algorithm. Numerical tests on synthetic and real data corroborate the effectiveness of the proposed algorithms.

Finally, Chapter 5 presents a concluding discussion of the proposed approaches, along with future research directions.

1.4 Notational Conventions

Unless otherwise noted, lowercase bold letters, $\boldsymbol{x}$, denote vectors, uppercase bold letters, $\mathbf{X}$, represent matrices, and calligraphic uppercase letters, $\mathcal{X}$, stand for sets. The (i, j)th entry of

matrix $\mathbf{X}$ is denoted by $[\mathbf{X}]_{ij}$; and its rank by rank$(\mathbf{X})$; $\mathbf{X}^\top$ denotes the transpose of matrix $\mathbf{X}$; $\mathbb{R}^D$ stands for the D-dimensional real Euclidean space, $\mathbb{R}_+$ for the set of positive real numbers. $\Pr$ denotes probability, or the probability mass function; $\sim$ denotes "distributed as, $\mathbb{E}[\cdot]$ stands for expectation, and $\|\cdot\|$ for the ℓ_2-norm. Underlined capital letters $\underline{X}$ denote tensors; while $[[\mathbf{A}, \mathbf{B}, \mathbf{C}]]_K$ is used to denote compactly a K-factor PARAFAC tensor [56, 126] with factor matrices $\mathbf{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_K], \mathbf{B} = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_K], \mathbf{C} = [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_K]$, that is $[[\mathbf{A}, \mathbf{B}, \mathbf{C}]]_K = \sum_{k=1}^K \boldsymbol{a}_k \circ \boldsymbol{b}_k \circ \boldsymbol{c}_k$, where $\circ$ denotes the outer product. Symbol $\mathcal{I}(A)$ denotes the indicator function of event A, i.e. $\mathcal{I}(A) = 1$ if A occurs, and is 0, otherwise.

Chapter 2

Large-scale Subspace Clustering

The immense amount of daily generated and communicated data presents unique challenges in their processing. Clustering, the grouping of data without the presence of ground-truth labels, is an important tool for drawing inferences from data. Subspace clustering (SC) is a relatively recent method that is able to successfully classify nonlinearly separable data in a multitude of settings. In spite of their high clustering accuracy, SC methods incur prohibitively high computational complexity when processing large volumes of high-dimensional data. Inspired by random sketching approaches for dimensionality reduction, the present paper introduces a randomized scheme for SC, termed Sketch-SC, tailored for large volumes of high-dimensional data. Sketch-SC accelerates the computationally heavy parts of state-of-the-art SC approaches by compressing the data matrix across both dimensions using random projections, thus enabling fast and accurate large-scale SC. Performance analysis as well as extensive numerical tests on real data corroborate the potential of Sketch-SC and its competitive performance relative to state-of-the-art scalable SC approaches.

For the remainder of this chapter $\mathbf{X} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top$ denotes the singular value decomposition (SVD) of a rank ρ, $D \times N$ matrix $\mathbf{X}$, where $\mathbf{U}_\rho$ is $D \times \rho$, $\mathbf{\Sigma}_\rho$ is $\rho \times \rho$, and $\mathbf{V}_\rho$ is $N \times \rho$. For a positive integer $r < \rho$, the SVD of $\mathbf{X}$ can be rewritten as

$$\begin{aligned} \mathbf{X} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top &= [\mathbf{U}_r \bar{\mathbf{U}}_r] \begin{bmatrix} \mathbf{\Sigma}_r & \\ & \bar{\mathbf{\Sigma}}_r \end{bmatrix} \begin{bmatrix} \mathbf{V}_r^\top \\ \bar{\mathbf{V}}_r^\top \end{bmatrix} \\ &= \mathbf{X}_r + \bar{\mathbf{X}}_r \end{aligned} \tag{2.1}$$

where $\mathbf{\Sigma}_r$ is an $r \times r$ diagonal matrix with the largest r singular values of $\mathbf{X}$ in descending order, and $\mathbf{X}_r = \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^\top$ is the best rank-r approximation of $\mathbf{X}$ in the sense that $\mathbf{X}_r$ minimizes $\|\mathbf{X} - \mathbf{X}_r\|_F$. Accordingly, $\bar{\mathbf{\Sigma}}_r$ is a $(\rho - r) \times (\rho - r)$ diagonal matrix containing the remaining singular values of $\mathbf{X}$ and $\bar{\mathbf{X}}_r = \bar{\mathbf{U}}_r \bar{\mathbf{\Sigma}}_r \bar{\mathbf{V}}_r^\top$. The D-dimensional real Euclidean space is denoted by $\mathbb{R}^D$, the set of positive real numbers by $\mathbb{R}_+$, the set of positive integers by $\mathbb{Z}_+$, the expectation operator by $\mathbb{E}[\cdot]$, and the ℓ_2-norm by $\|\cdot\|$.

2.1 Preliminaries

2.1.1 SC problem statement

Consider N vectors $\{\boldsymbol{x}_i\}_{i=1}^N$ of size $D \times 1$ drawn from a union of K affine subspaces, each denoted by $\mathcal{S}_k$, adhering to the model

$$\boldsymbol{x}_i = \mathbf{C}^{(k)} \boldsymbol{\psi}_i^{(k)} + \boldsymbol{\mu}^{(k)} + \boldsymbol{v}_i \,, \quad \forall \boldsymbol{x}_i \in \mathcal{S}_k \tag{2.2}$$

where d_k (possibly with $d_k \ll D$) is the dimensionality of $\mathcal{S}_k$; $\mathbf{C}^{(k)}$ is a $D \times d_k$ matrix whose columns form a basis of $\mathcal{S}_k$; the d_k-dimensional vector $\boldsymbol{\psi}_i^{(k)}$ is the low-dimensional representation of $\boldsymbol{x}_i$ in $\mathcal{S}_k$ with respect to (w.r.t.) $\mathbf{C}^{(k)}$; the $D \times 1$ vector $\boldsymbol{\mu}^{(k)}$ is the "centroid" or intercept of $\mathcal{S}_k$; and, $\boldsymbol{v}_i$ denotes the $D \times 1$ noise vector capturing unmodeled effects. If $\mathcal{S}_k$ is linear, then $\boldsymbol{\mu}^{(k)} = \mathbf{0}$.

Let also $\boldsymbol{p}_i$ denote the cluster assignment vector of $\boldsymbol{x}_i$, and $[\boldsymbol{p}_i]_k$ the kth entry of $\boldsymbol{p}_i$ that is constrained to satisfy $[\boldsymbol{p}_i]_k \geq 0$ and $\sum_{k=1}^K [\boldsymbol{p}_i]_k = 1$. If $\boldsymbol{p}_i \in \{0,1\}^K$, then $\boldsymbol{x}_i$ lies in only one subspace (hard clustering), while if $\boldsymbol{p}_i \in [0,1]^K$, then $\boldsymbol{x}_i$ can belong to multiple clusters (soft clustering). In the latter case, $[\boldsymbol{p}_i]_k$ can be thought of as the probability that $\boldsymbol{x}_i$ belongs to $\mathcal{S}_k$. Clearly in the case of hard clustering, (2.2) can be rewritten as

$$\boldsymbol{x}_i = \sum_{k=1}^K [\boldsymbol{p}_i]_k \left(\mathbf{C}^{(k)} \boldsymbol{\psi}_i^{(k)} + \boldsymbol{\mu}^{(k)} \right) + \boldsymbol{v}_i. \tag{2.3}$$

Given the $D \times N$ data matrix $\mathbf{X} := [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]$ and the number of subspaces K, the goal is to find the data-to-subspace assignment vectors $\{\boldsymbol{p}_i\}_{i=1}^N$, the subspace bases $\left\{\mathbf{C}^{(k)}\right\}_{k=1}^K$,

their dimensions $\{d_k\}_{k=1}^K$, the low-dimensional representations $\{\boldsymbol{\psi}_i^{(k)}\}_{i=1}^N$, as well as the centroids $\{\boldsymbol{\mu}^{(k)}\}_{k=1}^K$ [145]. SC can be formulated as follows

$$\begin{aligned} \min_{\mathbf{P},\{\mathbf{C}^{(k)}\},\{\boldsymbol{\psi}_i^{(k)}\},\mathbf{M}} \quad & \sum_{k=1}^{K}\sum_{i=1}^{N}[\boldsymbol{p}_i]_k\|\boldsymbol{x}_i-\mathbf{C}^{(k)}\boldsymbol{\psi}_i^{(k)}-\boldsymbol{\mu}^{(k)}\|_2^2 \\ \text{subject to (s.to)} \quad & \mathbf{P}^\top\mathbf{1}=\mathbf{1}; \quad [\boldsymbol{p}_i]_k\geq 0,\ \forall(i,k) \end{aligned} \tag{2.4}$$

where $\mathbf{P}:=[\boldsymbol{p}_1,\ldots,\boldsymbol{p}_N]$, $\mathbf{M}:=[\boldsymbol{\mu}^{(1)},\boldsymbol{\mu}^{(2)},\ldots,\boldsymbol{\mu}^{(k)}]$, and $\mathbf{1}$ denotes the all-ones vector of matching dimensions.

The problem in (2.4) is non-convex as all of $\mathbf{P},\{\mathbf{C}^{(k)}\}_{k=1}^K,\{d_k\}_{k=1}^K,\{\boldsymbol{\psi}_i^{(k)}\}$, and $\mathbf{M}$ are unknown. It is known that when $K=1$ and $\mathbf{C}$ is orthonormal, (2.4) boils down to PCA [68]

$$\begin{aligned} \min_{\mathbf{C},\{\boldsymbol{\psi}_i\},\boldsymbol{\mu}} \quad & \sum_{i=1}^{N}\|\boldsymbol{x}_i-\mathbf{C}\boldsymbol{\psi}_i-\boldsymbol{\mu}\|_2^2 \\ \text{s.to} \quad & \mathbf{C}^\top\mathbf{C}=\mathbf{I} \end{aligned} \tag{2.5}$$

where $\mathbf{I}$ denotes the identity matrix of appropriate dimension. Notice that for $K=1$, it holds that $[\boldsymbol{p}_i]_k=1$. Moreover, if $\mathbf{C}^{(k)}:=\mathbf{0}$, $\forall k$, looking for $\{\boldsymbol{\mu}^{(k)}\}_{k=1}^K$, $\{\boldsymbol{p}_i\}_{i=1}^N$ with $K>1$, amounts to K-means clustering

$$\begin{aligned} \min_{\mathbf{P},\mathbf{M}} \quad & \sum_{k=1}^{K}\sum_{i=1}^{N}[\boldsymbol{p}_i]_k\|\boldsymbol{x}_i-\boldsymbol{\mu}^{(k)}\|_2^2 \\ \text{s.to} \quad & \mathbf{P}^\top\mathbf{1}=\mathbf{1}\,. \end{aligned} \tag{2.6}$$

2.1.2 Prior work

Various algorithms have been developed by the machine learning [145] and data-mining community [108] to solve (2.4). Generalizing the ubiquitous K-means [87] the K-subspaces algorithm [2] builds on alternating optimization to solve (2.4). For $\boldsymbol{\Pi}$ and $\{d_k\}_{k=1}^K$ fixed, bases of the subspaces can be recovered using the SVD on the data associated with each subspace. Indeed, given $\mathbf{X}^{(k)}:=[\boldsymbol{x}_{i_1},\ldots,\boldsymbol{x}_{i_{N_k}}]$, belonging to $\mathcal{S}_k$ ($\sum_{k=1}^K N_k=N$), a basis $\mathbf{C}^{(k)}$ can be obtained from the first d_k (from the left) singular vectors of $\mathbf{X}^{(k)}-[\boldsymbol{\mu}^{(k)},\ldots,\boldsymbol{\mu}^{(k)}]$, where $\boldsymbol{\mu}^{(k)}=(1/N_k)\sum_{i\in\mathcal{S}_k}\boldsymbol{x}_i$. On the other hand, when $\{\mathbf{C}^{(k)},\boldsymbol{\mu}^{(k)}\}_{k=1}^K$ are given, the assignment matrix $\boldsymbol{\Pi}$ can be recovered in the case of hard clustering by finding the closest subspace to each

datapoint; that is, $\forall i \in \{1, 2, \ldots, N\}$, $\forall k \in \{1, \ldots, K\}$, we obtain

$$[\boldsymbol{p}_i]_k = \begin{cases} 1, & \text{if } k = \underset{k' \in \{1,\ldots,K\}}{\arg\min} \left\| \tilde{\boldsymbol{x}}_i^{(k')} - \mathbf{C}^{(k')} \mathbf{C}^{(k')\top} \tilde{\boldsymbol{x}}_i^{(k')} \right\|_2^2 \\ 0, & \text{otherwise} \end{cases} \tag{2.7}$$

where $\tilde{\boldsymbol{x}}_i^{(k)} := \boldsymbol{x}_i - \boldsymbol{\mu}^{(k)}$ and $\|\tilde{\boldsymbol{x}}_i^{(k)} - \mathbf{C}^{(k)} \mathbf{C}^{(k)\top} \tilde{\boldsymbol{x}}_i^{(k)}\|_2$ is the distance of $\boldsymbol{x}_i$ from $\mathcal{S}_k$. Thus, the K-subspaces algorithm operates as follows: (i) Fix $\mathbf{P}$ and solve for the remaining unknowns; and (ii) fix $\{\mathbf{C}^{(k)}, \boldsymbol{\mu}^{(k)}\}_{k=1}^K$, and solve for $\mathbf{P}$. Since SVD is involved, SC entails high computational complexity, whenever d_k and/or N_k are massive.

A probabilistic (soft) counterpart of K-subspaces is the mixture of probabilistic PCA [131], which assumes that data are drawn from a mixture of degenerate (zero-variance) Gaussians. Building on the same assumption, the agglomerative lossy compression (ALC) minimizes the required number of bits to "encode" each cluster, up to a certain distortion level [92]. Algebraic schemes, such as generalized (G)PCA approach SC from a linear algebra point of view, but generally their performance is guaranteed only for independent and noise-less subspaces [146]. Additional interesting methods recover subspaces by finding local linear subspace approximations [161]; by thresholding the correlations between data [58]; or by identifying the subspaces one by one [116]. Recently, multilinear methods for SC of tensor data have also been advocated [137]; see also [124, 136, 160] for online clustering approaches to handle streaming data.

Arguably the most successful class of solvers for (2.4) relies on *spectral clustering* [147] to find the data-to-subspace assignments. Algorithms in this class generate first an $N \times N$ symmetric weighted adjacency matrix $\mathbf{W}$ to capture the non-directional similarity between data vectors, and then perform spectral clustering on $\mathbf{W}$. Matrix $\mathbf{W}$ implies a graph $\mathcal{G}$ whose vertices correspond to data and the weight of the edge connecting vertex i and vertex j is given by $[\mathbf{W}]_{ij}$. Spectral clustering algorithms form the graph Laplacian matrix

$$\mathbf{L} := \text{diag}(\mathbf{W}\mathbf{1}) - \mathbf{W} \tag{2.8}$$

where $\text{diag}(\mathbf{W}\mathbf{1})$ is a diagonal matrix holding $\mathbf{W}\mathbf{1}$ on its diagonal. The algebraic multiplicity of the 0 eigenvalue of $\mathbf{L}$ yields the number of connected components in $\mathcal{G}$, while the corresponding eigenvectors are indicator vectors of these connected components [147]. Afterwards, having

formed $\mathbf{L}$, the K eigenvectors $\{\mathbf{v}_k\}_{k=1}^K$ corresponding to the trailing eigenvectors of $\mathbf{L}$ are found, and K-means is performed on the rows of the $N \times K$ matrix $\mathbf{V} := [\mathbf{v}_1, \ldots, \mathbf{v}_K]$ to obtain clustering assignments [147].

Sparse subspace clustering (SSC) [37] exploits the fact that under the union of subspaces model (2.4), only a small percentage of data suffices to provide a low-dimensional affine representation of $\boldsymbol{x}_i$; that is, $\boldsymbol{x}_i = \sum_{j=1, j\neq i}^N w_{ij}\boldsymbol{x}_j$, $\forall i \in \{1, 2, \ldots, N\}$. Specifically, SSC solves the following sparsity-promoting optimization problem

$$\begin{aligned} \min_{\mathbf{Z}} \quad & \|\mathbf{Z}\|_1 + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}\|_F^2 \\ \text{s.to} \quad & \mathbf{Z}^\top \mathbf{1} = \mathbf{1}; \quad \text{diag}(\mathbf{Z}) = \mathbf{0} \end{aligned} \tag{2.9}$$

where $\mathbf{Z} := [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N]$; column $\boldsymbol{z}_i$ is sparse and contains the coefficients for the representation of $\boldsymbol{x}_i$; $\lambda > 0$ is the regularization coefficient; and $\|\mathbf{Z}\|_1 := \sum_{i,j=1}^N |[\mathbf{Z}]_{i,j}|$. The constraint $\text{diag}(\mathbf{Z}) = \mathbf{0}$ ensures that the solution of the optimization problem is not a trivial one ($\mathbf{Z} = \mathbf{I}$), while $\mathbf{Z}^\top \mathbf{1} = \mathbf{1}$ is employed to guarantee that the $\mathbf{Z}$ found is invariant to shifting the data by a constant vector [145]. Matrix $\mathbf{Z}$ is used to create the weighted adjacency matrix $[\mathbf{W}]_{ij} := |[\mathbf{Z}]_{ij}| + |[\mathbf{Z}]_{ji}|$. Finally, spectral clustering, is performed on $\mathbf{W}$ and cluster assignments are identified. Using those assignments, $\mathbf{M}$ is found by taking sample means per cluster, and $\{\mathbf{C}^{(k)}\}_{k=1}^K$, $\{\boldsymbol{\psi}_i^{(k)}\}_{i=1}^N$ are obtained by applying SVD on $\mathbf{X}^{(k)} - [\boldsymbol{\mu}^{(k)}, \ldots, \boldsymbol{\mu}^{(k)}]$.

The low-rank representation (LRR) approach to SC is similar to SSC, but replaces the ℓ_1-norm in (2.9) with the nuclear one: $\|\mathbf{Z}\|_* := \sum_{i=1}^\rho \sigma_i(\mathbf{Z})$, where ρ stands for the rank and $\sigma_i(\mathbf{Z})$ for the ith singular value of $\mathbf{Z}$. Specifically, LRR solves the following optimization problem [85]

$$\min_{\mathbf{Z}} \quad \|\mathbf{Z}\|_* + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}\|_{2,1} \tag{2.10}$$

where $\|\mathbf{X}\|_{2,1} := \sum_{j=1}^N \|\boldsymbol{x}_j\|_2$, and $\boldsymbol{x}_j$ denotes the j-th column of $\mathbf{X}$.

Another popular algorithm is termed least-squares regression (LSR) [90]. It solves an optimization problem similar to (2.10), but replaces the ℓ_1/nuclear norm with the Frobenius one. Specifically, LSR solves

$$\min_{\mathbf{Z}} \quad \frac{1}{2}\|\mathbf{Z}\|_F^2 + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{X}\mathbf{Z}\|_F^2 \tag{2.11}$$

which admits the following closed-form solution $\mathbf{Z}^* = \lambda \left(\lambda \mathbf{X}^\top \mathbf{X} + \mathbf{I}\right)^{-1} \mathbf{X}^\top \mathbf{X}$. Combining SSC with LSR, the elastic net SC (EnSC) approaches employ a convex combination of ℓ_1-

and Frobenius-norm regularizers [38, 103]. The high clustering accuracy achieved by these self-dictionary methods comes at the price of high complexity. Solving (2.9), (2.10) or (2.11) scales cubically with the number of data N, on top of performing spectral clustering across K clusters, which renders these methods computationally prohibitive for large-scale SC. When data are high-dimensional ($D \gg$), methods based on (statistical) leverage scores, random projections [18, 59, 110, 149], preconditioning and sampling [112], or our recent sketching and validation (SkeVa) [135] approach can be employed to reduce complexity to an affordable level. Random projection based methods left multiply the data matrix $\mathbf{X}$, with a $d \times D$ data-agnostic random matrix, thereby reducing the dimensionality of the data vectors from D to d. This type of dimensionality reduction has been shown to reduce computational costs while not incurring significant clustering performance degradation when $d = \mathcal{O}(\sum_{k=1}^{K} d_k)$ [59]. When the number of data vectors is large ($N \gg$), the scalable SSC/LRR/LSR approach [109] involves drawing randomly $n < N$ data, performing SSC/LRR/LSR on them, and expressing the rest of the data according to the clusters identified by that random draw of samples. While this approach clearly reduces complexity, performance can potentially suffer as the random sample may not be representative of the entire dataset, especially when $n \ll N$ and clusters are unequally populated. Other approaches focus on greedy methods, such as orthogonal matching pursuit (OMP), for solving (2.9) [36, 156]. More recently, an active set method, termed Oracle guided Elastic Net (ORGEN) [157], can be used to reduce the complexity of SSC and EnSC tasks, by solving only for the entries of $\mathbf{Z}$ that correspond to data vectors that are highly correlated.

The present thesis introduces a novel approach based on random projections that creates a compact yet expressive dictionary that can be employed by SSC/LRR/LSR to reduce the number of optimization variables to $\mathcal{O}(nN)$ for $n < N$, thus yielding low computational complexity. In addition, the proposed approach can be combined with random projection methods to reduce data dimensionality, which further scales down computational costs.

2.2 Sketched Subspace Clustering

Consider the following unifying optimization problem

$$\min_{\mathbf{A} \in \mathcal{C}} h(\mathbf{A}) + \lambda L(\mathbf{X} - \mathbf{B}\mathbf{A}) \tag{2.12}$$

Algorithm 2.1 Linear sketched data model for Sketch-SC

Input: $D \times N$ data matrix $\mathbf{X}$; Number of columns of $\mathbf{R}$ n; regularization parameter λ;
Output: Model matrix $\mathbf{A}$;
1: Generate $N \times n$ JLT matrix $\mathbf{R}$.
2: Form $D \times n$ dictionary $\mathbf{B} = \mathbf{X}\mathbf{R}$.
3: Solve (2.12) for the cost in (2.14), (2.15), (2.16) to obtain $\mathbf{A}$.

where $\mathbf{B}$ is an appropriate $D \times n$ known basis matrix (dictionary), $h(\mathbf{A})$ is a regularization function of the $n \times N$ matrix $\mathbf{A}$, $L(\cdot)$ is an appropriate loss function, and $\mathcal{C}$ is a constraint set for $\mathbf{A}$. Eq. (2.12) will henceforth be referred to as *Sketch-SC objective*. As mentioned in Sec.2.1.2, the ability of $\mathbf{A}$, obtained from (2.12) to distinguish data for clustering depends on the choice of $h(\cdot)$, and on $\mathbf{B}$. For SSC, LSR and LRR, $\mathbf{B} = \mathbf{X}$, $n = N$ and $h(\cdot)$ is $\|\cdot\|_1$, $\frac{1}{2}\|\cdot\|_F^2$, $\|\cdot\|_*$, and $L(\cdot)$ is $\frac{1}{2}\|\cdot\|_F^2$, $\frac{1}{2}\|\cdot\|_F^2$ and $\frac{1}{2}\|\cdot\|_F^2$ or $\frac{1}{2}\|\cdot\|_{2,1}$ respectively. The constraint set for SSC is $\mathcal{C} = \{\mathbf{A} \in \mathbb{R}^{N\times N} : \mathbf{A}^\top \mathbf{1} = \mathbf{1}; \text{diag}(\mathbf{A}) = \mathbf{0}\}$, while for LSR and LRR, we have $\mathcal{C} = \mathbb{R}^{N\times N}$.

2.2.1 High volume of data

As the aim of the present thesis is to introduce scalable methods for subspace clustering, the dictionaries considered from now on will have $n \ll N$, bringing the number of variables to $\mathcal{O}(nN)$. In particular, the dictionaries employed will have the form, $\mathbf{B} := \mathbf{X}\mathbf{R}$, where $\mathbf{R}$ is a $N \times n$ sketching matrix. The role of $\mathbf{R}$ is to "compress" $\mathbf{X}$, while retaining as much information from it as possible. To this end, the celebrated Johnson-Lindenstrauss lemma [67] will be invoked.

Lemma 2.1. *[67]* Given $\varepsilon > 0$, for any subset $\mathcal{V} \subset \mathbb{R}^N$ containing d vectors of size $N \times 1$, there exists a map $q : \mathbb{R}^N \to \mathbb{R}^n$ such that for $n \geq n_0 = \mathcal{O}(\varepsilon^{-2} \log d)$, it holds for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{V}$

$$(1-\varepsilon)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2 \leq \|q(\boldsymbol{x}) - q(\boldsymbol{y})\|_2^2 \leq (1+\varepsilon)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2. \tag{2.13}$$

In particular, random matrices known as Johnson-Lindenstrauss transforms will be employed since they exhibit useful properties.

Definition 2.1. *[151, Def. 2.3], [18]* An $N \times n$ random matrix $\mathbf{R}$ forms a Johnson-Lindenstrauss transform (JLT(ε, δ, d)) with parameters ε, δ, d if there exists a function f, such that for any

$\varepsilon > 0, \delta < 1, d \in \mathbb{Z}_+$ and d-element subset $\mathcal{V} \subset \mathbb{R}^N$, with $n = \Omega(\frac{\log d}{\varepsilon^2} f(\delta))$, it holds that

$$\Pr\left\{(1-\varepsilon)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{x}^\top \mathbf{R}\|_2^2 \leq (1+\varepsilon)\|\boldsymbol{x}\|_2^2\right\} \geq 1-\delta$$

for any $1 \times N$ vector $\boldsymbol{x}^\top \in \mathcal{V}$.

One example of a random JLT matrix is a matrix with independent and identically distributed (i.i.d.) entries drawn from a normal $\mathcal{N}(0,1)$ distribution scaled by a factor $1/\sqrt{n}$ [151]. Rescaled random sign matrices, that is matrices with i.i.d. ± 1 entries multiplied by $1/\sqrt{n}$ are also JLTs [1, 18], and matrix products involving these matrices can be computed fast [82]. Another class of JLTs that allows for efficient matrix multiplication includes the so-called Fast (F)JLTs. This class of FJLTs samples randomly and rescales rows of a fixed orthonormal matrix, such as the discrete Fourier transform (DFT) matrix, or, the Hadamard matrix [3,4]; see also [27,113,151] where sparse JLT matrices have been advocated.

The following proposition proved in the appendix justifies the use of JLTs for constructing our dictionary $\mathbf{B}$ in (2.12).

Proposition 2.1. Let $\mathbf{X}$ be a $D \times N$ matrix such that rank$(\mathbf{X}) = \rho$, and define the $D \times n$ matrix $\mathbf{B} := \mathbf{X}\mathbf{R}$, where $\mathbf{R}$ is a JLT(ε, δ, D) of size $N \times n$. If $n = \mathcal{O}(\rho \frac{\log(\rho/\varepsilon)}{\varepsilon^2} f(\delta))$ then w.p. at least $1-\delta$, it holds that

$$\text{range}(\mathbf{X}) = \text{range}(\mathbf{B}).$$

This proposition asserts that with a proper choice of the sketching matrix $\mathbf{R}$, the dictionary $\mathbf{B}$ is as expressive as $\mathbf{X}$ for solving (2.12), as it preserves the column space of $\mathbf{X}$ with high probability. The next proposition provides a similar bound on the reduced dimension n, when $n < \text{rank}(\mathbf{X}) := \rho$.

Proposition 2.2. Let $\mathbf{X}$ be a $D \times N$ matrix such that rank$(\mathbf{X}) = \rho$, and define the $D \times n$ matrix $\mathbf{B} := \mathbf{X}\mathbf{R}$, where $\mathbf{R}$ is a JLT(ε, δ, D) of size $N \times n$. If $n = \mathcal{O}(r \frac{\log(r/\varepsilon)}{\varepsilon^2} f(\delta))$, then w.p. at least $1-2\delta$ it holds that

$$\|\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \mathbf{U}_r \mathbf{\Sigma}_r\|_F \leq (\varepsilon \frac{\sqrt{1+\varepsilon}}{\sqrt{1-\varepsilon}} + 1 + \varepsilon)\|\bar{\mathbf{X}}_r\|_F.$$

Prop. 2.2 suggests that $\mathbf{B}$ approximately inherits the range of $\mathbf{X}_r$.

Upon constructing a $\mathbf{B}$ adhering to Prop. 2.1 or Prop. 2.2, (2.12) can be solved for different choices of h. When $h(\mathbf{A}) = \frac{1}{2}\|\mathbf{A}\|_F^2$, the optimization task (termed henceforth *Sketch-LSR*)

$$\min_{\mathbf{A}} \ \frac{1}{2}\|\mathbf{A}\|_F^2 + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{B}\mathbf{A}\|_F^2 \tag{2.14}$$

is solved by $\mathbf{A}^* = \lambda \left(\lambda \mathbf{B}^\top \mathbf{B} + \mathbf{I}\right)^{-1} \mathbf{B}^\top \mathbf{X}$, incurring complexity $\mathcal{O}(n^3 + n^2 D + nDN)$. Accordingly, our *Sketch-SSC* corresponds to $h(\mathbf{A}) = \|\mathbf{A}\|_1 = \sum_{ij} |[\mathbf{A}]_{ij}|$ and relies on the objective

$$\min_{\mathbf{A}} \|\mathbf{A}\|_1 + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{B}\mathbf{A}\|_F^2 \tag{2.15}$$

that can be solved efficiently to obtain $\mathbf{A}$ using the alternating direction method of multipliers (ADMM) [49], as per [37], or any other efficient LASSO solver. The ADMM solver for (2.15) incurs complexity $\mathcal{O}(n^3 + n^2 D + nDN + n^2 NI)$, where I is the required number of iterations until convergence, and the constraint $\text{diag}(\mathbf{A}) = \mathbf{0}$ is no longer required as $\mathbf{I}$ is not a trivial solution of (2.15). Proceeding along similar lines, our *Sketch-LRR* objective, for $h(\mathbf{A}) = \|\mathbf{A}\|_*$ aims at

$$\min_{\mathbf{A}} \ \|\mathbf{A}\|_* + \frac{\lambda}{2}\|\mathbf{X} - \mathbf{B}\mathbf{A}\|_F^2 \tag{2.16}$$

that can be solved using the augmented Lagrange multiplier (ALM) method of [85], which incurs complexity $\mathcal{O}(n^3 + n^2 D + nDN + (nDN + nN^2 + n^2 N)I)$, where I is the number of iterations until convergence. In addition, (2.16) can be solved using the $\ell_{2,1}$ norm instead of the Frobenius norm for the fitting term $\mathbf{X} - \mathbf{B}\mathbf{A}$. The entire process to obtain the data model $\mathbf{A}$ is outlined in Alg. 2.1. Detailed algorithms for solving (2.15) and (2.16) are described in Appendix D.

Remark 2.1. An optimal data-driven choice of $\mathbf{R}$ would be interesting only if finding it incurs manageable complexity - a topic which goes beyond the scope of this submission and constitutes a worthy future research direction.

Remark 2.2. Upon computing $\mathbf{B}$, (2.14) and (2.15) can be readily parallelized across columns of $\mathbf{X}$. In the nuclear norm case of (2.16) one can employ the following identity [124, 132]

$$\|\mathbf{A}\|_* = \min_{\mathbf{Z} = \mathbf{P}\mathbf{Q}^\top} \frac{1}{2}(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2) \tag{2.17}$$

where $\mathbf{A}$ is some $n \times N$ matrix of rank ρ and $\mathbf{P}$ and $\mathbf{Q}$ are $n \times \rho$ and $N \times \rho$ matrices respectively. This is especially useful when multiple computing nodes are available, or the data is scattered

across multiple devices. Without (2.17), distributed solvers of (2.16) are challenged because as columns of $\mathbf{A}$ are added the SVD needed to find the nuclear norm has to be recomputed, which is not the case with (2.17).

Remark 2.3. Existing general guidelines for choosing the regularization parameter λ for SSC and LRR [37, 85] rely on cross-validation and apply also to the proposed Algs. 2.1 and 2.2 here.

2.2.2 High-dimensional data

The complexity of all the aforementioned algorithms depends on the data dimensionality D. As such, datasets containing high-dimensional vectors will certainly increase the computational complexity. As mentioned in Sec. 2.1.2, dimensionality reduction techniques can be employed to reduce the computational burden of SC approaches. Using PCA for instance, a $d < D$-dimensional subspace that describes most of the data variance can be found. This, however, can be prohibitively expensive for large-scale datasets where $N \gg$. For such cases, our idea is to combine the method described in the previous section with randomized dimensionality reduction techniques [59]. Let $\check{\mathbf{R}}$ be a $d \times D$ JLT matrix, where $d \ll D$ is the target dimensionality, and consider the $d \times N$ matrix $\check{\mathbf{X}} := \check{\mathbf{R}}\mathbf{X}$, which is a reduced dimensionality version of the original data $\mathbf{X}$. The Sketch-SC objective then becomes

$$\min_{\mathbf{A}} h(\mathbf{A}) + \lambda L(\check{\mathbf{X}} - \check{\mathbf{B}}\mathbf{A}) \tag{2.18}$$

where $\check{\mathbf{B}} := \check{\mathbf{X}}\mathbf{R}$ is a $d \times n$ dictionary of reduced dimension with $\mathbf{R}$ being an $N \times n$ JLT matrix as in (2.12). Upon forming $\check{\mathbf{X}}$ and $\check{\mathbf{B}}$, (2.18) can be solved for different choices of h as in Sec. 2.2.1. The steps of our algorithm for high-dimensional data are summarized in Alg. 2.2.

Remark 2.4. While carrying out the products $\mathbf{XR}$, $\check{\mathbf{R}}\mathbf{X}$ or $\check{\mathbf{X}}\mathbf{R}$ can be computationally expensive in cases, they can be accelerated using modern numerical linear algebra tools, such as the Mailman algorithm [82] or by employing the Welsh-Hadamard transform [77, 112].

2.2.3 Obtaining cluster assignments

After obtaining the $N \times N$ matrix $\mathbf{Z}$ in (2.9), (2.10) or (2.11), a typical post-processing step for SSC, LSR, and LRR, is to perform spectral clustering, using $\mathbf{W} := |\mathbf{Z}| + |\mathbf{Z}^\top|$ as the adjacency

Algorithm 2.2 Linear sketched data model for Sketch-SC and $D \gg$

Input: $D \times N$ data matrix $\mathbf{X}$; Lower dimension d; Number of columns of $\mathbf{R}$ n; regularization parameter λ;

Output: Model matrix $\mathbf{A}$;

1: Generate $d \times D$ JLT matrix $\check{\mathbf{R}}$.
2: Generate $N \times n$ JLT matrix $\mathbf{R}$.
3: Form $d \times N$ matrix $\check{\mathbf{X}} = \check{\mathbf{R}}\mathbf{X}$.
4: Create $d \times n$ dictionary $\check{\mathbf{B}} = \check{\mathbf{X}}\mathbf{R}$.
5: Solve (2.18) to obtain sketched data model $\mathbf{A}$.

matrix. This step however, is not possible for the matrix $\mathbf{A}$ obtained from (2.14), (2.15) or (2.16), because it has size $n \times N$, with $n < N$.

While $\mathbf{A}$ cannot be directly used for spectral clustering, a k-nearest neighbor graph [57] can be constructed from the columns of $\mathbf{A}$. Let $\boldsymbol{a}_i$ denote the i-th column of $\mathbf{A}$, and $\mathcal{K}_i$ the set of the k columns of $\mathbf{A}$ that are closest to $\boldsymbol{a}_i$, in the Euclidean distance sense. The $N \times N$ adjacency matrix $\mathbf{W}$ can then be constructed with entries

$$[\mathbf{W}]_{ij} = \begin{cases} 1, & \text{if } \boldsymbol{a}_j \in \mathcal{K}_i \text{ or } \boldsymbol{a}_i \in \mathcal{K}_j \\ 0, & \text{otherwise.} \end{cases} \tag{2.19}$$

In addition, non-binary edge weights can be assigned as

$$[\mathbf{W}]_{ij} = \begin{cases} w_{ij}, & \text{if } \boldsymbol{a}_j \in \mathcal{K}_i \text{ or } \boldsymbol{a}_i \in \mathcal{K}_j \\ 0, & \text{otherwise.} \end{cases} \tag{2.20}$$

where w_{ij} is some scalar that depends on $\boldsymbol{a}_i$ and $\boldsymbol{a}_j$. For instance, if heat kernel weights are used, then $w_{ij} = \exp(-\|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2^2/\sigma^2)$, for some $\sigma > 0$. The resultant mutual k-nearest neighbor matrix $\mathbf{W}$ can then be employed for spectral clustering. Note that the $N \times N$ matrix $\mathbf{W}$ emerging from (2.19) or (2.20) will be sparse with $\mathcal{O}(N)$ nonzero entries, which can accelerate the eigendecomposition schemes employed for spectral clustering [69, 81]. The overall scheme is tabulated in Alg. 2.3.

Remark 2.5. When N and n are large, computation of the k nearest neighbors can be computationally taxing. Many efficient algorithms are available to accelerate the construction of

Algorithm 2.3 Obtaining clustering assignments from $\mathbf{A}$

Input: $n \times N$ matrix $\mathbf{A}$; Number of nearest neighbors k; Number of clusters K
Output: Clustering assignments
1: Find k-nearest neighbors for each column of $\mathbf{A}$.
2: Create matrix $\mathbf{W}$ using (2.19) or (2.20).
3: Apply spectral clustering on $\mathbf{W}$.

the k nearest neighbor graph [5, 107]. In addition, approximate nearest neighbor (ANN) methods [51, 63, 127] can be employed to speed up the post-processing step even further. Finally, this post-processing step can be employed for regular SSC, LSR, and LRR.

2.3 Distributed sketched subspace clustering

In many cases it might be desirable to distribute the computational load of the subspace clustering task to multiple computing nodes. Methods such as SSC and LSR can be easily parallelized, if $\mathbf{X}$ is available to all computing nodes. However, if the data is scattered across multiple nodes, this approach could incur prohibitive communication and storage costs. Here, we show how the Sketched SC approach can be used to provide communication efficient distributed SC.

Note that upon obtaining $\mathbf{B}$, (2.12) can be readily parallelized across columns of $\mathbf{X}$. To account for misses, (2.12) is rewritten as

$$\min_{\mathbf{A}\in\mathcal{C}} \; h(\mathbf{A}) + \frac{\lambda}{2}\|\mathcal{P}_{\Omega}(\mathbf{X} - \mathbf{B}\mathbf{A})\|_F^2 \tag{2.21}$$

where $\Omega \subseteq (1, 2, \ldots, D) \times (1, 2, \ldots, N)$ is the set of indices of the observed (non-missing) entries of $\mathbf{X}$, and $\mathcal{P}_{\Omega}$ is a projection operator such that

$$[\mathcal{P}_{\Omega}(\mathbf{X})]_{ij} := \begin{cases} [\mathbf{X}]_{ij} & \text{if } (i,j) \in \Omega \\ 0 & \text{otherwise.} \end{cases} \tag{2.22}$$

Suppose now that data are scattered across L computing nodes. Each node ℓ has a different subset of the data $\mathbf{X}_\ell \subset \mathbf{X}$ of size N_ℓ ($\sum_{\ell=1}^{L} N_\ell = N$). Suppose also without loss of generality

that $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_L]$, in which case

$$\mathbf{B} = \mathbf{X}\mathbf{R} = \sum_{\ell=1}^{L} \mathbf{X}_\ell \mathbf{R}_\ell = \sum_{\ell=1}^{L} \mathbf{B}_\ell \tag{2.23}$$

where $\mathbf{R}_\ell$ is a $N_\ell \times n$ subset of the rows of $\mathbf{R}$, and $\mathbf{B}_\ell := \mathbf{X}_\ell \mathbf{R}_\ell$ is the $D \times n$ subdictionary corresponding to the ℓ-th computing unit. In order to facilitate a distributed algorithm, each computing node ℓ must find its own subdictionary $\mathbf{B}_\ell$ and communicate it to the remaining nodes, thus incurring communication complexity $\mathcal{O}(LDn)$. For certain classes of JLT, such as those with i.i.d. Gaussian or Rademacher entries, each computing node can generate its own $N_\ell \times n$ matrix $\mathbf{R}_\ell$, without extra communication overhead. In other cases, a predetermined seed can be used to generate $\mathbf{R}_\ell$. Upon constructing $\mathbf{B}$ as in (2.23), each node ℓ can solve the following optimization problem

$$\min_{\mathbf{A}_\ell \in \mathcal{C}_\ell} \; h(\mathbf{A}_\ell) + \frac{\lambda}{2} \|\mathcal{P}_{\Omega_\ell}(\mathbf{X}_\ell - \sum_{\ell'=1}^{L} \mathbf{X}_{\ell'} \mathbf{R}_{\ell'} \mathbf{A}_\ell)\|_F^2, \quad \ell = 1 \ldots, L \tag{2.24}$$

where the $n \times N_\ell$ matrix $\mathbf{A}_\ell$ is the subset of columns of $\mathbf{A}$ corresponding to the ℓ-th computing node, $\mathcal{C}_\ell$ is the constraint set for $\mathbf{A}_\ell$, and $\Omega_\ell \subseteq (1, 2, \ldots, D) \times (1, 2, \ldots, N_\ell)$ is the set of indices of the observed entries of $\mathbf{X}_\ell$. The number of variables in (2.24) is $\mathcal{O}(nN_\ell)$, which dramatically reduces the computational burden per node; and, if h is separable across the columns of $\mathbf{A}$, then solving (2.24) for $\ell = 1, \ldots, L$ is equivalent to solving the full problem (2.21). This holds for $h = \|\cdot\|_F^2$ and $h = \|\cdot\|_1$, whereas in the nuclear norm case one can employ the following identity [79, 124, 132]

$$\|\mathbf{Z}\|_* = \min_{\mathbf{Z}=\mathbf{P}\mathbf{Q}^\top} \frac{1}{2}(\|\mathbf{P}\|_F^2 + \|\mathbf{Q}\|_F^2) \tag{2.25}$$

where $\mathbf{Z}$ is some $M \times I$ matrix of rank ρ and $\mathbf{P}$ and $\mathbf{Q}$ are $M \times \rho$ and $I \times \rho$ matrices respectively.

After finding $\{\mathbf{A}_\ell\}_{\ell=1}^L$, columns of $\mathbf{A} := [\mathbf{A}_1, \ldots, \mathbf{A}_L]$ can be clustered by collecting $\mathbf{A}$ at a fusion center and performing spectral clustering, or, using distributed schemes, such as distributed K-means [8, 39] or distributed spectral clustering [25]. The entire distributed SC process is summarized in Alg.2.4.

Algorithm 2.4 Distributed Sketched Subspace Clustering(SC)

Input: Data matrix per computing node $\{\mathbf{X}_\ell\}_{\ell=1}^L$; Number of columns of random matrix n; regularization parameter λ;

Output: Clustered data;

1: **for** computing node ℓ **do**
2: Generate $N_\ell \times n$ JLT matrix $\mathbf{R}_\ell$.
3: Create $D \times n$ subdictionary $\mathbf{D}_\ell = \mathbf{X}_\ell \mathbf{R}_\ell$.
4: Transmit $\mathbf{D}_\ell$ to other nodes. Receive $\{\mathbf{D}_{\ell'}\}_{\ell' \neq \ell}$.
5: Form $\mathbf{D} = \sum_{\ell'=1}^L \mathbf{D}_{\ell'}$
6: Solve (2.24) and obtain $\mathbf{A}_\ell$.
7: **end for**
8: Perform spectral clustering on columns of $\mathbf{A} = [\mathbf{A}_1, \ldots, \mathbf{A}_L]$.

2.4 Performance Analysis

In this section, performance of the proposed method will be quantified analytically. Albeit not the tightest, the bounds to be derived will provide nice intuition on why the proposed methods work. The following theorem bounds the representation error of Sketch-LSR in the noise less case.

Theorem 2.1. Consider noise-free and normalized data vectors obeying (2.3) with $\boldsymbol{v}_i \equiv \mathbf{0}$, to form columns of a $D \times N$ data matrix $\mathbf{X}$, with unit ℓ_2 norm per column, and rank$(\mathbf{X}) = \rho$. Let also $\mathbf{R}$ denote a JLT(ε, δ, D) of size $N \times n$. Let $\boldsymbol{g}^*(\boldsymbol{x}) := \mathbf{X}\boldsymbol{z}^* = \boldsymbol{x}$ denote the representation of $\boldsymbol{x}$ provided by LSR, and $\hat{\boldsymbol{g}}(\boldsymbol{x}) := \mathbf{X}\mathbf{R}\hat{\boldsymbol{a}}$ the representation given by Sketch-LSR. If $n = \mathcal{O}(r \frac{\log(r/\varepsilon)}{\varepsilon^2} f(\delta))$, then the following bound holds w.p. at least $1 - 2\delta$

$$\|\boldsymbol{g}^*(\boldsymbol{x}) - \hat{\boldsymbol{g}}(\boldsymbol{x})\|_2 \leq \lambda \, (1 + \sqrt{\frac{1+\varepsilon}{1-\varepsilon}} \, \sqrt{\rho - r} \, \sigma_{r+1}^2) + \frac{1}{\sqrt{1+\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denotes the $(r+1)$st singular value of $\mathbf{X}$.

Theorem 2.1 implies that the larger n is, the smaller the upper bound becomes as a smaller singular value of $\mathbf{X}$ is selected. This also suggests that datasets exhibiting lower rank can be compressed more (with smaller n), while retaining representation accuracy. The following corollaries extend the result of Thm. 2.1 to the Sketch-SSC and Sketch-LRR cases.

Corollary 2.1. Consider the setting of Thm. 2.1, and let $\hat{\boldsymbol{g}}(\boldsymbol{x}) := \mathbf{X}\mathbf{R}\hat{\boldsymbol{a}}$ be the representation

of a datum given by Sketch-SSC. The following bound holds w.p. at least $1-2\delta$

$$\|\boldsymbol{g}^*(\boldsymbol{x}) - \hat{\boldsymbol{g}}(\boldsymbol{x})\|_2 \leq \lambda\,(1+\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\,\sqrt{\rho-r}\,\sigma_{r+1}^2) + \sqrt{\frac{n}{1-\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denotes the $(r+1)$st singular value of $\mathbf{X}$.

This corollary is a direct consequence of the fact that for any $n \times 1$ vector $\boldsymbol{x}$, it holds that $\|\boldsymbol{x}\|_1 \leq \sqrt{n}\|\boldsymbol{x}\|_2$. Accordingly, the following corollary for Sketch-LRR holds because for any rank n matrix $\mathbf{X}$ we have $\|\mathbf{X}\|_* \leq \sqrt{n}\|\mathbf{X}\|_F$.

Corollary 2.2. Consider the setting of Thm. 2.1, and let $\boldsymbol{g}^*(\mathbf{X}) := \mathbf{XZ}$ and $\hat{\boldsymbol{g}}(\mathbf{X}) := \mathbf{XR}\hat{\mathbf{A}}$ be the representations of all the data given by LRR and Sketch-LRR respectively. The following bound holds w.p. at least $1-2\delta$

$$\|\boldsymbol{g}^*(\mathbf{X}) - \hat{\boldsymbol{g}}(\mathbf{X})\|_F \leq \lambda\,(\sqrt{N}+\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\,\sqrt{\rho-r}\,\sigma_{r+1}^2) + \sqrt{\frac{n}{1-\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denotes the $(r+1)$st singular value of $\mathbf{X}$.

For the Sketch-SSC and Sketch-LRR, tighter bounds could possibly be derived by taking into account the special structures of the ℓ_1 and nuclear norms, instead of invoking norm inequalities.

For a dataset $\mathbf{X}$ drawn from a union of subspaces model, batch methods such as SSC, LSR and LRR, should produce a matrix of representations $\mathbf{Z}$ that is block-diagonal, under certain conditions on the separability of subspaces [85, 90]. This, in turn, implies that for data $\boldsymbol{x}_i, \boldsymbol{x}_j \in \mathcal{S}_k, \boldsymbol{x}_\ell \in \mathcal{S}_{k'}$ for $k \neq k'$, it holds that

$$\|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 \leq \|\boldsymbol{z}_i - \boldsymbol{z}_\ell\|_2 \tag{2.26}$$

that is the representations of two points in the same subspace, are closer than the representations of two points that lie in different subspaces. The following proposition suggests that this property is approximately inherited by the Sketch-SC algorithms of Sec. 3.2, with high probability.

Proposition 2.3. Consider $\boldsymbol{x}_i = \mathbf{X}\boldsymbol{z}_i$ and $\boldsymbol{x}_j = \mathbf{X}\boldsymbol{z}_j$, and their representation provided by SSC, LRR or LSR $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, respectively. Let $\rho = \text{rank}(\mathbf{X})$ and $\boldsymbol{a}_i$, $\boldsymbol{a}_j$ be the representation obtained by the corresponding Sketch algorithm of Section 3.2; that is, $\boldsymbol{x}_i = \mathbf{XR}\boldsymbol{a}_i$, where the $N \times n$

matrix $\mathbf{R}$ is a JLT(ε, δ, D). If $n = \mathcal{O}(\rho \frac{\log(\rho/\varepsilon)}{\varepsilon^2} f(\delta))$, then w.p. at least $1 - \delta$ it holds that

$$\frac{1}{\sqrt{1+\varepsilon}} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 \leq \|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2.$$

Proposition 2.3 also justifies the use of the k-nearest neighbor graph as a post-processing step in Sec. 2.2.3.

As will be seen in the ensuing section, the proposed approach has comparable performance to other high-accuracy SC approaches while requiring markedly less time.

2.5 Numerical Tests

The proposed method is validated in this section using real datasets. Sketch-SC methods (termed throught this section as *Sketch-SSC, Sketch-LSR* and *Sketch-LRR*) are compared to SSC, LSR, LRR, the orthogonal matching pursuit method (OMP) for large-scale SC [156], as well as ORGEN [157]. When datasets are large ($N \gg$), the proposed methods are only compared to OMP and ORGEN. The figures of merit evaluated are following.

- Accuracy, i.e., percentage of correctly clustered data:

$$\text{Accuracy} := \frac{\text{number of data correctly clustered}}{N}.$$

- Time (in seconds) required for clustering all data. For Algs. 2.1 and 2.2 this includes the time required to generate the JLT matrices $\mathbf{R}$, the time required for computing the products $\mathbf{B} = \mathbf{X}\mathbf{R}$, and in the case of Alg. 2.2 $\check{\mathbf{X}} = \check{\mathbf{R}}\mathbf{X}$, $\check{\mathbf{B}} = \check{\mathbf{X}}\mathbf{R}$, as well as the time required for Alg. 2.3.

All experiments were performed on a machine with an Intel Core-i5 4570 CPU with 16GB of RAM. The software used to conduct all experiments is MATLAB [94]. K-means and ANN were implemented using the VLfeat package [143]. All results represent the averages of 10 independent Monte Carlo runs. The regularization scalar λ [cf. (2.9)] of SSC and Sketch-SSC is computed as per [37, Prop. 1], and it is controlled by a parameter α. ORGEN has two parameters that need to be specified, namely λ and α. LRR and Sketch-LRR employ the $\ell_{2,1}$ norm for the residual $\mathbf{X} - \mathbf{X}\mathbf{Z}$. For LRR, LSR, Sketch-LRR, Sketch-LSR, OMP and ORGEN the parameters are tuned to optimize empirically the performance of each method considered.

The real datasets tested are Hopkins 155 [140], the Extended Yale Face dataset [47], the COIL-100 database [98], and the MNIST handwritten digits dataset [78].

2.5.1 Assessing the effect of different JLTs

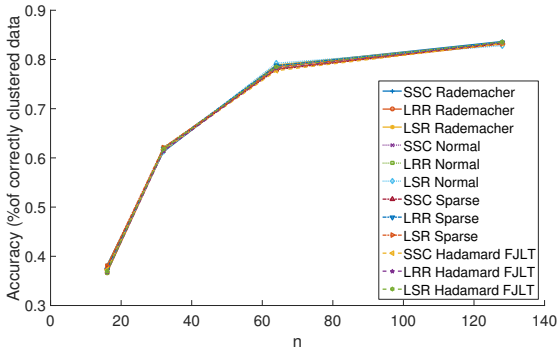
Before comparing the proposed scheme with state-of-the-art competing alternatives, the effect of different JLT matrices on the SC task was tested on two datasets: the Extended Yale Face dataset and the COIL-100 database. The different $N \times n$ JLT matrices assessed are: matrices with i.i.d. ± 1 entries rescaled by $1/\sqrt{n}$ (denoted as *Rademacher*); matrices with i.i.d. $\mathcal{N}(0,1)$ entries rescaled by $1/\sqrt{n}$ (denoted as *Normal*); Sparse embedding matrices as described in [27, 151] (denoted as *Sparse*); Fast JLTs using the Hadamard matrix as described in [3] (denoted as *Hadamard FJLT*). Fig. 2.1 depicts the performance of Alg. 2.1 for different choices of JLT for the two aforementioned datasets. All JLT matrices achieve comparable performance for the Yale Face database. However, this is not true for the COIL-100 dataset, where the Rademacher JLT seem to provide the most consistent performance.

For all tests in the rest of this section Algs. 2.1 and 2.2 use random matrices $\mathbf{R}$, and $\check{\mathbf{R}}$ that are generated having i.i.d. ± 1 entries rescaled by $1/\sqrt{n}$.

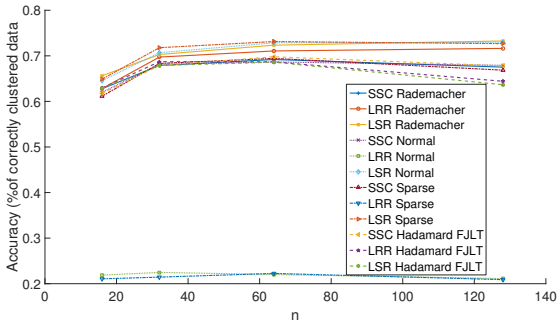
2.5.2 High volume of data

In this section the performance of Sketch-SC (Alg. 2.1) is assessed on all datasets. Hopkins 155 is a popular benchmark dataset for subspace clustering and motion segmentation. It contains 155 video sequences, with N points tracked in each frame of a video sequence. Clusters ($K = 2$ or $K = 3$) represent different objects moving in the video sequence. The results for the Hopkins 155 dataset are listed in Tab. 2.1 for $K = 2$ and $K = 3$ clusters, with $n = 0.15N$ for the proposed methods. Here $\alpha = 800$ was used for SSC and $\alpha = 100$ for Sketch-SSC, $\lambda = 1$ for LRR and $\lambda = 10$ for Sketch-LRR, $\lambda = 4.6 \cdot 10^{-3}$ for LSR and Sketch-LSR. The number of nearest neighbors for Alg. 2.3 is set to $k = 5$. As the size of the dataset is small, large computational gains are not expected by using Alg. 2.1. Nevertheless, the Sketch-SC methods achieve comparable accuracy to their batch counterparts, while in most cases (except one) requiring less time.

The Extended Yale Face database contains $N = 2,414$ face images of $K = 38$ people, each of dimension $D = 2,016$. Fig. 2.2 shows the results for this dataset for varying n, where



(a) Extended Yale Face Database

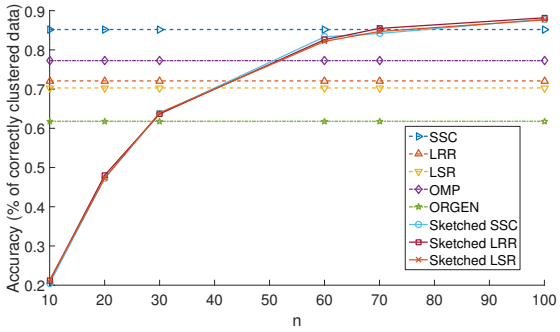


(b) COIL-100

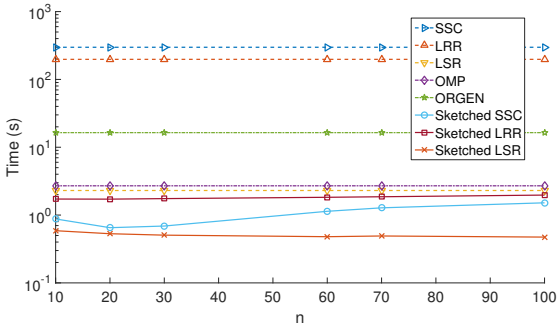
Figure 2.1: Simulated tests on real datasets Extended Yale Face Database and COIL-100, evaluating the clustering performance with different JLT matrix $\mathbf{R}$.

$\alpha = 30$ for SSC and $\alpha = 50$ for Sketch-SSC, $\lambda = 0.15$ for LRR and Sketch-LRR, $\lambda = 10^6$ for LSR and Sketch-LSR, the number of non-zeros per column of $\mathbf{Z}$ for OMP is set to 5, while $\lambda = 0.7$ and $\alpha = 200$ for ORGEN. The number of nearest neighbors for Alg. 2.3 is set to $k = 5$. The proposed algorithms exhibit comparable accuracy to their batch counterparts, in particular SSC, and also achieve higher accuracy than the state-of-the-art large-scale algorithms OMP and ORGEN, as n increases. Interestingly, with $n \approx 0.03 \cdot N$ the proposed methods achieve the accuracy of batch SSC. In addition, the proposed approach requires markedly less time than the batch methods, and less time than OMP and ORGEN as well.

The Columbia object-image dataset (COIL-100) contains $N = 7,200$ images of size 32×32 corresponding to $K = 100$ objects. Each cluster corresponds to one object, and contains images of it from 72 different angles. Fig. 2.3 shows the comparisons on this dataset for varying n, where $\alpha = 25$ for SSC and $\alpha = 500$ for Sketch-SSC, $\lambda = 0.9$ for LRR and $\lambda = 10^{-4}$ for Sketch-LRR,



(a) Clustering accuracy



(b) Clustering time

Figure 2.2: Simulated tests on real dataset Extended Yale Face Database B, with $N = 2,414$ data dimension $D = 2,016$ and $K = 38$ clusters for varying n.

$\lambda = 10^2$ for LSR and Sketch-LSR, the number of non-zeros per column of $\mathbf{Z}$ for OMP is set to 2, while $\lambda = 0.95$ and $\alpha = 3$ for ORGEN. The number of nearest neighbors for Alg. 2.3 is set to $k = 5$. The proposed approaches exhibit performance comparable to the state-of-the-art as n increases, while requiring significantly less time. Note that, OMP requires almost the same time as the proposed approaches, however its clustering performance is significantly lower.

Fig. 2.4 plots the singular values of the Extended Yale Face Database and the COIL-100 dataset. For both, the largest singular values are approximately the first 70 ones. Note that for the Extended Yale face database our proposed approaches attain their best performance for approximately $n = 70$ yielding a compression ratio of $\frac{2414}{70} \approx 34.5$, while for the COIL-100 database our proposed approaches reach their peak performance again for $n = 70$, but this time the compression ratio is $\frac{7200}{70} \approx 102.85$. This suggests that, indeed, datasets that exhibit low rank can be compressed with a lower n.

Due to their large size, tests on the following three datasets compare Alg. 2.1 only to OMP and

$K = 2$						
Algorithm	SSC	LRR	LSR	Sketch-SSC	Sketch-LRR	Sketch-LSR
Accuracy	0.9839	0.9723	0.982	**0.946**	**0.9435**	**0.9319**
Time (s)	0.6902	0.9478	0.093	**0.0795**	**0.0808**	**0.0787**
$K = 3$						
Algorithm	SSC	LRR	LSR	Sketch-SSC	Sketch-LRR	Sketch-LSR
Accuracy	0.9747	0.9253	0.9654	**0.8942**	**0.9415**	**0.9242**
Time (s)	1.566	1.295	0.1797	**0.1755**	**0.1459**	**0.1829**

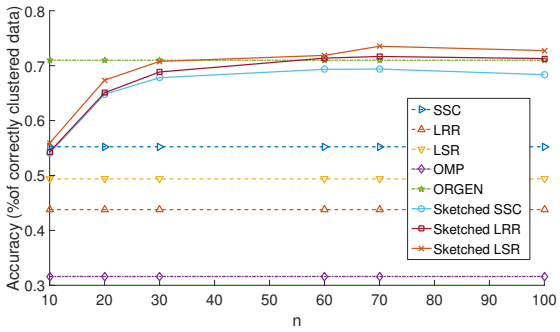
Table 2.1: Results for $K = 2$ and $K = 3$ motions for the Hopkins155 dataset

Dataset		OMP	ORGEN	Sketch-SSC	Sketch-LRR	Sketch-LSR
MNIST	Accuracy	0.47049	0.93788	**0.85825**	**0.90644**	**0.90784**
	Time (s)	502.91	801.3954	**155.1017**	**156.7709**	**99.4724**
CoverType	Accuracy	0.4870	0.4873	**0.42387**	**0.3277**	**0.4860**
	Time (s)	$1.8947 * 10^4$	$2.9893 * 10^4$	**6064.8403**	**4468.5274**	**392.916**
PokerHand	Accuracy	0.5009	-	**0.5008**	**0.1833**	**0.44225**
	Time (s)	$4.6654 * 10^4$		$\mathbf{7.8 * 10^3}$	$\mathbf{3.6 * 10^4}$	$\mathbf{2.71 * 10^3}$

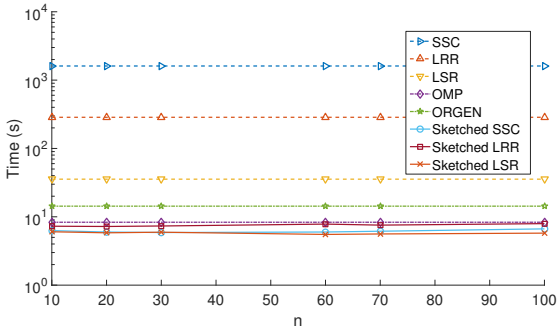
Table 2.2: Results for the Preprocessed MNIST dataset ($N = 70,000$), the CoverType dataset ($N = 581,012$) and the PokerHand dataset ($N = 1,000,000$)

ORGEN. The results for the following three datasets are listed in Tab. 2.2. The MNIST dataset contains $70,000$ images of handwritten digits, each of dimension 28×28, with $K = 10$ clusters, one per digit. Here the dataset is preprocessed with a scattering convolutional network [21] and PCA to bring each image dimension down to $D = 500$, as per [156, 157]. Here $n = 200$, $\alpha = 12,000$ for Sketch-SSC, $\lambda = 1$ for Sketch-LRR, $\lambda = 10^{-1}$ for Sketch-LSR, the number of non-zeros per column of $\mathbf{Z}$ for OMP is set to 10, while $\lambda = 0.95$ and $\alpha = 120$ for ORGEN. The number of nearest neighbors for Alg. 2.3 is set to $k = 3$, and the set of nearest neighbors for each datum is found using the ANN implementation of the VLfeat package. In this scenario ORGEN showcases the best clustering performance, however Sketch-LRR and Sketch-LSR exhibit comparable accuracy, while requiring markedly less time.

The CoverType dataset consists of $N = 581,012$ data belonging to $K = 7$ clusters. Each cluster corresponds to a different forest cover type. Data are vectors of dimension $D = 54$ that contain cartographic variables, such as soil type, elevation, hillshade etc. Here $n = 150$, $\alpha = 1$ for Sketch-SSC, $\lambda = 10^{-8}$ for Sketch-LRR, $\lambda = 10^4$ for Sketch-LSR, the number of non-zeros per column of $\mathbf{Z}$ for OMP is set to 15, while $\lambda = 0.95$ and $\alpha = 500$ for ORGEN. The number of



(a) Clustering accuracy



(b) Clustering time

Figure 2.3: Simulated tests on real dataset COIL-100, with $N = 7,200$ data dimension $D = 1,025$ and $K = 100$ clusters for varying n.

nearest neighbors for Alg. 2.3 is set to $k = 10$, and the set of nearest neighbors for each datum is found using the ANN implementation of the VLfeat package.

The PokerHand database contains $N = 10^6$ data, belonging to $K = 10$ classes. Each datum is a 5-card hand drawn from a deck of 52 cards, with each card being described by its suit (spades, hearts, diamonds, and clubs) and rank (Ace, 2, 3, ..., Queen, King). Each class represents a valid Poker hand. Here $n = 30$, $\alpha = 10$ for Sketch-SSC, $\lambda = 1$ for Sketch-LRR, $\lambda = 10^2$ for Sketch-LSR, the number of non-zeros per column of $\mathbf{Z}$ for OMP is set to 10. The number of nearest neighbors for Alg. 2.3 is set to $k = 20$, and the set of nearest neighbors for each datum is found using the ANN implementation of the VLfeat package. Results are not reported for ORGEN as the algorithm did not converge within 24 hours. For both the CoverType and PokerHand datasets, most algorithms exhibit comparable accuracy, while Alg. 2.1 requires again less time than OMP or ORGEN.

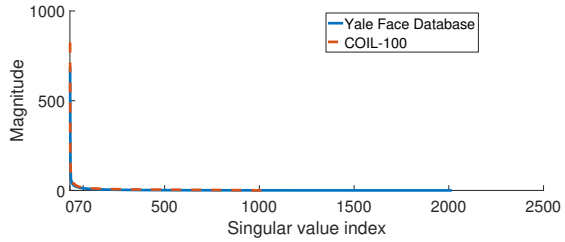


Figure 2.4: Singular value plots for the Extended Yale Face database and the COIL-100 dataset.

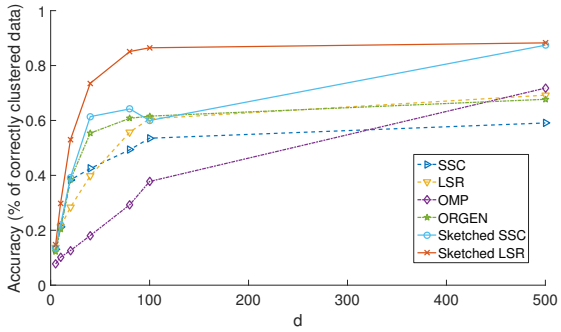
2.5.3 High-dimensional data

In this section, the performance of Sketch-SC approaches combined with randomized dimensionality reduction (Alg. 2.2) is assessed, for the Extended Yale Face database.

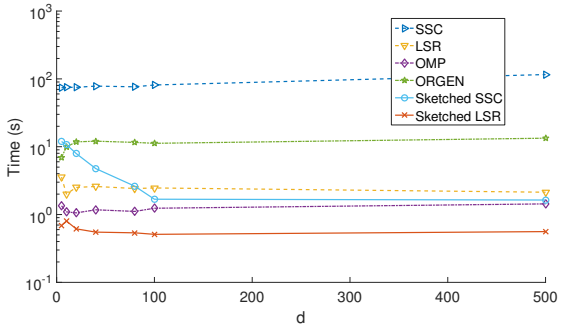
Fig. 2.5 depicts the simulation results on the Extended Yale Face database, when performing dimensionality reduction, for varying d. Here Alg. 2.2, with fixed $n = 70$ is compared to its batch counterparts, OMP and ORGEN. LRR and Sketch-LRR are not included in this simulation as the algorithm failed for small values of d. All parameters are the same as the corresponding experiment in Sec. 2.5.2. In this experiment, Sketch-LSR and Sketch-SSC outperform their competing alternatives in terms of clustering accuracy, while maintaining a low computational overhead. OMP also exhibits low computational time, at the expense of clustering accuracy.

2.5.4 Distributed SC

Next we evaluate the performance of the proposed Distributed sketched SC scheme, using real datasets. Let d denote the number of observed entries per column of $\mathbf{X}$. Distributed sketched SC methods (denoted by *DS-LSR, DS-SSC* and *DS-LRR*) are compared to batch LSR [cf. (2.21)] across variable percentages of available data d/D. The metrics evaluated are the *clustering accuracy*, expressed as the percentage of correctly clustered data given by the rate $(\text{number of data correctly clustered})/N$, and the computational time per node in seconds, that is the average time required by a single node to solve (2.24). The $D \times N$ dataset $\mathbf{X}$ is distributed across L different computational nodes, each required to process approximately $N_\ell = \lfloor N/L \rfloor$ data. The $N \times n$ random matrices $\mathbf{R}$ in all experiments have i.i.d. normal entries rescaled by a factor $1/\sqrt{n}$. As matrix $\mathbf{A}$ resulting from Alg. 2.4 is not $N \times N$, a k-nearest neighbor (k-NN) graph, with adjacency matrix $\mathbf{W}$, is created using the columns of $\mathbf{A}$. The edge weights of this graph are given by $[\mathbf{W}]_{ij} = \exp(-\|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2^2/2\sigma^2)$, where $\boldsymbol{a}_i$ is one of the k nearest neighbors



(a) Clustering accuracy



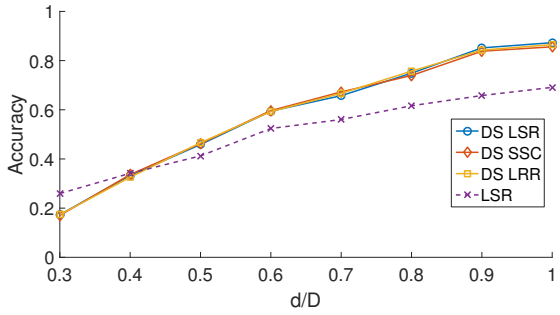
(b) Clustering time

Figure 2.5: Simulated tests on real dataset Extended Yale Face Database B, with $N = 2,414$ data dimension $D = 2,016$ and $K = 38$ clusters for varying d and fixed $n = 70$.

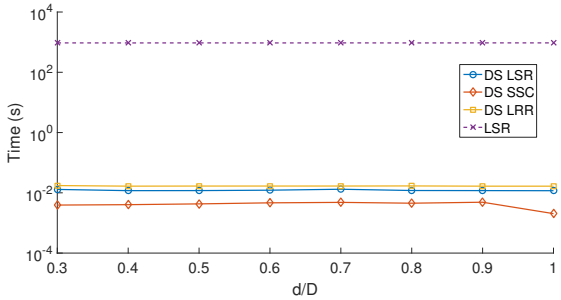
of $\boldsymbol{a}_j$. Here σ is the average distance between all the columns of $\mathbf{A}$ involved in the k nearest neighbor computation. For all tests, the number of nearest neighbors for the graph construction is set to $k = 5$. Tests were performed using 2 real datasets of the previous subsections: Extended Yale Face database B [47], and the MNIST handwritten digits dataset [78].

Fig. 2.6 shows the results for the Extended Yale Face database. Here, $L = 50$ computing nodes are considered and $n = 150$, $\lambda = 100$ for DS-SSC, $\lambda = 0.15$ for DS-LRR, and $\lambda = 10^6$ for LSR and DS-LSR. Fig. 2.7 shows the results for the preprocessed MNIST dataset. Again, here $L = 50$, while $\lambda = 0.6$ for DS-SSC $\lambda = 0.3$ for DS-LRR, and $\lambda = 1.2 \cdot 10^4$ for LSR and DS-LSR.

From all the tests it can be seen that distributed sketched SC approaches exhibit comparable clustering accuracy with their batch counterparts, even in the presence of missing data, while requiring markedly less computational time per node, thus corroborating their merits in large-scale and distributed settings. Also note that as L increases, computational time per node is



(a) Clustering accuracy



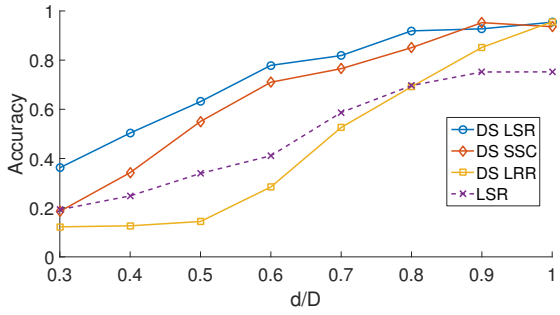
(b) Computational time per node (seconds)

Figure 2.6: Simulated tests on 'Extended Yale Face Database B,' with $N = 2,414$ data; $D = 2,016$; and $K = 38$.

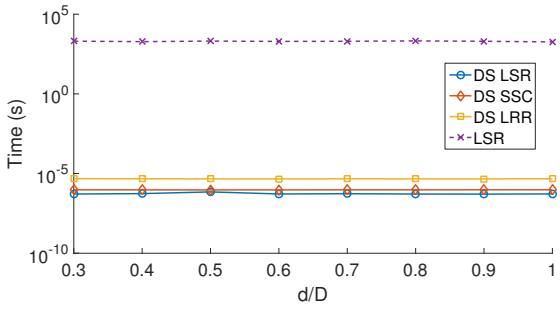
expected to decrease.

2.6 Conclusion

The present chapter introduced a novel data-reduction scheme for subspace clustering, namely Sketch-SC, that enables grouping of data drawn from a union of subspaces based on a random sketching approach for fast, yet-accurate subspace clustering. Performance of the proposed scheme was evaluated both analytically and through simulated tests on multiple real datasets.



(a) Clustering accuracy



(b) Computational time per node (seconds)

Figure 2.7: Simulated tests on a preprocessed subset of MNIST dataset with $N = 5,000$ data; $D = 500$; and $K = 10$.

Chapter 3

Learning with Blind Ensembles of Classifiers for iid data

The rising interest in pattern recognition and data analytics has spurred the development of innovative machine learning algorithms and tools. However, as each algorithm has its strengths and limitations, one is motivated to judiciously fuse multiple algorithms in order to find the "best" performing one, for a given dataset. The present chapter introduces a blind scheme for learning from ensembles of classifiers, using a moment matching method that leverages joint tensor and matrix factorization. Blind refers to the combiner who has no knowledge of the ground-truth labels that each classifier has been trained on. A rigorous performance analysis is derived and the proposed scheme is evaluated on synthetic and real datasets.

3.1 Problem Statement and Preliminaries

Consider a dataset consisting of N data (possibly vectors) $\{x_n\}_{n=1}^N$ each belonging to one of K possible classes with corresponding labels $\{y_n\}_{n=1}^N$, e.g. $y_n = k$ if x_n belongs to class k. The pairs $\{(x_n, y_n)\}$ are drawn independently from an unknown joint distribution D, and X and Y denote random variables such that $(X, Y) \sim \mathrm{D}$. Consider now M annotators that observe $\{x_n\}_{n=1}^N$, and provide estimates of labels. Let $f_m(x_n) \in \{1, \ldots, K\}$ denote the label assigned to datum x_n by the m-th annotator. All annotator responses are then collected at a centralized meta-learner or fusion center. Collect all annotator responses in the $M \times N$ matrix $\mathbf{F}$, that has entries $[\mathbf{F}]_{mn} = f_m(x_n)$, and all ground-truth labels in the $N \times 1$ vector $\boldsymbol{y} = [y_1, \ldots, y_N]^\top$

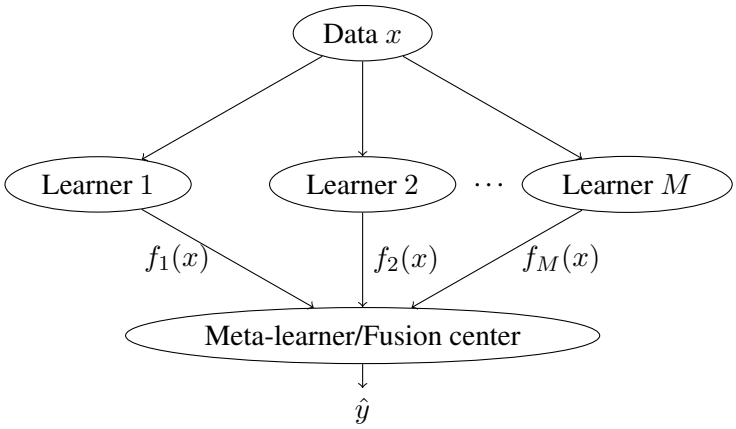


Figure 3.1: Unsupervised ensemble classification setup, where the outputs of learners are combined in parallel.

. The task of *unsupervised ensemble classification* is: Given only the annotator responses $\{f_m(x_n), m = 1, \ldots, M\}_{n=1}^N$, we wish to estimate the ground-truth labels of the data $\{y_n\}$; see Fig. 3.1.

Similar to unsupervised ensemble classification, crowdsourced classification seeks to estimate ground-truth labels of the data $\{y_n\}$ from annotator responses $\{f_m(x_n)\}$, with the additional caveat that each annotator m may choose to provide labels for only a subset $N_m < N$ of data.

3.1.1 Prior work

Probably the simplest scheme for blind or unsupervised ensemble classification is majority voting, where the estimated label of a datum is the one that most annotators agree upon. Majority voting has been used in popular ensemble schemes such as bagging, and random forests [20]. While relatively easy to implement, majority voting presumes that all annotators are equally "reliable," which is rather unrealistic, both in crowdsourcing as well as in ensemble learning setups. Other blind ensemble methods aim to estimate the parameters that characterize the annotators' performance. A joint maximum likelihood (ML) estimator of the unknown labels and these parameters has been reported using the expectation-maximization (EM) algorithm [31]. As the EM algorithm does not guarantee convergence to the ML solution, recent works pursue alternative estimation methods. For binary classification, [48] assumes that annotators adhere to the "one-coin" model, meaning each annotator m provides the correct (incorrect) label with

probability δ_m $(1-\delta_m)$; see also [30] when annotators do not label all the data, and [71] for an iterative method. Recently, [106], [65] advocated a spectral decomposition technique of the second-order statistics of annotator responses for binary classification, that yields the reliability parameters of annotators, when class probabilities are unknown, while [16] introduced a minimax optimal algorithm that can infer annotator reliabilities. In the multiclass setting, [71] solves multiple binary classification problems. In addition, [66] and [162] utilize third-order moments and orthogonal tensor decomposition to estimate the unknown reliability parameters and then initialize the EM algorithm of [31]. This procedure however, can be numerically unstable, especially when the number of classes K is large, and classes are unequally populated. Finally, all the methods mentioned in this section employ ML estimation, which implicitly assumes that the dataset is balanced, meaning classes are roughly equiprobable. Another interesting approach is presented in [74], where a joint moment matching and maximum likelihood optimization problem is solved.

The present work puts forth a novel scheme for *multiclass blind ensemble classification*, built upon simple concepts from probability and detection theory. It relies on a joint PARAFAC decomposition approach, which lends itself to a numerically stable algorithm. At the same time, our novel approach takes into account class prior probabilities to yield accurate estimates of class labels. Compared to our conference precursor in [138], here we do not require the prior probabilities to be known, and we present comprehensive numerical tests, along with a rigorous performance analysis.

3.1.2 Canonical Polyadic Decomposition/PARAFAC

This subsection will outline tensor decompositions, which will be used in the following sections to derive the proposed scheme. Consider a 3-mode $I \times J \times L$ tensor $\underline{X}$, which can be described by a matrix in 3 different ways

$$\mathbf{X}^{(1)} := [\text{vec}(\underline{X}(1,:,:)), \ldots, \text{vec}(\underline{X}(I,:,:))] \tag{3.1a}$$

$$\mathbf{X}^{(2)} := [\text{vec}(\underline{X}(:,1,:)), \ldots, \text{vec}(\underline{X}(:,J,:))] \tag{3.1b}$$

$$\mathbf{X}^{(3)} := [\text{vec}(\underline{X}(:,:,1)), \ldots, \text{vec}(\underline{X}(:,:,L))] \tag{3.1c}$$

where $\mathbf{X}^{(1)}$ is of dimension $JL \times I$, $\mathbf{X}^{(2)}$ is $IL \times J$ and $\mathbf{X}^{(3)}$ is $IJ \times L$. Under the Canonical Polyadic Decomposition(CPD)/Parallel Factor Analysis (PARAFAC) model [56], $\underline{X}$ can be written as a sum of R rank one tensors (a.k.a. *factors*)

$$\underline{X} = \sum_{r=1}^{R} \boldsymbol{a}_r \circ \boldsymbol{b}_r \circ \boldsymbol{c}_r \tag{3.2}$$

where $\boldsymbol{a}_r, \boldsymbol{b}_r, \boldsymbol{c}_r$ are $I \times 1$, $J \times 1$ and $L \times 1$ vectors, respectively. Letting $\mathbf{A} := [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_R], \mathbf{B} := [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_R]$, and $\mathbf{C} := [\boldsymbol{c}_1, \ldots, \boldsymbol{c}_R]$ be the so-called factor matrices of the CPD model, we write (3.2) compactly as

$$\underline{X} = [[\mathbf{A}, \mathbf{B}, \mathbf{C}]]_R \tag{3.3}$$

and (3.1) can be equivalently written as

$$\mathbf{X}^{(1)} = (\mathbf{C} \odot \mathbf{B})\, \mathbf{A}^\top \tag{3.4a}$$

$$\mathbf{X}^{(2)} = (\mathbf{C} \odot \mathbf{A})\, \mathbf{B}^\top \tag{3.4b}$$

$$\mathbf{X}^{(3)} = (\mathbf{B} \odot \mathbf{A})\, \mathbf{C}^\top \tag{3.4c}$$

where we have used the fact that for matrices $\mathbf{A}, \mathbf{B}$ and a vector $\boldsymbol{c}$ of appropriate dimensions, it holds that $\text{vec}(\mathbf{A}\text{diag}(\boldsymbol{c})\mathbf{B}^\top) = (\mathbf{B} \odot \mathbf{A})\boldsymbol{c}$. By vectorizing $\mathbf{X}^{(3)}$, it is easy to show that the vectorization of the entire tensor will be of the form $\boldsymbol{x} := \text{vec}(\underline{X}) = \text{vec}(\mathbf{X}^{(3)}) = (\mathbf{C} \odot \mathbf{B} \odot \mathbf{A})\, \mathbf{1}$. Accordingly, vectorizing $\mathbf{X}^{(1)}$ or $\mathbf{X}^{(2)}$ produces different vectorizations of the entire tensor, where the order of factor matrices in the Khatri-Rao product is permuted. Recovery of the factor matrices $\mathbf{A}, \mathbf{B}$ and $\mathbf{C}$, can be done by solving the following non-convex optimization problem

$$[\hat{\mathbf{A}}, \hat{\mathbf{B}}, \hat{\mathbf{C}}] = \underset{\mathbf{A},\mathbf{B},\mathbf{C}}{\arg\min} \|\underline{X} - [[\mathbf{A}, \mathbf{B}, \mathbf{C}]]_R\|_F^2. \tag{3.5}$$

Similar to the matrix case, the Frobenius norm here can be defined as $\|\underline{X}\|_F := \sqrt{\sum_{i,j,l} \underline{X}(i,j,l)^2}$, and as (3.4) is just a rearrangement of the terms in $\underline{X}$, it holds that

$$\|\underline{X}\|_F = \|\mathbf{X}^{(1)}\|_F = \|\mathbf{X}^{(2)}\|_F = \|\mathbf{X}^{(3)}\|_F. \tag{3.6}$$

Typically, (3.5) is solved using alternating optimization (AO) or gradient descent [126]. Multiple off-the-shelf solvers are available for PARAFAC tensor decomposition; see e.g. [6, 144]. Furthermore, depending on extra properties of $\underline{X}$, constraints can be enforced on the factor matrices, such as nonnegativity and sparsity to name a few, which can be effectively handled by popular solvers such as AO-ADMM [62]. Under certain conditions, the factorization of $\underline{X}$ into $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$, is *essentially unique*, or *essentially identifiable*, that is $\hat{\mathbf{A}}$, $\hat{\mathbf{B}}$, and $\hat{\mathbf{C}}$ can be expressed as

$$\hat{\mathbf{A}} = \mathbf{A}\mathbf{P}\mathbf{\Lambda}_a, \quad \hat{\mathbf{B}} = \mathbf{B}\mathbf{P}\mathbf{\Lambda}_b, \quad \hat{\mathbf{C}} = \mathbf{C}\mathbf{P}\mathbf{\Lambda}_c \tag{3.7}$$

where $\mathbf{P}$ is a common permutation matrix, and $\mathbf{\Lambda}_a, \mathbf{\Lambda}_b, \mathbf{\Lambda}_c$ are diagonal scaling matrices such that $\mathbf{\Lambda}_a\mathbf{\Lambda}_b\mathbf{\Lambda}_c = \mathbf{I}$ [126]. For more details regarding the PARAFAC decomposition and tensors with more than 3 modes, interested readers are referred to the comprehensive tutorial in [126] and references therein.

3.2 Unsupervised Ensemble Classification

Each annotator in our model has a fixed probability of deciding that a datum belongs to class k', when presented with a datum of class k. Thus, each annotator m can be characterized by a so called *confusion* matrix $\mathbf{\Gamma}_m$, whose (k', k)-th entry is

$$[\mathbf{\Gamma}_m]_{k'k} := \Gamma_m(k', k) = \Pr\left(f_m(X) = k' | Y = k\right). \tag{3.8}$$

The $K \times K$ matrix $\mathbf{\Gamma}_m$ has non-negative entries that obey the simplex constraint, since $\sum_{k'=1}^{K} \Pr\left(f_m(X) = k' | Y = k\right) = 1$, for $k = 1, \ldots, K$; hence, entries of each $\mathbf{\Gamma}_m$ column sum up to 1, that is, $\mathbf{\Gamma}_m^\top \mathbf{1} = \mathbf{1}$ and $\mathbf{\Gamma}_m \geq \mathbf{0}$. The confusion matrix showcases the statistical behavior of an annotator, as each column provides the annotator's probability of deciding the correct class, when presented with a datum from each class. Before proceeding, we adopt the following assumptions.

As1. Responses of different annotators per datum, are conditionally independent, given the ground-truth label Y of the same datum X; that is,

$$\Pr\left(f_1(X) = k_1, \ldots, f_M(X) = k_M | Y = k\right) = \prod_{m=1}^{M} \Pr\left(f_m(X) = k_m | Y = k\right)$$

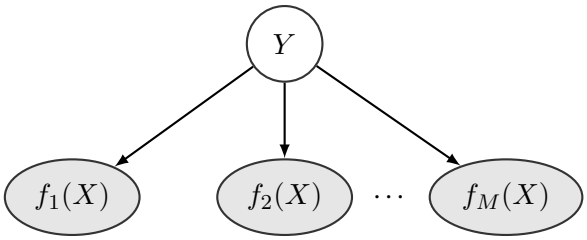


Figure 3.2: Graphical representation of the Dawid and Skene model for i.i.d. data. Shaded ellipses indicate observed variables, i.e. annotator responses.

As2. Most annotators are better than random; e.g., most have probability of correct detection exceeding 0.5 for $K = 2$.

Clearly, for annotators that are better than random, the largest elements of each column of their confusion matrix will be those on the diagonal of $\mathbf{\Gamma}_m$; that is

$$[\mathbf{\Gamma}_m]_{kk} \geq [\mathbf{\Gamma}_m]_{k'k}, \text{ for } k', k = 1, \ldots, K.$$

As1, which is also known as the Dawid and Skene model, suggests that annotators make decisions independently of each other, which is rather a standard assumption [31, 65, 162]. A graphical representation of the resulting model is shown in Fig. 3.2. Likewise, As2 is another standard assumption, used to alleviate the inherent permutation ambiguity of the confusion matrix estimates provided by our algorithm. Note that As2 is slightly more relaxed than the corresponding assumption in [162], which splits annotators into 3 groups and requires most annotators in each group to be better than random.

3.2.1 Maximum a posteriori label estimation

Given only annotator responses for all data, a straightforward approach to estimating their ground-truth labels is through a maximum a posteriori (MAP) classifier [73]. In particular, for datum X the MAP classifier is

$$\hat{y}_{\text{MAP}}(X) = \arg \max_{k \in \{1,\ldots,K\}} L(X|k) \Pr(Y = k) \tag{3.9}$$

where $L(X|k) := \Pr\left(f_1(X) = k_1, \ldots, f_M(X) = k_M | Y = k\right)$ is the conditional likelihood of X. As annotators make independent decisions, it holds that $L(X|k) = \prod_{m=1}^{M} \Pr\left(f_m(X) = k_m | Y = k\right)$,

and thus the MAP classifier can be rewritten as

$$\hat{y}_{\mathrm{MAP}}(X) = \underset{k\in\{1,\ldots,K\}}{\arg\max} \log \pi_k + \sum_{m=1}^{M} \log(\Gamma_m(k_m, k)) \tag{3.10}$$

where $\pi_k := \Pr(Y = k)$. It is well known from detection theory [73] that the MAP classifier (3.10) minimizes the average probability of error $\mathrm{P_e}$, given by

$$\mathrm{P_e} = \sum_{k=1}^{K} \pi_k \Pr(\hat{y}_{\mathrm{MAP}} = k' \neq k | Y = k). \tag{3.11}$$

If all classes are equiprobable, that is $\pi_k = 1/K$ for all $k = 1, \ldots, K$, then (3.10) reduces to the ML classifier. In order to obtain the MAP or ML classifier, $\{\mathbf{\Gamma}_m\}_{m=1}^{M}$ must be available, while in the MAP classifier case $\boldsymbol{\pi} := [\pi_1, \ldots, \pi_K]^\top$ is also required. Interestingly, the next section will illustrate that $\{\mathbf{\Gamma}_m\}_{m=1}^{M}$ and $\boldsymbol{\pi}$ show up in (and can thus be estimated from) the moments of annotator responses.

3.2.2 The Expectation Maximization algorithm

As mentioned in Sec. 3.1.1, a popular method for ML estimation of the unknown labels and annotator performance parameters is the EM algorithm. The EM algorithm seeks to iteratively maximize the marginal log-likelihood of the observed annotator responses, that is $\log \Pr(\mathbf{F}|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is used to simplify notation, and concatenates all the annotator confusion matrices. Each iteration of the EM algorithm consists of two steps, the Expectation (or E-)step and the Maximization (or M-)step. At the E-step of the $i+1$-th iteration, and given current parameter estimates $\boldsymbol{\theta}^{(i)}$, the so-called Q-function is derived, defined as

$$\begin{aligned} Q(\boldsymbol{\theta}; \boldsymbol{\theta}^{(i)}) &= \mathbb{E}_{\boldsymbol{y}|\mathbf{F};\boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y}, \mathbf{F}; \boldsymbol{\theta})] \\ &= \mathbb{E}_{\boldsymbol{y}|\mathbf{F};\boldsymbol{\theta}^{(i)}}[\log \Pr(\mathbf{F}|\boldsymbol{y}; \boldsymbol{\theta})] + \mathbb{E}_{\boldsymbol{y}|\mathbf{F};\boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y}; \boldsymbol{\theta})]. \end{aligned} \tag{3.12}$$

Since the data are i.i.d. and using As 1 we have that

$$\mathbb{E}_{\boldsymbol{y}|\mathbf{F};\boldsymbol{\theta}^{(i)}}[\log \Pr(\mathbf{F}|\boldsymbol{y}; \boldsymbol{\theta})] = \sum_{n=1}^{N} \sum_{k=1}^{K} \sum_{m=1}^{M} \log \Gamma_m(f_m(x_n), k) q_{nk}$$

and

$$\mathbb{E}_{\boldsymbol{y}|\mathbf{F};\boldsymbol{\theta}^{(i)}}[\log \Pr(\boldsymbol{y};\boldsymbol{\theta})] = \sum_{n=1}^{N}\sum_{k=1}^{K} \log \Pr(y_n = k;\boldsymbol{\theta}) q_{nk}$$

where we have defined $q_{nk} := \Pr(y_n = k|\mathbf{F};\boldsymbol{\theta}^{(i)}) = \Pr(y_n = k|\{f_m(x_n)\}_{m=1}^M;\boldsymbol{\theta}^{(i)})$. Under our model, it can be shown [31, 162] that

$$q_{nk}^{(i+1)} = \frac{1}{Z}\exp\left(\sum_{m=1}^{M}\sum_{k'=1}^{K} \mathcal{I}(f_m(x_n) = k') \log(\Gamma_m^{(i)}(k',k))\right) \tag{3.13}$$

where Z is the normalization constant. At the M-step, annotator confusion matrices are updated by maximizing the Q-function, that is

$$\boldsymbol{\theta}^{(i+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta};\boldsymbol{\theta}^{(i)}). \tag{3.14}$$

Accordingly, it can be shown that, per annotator m, (3.14) boils down to

$$[\boldsymbol{\Gamma}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^{K}\sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')}. \tag{3.15}$$

The E- and M-steps are then repeated until convergence. Afterwards, ML estimates of data labels can be obtained as follows:

$$\hat{y}(x_n) = \underset{k\in\{1,\ldots,K\}}{\arg\max}\ \Pr(\{f_m(x_n)\}_{m=1}^M, y_n = k) \Rightarrow \hat{y}(x_n) = \underset{k\in\{1,\ldots,K\}}{\arg\max}\, q_{nk}.$$

As the EM algorithm solves a nonconvex optimization problem, the accuracy of its results depends on the initialization. Interestingly, the next section will illustrate that $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ and $\boldsymbol{\pi}$ show up in (and can thus be estimated from) the moments of annotator responses. The parameters estimated from these moments can then be used directly with the MAP estimator of Sec. 3.2.1 or to initialize the aforementioned EM algorithm.

3.2.3 Statistics of annotator responses

Consider each label represented by the annotators using the canonical $K \times 1$ vector $\boldsymbol{e}_k$, denoting the k-th column of the $K \times K$ identity matrix $\mathbf{I}$. Let $\mathbf{f}_m(X)$ denote the m-th annotator's response in vector format. Since $\mathbf{f}_m(X)$ is just a vector representation of $f_m(X)$, it holds that

$\Pr\left(f_m(X)=k'|Y=k\right) \equiv \Pr\left(\mathbf{f}_m(X)=\boldsymbol{e}_{k'}|Y=k\right)$. With $\boldsymbol{\gamma}_{m,k}$ denoting the k-th column of $\boldsymbol{\Gamma}_m$, it thus holds that

$$\mathbb{E}[\mathbf{f}_m(X)|Y=k]=\sum_{k'=1}^{K}\boldsymbol{e}_{k'}\Pr\left(f_m(X)=k'|Y=k\right)=\boldsymbol{\gamma}_{m,k} \tag{3.16}$$

where the first equality comes from the definition of conditional expectation, and the second one because $\boldsymbol{e}_k$'s are columns of $\mathbf{I}$. Using (3.16) and the law of total probability, the mean vector of responses from annotator m, is hence

$$\mathbb{E}[\mathbf{f}_m(X)]=\sum_{k=1}^{K}\mathbb{E}[\mathbf{f}_m(X)|Y=k]\Pr\left(Y=k\right)=\boldsymbol{\Gamma}_m\boldsymbol{\pi}. \tag{3.17}$$

Upon defining the diagonal matrix $\boldsymbol{\Pi}:=\text{diag}(\boldsymbol{\pi})$, the $K\times K$ cross-correlation matrix between the responses of annotators m and $m'\neq m$, can be expressed as

$$\begin{aligned}\mathbf{R}_{mm'}:=\mathbb{E}[\mathbf{f}_m(X)\mathbf{f}_{m'}^{\top}(X)]&=\sum_{k=1}^{K}\mathbb{E}[\mathbf{f}_m(X)|Y=k]\,\mathbb{E}[\mathbf{f}_{m'}^{\top}(X)|Y=k]\Pr\left(Y=k\right)\\&=\boldsymbol{\Gamma}_m\text{diag}(\boldsymbol{\pi})\boldsymbol{\Gamma}_{m'}^{\top}=\boldsymbol{\Gamma}_m\boldsymbol{\Pi}\boldsymbol{\Gamma}_{m'}^{\top}\end{aligned} \tag{3.18}$$

where we successively relied on the law of total probability, As1, and (3.16). Consider now the $K\times K\times K$ cross-correlation tensor between the responses of annotators m, $m'\neq m$ and $m''\neq m', m$, namely

$$\underline{\Psi}_{mm'm''}=\mathbb{E}[\mathbf{f}_m(X)\circ\mathbf{f}_{m'}(X)\circ\mathbf{f}_{m''}(X)]. \tag{3.19}$$

It can be shown that $\underline{\Psi}_{mm'm''}$ obeys a CPD/PARAFAC model [cf. Sec. 3.1.2] with factor matrices $\boldsymbol{\Gamma}_m, \boldsymbol{\Gamma}_{m'}$ and $\boldsymbol{\Gamma}_{m''}$; that is,

$$\underline{\Psi}_{mm'm''}=\sum_{k=1}^{K}\pi_k\boldsymbol{\gamma}_{m,k}\circ\boldsymbol{\gamma}_{m',k}\circ\boldsymbol{\gamma}_{m'',k}=[[\boldsymbol{\Gamma}_m\boldsymbol{\Pi},\boldsymbol{\Gamma}_{m'},\boldsymbol{\Gamma}_{m''}]]_K. \tag{3.20}$$

Note here that the diagonal matrix $\boldsymbol{\Pi}$ can multiply any of the factor matrices $\boldsymbol{\Gamma}_m, \boldsymbol{\Gamma}_{m'}$, or, $\boldsymbol{\Gamma}_{m''}$.

With $\bar{\mathbf{F}}_m := [\mathbf{f}_m(x_1), \mathbf{f}_m(x_2), \ldots, \mathbf{f}_m(x_N)]$ the sample mean of the m-th annotator responses can be readily obtained as

$$\boldsymbol{\mu}_m = \frac{1}{N}\sum_{n=1}^{N} \mathbf{f}_m(x_n) = \frac{1}{N}\bar{\mathbf{F}}_m \mathbf{1}. \tag{3.21}$$

Accordingly, the $K \times K$ sample cross-correlation $\mathbf{S}_{mm'}$ matrices between the responses of annotators m and $m' \neq m$, are given by

$$\mathbf{S}_{mm'} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{f}_m(\boldsymbol{x}_n)\mathbf{f}_{m'}^{\top}(\boldsymbol{x}_n) = \frac{1}{N}\bar{\mathbf{F}}_m \bar{\mathbf{F}}_{m'}^{\top}. \tag{3.22}$$

Lastly, the sample $K \times K \times K$ cross-correlation tensors $\underline{T}_{mm'm''}$ between the responses of annotators $m, m' \neq m$ and $m'' \neq m, m'$ are

$$\underline{T}_{mm'm''} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{f}_m(\boldsymbol{x}_n) \circ \mathbf{f}_{m'}(\boldsymbol{x}_n) \circ \mathbf{f}_{m''}(\boldsymbol{x}_n) = \frac{1}{N}\bar{\mathbf{F}}_m \circ \bar{\mathbf{F}}_{m'} \circ \bar{\mathbf{F}}_{m''}. \tag{3.23}$$

Clearly, $\mathbf{S}_{mm'} = \mathbf{S}_{m'm}^{\top}$, $\mathbf{T}^{(2)}_{m'mm''} = \mathbf{T}^{(3)}_{m'm''m} = \mathbf{T}^{(1)}_{mm'm''}$. In addition, as N increases, the law of large numbers (LLN) implies that, $\{\boldsymbol{\mu}_m\}$, $\{\mathbf{S}_{mm'}\}$, and $\{\underline{T}_{mm'm''}\}$ approach their ensemble counterparts in (3.17), (3.18), and (3.19).

Having available first-, second-, and third-order statistics of annotator responses, namely $\{\boldsymbol{\mu}_m\}_{m=1}^M$, $\{\mathbf{S}_{mm'}\}_{m,m'=1}^M$, and $\{\underline{T}_{mm'm''}\}_{m,m',m''=1}^M$, estimates of $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ and $\boldsymbol{\pi}$ can be readily extracted from them [cf. (3.17), (3.18), (3.19)]. This procedure corresponds to the method-of-moments estimation [72]. Upon obtaining $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$ and $\hat{\boldsymbol{\pi}}$, the MAP classifier of Sec. 3.2.1 can be subsequently employed to estimate the label for each datum. That is, for $n = 1, \ldots, N$,

$$\hat{y}_{\mathrm{MAP}}(x_n) = \underset{k\in\{1,\ldots,K\}}{\arg\max} \log \hat{\pi}_k + \sum_{m=1}^{M} \log \hat{\Gamma}_m(f_m(x_n), k) \tag{3.24}$$

where $\hat{\Gamma}_m(k', k) = [\hat{\boldsymbol{\Gamma}}_m]_{k'k}$, and $\hat{\pi}_k = [\hat{\boldsymbol{\pi}}]_k$. The following section provides an algorithm to estimate these unknown quantities.

3.2.4 Moment matching for confusion matrix and prior probability estimation

To estimate the unknown confusion matrices and prior probabilities consider the following non-convex constrained optimization problem,

$$\begin{aligned} \min_{\substack{\boldsymbol{\pi} \\ \{\boldsymbol{\Gamma}_m\}_{m=1}^M}} \quad & h_N(\{\boldsymbol{\Gamma}_m\}_{m=1}^M, \boldsymbol{\pi}) \\ \text{s.to} \quad & \boldsymbol{\Gamma}_m \geq \mathbf{0}, \quad \boldsymbol{\Gamma}_m^\top \mathbf{1} = \mathbf{1}, \quad m = 1, \ldots, M \\ & \boldsymbol{\pi} \geq \mathbf{0}, \quad \boldsymbol{\pi}^\top \mathbf{1} = 1 \end{aligned} \tag{3.25}$$

where

$$\begin{aligned} h_N(\{\boldsymbol{\Gamma}_m\}, \boldsymbol{\pi}) := & \frac{1}{2} \sum_{m=1}^M \|\boldsymbol{\mu}_m - \boldsymbol{\Gamma}_m \boldsymbol{\pi}\|_2^2 \\ & + \frac{1}{2} \sum_{\substack{m=1 \\ m'>m}}^M \|\mathbf{S}_{mm'} - \boldsymbol{\Gamma}_m \boldsymbol{\Pi} \boldsymbol{\Gamma}_{m'}^\top\|_F^2 \\ & + \frac{1}{2} \sum_{\substack{m=1 \\ m'>m \\ m''>m'}}^M \|\underline{T}_{mm'm''} - [[\boldsymbol{\Gamma}_m \boldsymbol{\Pi}, \boldsymbol{\Gamma}_{m'}, \boldsymbol{\Gamma}_{m''}]]_K\|_F^2 \end{aligned}$$

and the subscript N in h_N denotes the number of data used to obtain annotator statistics. Collect the set of constraints per matrix to the convex set $\mathcal{C} := \{\boldsymbol{\Gamma} \in \mathbb{R}^{K \times K} : \boldsymbol{\Gamma} \geq \mathbf{0}, \boldsymbol{\Gamma}^\top \mathbf{1} = \mathbf{1}\}$, where essentially each column lies on a probability simplex, and let $\mathcal{C}_p := \{\boldsymbol{u} \in \mathbb{R}^K : \boldsymbol{u} \geq \mathbf{0}, \boldsymbol{u}^\top \mathbf{1} = 1\}$ denote the constraint set for $\boldsymbol{\pi}$.

As (3.25) is a non-convex problem, alternating optimization will be employed to solve it. Specifically the alternating optimization-alternating direction method of multipliers (AO-ADMM) will be employed; see [62], and also [70] where a similar formulation appears. Under the AO-ADMM paradigm, h_N is minimized per block of unknown variables $\{\boldsymbol{\Gamma}_m\}$ or $\boldsymbol{\pi}$ while the other blocks remain fixed, as in block coordinate descent schemes. Solving for one block of variables with the remaining fixed is a convex constrained optimization problem under convex $\mathcal{C}$ and $\mathcal{C}_p$ constraint sets. These optimization problems are pretty standard and several solvers are available, including proximal splitting methods, projected gradient descent or ADMM [13, 49, 86, 105]. Here, the solver of choice for each block of variables will be ADMM.

Algorithm 3.1 Confusion matrix and prior probability estimation algorithm

Input: Annotator responses $\{\mathbf{F}_m\}_{m=1}^M$, $\lambda > 0,\ \nu > 0$; maximum number of iterations $I \in \mathbb{Z}_+$
Output: Estimates of $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$ and $\hat{\boldsymbol{\pi}}$

Compute $\{\boldsymbol{\mu}_m\}, \{\mathbf{S}_{mm'}\}, \{\underline{T}_{mm'm''}\}$ using (3.21), (3.22), and (3.23).
Initialize $\{\mathbf{\Gamma}_m\}$ and $\boldsymbol{\pi}$ randomly.
do
  for $m = 1, \ldots, M$ **do**
    Update $\mathbf{\Gamma}_m$ using (3.27)
    $\mathbf{\Gamma}_m^{(\text{prev})} \leftarrow \mathbf{\Gamma}_m$
  end for
  Update $\boldsymbol{\pi}$ using (3.26)
  $\boldsymbol{\pi}^{(\text{prev})} \leftarrow \boldsymbol{\pi}$
  $i \leftarrow i + 1$
while not converged and $i < I_T$
Find permutation matrix $\hat{\mathbf{P}}$, such that the majority of $\{\hat{\mathbf{\Gamma}}_m \hat{\mathbf{P}}\}_{m=1}^M$ satisfy As2.

Algorithm 3.2 Unsupervised multiclass ensemble classification

Input: Annotator responses $\{\mathbf{F}_m\}_{m=1}^M$
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^N$

Find estimates $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$ and $\hat{\boldsymbol{\pi}}$ using Alg. 3.1
for $n = 1, \ldots, N$ **do**
  Estimate label y_n using (3.24).
end for

The update for $\boldsymbol{\pi}$ involves minimizing h_N with $\{\mathbf{\Gamma}_m\}_{m=1}^M$ fixed. Specifically, the following problem is solved

$$\min_{\boldsymbol{\pi} \in \mathcal{C}_p} \quad g_{N,\boldsymbol{\pi}}(\boldsymbol{\pi}) \tag{3.26}$$

where

$$g_{N,\boldsymbol{\pi}}(\boldsymbol{\pi}) := \frac{1}{2} \sum_{m=1}^{M} \|\boldsymbol{\mu}_m - \mathbf{\Gamma}_m \boldsymbol{\pi}\|_2^2 + \frac{\nu}{2} \|\boldsymbol{\pi} - \boldsymbol{\pi}^{(\text{prev})}\|_2^2$$
$$+ \frac{1}{2} \sum_{\substack{m=1 \\ m'>m}}^{M} \|\boldsymbol{s}_{mm'} - (\mathbf{\Gamma}_{m'} \odot \mathbf{\Gamma}_m)\, \boldsymbol{\pi}\|_2^2$$

$$+\frac{1}{2}\sum_{\substack{m=1\\ m'>m\\ m''>m'}}^{M}\|\boldsymbol{t}_{mm'm''}-(\boldsymbol{\Gamma}_{m''}\odot\boldsymbol{\Gamma}_{m'}\odot\boldsymbol{\Gamma}_{m})\boldsymbol{\pi}\|_2^2$$

$\boldsymbol{s}_{mm'}=\text{vec}(\mathbf{S}_{mm'})$, $\boldsymbol{t}_{mm'm''}=\text{vec}(\mathbf{T}^{(3)}_{mm'm''})$ [cf. (3.4)], ν is a positive scalar, and ewe have used $\text{vec}(\boldsymbol{\Gamma}_m\text{diag}(\boldsymbol{\pi})\boldsymbol{\Gamma}_{m'}^\top)=(\boldsymbol{\Gamma}_{m'}\odot\boldsymbol{\Gamma}_m)\,\boldsymbol{\pi}$ and $\text{vec}([[\boldsymbol{\Gamma}_m\text{diag}(\boldsymbol{\pi}),\boldsymbol{\Gamma}_{m'},\boldsymbol{\Gamma}_{m''}]]_K)=(\boldsymbol{\Gamma}_{m''}\odot\boldsymbol{\Gamma}_{m'}\odot\boldsymbol{\Gamma}_m)\boldsymbol{\pi}$. Note that $g_{N,\boldsymbol{\pi}}$ contains all of the terms in h_N along with $(\nu/2)\|\boldsymbol{\pi}-\boldsymbol{\pi}^{(\text{prev})}\|_2^2$, which is included to ensure convergence of the AO-ADMM iterations to a stationary point of (3.25) [62, 117]. Here, $\boldsymbol{\pi}^{(\text{prev})}$ denotes the estimate of $\boldsymbol{\pi}$ obtained by the previous solutions of (3.26).

Accordingly per $\boldsymbol{\Gamma}_m$, the following subproblem is solved with $\{\boldsymbol{\Gamma}_{m'}\}_{m'\neq m}^M$ and $\boldsymbol{\pi}$ fixed

$$\min_{\boldsymbol{\Gamma}_m\in\mathcal{C}}\quad g_{N,m}(\boldsymbol{\Gamma}_m) \tag{3.27}$$

where

$$\begin{aligned}g_{N,m}(\boldsymbol{\Gamma}_m):=&\frac{1}{2}\|\boldsymbol{\mu}_m-\boldsymbol{\Gamma}_m\boldsymbol{\pi}\|_2^2+\frac{\nu}{2}\|\boldsymbol{\Gamma}_m-\boldsymbol{\Gamma}_m^{(\text{prev})}\|_F^2\\&+\frac{1}{2}\sum_{m'\neq m}^{M}\|\mathbf{S}_{m'm}-\boldsymbol{\Gamma}_{m'}\boldsymbol{\Pi}\boldsymbol{\Gamma}_m^\top\|_F^2\\&+\frac{1}{2}\sum_{\substack{m'>m\\ m''>m'}}^{M}\|\mathbf{T}^{(1)}_{mm'm''}-(\boldsymbol{\Gamma}_{m''}\odot\boldsymbol{\Gamma}_{m'})\boldsymbol{\Pi}\boldsymbol{\Gamma}_m^\top\|_F^2\end{aligned}$$

$\mathbf{T}^{(1)}_{mm'm''}=[\text{vec}(\underline{T}(1,:,:)),\ldots,\text{vec}(\underline{T}(K,:,:))]$, $\boldsymbol{\Gamma}_m^{(\text{prev})}$ denotes the estimate of $\boldsymbol{\Gamma}_m$ obtained by the previous solution of (3.27), ν is a positive scalar, and we have used (3.6). Here, $g_{N,m}$ contains all the terms of h_N that involve $\boldsymbol{\Gamma}_m$ with the additional term $(\nu/2)\|\boldsymbol{\Gamma}_m-\boldsymbol{\Gamma}_m^{(\text{prev})}\|_F^2$, which ensures convergence of the AO-ADMM iterations.

Detailed derivations of the ADMM iterations for solving (3.27) and (3.26) are provided in Appendix D, while the AO-ADMM is summarized in Alg. 3.1. The computational complexity of the entire AO-ADMM scheme is approximately $\mathcal{O}(I_T M^3 K^4)$, where I_T is the number of required iterations until convergence (see Appendix D.3). The entire unsupervised ensemble classification procedure is listed in Alg. 3.2.

3.2.5 Convergence and identifiability

Convergence of the entire AO-ADMM scheme for (3.25), follows readily from results in [62, Prop. 1], stated next for our setup.

Proposition 3.1. [62, Prop. 1] *Alg. 3.1 for $M \geq 3$, and $\nu > 0$ converges to a stationary point of* (3.25).

Having established the convergence of Alg. 3.1 to a stationary point of (3.25) using Prop. 3.1, the suitability of the estimates provided by Alg. 3.1 for the ensemble classification task needs to be assessed. As (3.25) involves joint tensor decompositions, under certain conditions the solutions $\{\hat{\mathbf{\Gamma}}_m\}, \hat{\boldsymbol{\pi}}$ of (3.25) will be, similar to the PARAFAC decomposition of Sec. 3.1.2, *essentially unique*. Thus, in order to assess the suitability of the estimates provided by Alg. 3.1 the conditions under which the model employed in (3.25) is identifiable have to be established. Luckily, identifiability claims for the present problem can be easily derived from recent results in joint PARAFAC factorization [70, 129].

Lemma 3.1. *Let $\{\mathbf{\Gamma}_m^*\}$, $\boldsymbol{\pi}^*$ be the optimal solutions of* (3.25)*, and $\{\hat{\mathbf{\Gamma}}_m\}$, $\hat{\boldsymbol{\pi}}$ the estimates provided by Alg. 3.1. If at least three $\{\mathbf{\Gamma}_m\}_{m=1}^M$ have full column rank, there exists a permutation matrix $\hat{\mathbf{P}}$ such that*

$$\hat{\mathbf{\Gamma}}_m \hat{\mathbf{P}} = \mathbf{\Gamma}_m^*, \quad m = 1, \ldots, M, \quad \hat{\mathbf{P}}^\top \hat{\boldsymbol{\pi}} = \boldsymbol{\pi}^*.$$

Lemma 3.1 essentially requires that at least three annotators respond differently to different classes, that is no two columns of at least three confusion matrices are colinear. Possibly more relaxed identifiability conditions could be derived using techniques mentioned in [129]. Unlike the tensor decomposition mentioned in Sec. 3.1.2, here we have no scaling ambiguity on the confusion matrices or prior probabilities. This is important because there are infinite scalings, but finite permutation matrices since K is finite. Under As2, $\hat{\mathbf{P}}$ can be easily obtained since the largest elements of each column of a confusion matrix must lie on the diagonal for the majority of annotators. Each $\hat{\mathbf{\Gamma}}_m$ can be multiplied by a permutation matrix $\hat{\mathbf{P}}_m$, such that the largest elements are located on the diagonal. The final $\hat{\mathbf{P}}$ can be derived as the most commonly occurring permutation matrix out of $\{\hat{\mathbf{P}}_m\}_{m=1}^M$.

Remark 3.1. While we relied on statistics of annotator responses up to order three, higher-order statistics can also be employed. Higher-order moments however, will increase the complexity of

the algorithm, as well as the number of data required to obtain reliable (low-variance) estimates.

Remark 3.2. Estimates of annotator confusion matrices $\{\hat{\mathbf{\Gamma}}_m\}$ and data labels $\{\hat{y}_n\}$, provided by Alg. 3.2, can be used to initialize the EM algorithm of Sec. 3.2.2 [31].

Remark 3.3. The orthogonal tensor decomposition used by [66, 162] is a special case of the PARAFAC decomposition employed in this work.

Remark 3.4. When $\boldsymbol{\pi}$ is known, (3.26) can be skipped, and correspondingly steps 8 and 9 of Alg. 3.1.

3.2.6 Reducing complexity

When K and M are large Alg. 3.1 may require long computational time to converge. Our idea in this case is to split the annotators into L groups, and solve (3.25) for each group. For simplicity of exposition, consider non-overlapping groups, each with $M_\ell \geq 3$ annotators ($\sum_{\ell=1}^{L} M_\ell = M$). Let $\boldsymbol{\mu}_m^{(\ell)}, \mathbf{S}_{mm'}^{(\ell)}$ and $\underline{T}_{mm'm''}^{(\ell)}$ denote the sample statistics for annotators in group ℓ, and $\{\mathbf{\Gamma}_m^{(\ell)}\}_{m=1}^{M_\ell}$ the confusion matrices in group ℓ.

For each group $\ell \in \{1, \ldots, L\}$ confusion matrices $\{\hat{\mathbf{\Gamma}}_m^{(\ell)}\}_{m=1}^{M_\ell}$ and prior probabilities $\boldsymbol{\pi}^{(\ell)}$ are estimated by solving a smaller version of (3.25), namely

$$\begin{aligned} \min_{\substack{\boldsymbol{\pi}^{(\ell)} \\ \{\mathbf{\Gamma}_m^{(\ell)}\}_{m=1}^{M}}} \quad & h_N^{(\ell)}(\{\mathbf{\Gamma}_m^{(\ell)}\}_{m=1}^{M}, \boldsymbol{\pi}^{(\ell)}) \\ \text{s.to} \quad & \mathbf{\Gamma}_m^{(\ell)} \geq \mathbf{0}, \quad \mathbf{1}^\top \mathbf{\Gamma}_m^{(\ell)} = \mathbf{1}^\top, \quad m = 1, \ldots, M_\ell \\ & \boldsymbol{\pi}^{(\ell)} \geq \mathbf{0}, \quad \mathbf{1}^\top \boldsymbol{\pi}^{(\ell)} = 1 \end{aligned} \tag{3.28}$$

where

$$\begin{aligned} h_N^{(\ell)}(\{\mathbf{\Gamma}_m\}, \boldsymbol{\pi}) := & \frac{1}{2} \sum_{m=1}^{M_\ell} \|\boldsymbol{\mu}_m^{(\ell)} - \mathbf{\Gamma}_m \boldsymbol{\pi}\|_2^2 \\ & + \frac{1}{2} \sum_{\substack{m=1 \\ m'>m}}^{M_\ell} \|\mathbf{S}_{mm'}^{(\ell)} - \mathbf{\Gamma}_m \mathbf{\Pi} \mathbf{\Gamma}_{m'}^\top\|_F^2 \\ & + \frac{1}{2} \sum_{\substack{m=1 \\ m'>m \\ m''>m'}}^{M} \|\underline{T}_{mm'm''}^{(\ell)} - [[\mathbf{\Gamma}_m \mathbf{\Pi}, \mathbf{\Gamma}_{m'}, \mathbf{\Gamma}_{m''}]]_K\|_F^2. \end{aligned}$$

Upon solving (3.28) for all L groups, estimates of $\{\mathbf{\Gamma}_m\}_{m=1}^M$ are readily obtained, since we have assumed non-overlapping groups. A final estimate of the prior probabilities $\boldsymbol{\pi}$ can be obtained by averaging the L estimates $\{\boldsymbol{\pi}^\ell\}_{\ell=1}^L$.

As (3.28) incurs a complexity of $\mathcal{O}(IM_\ell^3K^3)$, the worst-case complexity of this approach is $\mathcal{O}(I_MK^3\sum_{\ell=1}^L M_\ell^3)$, where I_M is the largest number of iterations required to converge among all L groups. Since $M^3=(\sum_{\ell=1}^L M_\ell)^3>\sum_{\ell=1}^L M_\ell^3$ this approach reduces the computational and memory overhead significantly compared to Alg. 3.1. Note however, that this method is expected to perform well when As1 and As2, as well as the conditions outlined in Lemma 3.1 are satisfied for all L groups of annotators, and N is sufficiently large. The effectiveness of this complexity reduction scheme is tested in Sec. 2.5.

3.2.7 Application to crowdsourcing

While crowdsourced classification is a task related to ensemble classification, it presents additional challenges. So far it has been implicitly assumed that all annotators provide labels for all $\{x_n\}_{n=1}^N$. In the crowdsourcing setup however, an annotator m could provide labels just for a subset of $N_m<N$ data.

Next, we outline a computationally attractive approach, that takes into account only the available annotator responses. If an annotator m does not provide a label for a datum, his/her response is $f_m(x)=0$ or $\mathbf{f}_m(x)=\mathbf{0}$ in vector format. Let $J_m(x_n)$ be an indicator function that takes the value 1 when annotator m provides a label for x_n, and 0 when $f_m(x_n)=0$. To account for such cases, the annotator sample statistics become

$$\boldsymbol{\mu}_m=\frac{1}{\sum_{n=1}^N J_m(x_n)}\sum_{n=1}^N J_m(x_n)\mathbf{f}_m(x_n) \tag{3.29a}$$

$$\mathbf{S}_{mm'}=\frac{\sum_{n=1}^N J_m(x_n)J_{m'}(x_n)\mathbf{f}_m(\boldsymbol{x}_n)\mathbf{f}_{m'}^\top(\boldsymbol{x}_n)}{\sum_{n=1}^N J_m(x_n)J_{m'}(x_n)} \tag{3.29b}$$

$$\underline{T}_{mm'm''}=\frac{\sum_n J_m(x_n)J_{m'}(x_n)J_{m''}(x_n)\mathbf{f}_m(\boldsymbol{x}_n)\circ\mathbf{f}_{m'}(\boldsymbol{x}_n)\circ\mathbf{f}_{m''}(\boldsymbol{x}_n)}{\sum_{n=1}^N J_m(x_n)J_{m'}(x_n)J_{m''}(x_n)}. \tag{3.29c}$$

Upon computing the modified sample statistics of (3.29), we can obtain estimates of the confusion

matrices and prior probabilities in the crowdsourcing setup, via Alg. 3.1. Finally, the MAP classifier in (3.24) has to be modified as follows

$$\hat{y}_{\mathrm{MAP}}(x) = \underset{k\in\{1,\ldots,K\}}{\arg\max} \log \hat{\pi}_k + \sum_{m=1}^{M} J_m(x) \log \hat{\Gamma}_m(f_m(x), k) \tag{3.30}$$

to take into account only the available annotator responses for each x.

Having completed the algorithmic aspects of our approach, we proceed with performance analysis.

3.3 Performance Analysis

In this section, performance of the proposed method will be quantified analytically. First, the consistency of the estimates provided by Alg. 3.1 as $N \to \infty$ will be established, followed by a performance analysis for the MAP classifier of Sec. 3.2.1.

3.3.1 Consistency of Alg. 3.1 estimates

As $N \to \infty$, the sample statistics in (3.21), (3.22), and (3.23) approach their ensemble counterparts, and we end up with the following optimization problem for extracting annotator confusion matrices and prior probabilities

$$\begin{aligned} &\min_{\substack{\boldsymbol{\pi} \\ \{\boldsymbol{\Gamma}_m\}_{m=1}^M}} \quad h_\infty(\{\boldsymbol{\Gamma}_m\}_{m=1}^M, \boldsymbol{\pi}) \\ &\text{s.to} \qquad \boldsymbol{\Gamma}_m \in \mathcal{C}, \quad m = 1, \ldots, M, \quad \boldsymbol{\pi} \in \mathcal{C}_p. \end{aligned} \tag{3.31}$$

Clearly, the optimal solutions to (3.31) are the true confusion matrices and prior probabilities. As N increases, it is desirable to show that the solutions obtained from Alg. 3.1 converge to the true confusion matrices and prior probabilities. To this end, techniques from statistical learning theory and stochastic optimization will be employed [123, 142]. Specifically, we will establish the uniform convergence of h_N to h_∞, which implies the consistency of the solutions. Define the distance between two sets $\mathcal{A}, \mathcal{B} \subseteq \mathbb{R}^q$, for some $q > 0$, as $D(\mathcal{A}, \mathcal{B}) = \sup_{x\in\mathcal{A}}\{\inf_{y\in\mathcal{B}} \|x-y\|_2\}$. The following theorem shows that as N increases, the solutions of (3.25) approach those of (3.31).

Theorem 3.1. *If $\mathcal{S}_*$ and $\mathcal{S}_N$ denote the sets of solutions of problems* (3.31) *and* (3.25)*, respectively, then $D(\mathcal{S}_N, \mathcal{S}_*) \to 0$, as $N \to \infty$ almost surely.*

Under As2 and the conditions outlined in Lemma 3.1, Alg. 3.1 can recover the true solutions of (3.25) or (3.31). Then, by Thm. 3.1 we know that as $N \to \infty$ the solutions of (3.25) converge to the solutions of (3.31), which together with the result of Lemma 3.1 implies the statistical consistency of the solutions of Alg. 3.1. As a result, the estimates $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$, and $\hat{\boldsymbol{\pi}}$ from Alg. 3.1 will converge to their true values w.p. 1 as $N \to \infty$.

3.3.2 MAP classifier performance

With consistency of the confusion matrix and prior probability estimates established, the performance of the final component of the proposed algorithm has to be studied. The behavior of the MAP classifier of Sec. 3.2.1 can be quantified in terms of its average probability of error

$$\mathrm{P_e} = \sum_{k=1}^{K} \Pr(\hat{y}_{\mathrm{MAP}} = k' \neq k | Y = k) \Pr(Y = k)$$

Here, a well-known asymptotic result for distributed binary detection under the MAP detector [141] is extended to the multiclass case.

Theorem 3.2. *Under As1, and given $\{\mathbf{\Gamma}_m\}_{m=1}^M$ and $\boldsymbol{\pi}$, there exist constants $\alpha > 0, \beta > 0$ such that the MAP classifier of Sec. 3.2.1 satisfies*

$$\mathrm{P_e} \leq \alpha e^{-\beta M}.$$

In words, Theorem 3.2 suggests that when accurate estimates of $\{\mathbf{\Gamma}_m\}_{m=1}^M$ and $\boldsymbol{\pi}$ are available, the error rate decreases at an exponential rate with the number of annotators M.

In order to validate our theoretical results and evaluate the performance of the proposed scheme, the following section 3.4 presents numerical tests with synthetic and real data.

3.4 Numerical Tests

For $K \geq 2$, Alg. 3.2, using both MAP and ML criteria in step 3, (denoted as *Alg. 3.2 MAP* and *Alg. 3.2 ML* respectively) is compared to majority voting, the algorithm of [71] (denoted as *KOS*), and the EM algorithm initialized both with majority voting and with the spectral method of [162] (denoted as *EM + MV* and *EM + Spectral*, respectively). For $K = 2$, Alg. 3.2 is also compared to the binary ensemble learning methods of [65], [16] and [30], denoted as *SML*, *TE* and *EigenRatio*, respectively. For synthetic data, the performance of "oracle" estimators, that is MAP/ML classifiers with true confusion matrices of the annotators, and the true class priors, is also evaluated for benchmarking purposes. The metric utilized in all experiments is the classification error rate (ER), defined as the percentage of misclassified data,

$$ER = \frac{\text{\# of misclassified data}}{N} \times 100\%,$$

where $ER = 100\%$ indicates that all N data have been misclassified, and $ER = 0\%$ indicates perfect classification accuracy. For synthetic data, the average confusion matrix and prior probability estimation error is also evaluated

$$\bar{\varepsilon}_{CM} := \frac{1}{M} \sum_{m=1}^{M} \frac{\|\boldsymbol{\Gamma}_m - \hat{\boldsymbol{\Gamma}}_m\|_1}{\|\boldsymbol{\Gamma}_m\|_1} = \frac{1}{M} \sum_{m=1}^{M} \|\boldsymbol{\Gamma}_m - \hat{\boldsymbol{\Gamma}}_m\|_1$$

$$\bar{\varepsilon}_{\boldsymbol{\pi}} := \|\boldsymbol{\pi} - \hat{\boldsymbol{\pi}}\|_1.$$

All results represent averages over 10 independent Monte Carlo runs, using MATLAB [94]. In all experiments, the parameters λ and ν of Alg. 3.1 are set as suggested in [62, 117]. Vertical lines in some figures indicate standard deviation. For some experiments, classification times (in seconds) required by the ensemble algorithms are also reported. Note that classification times for majority voting and oracle estimators are not reported as the time required by these methods is negligible compared to the rest of the algorithms.

3.4.1 Synthetic data

For the synthetic data tests, N ground-truth labels $\{y_n\}_{n=1}^{N}$, each corresponding to one out of K possible classes, were generated i.i.d. at random according to $\boldsymbol{\pi}$, that is $y_n \sim \boldsymbol{\pi}$, for $n = 1, \ldots, N$. Afterwards, $\{\boldsymbol{\Gamma}_m\}_{m=1}^{M}$ were generated at random, such that $\boldsymbol{\Gamma}_m \in \mathcal{C}$, for all

$m = 1, \ldots, M$, and annotators are better than random, as per As2. Then annotators' responses were generated as follows: if $y_n = k$, then the response of annotator m will be generated randomly according to the k-th column of its confusion matrix, $\boldsymbol{\gamma}_{m,k}$ [cf. Sec. 3.1], that is $f_m(x_n) \sim \boldsymbol{\gamma}_{m,k}$.

Tab. 3.1 lists the classification ER of different algorithms, for a synthetic dataset with $K = 2$ classes with prior probabilities $\boldsymbol{\pi} = [0.9003, 0.0997]^\top$, and $M = 10$ annotators. Tab. 3.2 lists the results for a similar experiment, with $K = 2$ classes, priors $\boldsymbol{\pi} = [0.5856, 0.4144]^\top$, and $M = 10$ annotators, while Tab. 3.3 shows the clustering time required by all algorithms. Note that when the class probabilities are similar, the ML and MAP classifiers perform comparably as expected. Furthermore, majority voting gives good results for a reduced number of instances N. Fig. 3.3 depicts the average estimation errors for the confusion matrices and prior probabilities in the two aforementioned experiments. Clearly, as N increases, the proposed classifiers approach the performance of the oracle ones, and as suggested by Thm. 3.1, the estimation error for the confusion matrices and prior probabilities approaches 0.

The next synthetic data experiment investigates how the proposed method performs when presented with multiclass data. Furthermore, to showcase that accurate estimation of $\boldsymbol{\pi}$ is beneficial, we also compare against Alg. 3.2 with $\boldsymbol{\pi}$ fixed to the uniform distribution, i.e. $\boldsymbol{\pi} = \mathbf{1}/K$ (denoted as *Alg. 3.2 - fixed* $\boldsymbol{\pi}$.) Fig. 3.4 shows the simulation results for a synthetic dataset with $K = 5$ classes, prior probabilities $\boldsymbol{\pi} = [0.2404, 0.2679, 0.0731, 0.1950, 0.2236]^\top$, and $M = 10$ annotators, while Fig. 3.5 shows the simulation results for a synthetic dataset with $K = 7$ classes, priors $\boldsymbol{\pi} = [0.2347, 0.0230, 0.0705, 0.1477, 0.2659, 0.0043, 0.2539]^\top$ and $M = 10$ annotators. Tabs. 3.4 and 3.5 show classification times for the $K = 5$ and $K = 7$ experiments, respectively. Fig. 3.6 shows the average estimation errors for the confusion matrices and prior probabilities in the two aforementioned multiclass experiments. Note that for $K = 5$ for small values of N and $K = 7$ the EM+Spectral approach of [162] suffers from numerical issues during the tensor whitening procedure, which explains its worst classification ER and slow runtimes. Here, the proposed approaches exhibit similar behavior to the binary case, as expected from Thm. 3.1; *as the number of data increases, their performance approaches the clairvoyant "oracle" estimators, and the estimation accuracy of the confusion matrices and prior probabilities increases.* In addition, our methods outperform the competing alternatives for almost all values of N. Here we also see that running Alg. 3.2 with fixed $\boldsymbol{\pi} = \mathbf{1}/K$ produces lower quality estimates than Alg. 3.2 that solves for $\boldsymbol{\pi}$. Specifically, Alg. 3.2 with fixed $\boldsymbol{\pi}$ performs similarly

to the EM algorithm when initialized with majority voting.

Next, we evaluate how the number of annotators M affects the classification ER, for fixed $N = 10^6$. Fig. 3.7 depicts an experiment for $K = 3$ classes with priors $\boldsymbol{\pi} = [0.2318, 0.4713, 0.2969]^\top$, while Fig. 3.8 shows an experiment for $K = 5$ classes with priors $\boldsymbol{\pi} = [0.3596, 0.1553, 0.1229, 0.3258, 0.0364]^\top$. Tabs. 3.6 and 3.7 list classification times for the $K = 3$ and $K = 5$ experiments, respectively. Fig. 3.9 plots the results of an experiment with $K = 5$ classes with the same priors as those in Fig. 3.8 and $N = 5,000$ data, for varying number of annotators. The average estimation error for the confusion matrices and prior probabilities, for the aforementioned tests, is shown in Fig. 3.10. As expected from Thm. 3.2, *the classification ER decreases as the number of annotators increase*, for all methods considered. In addition, our proposed algorithm outperforms the competing alternatives for all values of M. Furthermore, the results of Fig. 3.9 indicate that *when the number of data is small, increasing the number of annotators provides a boost to the classification performance.* Fig. 3.10 shows another interesting feature: *as the number of annotators increases the estimation accuracy of* $\{\boldsymbol{\Gamma}_m\}$ *and* $\boldsymbol{\pi}$ *also increases.*

The following experiment evaluates the effectiveness of the complexity reduction scheme of Sec. 3.2.6, for a dataset with $M = 30$ annotators with $K = 3$ classes with priors $\boldsymbol{\pi} = [0.3096, 0.3416, 0.3488]^\top$, and a varying number of data N. Annotators are split into $L = \{1, 2, 4, 5\}$ non-overlapping groups. Fig. 3.11 shows the classifcation ER and time (in seconds) required for the ensemble classification task, for different group sizes. When N is large we observe similar ER for all L, however larger number of groups require significantly less time than $L = 1$.

In all aforementioned experiments, all annotators were generated to be better than random. The next experiment, investigates the effect of *adversarial annotators*, that is annotators for who the largest values of the confusion matrix are not located on its diagonal. Let α denote the percentage of adversarial annotators. Fig. 3.12 shows the classification ER on a synthetic dataset with $K = 3$, $N = 10^6$, $\boldsymbol{\pi} = [0.31, 0.34, 0.35]^\top$ and $M = 10$ annotators, for varying α. While all approaches, with the exception of majority voting, seem to be robust to a small number of adversarial annotators, Alg. 3.2 can handle values of α of up to 50%, which speaks for the potential of the novel approach in adversarial learning setups [15, 26].

Algorithm	$N = 100$	$N = 1000$	$N = 10^4$	$N = 10^5$
Majority Voting	6.3	7.08	7.04	7.13
KOS	27.70	33.33	32.21	32.53
EigenRatio	6.30	5.75	5.69	5.64
TE	4.20	4.91	4.61	4.67
SML	15.80	11.38	11.82	12.26
EM + MV	21.2	27.67	26.50	27.01
EM + Spectral	17.7	27.72	26.50	27.01
Alg. 3.2 ML	6.30	**2.70**	**1.97**	**1.87**
Alg. 3.2 MAP	**2.40**	**1.40**	**1.13**	**1.11**
Oracle ML	1.6	2.05	1.81	1.86
Oracle MAP	1.1	1.31	1.11	1.11

Table 3.1: Classification ER for a synthetic dataset with $K = 2$, prior probabilities $\boldsymbol{\pi} = [0.9003, 0.0997]^\top$ and $M = 10$ annotators.

3.4.2 Real data

Further tests were conducted using real datasets. In this case, in addition to other ensemble learning algorithms, the proposed methods are also compared to the single best annotator, that is the classifier that exhibited the highest accuracy. For all experiments, a collection of $M = 15$ classification algorithms from MATLAB's machine learning toolbox were trained, each on a different randomly selected subset of the dataset. Afterwards, the algorithms provided labels for all data in each dataset. The classification algorithms considered were k-nearest neighbor classifiers, for varying number of neighbors k and different distance measures; support vector machine classifiers, utilizing different kernels; and decision trees with varying depth. The real datasets considered are the MNIST dataset [78], and 5 UCI datasets [83]: the CoverType database, the PokerHand dataset, the Connect-4 dataset, the Magic dataset and the Dota 2 dataset. MNIST contains $N = 70,000$ 28×28 images of handwritten digits, each belonging to one of $K = 10$ classes (one per digit). For this dataset, each classification algorithm was trained on subsets of $2,000$ instances. The CoverType dataset consists of $N = 581,012$ data belonging to $K = 7$ classes. Each cluster corresponds to a different forest cover type. Data are vectors of dimension $D = 54$ that contain cartographic variables, such as soil type, elevation, hillshade etc. Here, each classification algorithm was trained on a subset of $1,000$ instances. The PokerHand database contains $N = 10^6$ data belonging to $K = 10$ classes. Each datum is a 5-card hand

Algorithm	$N = 100$	$N = 1000$	$N = 10^4$	$N = 10^5$
Majority Voting	8.10	8.27	8.27	8.19
KOS	8.30	6.46	6.65	6.58
EigenRatio	7.40	6.35	6.39	6.21
TE	10.20	6.04	6.35	6.20
SML	13.10	8.47	**4.66**	**4.61**
EM + MV	6.60	5.15	4.93	4.87
EM + Spectral	6.60	5.15	4.93	4.87
Alg. 3.2 ML	**6.50**	**4.86**	**4.66**	**4.61**
Alg. 3.2 MAP	**6.20**	**4.85**	**4.59**	**4.51**
Oracle ML	4.10	4.86	4.66	4.61
Oracle MAP	3.90	4.81	4.58	4.50

Table 3.2: Classification ER for a synthetic dataset with $K = 2$, prior probabilities $\boldsymbol{\pi} = [0.5856, 0.4144]^\top$ and $M = 10$ annotators.

drawn from a deck of 52 cards, with each card being described by its rank and suit (spades, hearts, diamonds, and clubs). Each class represents a valid Poker hand. For this experiment the 3 most prevalent classes are considered. Here, each classification algorithm was trained on a subset of $10,000$ instances. Connect-4 contains $N = 67,557$ vectors of size 42×1, each representing the possible positions in a connect-4 game. These vectors belong to one of $K = 3$ classes, indicating whether the first player is in a position to win, lose, or, tie the game. Here, each classification algorithm was trained on a subset of 300 instances. The Magic dataset contains $N = 19,020$ data captured by ground-based atmospheric Cherenkov gamma-ray detector. The

Algorithm	$N = 100$	$N = 1000$	$N = 10^4$	$N = 10^5$
KOS	0.013	0.004	0.005	0.05
EigenRatio	0.003	0.002	0.005	0.03
TE	0.003	0.001	0.012	0.10
SML	0.04	0.09	0.76	11.98
EM + MV	0.01	0.02	0.12	1.47
EM + Spectral	1.48	1.55	1.58	3.00
Alg. 3.2	1.82	2.32	2.05	3.01

Table 3.3: Classification time (in seconds) for a synthetic dataset with $K = 2$, prior probabilities $\boldsymbol{\pi} = [0.5856, 0.4144]^\top$ and $M = 10$ annotators.

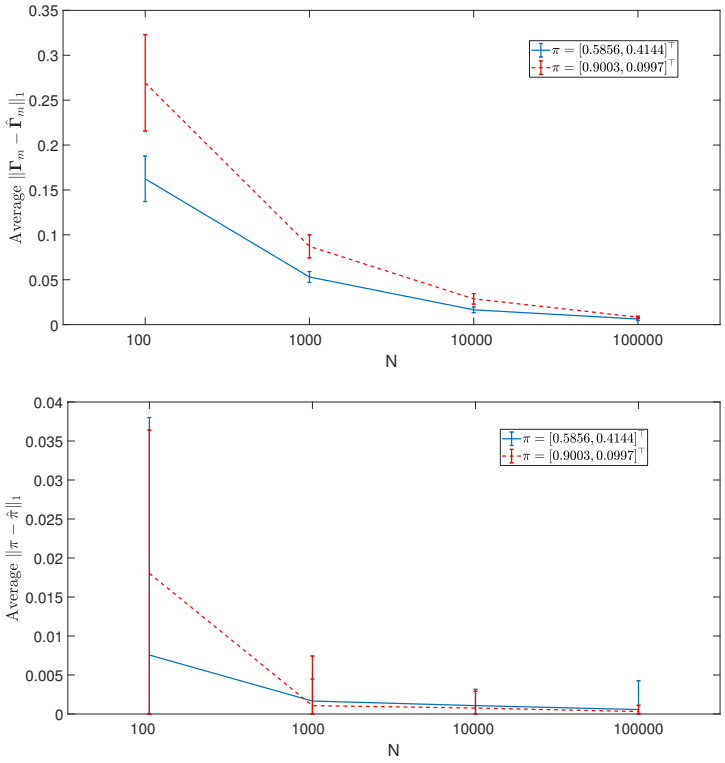


Figure 3.3: Average estimation errors of confusion matrices (top); and prior probabilities (bottom), for two synthetic datasets with $K = 2$ and $M = 10$ annotators

dataset contains $K = 2$ classes, each indicating the presence or abscence of Gamma rays. For this dataset, each classification algorithm was trained on subsets of 100 instances. The Dota 2 dataset contains $N = 102,944$ data, corresponding to different Dota 2 games played, between two teams of 5 players. The dataset is split into $K = 2$ classes, corresponding to the team that won the game. Each datum consists of the starting parameters of each game, such as the game type (ranked or amateur) and which heroes were chosen from the players. Finally, for this dataset, each classification algorithm was trained on subsets of $5,000$ instances.

Table 3.8 lists classification ER results for the real data experiments. For most datasets, the proposed approaches outperform the competing alternatives, as well as the single-best classifier. For the MNIST dataset the EM methods of [162] outperform our approaches. Nevertheless, Alg. 3.1 comes very close to the performance of the EM schemes and if the confusion matrix estimates $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$ of Alg. 3.2 are refined using EM, we also reach a classification ER of 6.23%.

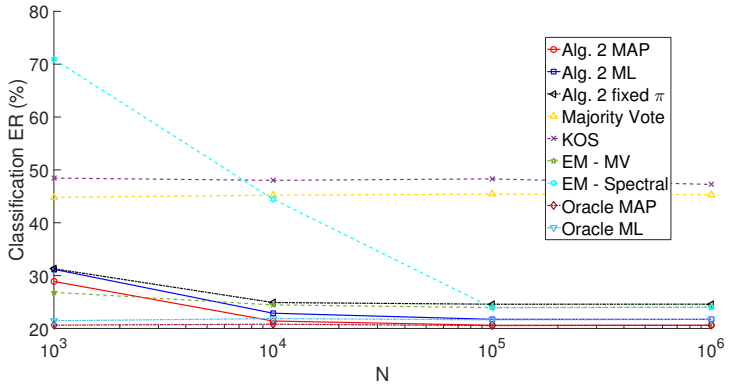


Figure 3.4: Classification ER for a synthetic dataset with $K = 5$ classes, priors $\boldsymbol{\pi} = [0.2404, 0.2679, 0.0731, 0.1950, 0.2236]^\top$ and $M = 10$ annotators.

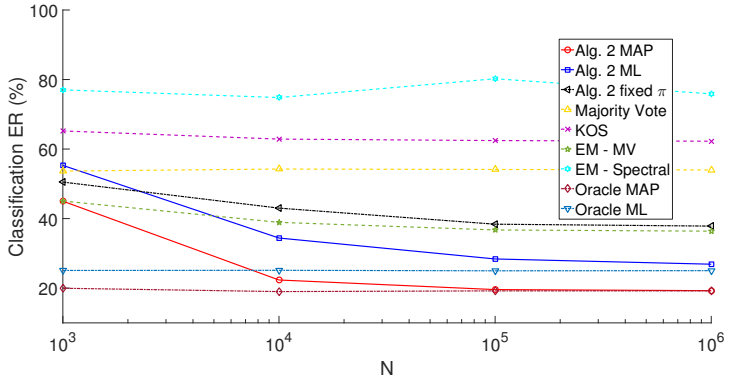


Figure 3.5: Classification ER for a synthetic dataset with $K = 7$ classes, priors $\boldsymbol{\pi} = [0.2347, 0.0230, 0.0705, 0.1477, 0.2659, 0.0043, 0.2539]^\top$ and $M = 10$ annotators.

3.4.3 Crowdsourcing data

In this section, the proposed scheme of Sec. 3.2.7 is evaluated on crowdsourcing data. The datasets considered are the Adult dataset [125], the TREC dataset [22] and the Bird dataset [150]. In most datasets, only a small set of ground-truth labels was available, and the performance of each method was evaluated on this set.

For the Adult dataset, annotators were tasked with classifying $N = 11,028$ websites into $K = 4$ different classes, using Amazon's Mechanical Turk [75]. The 4 classes correspond to different levels of adult content of a website. To maintain reasonable computational complexity, we only considered annotators that had given labels for all 4 classes and provided labels for more than 370 websites.

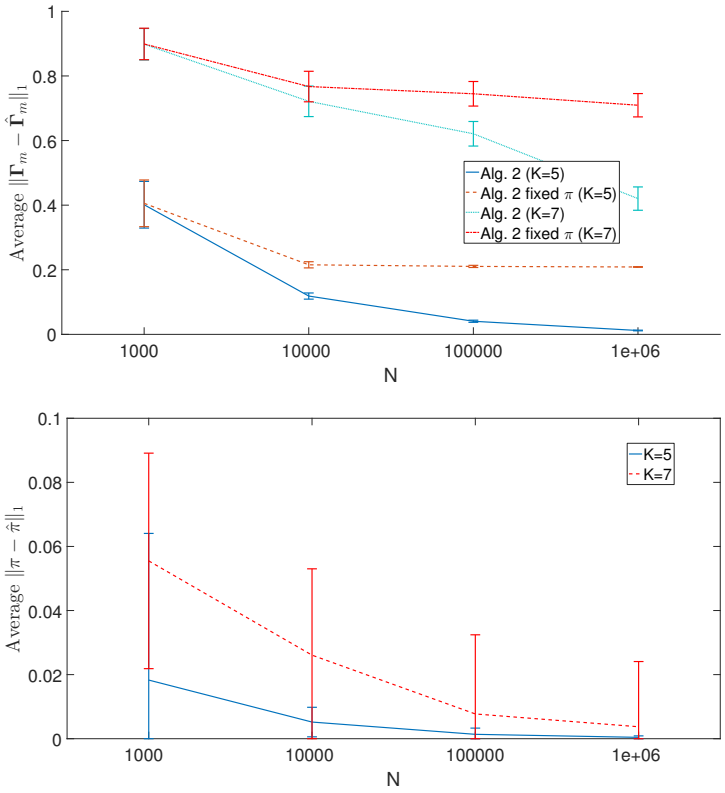


Figure 3.6: Average estimation errors of confusion matrices (top); and prior probabilities (bottom) for two synthetic datasets with $K = 5$ and $K = 7$ classes and $M = 10$ annotators

For the TREC dataset, annotators from Amazon's Mechanical Turk [75] were tasked with classifying $N = 19,033$ websites into $K = 2$ classes: "relevant" or "irrelevant" to some search queries. Again, to maintain reasonable computational complexity for our approach, we only considered annotators that had given labels for both classes and provided labels for more than 708 websites.

For the bird dataset, annotators from Amazon's Mechanical Turk were tasked with classifying $N = 108$ images of birds into $K = 2$ classes: "Indigo Bunting" or "Blue Grosbeak".

Table 3.9 lists classification ER for the two crowdsourcing experiments. The column "Labels" denotes the number of ground-truth labels available. As with the previous experiments, our approach exhibits lower classification ER than the competing alternatives, in both multiclass and binary classification settings.

Algorithm	$N = 1000$	$N = 10^4$	$N = 10^5$	$N = 10^6$
KOS	0.016	0.02	0.17	2.03
EM + MV	0.04	0.27	3.43	37.27
EM + Spectral	119.35	124.94	119.35	160.54
Alg. 3.2	28.27	40.23	36.08	47.17
Alg. 3.2 fixed $\boldsymbol{\pi}$	13.34	6.23	6.11	18.16

Table 3.4: Classification time (in seconds) for a synthetic dataset with $K = 5$ classes, priors $\boldsymbol{\pi} = [0.2404, 0.2679, 0.0731, 0.1950, 0.2236]^\top$ and $M = 10$ annotators.

Algorithm	$N = 1000$	$N = 10^4$	$N = 10^5$	$N = 10^6$
KOS	0.017	0.025	0.23	2.83
EM + MV	0.05	0.30	4.80	48.87
EM + Spectral	619.61	616.47	621.30	676.95
Alg. 3.2	46.19	52.66	54.50	69.99
Alg. 3.2 fixed $\boldsymbol{\pi}$	34.94	38.88	39.11	40.17

Table 3.5: Classification time (in seconds) for a synthetic dataset with $K = 7$ classes, priors $\boldsymbol{\pi} = [0.2347, 0.0230, 0.0705, 0.1477, 0.2659, 0.0043, 0.2539]^\top$ and $M = 10$ annotators.

3.5 Conclusions

This chapter introduced a novel approach to blind ensemble and crowdsourced classification that relies solely on annotator responses to assess their quality and combine their answers. Compact expressions of annotator moments, based on PARAFAC tensor decompositions were derived, and a novel moment matching scheme was developed using AO-ADMM. The performance of the novel algorithm was evaluated on real and synthetic data.

Algorithm	$M = 5$	$M = 10$	$M = 20$	$M = 30$
KOS	0.44	0.96	4.13	5.29
EM + MV	11.48	21.67	41.88	62.19
EM + Spectral	21.92	32.77	53.88	75.24
Alg. 3.2	4.85	15.43	83.73	271.71

Table 3.6: Classification time (in seconds) for a synthetic dataset with $K = 3$ classes, priors $\boldsymbol{\pi} = [0.2318, 0.4713, 0.2969]^\top$ and $N = 10^6$ data.

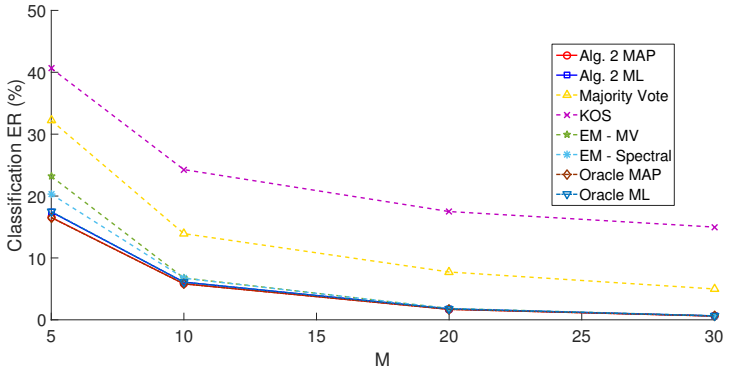


Figure 3.7: Classification ER for a synthetic dataset with $K = 3$ classes, priors $\boldsymbol{\pi} = [0.2318, 0.4713, 0.2969]^\top$ and $N = 10^6$ data.

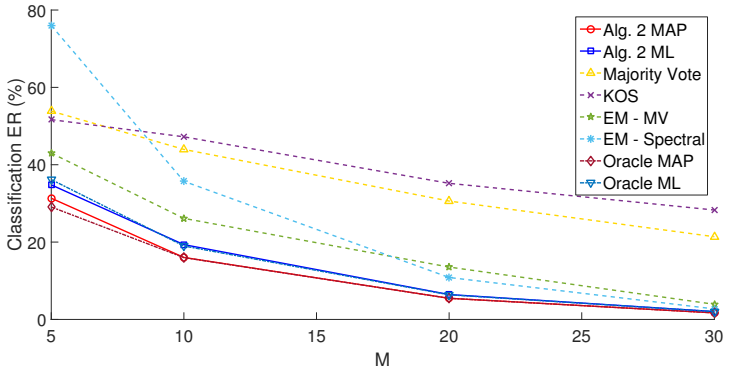


Figure 3.8: Classification ER for a synthetic dataset with $K = 5$ classes, priors $\boldsymbol{\pi} = [0.3596, 0.1553, 0.1229, 0.3258, 0.0364]^\top$ and $N = 10^6$ data.

Algorithm	$M = 5$	$M = 10$	$M = 20$	$M = 30$
KOS	0.85	1.90	8.99	11.11
EM + MV	18.47	34.68	67.14	99.82
EM + Spectral	136.30	153.35	186.99	221.50
Alg. 3.2	12.92	28.89	150.33	471.22

Table 3.7: Classification time (in seconds) for a synthetic dataset with $K = 5$ classes, priors $\boldsymbol{\pi} = [0.3596, 0.1553, 0.1229, 0.3258, 0.0364]^\top$ and $N = 10^6$ data.

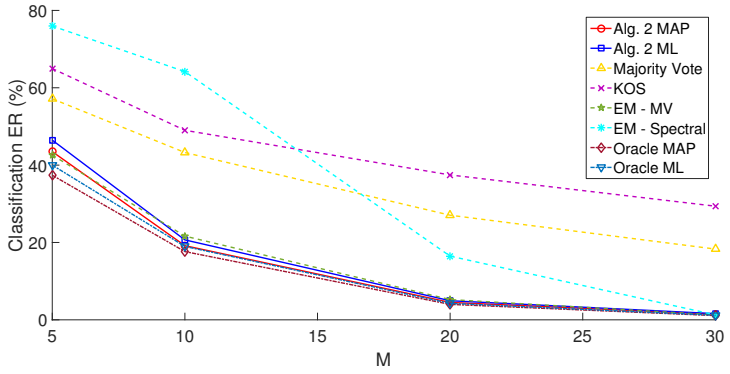


Figure 3.9: Classification ER for a synthetic dataset with $K = 5$ classes, priors $\boldsymbol{\pi} = [0.3596, 0.1553, 0.1229, 0.3258, 0.0364]^\top$ and $N = 5,000$ data.

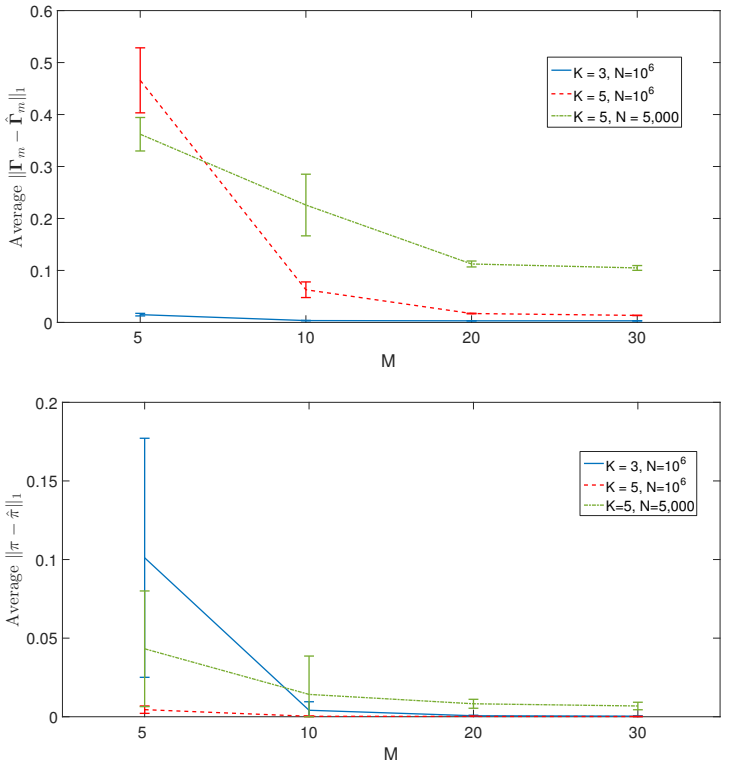


Figure 3.10: Average estimation errors of confusion matrices (top); and prior probabilities (bottom) for two synthetic datasets with $K = 3$ and $K = 5$ classes and $N = 10^6$ data, and a synthetic dataset with $K = 5$ classes and $N = 5,000$ data.

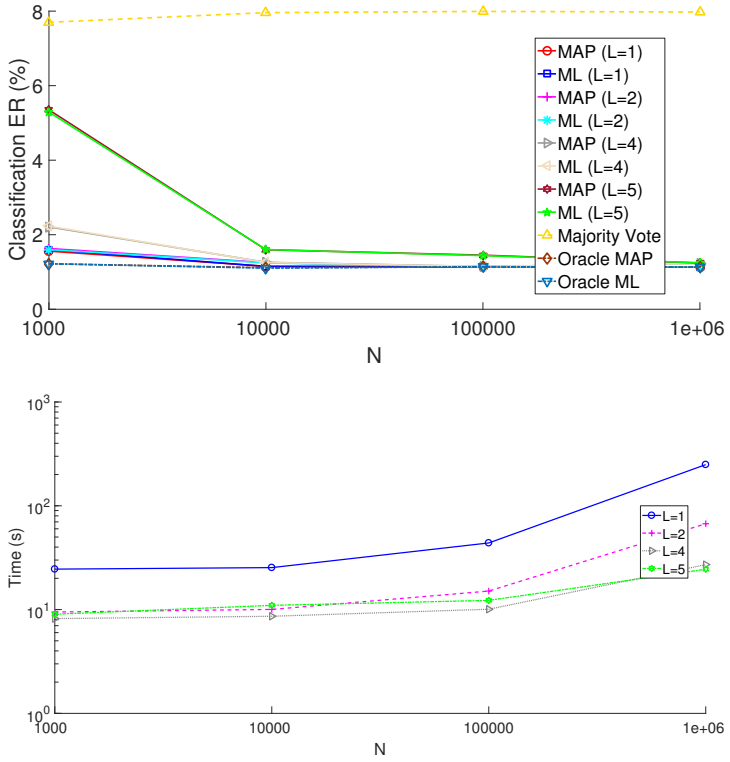


Figure 3.11: Classification ER (top); and time (in seconds) (bottom) for a synthetic dataset with $K = 3$ classes, priors $\boldsymbol{\pi} = [0.3096, 0.3416, 0.3488]^\top$, $M = 30$ annotators for varying number of data N and annotator groups L.

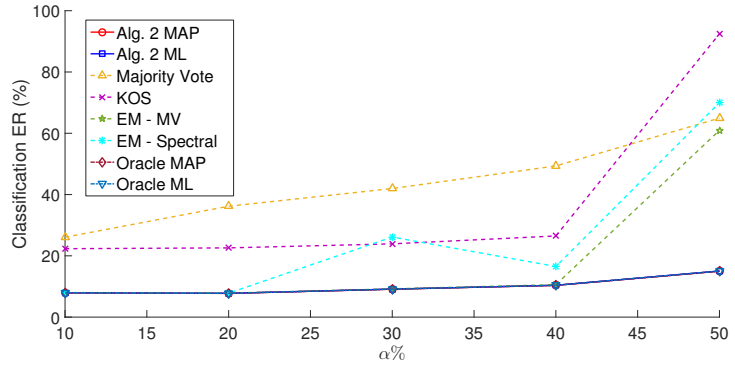


Figure 3.12: Classification ER for a synthetic dataset with $K = 3$ classes, priors $\boldsymbol{\pi} = [0.31, 0.34, 0.35]^\top$, $N = 10^6$, $M = 10$ annotators and varying percentage of adversarial annotators α.

Dataset	K	Single best	MV	EigenRatio	TE	SML	KOS	EM + MV	EM + Spectral	Alg. 3.2 MAP	Alg. 3.2 ML
MNIST	10	7.29	7.0986	-	-	-	9.84	**6.23**	**6.23**	6.3986	6.3843
CoverType	7	29.89	28.642	-	-	-	31.13	58.68	95.62	**28.574**	28.913
PokerHand	3	41.95	43.365	-	-	-	49.62	53.62	78.38	**39.436**	**39.339**
Connect-4	3	29.17	31.636	-	-	-	32.33	44.27	61.20	**26.176**	**26.86**
Magic	2	21.32	21.73	26.25	26.28	21.27	21.29	21.17	21.14	**20.77**	**20.98**
Dota 2	2	41.27	42.174	45.55	45.75	40.568	40.59	40.80	59.19	**40.497**	**40.549**

Table 3.8: Classification ER for Real data experiments with $M = 15$.

Dataset	N	K	M	Labels	MV	EigenRatio	TE	SML	KOS	EM + MV	EM + Spectral	Alg. 3.2 MAP	Alg. 3.2 ML
Adult	$11,028$	4	38	347	36.023	-	-	-	80.98	40.63	38.90	$\mathbf{33.429}$	$\mathbf{34.87}$
TREC	$19,033$	2	23	$2,275$	50.002	43.34	48.97	48.44	54.68	56.04	40.62	$\mathbf{37.846}$	$\mathbf{39.824}$
Bird	108	2	39	108	24.07	27.78	17.59	11.11	11.11	11.11	$\mathbf{10.19}$	$\mathbf{10.19}$	$\mathbf{10.19}$

Table 3.9: Classification ER for crowdsourcing data experiments.

Chapter 4

Blind Ensemble Classification for Dependent Data

While the assumption that data are iid is convenient and often leads to elegant algorithms, in many cases it is violated. Examples of such cases are data that form a sequence, e.g. data arising from natural language processing tasks, or graph data such as citation networks. The present chapter builds upon the algorithms and results of the previous chapter and shows how information regarding data dependencies can be incorporated in the blind ensemble classification task and enhance its performance.

4.1 Prior work

Most works have attacked the blind ensemble classification problem under the assumption that data are independent and identically distributed (i.i.d.), see Chap. 3.1.1. Recent works advocate blind ensemble approaches for sequential data. [119] proposed a method for aggregating annotator labels for sequential data using conditional random fields (CRFs). However, this method operates under strong and possibly unrealistic assumptions, such as the assumption that only one annotator provides the correct label for each datum. Relaxing the assumptions of [119], [99] extended the standard hidden markov model (HMM) to incorporate annotator responses, and employs a variational EM [9] algorithm to aggregate them. As both aforementioned methods require tuning of hyperparameters, a cross-validation step is necessary, which might be unrealistic in unsupervised settings.

Regarding networked data, recent works have employed Gaussian Processes to classify the data based on annotator responses [120, 121, 154]. In addition to requiring the data features at the meta-learner, these methods need extensive training and as such are not suited for an unsupervised setting.

The present work puts forth two novel schemes for *blind ensemble learning of dependent data*. For sequential data, the proposed scheme presumes data are drawn from a hidden markov model (HMM) and employs decoupling and moment-matching to enable the assessment of annotator reliability and judiciously fuse their responses, in a two-step approach. In addition, an EM algorithm is developed. For networked data, the proposed scheme presumes data are drawn from a hidden markov random field (HMRF.) The dependencies are captured through a graph, and an EM algorithm is developed to infer annotator reliability and data labels. Our novel approaches do not require the tuning of hyperparameters, making them suitable for truly unsupervised settings, and the presence of dependencies between data markedly extends the scope of our previous work in Chap. 3. The following two sections of this chapter will introduce our approaches for ensemble classification with sequential and networked data. Throughout this chapter we assume that As1 and As2 from Chap. 3 hold.

4.2 Blind Ensemble Classification of Sequential Data

Suppose now, that for our dataset, it is known that the data are sequential, i.e., the n-th datum depends on the $n-1$-st datum. In order to take advantage of this fact, we will encode this knowledge in the marginal pmf of data labels $\Pr(\boldsymbol{y})$. Specifically, the sequence of labels $\{y_n\}_{n=1}^N$ forms a one-step time-homogeneous Markov chain; that is, variable y_n depends only on its immediate predecessor y_{n-1}. This Markov chain is characterized by a $K \times K$ transition matrix $\mathbf{T}$, whose (k, k')-th entry is given by

$$[\mathbf{T}]_{kk'} = T(k, k') = \Pr(y_n = k | y_{n-1} = k').$$

Matrix $\mathbf{T}$ has non-negative entries that satisfy the simplex constraint; hence, $\mathbf{T} \in \mathcal{C}$. Then the marginal probability of $\{y_n\}_{n=1}^N$ can be written as

$$\Pr(\boldsymbol{y} = \boldsymbol{k}) = \Pr(y_1 = k_1) \prod_{n=2}^{N} T(k_n, k_{n-1}) \tag{4.1}$$

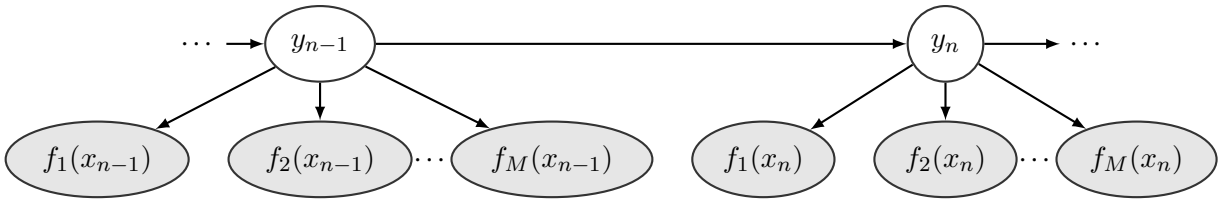


Figure 4.1: Graphical representation of the proposed model for sequential data. Shaded ellipses indicate observed variables, i.e. annotator responses.

where $\boldsymbol{k} := [k_1, \ldots, k_N]^\top$. Accordingly, the data $\{x_n\}_{n=1}^N$ depend only on their corresponding y_n, and are generated from an unknown conditional pdf as $x_n \sim \Pr(x_n|y_n = k_n)$. The data pairs $\{(x_n, y_n)\}_{n=1}^N$ form a hidden markov model (HMM), where the labels $\{y_n\}_{n=1}^N$ correspond to the hidden variables of the HMM, while $\{x_n\}_{n=1}^N$ correspond to the observed variables of the HMM.

As with the i.i.d. case of Chap. 3, M annotators observe $\{x_n\}_{n=1}^N$, and provide estimates of their labels $f_m(x_n)$. Under As1, the responses of different annotators per datum are conditionally independent, given the ground-truth label y_n of the same datum x_n; that is

$$\begin{aligned}&\Pr\left(f_1(x_n) = k_1, \ldots, f_M(x_n) = k_M | y_n = k\right)\\ &= \prod_{m=1}^{M} \Pr\left(f_m(x_n) = k_m | y_n = k\right) \text{ for } n = 1, \ldots, N. \end{aligned} \tag{4.2}$$

A graphical representation of this model is provided in Fig. 4.1. At this point, we require an additional assumption.

As3. The Markov chain formed by the labels $\{y_n\}$ has a unique stationary distribution $\boldsymbol{\pi} := [\pi_1, \ldots, \pi_K]^\top = [\Pr(Y = 1), \ldots, \Pr(Y = K)]^\top$, and is also irreducible.

As3 here will lead to our two-step moment matching algorithm.

Building on the aforementioned model, we will now present our novel moment-matching approach to blind ensemble learning for classifying sequential data. Similar to [76], our method decouples the problem of learning the parameters of interest in two steps. First, estimates of the confusion matrices $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$ and stationary distribution $\hat{\boldsymbol{\pi}}$ are obtained; and subsequently, the transition matrix is estimated as $\hat{\mathbf{T}}$ before obtaining an estimate of the labels $\{\hat{y}_n\}_{n=1}^N$.

4.2.1 Label estimation

Given only annotator responses for all data in a sequence, an approach to estimating the labels of each datum, meaning the hidden variables of the HMM, is to find the sequence $\boldsymbol{k}$ that maximizes the joint probability of the labels $\boldsymbol{y}$ and the annotator responses $\mathbf{F}$, namely

$$\Pr\left(\boldsymbol{y}=\boldsymbol{k},\mathbf{F}\right)=\Pr(y_1=k_1)\prod_{n=2}^{N}T(k_n,k_{n-1})\prod_{m=1}^{M}\Gamma_m(f_m(x_n),k_n) \tag{4.3}$$

where the equality is due to (4.1) and (4.2). This can be done efficiently using the Viterbi algorithm [40,114]. In order to obtain estimates of the labels, $\{\mathbf{\Gamma}_m\}_{m=1}^M$ and $\mathbf{T}$ must be available. The next subsection will show that $\{\mathbf{\Gamma}_m\}_{m=1}^M$ and $\mathbf{T}$ can be recovered by the statistics of annotator responses, using the aforementioned two-step procedure.

4.2.2 Confusion and Transition matrix estimation

Under As3, the HMM is mixing and assuming that y_0 is drawn from the stationary distribution $\boldsymbol{\pi}$, the responses of an annotator m can be considered to be generated from a mixture model, i.e [76].

$$f_m(X)\sim\sum_{k=1}^{K}\pi_k\Pr(f_m(X)|Y=k)$$

Based on this, for the remainder of this subsection we will treat the labels $\{y_n\}_{n=1}^N$, as if they had been drawn i.i.d. from the stationary distribution $\boldsymbol{\pi}$, that is $y_n\sim\boldsymbol{\pi}$ for $n=1,\ldots,N$. Then, the procedure presented in Chap. 3 can be readily applied to obtain estimates of the stationary distribution $\hat{\boldsymbol{\pi}}$ and the confusion matrices $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$.

With estimates of annotator confusion matrices $\{\hat{\mathbf{\Gamma}}_m\}$ and stationary probabilities $\hat{\boldsymbol{\pi}}$ at hand, we turn our attention to the estimation of the transition matrix $\mathbf{T}$. In order to estimate the transition matrix $\mathbf{T}$ of the HMM, consider the crosscorrelation matrix of subsequent vectorized observations between annotators m and m', namely $\tilde{\mathbf{R}}_{mm'}=\mathbb{E}[\mathbf{f}_m(x_n)\mathbf{f}_{m'}^\top(x_{n-1})]$. Under the HMM of Sec. 4.2, $\tilde{\mathbf{R}}_{mm'}$ can be written as

$$\tilde{\mathbf{R}}_{mm'}=\mathbf{\Gamma}_m\mathbf{T}\mathrm{diag}(\boldsymbol{\pi})\mathbf{\Gamma}_{m'}^\top=\mathbf{\Gamma}_m\mathbf{A}\mathbf{\Gamma}_{m'}^\top \tag{4.4}$$

where $\mathbf{A}:=\mathbf{T}\mathrm{diag}(\boldsymbol{\pi})$. Letting $\tilde{\mathbf{S}}_{mm'}$ denote the sample counterpart of (4.4), and with

$\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$ available, we can recover $\mathbf{T}$ as follows. First, we solve the convex moment-matching optimization problem

$$\min_{\mathbf{A}\in\mathcal{C}_S} \sum_{\substack{m=1 \\ m'>m}}^{M} \|\tilde{\mathbf{S}}_{mm'} - \hat{\mathbf{\Gamma}}_m \mathbf{A} \hat{\mathbf{\Gamma}}_{m'}^\top\|_F^2 \tag{4.5}$$

where $\mathcal{C}_S$ is the set of matrices whose entries are positive and sum to 1, namely $\mathcal{C}_S := \{\mathbf{X} \in \mathbb{R}^{K\times K} : \mathbf{X} \geq \mathbf{0}, \mathbf{1}^\top \mathbf{X} \mathbf{1} = 1\}$. The constraint is due to the fact that $\mathbf{1}^\top \mathbf{T} = \mathbf{1}^\top$, $\text{diag}(\boldsymbol{\pi})\mathbf{1} = \boldsymbol{\pi}$, and $\boldsymbol{\pi}^\top \mathbf{1} = 1$. Note that (4.5) is a standard constrained convex optimization problem that can be solved with off-the-shelf tools, such as CVX [54]. Having obtained $\hat{\mathbf{A}}$ from (4.5), we can then estimate the transition matrix as

$$\hat{\mathbf{T}} = \hat{\mathbf{A}}(\text{diag}(\hat{\boldsymbol{\pi}}))^{-1}. \tag{4.6}$$

Note here that explicit knowledge of $\boldsymbol{\pi}$ is not required, as its estimate can be recovered from $\hat{\mathbf{A}}$ as follows

$$\hat{\boldsymbol{\pi}} = \mathbf{1}^\top \hat{\mathbf{A}} = \mathbf{1}^\top \hat{\mathbf{T}} \text{diag}(\hat{\boldsymbol{\pi}}) = \mathbf{1}^\top \text{diag}(\hat{\boldsymbol{\pi}}).$$

The following proposition argues the consistency of the transition matrix estimates $\hat{\mathbf{T}}$.

Proposition 4.1. *Given accurate estimates of* $\{\mathbf{\Gamma}_m\}, \boldsymbol{\pi}$*, the estimate* $\hat{\mathbf{T}}$ *given by* (4.5) *and* (4.6) *approaches* $\mathbf{T}$ *as* $N \to \infty$.

Proof. By the LLN $\tilde{\mathbf{S}}_{mm'} \to \tilde{\mathbf{R}}_{mm'}$ as $N \to \infty$ for all m, m'. Since the objective function of (4.5) is convex, from [142], we have that $\hat{\mathbf{A}}$ will converge to $\mathbf{A} = \mathbf{T}\text{diag}(\boldsymbol{\pi})$ as $N \to \infty$. Finally, as $\hat{\mathbf{T}}$ can be recovered from $\hat{\mathbf{A}}$ in closed form [cf. (4.6)], the claim of the proposition follows. □

With estimates of $\{\hat{\mathbf{\Gamma}}_m\}, \hat{\boldsymbol{\pi}}$ and $\hat{\mathbf{T}}$ at hand, estimates of the labels $\{y_n\}_{n=1}^N$ can be obtained using the method described in Sec. 4.2.1. Futhermore, the estimates of $\{\hat{\mathbf{\Gamma}}_m\}, \hat{\boldsymbol{\pi}}$ and $\hat{\mathbf{T}}$ can be used to initialize an EM algorithm (a.k.a. Baum-Welch,) whose details are provided in the next subsection.

Remark 4.1. While here we employed the algorithm of [134] to estimate annotator confusion matrices $\{\mathbf{\Gamma}_m\}$, any other blind ensemble classification algorithm, such as [31, 162], can be utilized.

Algorithm 4.1 EM for Sequential Data

Input: Annotator responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, initial estimates $\mathbf{T}^{(0)}$, $\{\mathbf{\Gamma}_m^{(0)}\}_{m=1}^{M}$.
Output: Estimates $\hat{\mathbf{T}}$, $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^{M}$.

1: **while** not converged **do**
2: Estimate $\hat{q}_{nk}^{(i+1)}$ and $\hat{\xi}_n^{(i+1)}(k,k')$ using the forward-backward algorithm (App. E).
3: Estimate $\{\hat{\mathbf{\Gamma}}_m^{(i+1)}\}_{m=1}^{M}$ via (4.10).
4: Estimate $\hat{\mathbf{T}}^{(i+1)}$ via (4.9).
5: $i \leftarrow i+1$
6: **end while**

Algorithm 4.2 Blind Ensemble Classifier for Sequential Data

Input: Annotator responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$.
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^{N}$.

1: Estimate $\boldsymbol{\pi}$, $\{\mathbf{\Gamma}_m\}_{m=1}^{M}$ via Alg. 3.1.
2: Estimate $\hat{\mathbf{T}}$ via (4.5) and (4.6).
3: Estimate $\hat{y}_n$ using the Viterbi algorithm [cf. Chap. 4.2.1].
4: If needed refine estimates of $\hat{\mathbf{T}}$, $\{\hat{\mathbf{\Gamma}}_m\}$ and $\{\hat{y}_n\}$ using Alg. 4.1.

4.2.3 EM algorithm for sequential data

As with the i.i.d. case of Chap. 3, the EM algorithm of this section seeks to iteratively maximize the marginal log-likelihood of observed annotator responses. In each iteration, in order to update the parameters of interest $\boldsymbol{\theta} = [\mathbf{T}, \mathbf{\Gamma}_1, \ldots, \mathbf{\Gamma}_M]$, the following quantities have to be computed:

$$q_{nk} = \Pr(y_n = k | \mathbf{F}, \boldsymbol{\theta}) \tag{4.7}$$

and

$$\xi_n(k,k') = \Pr(y_n = k, y_{n+1} = k' | \mathbf{F}, \boldsymbol{\theta}) \tag{4.8}$$

Luckily, due to the causal structure of $\Pr(\boldsymbol{y})$, the aforementioned quantities can be estimated efficiently using the forward-backward algorithm [114], whose details are provided in Appendix E.

At iteration i, after obtaining $q_{nk}^{(i+1)}, \xi_n^{(i+1)}(k,k')$ for $k,k' = 1,\ldots,K$ and $n = 1,\ldots,N$, via the forward-backward algorithm, the transition and confusion matrix estimates can be updated as follows

$$[\hat{\mathbf{T}}^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N-1} \xi_n^{(i+1)}(k',k)}{\sum_{n=1}^{N-1} q_{nk'}^{(i+1)}} \tag{4.9}$$

$$[\hat{\mathbf{\Gamma}}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^{K} \sum_{n=1}^{N} q_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')}. \tag{4.10}$$

The EM process for sequential data is summarized in Alg. 4.1, while the entire ensemble classification process for sequential data is tabulated in Alg. 4.2.

4.3 Blind Ensemble Classification of Networked data

In many cases, additional information pertaining to the data is available in the form of an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ denote the vertex (or node) and edge sets of $\mathcal{G}$ respectively. Each node of this graph corresponds to a data point, thus $|\mathcal{V}| = N$, and the edges encode pairwise relationships between the data. This graph can also be characterized using the (typically sparse) $N \times N$ adjacency matrix $\mathbf{A}$, whose (n, n')-th entry $[\mathbf{A}]_{nn'} = 1$ if there exists an edge between nodes n and n' and is 0 otherwise.

As with Sec. 4.2, we will encode data dependence, that is, the pairwise relationships provided by the graph $\mathcal{G}$, in the marginal pmf of the labels $\Pr(\boldsymbol{y})$. Specifically, we model the labels $\{y_n\}_{n=1}^N$ as being drawn from an MRF, and as such we require the following assumption

As4. The conditional pmf of y_n, for all $n = 1, \ldots, N$, satisfies the local Markov property

$$\Pr(y_n|\boldsymbol{y}_{-n}) = \Pr(y_n|\boldsymbol{y}_{\mathcal{N}_n}) \tag{4.11}$$

where $\boldsymbol{y}_{-n}$ is a vector containing all labels except y_n and $\boldsymbol{y}_{\mathcal{N}_n}$ is a vector containing the labels of the neighbors of node n.

Under As 4, the joint pmf of all labels can be expressed as

$$\Pr(\boldsymbol{y}) = \frac{1}{Z} \exp(-U(\boldsymbol{y})) \tag{4.12}$$

where $Z = \sum_{\boldsymbol{y}} \exp(-U(\boldsymbol{y}))$ is the normalization constant, and $U(\boldsymbol{y})$ is the so-called energy function. Note that, computing the normalization constant Z, involves all possible configurations

of $\boldsymbol{y}$, and thus is intractable for situations involving large numbers of data. By the Hammersley-Clifford theorem [55] this energy function can be written as

$$U(\boldsymbol{y}) = \frac{1}{2} \sum_{(n,n')\in\mathcal{E}} V(y_n, y_{n'}) \tag{4.13}$$

where $V(y_n, y_{n'})$ denotes the so-called clique potential of the (n, n')-th edge. Here, we define the clique potentials as

$$V(y_n, y_{n'}) = \begin{cases} 0 & \text{if } y_n = y_{n'} \\ \delta_n & \text{if } y_n \neq y_{n'} \end{cases}, \tag{4.14}$$

where $\delta_n > 0$ is some predefined parameter. The local energy at node (datum) n of the graph is then defined as

$$U_n(y_n) = \frac{1}{2} \sum_{n'\in\mathcal{N}_n} V(y_n, y_{n'}). \tag{4.15}$$

As with the previous sections, As1 holds, that is, annotator responses are conditionally independent given the label Y. Then, we can express the joint probability of the label y_n and corresponding annotator responses $\{f_m(x_n)\}_{m=1}^M$ given the neighborhood $\boldsymbol{y}_{\mathcal{N}_n}$ of node n as

$$\Pr\left(\{f_m(x_n)\}_{m=1}^M, y_n = k|\boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}\right) = \prod_{m=1}^M \Gamma_m(f_m(x_n), k) \Pr(y_n = k|\boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}) \tag{4.16}$$

and accordingly

$$\begin{aligned} &\Pr\left(y_n = k|\{f_m(x_n)\}_{m=1}^M, \boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}\right) \propto \prod_{m=1}^M \Gamma_m(f_m(x_n), k) \Pr(y_n = k|\boldsymbol{y}_{\mathcal{N}_n} = \boldsymbol{k}_{\mathcal{N}_n}) \\ &= \exp\left(-U_n(k) + \sum_{m=1}^M \log\Gamma_m(f_m(x_n), k)\right). \end{aligned} \tag{4.17}$$

4.3.1 Label estimation in MRFs

Finding ML estimates of the labels $\hat{\boldsymbol{y}}$, under the aforementioned model, involves the following optimization problem

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}} \Pr(\mathbf{F}, \boldsymbol{y}) = \arg\max_{\boldsymbol{y}} \Pr(\mathbf{F}|\boldsymbol{y}) \Pr(\boldsymbol{y}) \tag{4.18}$$

$$= \arg\max_{\boldsymbol{y}} \frac{1}{Z} \exp(-U(\boldsymbol{y})) \Pr(\mathbf{F}|\boldsymbol{y}). \tag{4.19}$$

Unfortunately, (4.18) is an intractable problem even for relatively small N, due to the structure of (4.12), and as such, we will have to rely on approximation techniques to obtain estimates of the labels.

Popular approximation methods include Gibbs sampling [24] and mean-field approximations [158]. Here, we opted for an iterative method called Iterated Conditional Modes (ICM), which has been used successfully in image segmentation [14]. Under the ICM paradigm, per iteration and given current estimates $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$, the label for datum n is updated by finding the k that maximizes its local posterior probability, that is

$$\tilde{y}_n^{(t)} = \arg\max_{k\in\{1,\ldots,K\}} \Pr\left(y_n = k | \{f_m(x_n)\}_{m=1}^M, \tilde{\boldsymbol{y}}_{\mathcal{N}_n}^{(t-1)}\right)$$

$$= \arg\min_{k\in\{1,\ldots,K\}} U_n(k) - \sum_{m=1}^M \log\left(\hat{\Gamma}_m(f_m(x_n), k)\right) \tag{4.20}$$

where the superscript denotes the iteration index, $\tilde{\boldsymbol{y}}_{\mathcal{N}_n}$ denotes the label estimates provided by the previous ICM iteration, and the second equality is due to (4.17). The optimization in (4.20) is carried out for $n = 1, \ldots, N$ until the values of $\tilde{\boldsymbol{y}}$ have converged or until a maximum number of iterations $T_{\max}$ has been reached.

The next subsection puts forth an EM-type algorithm for estimating $\{\hat{y}_n\}_{n=1}^N$ and $\{\hat{\mathbf{\Gamma}}_m\}_{m=1}^M$.

4.3.2 EM algorithm for networked data

As with the i.i.d. case of Chap. 3 and the sequential case of Sec. 4.2, the EM algorithm of this section seeks to iteratively maximize the marginal log-likelihood of observed annotator responses. Due, however, to the MRF structure of $\boldsymbol{y}$, the Q-function of the E-step [cf. Chap. 3.2.2], is hard to compute.

As such, we have to rely on the approximation technique of the previous subsection to compute estimates of $q_{nk} = \Pr(y_n = k|\{f_m(x_n)\}_{m=1}^M; \boldsymbol{\theta})$. Specifically, per EM iteration i, let $\hat{\boldsymbol{y}}^{(i)} := [\hat{y}_1^{(i)}, \ldots, \hat{y}_N^{(i)}]$ denote the estimates obtained by the iterative procedure of Sec. 4.3.1. Then, estimates $\hat{q}_{nk}^{(i+1)}$ are obtained as follows [cf. 4.17]

$$\hat{q}_{nk}^{(i+1)} = \frac{1}{Z'} \exp\left(-U_n^{(i+1)}(k) + \sum_{m=1}^{M} \log\left(\hat{\Gamma}_m^{(i)}(f_m(x_n), k)\right)\right) \tag{4.21}$$

where

$$Z' = \sum_k \exp\left(-U_n^{(i+1)}(k) + \sum_{m=1}^{M} \log\left(\hat{\Gamma}_m^{(i)}(f_m(x_n), k)\right)\right)$$

is the normalization constant, and $U_n^{(i+1)}(k)$ is defined using $\hat{\boldsymbol{y}}^{(i)}$ as

$$U_n^{(i+1)}(k) = \frac{1}{2} \sum_{n' \in \mathcal{N}_n} V(k, \hat{y}_{n'}^{(i+1)}). \tag{4.22}$$

Finally, the M-step, which involves finding estimates of $\{\boldsymbol{\Gamma}_m\}_{m=1}^M$ is identical to the M-step of the EM algorithm of Chap. 3.2.2 for i.i.d. data, i.e.

$$[\hat{\boldsymbol{\Gamma}}_m^{(i+1)}]_{k'k} = \frac{\sum_{n=1}^N \hat{q}_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k')}{\sum_{k''=1}^K \sum_{n=1}^N \hat{q}_{nk}^{(i+1)} \mathcal{I}(f_m(x_n) = k'')}. \tag{4.23}$$

Similar to the i.i.d. case, the aforementioned EM procedure tries to solve a non-convex problem. In addition, the ICM method outlined in Sec. 4.3.1 is a deterministic approach that performs greedy optimization. Therefore, proper initialization is crucial for obtaining accurate estimates of the labels and annotator confusion matrices.

Similarly to the decoupling approach of Sec. 4.2, here we first obtain estimates of annotator confusion matrices $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^M$ and labels $\hat{\boldsymbol{y}}$, using the moment-matching algorithm of Chap. 3. These values are then provided as initialization to Alg. 4.3. In cases where N is small to have accurate moment estimates, majority voting can be used instead to initialize Alg. 4.3. The entire procedure for blind ensemble classification with networked data is tabulated in 4.4.

The next section will evaluate the performance of our proposed schemes.

Algorithm 4.3 EM algorithm for MRFs

Input: Annotator responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, initial $\boldsymbol{y}^{(0)}$, $\{\boldsymbol{\Gamma}_m^{(0)}\}_{m=1}^{M}$, Data graph $\mathcal{G}(\mathcal{V},\mathcal{E})$.
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^{N}$.

while not converged **do**
  while not converged AND $t < T_{\max}$ **do**
    for $n = 1, \ldots, N$ **do**
      Update $\tilde{y}_n^{(t)}$ using (4.20).
    end for
    $t \leftarrow t + 1$
  end while
  Compute $\hat{q}_{nk}^{(i+1)}$ using (4.21).
  Compute $\{\hat{\boldsymbol{\Gamma}}_m^{(i+1)}\}_{m=1}^{M}$ using (4.23).
  $i \leftarrow i + 1$
end while

Algorithm 4.4 Blind Ensemble Classifier for networked data

Input: Annotator responses $\{f_m(x_n)\}_{m=1,n=1}^{M,N}$, Data graph $\mathcal{G}(\mathcal{V},\mathcal{E})$
Output: Estimates of data labels $\{\hat{y}_n\}_{n=1}^{N}$

Estimate initial values of $\{\boldsymbol{\Gamma}_m\}_{m=1}^{M}$ via Alg. 3.1.
Estimate initial values of $\{\hat{y}_n\}_{n=1}^{N}$ using (3.24).
Refine estimates of $\{\hat{y}_n\}_{n=1}^{N}$ and $\{\hat{\boldsymbol{\Gamma}}_m\}_{m=1}^{M}$ using Alg. 4.3.

4.4 Numerical Tests

The performance of the proposed algorithms for both sequential and networked data is evaluated in this section using synthetic and real datasets. The metric utilized in all experiments is the F-score, defined as follows

$$\text{F-score} = \frac{1}{K}\sum_{k=1}^{K} 2\frac{\text{Precision}_k * \text{Recall}_k}{\text{Precision}_k + \text{Recall}_k} \tag{4.24}$$

where we have defined the per-class Precision and Recall as

$$\text{Precision}_k = \frac{TP_k}{TP_k + FP_k} \tag{4.25}$$

$$\text{Recall}_k = \frac{TP_k}{TP_k + FN_k} \tag{4.26}$$

and the per-class True Positive (TP), False Positive (FP) and False Negative (FN) quantities as

$$TP_k = \sum_{n=1}^{N} \mathcal{I}(\hat{y}_n = k, y_n = k) \tag{4.27}$$

$$FP_k = \sum_{n=1}^{N} \mathcal{I}(\hat{y}_n \neq k, y_n = k) \tag{4.28}$$

$$FN_k = \sum_{n=1}^{N} \mathcal{I}(\hat{y}_n = k, y_n \neq k) \tag{4.29}$$

For synthetic data, the average confusion matrix estimation error is also evaluated

$$\bar{\varepsilon}_{CM} := \frac{1}{M} \sum_{m=1}^{M} \frac{\|\mathbf{\Gamma}_m - \hat{\mathbf{\Gamma}}_m\|_1}{\|\mathbf{\Gamma}_m\|_1} = \frac{1}{M} \sum_{m=1}^{M} \|\mathbf{\Gamma}_m - \hat{\mathbf{\Gamma}}_m\|_1 \tag{4.30}$$

All results represent averages over 10 independent Monte Carlo runs, using MATLAB [94]. Vertical lines in some figures indicate standard deviation.

4.4.1 Sequential data

For the sequential data case Alg. 4.2 with and without EM refinement (denoted as *Alg. 4.2 + Alg. 4.1* and *Alg. 4.2* respectively) is compared to majority voting (denoted as *MV*), the moment-matching method of [134] described in Chap. 3 (denoted as *MM*), Alg. 4.1 initialized with majority voting (denoted as *MV + Alg. 4.1*), and "oracle" classifiers, i.e. a Viterbi classifier that uses the ground-truth confusion and transition matrices. In addition, for synthetic data, the transition matrix estimation error $\|\mathbf{T} - \hat{\mathbf{T}}\|_1$ is also evaluated.

All datasets in this subsection are split into sequences of data. Here, we assume that per dataset these sequences have been drawn from the same ensemble HMM [cf. 4.2]. The reported F-score represents the averaged F-score of each sequence.

Synthetic data

For the synthetic data tests, S sequences of $N_s, s = 1, \dots S$, ground-truth labels each, were generated from a Markov chain, whose transition matrix was generated at random such that $\mathbf{T} \in \mathcal{C}$. Each of the $N = \sum_s N_s$ ground-truth labels $\{y_n\}_{n=1}^{N}$ correspond to one out of K

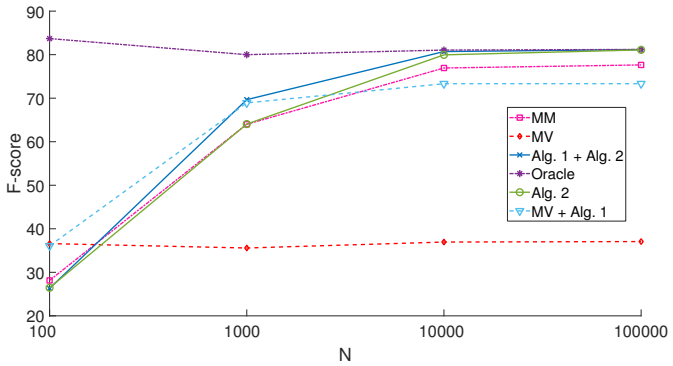
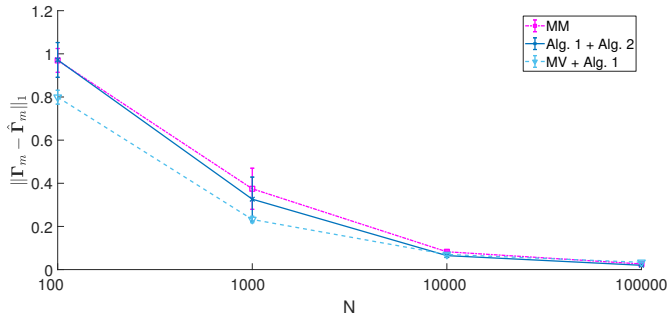


Figure 4.2: Average F-score for a synthetic dataset with $K = 4$ and $M = 10$ annotators

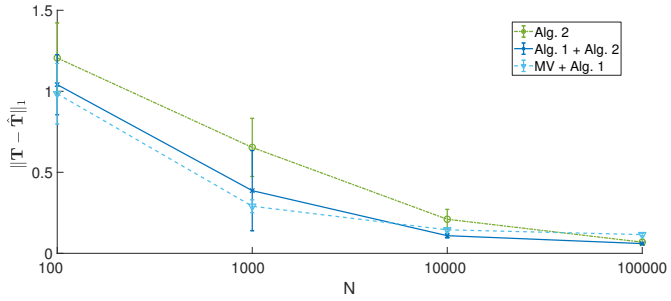
possible classes. Afterwards, $\{\mathbf{\Gamma}_m\}_{m=1}^M$ were generated at random, such that $\mathbf{\Gamma}_m \in \mathcal{C}$, for all $m = 1, \ldots, M$, and $\lfloor M/2 \rfloor + 1$ annotators are better than random, as per As2. Then annotators' responses were generated as follows: if $y_n = k$, then the response of annotator m will be generated randomly according to the k-th column of its confusion matrix, $\boldsymbol{\gamma}_{m,k}$ [cf. Sec. 3.1], that is $f_m(x_n) \sim \boldsymbol{\gamma}_{m,k}$.

Fig. 4.2 shows the average F-score for a synthetic dataset with $K = 4$, $M = 10$ annotators and varying number of data N. Fig. 4.3 shows the average confusion and transition matrix estimation errors for varying N. As the number of data N increases the performance of the proposed methods approaches the performance of the "oracle" one. Accordingly, the confusion and transition matrix estimates are approaching the true ones as N increases. This is to be expected, as noted in [134], since the estimated moments are more accurate for large N. Interestingly, Alg. 4.1 performs well when initialized with majority voting, even though it reaches a performance plateau as N increases. For small N however, it outperforms the other proposed methods. This suggests that, when N is small to have accurate moment estimation, it is preferred to initialize Alg. 4.1 with majority voting.

The next experiment evaluates the influence of the number of annotators M for the sequential classification task. Figs. 4.4 and 4.5 showcase results for an experiment with $K = 4$, fixed number of data $N = 10^3$ and varying number of annotators M. Clearly, the presence of multiple annotators is beneficial, as the F-score increases for all algorithms, while the confusion and transition matrix errors decrease. As with the previous experiment, Alg. 4.1 + Alg. 4.2 exhibits the best performance in terms of F-score, when M increases.



(a) Confusion matrix estimation error



(b) Transition matrix estimation error

Figure 4.3: Average estimation errors of confusion matrices and prior probabilities for a synthetic dataset with $K = 4$ and $M = 10$ annotators

Real data

Further tests were conducted on two real datasets, the Part-of-Speech (POS) tagging dataset and the Biomedical Information Extraction (IE) [99] dataset. For these datasets the moment matching method of Chap. 3 is used to initialize the EM algorithm of Chap. 3.2.2 and is denoted as *DS*.

For the POS dataset $M = 10$ classifiers were trained using NLTK [88] on subsets of the Brown coprus [41] to provide part-of-speech (POS) tags of text. The number of tags is $K = 12$. Then the classifiers provided POS tags for all words in the Penn Treebank corpus [93], which contains $N = 100,676$ words. Results for this dataset are tabulated in Tab. 4.1.

The Biomedical IE dataset consists of $5,000$ medical paper abstracts. $M = 91$ annotators were tasked with marking all text spans in a given abstract that identify the population of a randomized controlled trial. The dataset consists of $N = 304,629$ words belonging into $K = 2$ classes: in a span identifying the population or outside. For this particular dataset we evaluate

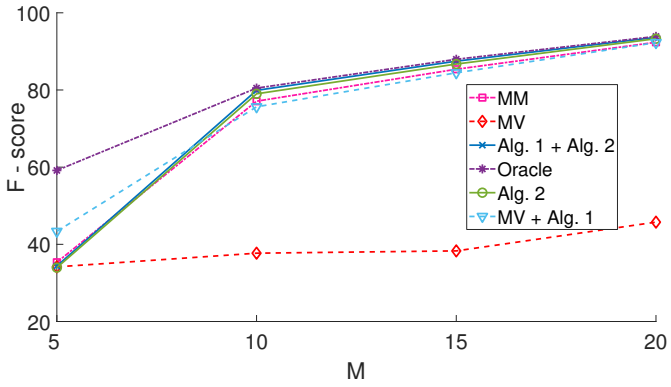


Figure 4.4

Metric	MV	DS	Alg. 4.2	Alg. 4.1 + Alg. 4.2	MV + Alg. 4.1
Precision	0.22916	0.23406	0.24785	**0.25856**	0.22972
Recall	0.23518	0.25629	0.24396	**0.26638**	0.24497
F-score	0.22598	0.22264	0.23352	**0.24735**	0.23007

Table 4.1: Results for the POS dataset with $N = 100,676$, $M = 10$ and $K = 12$.

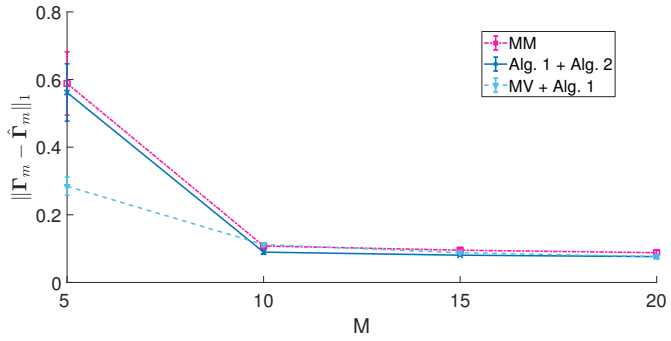
Precision and Recall per sequence in the following way, which was suggested in [99]

$$\text{Precision} = \frac{\text{\# true positive words}}{\text{\# words in a predicted span}}$$
$$\text{Recall} = \frac{\text{\# words in a predicted span}}{\text{\# words in ground-truth span}}$$

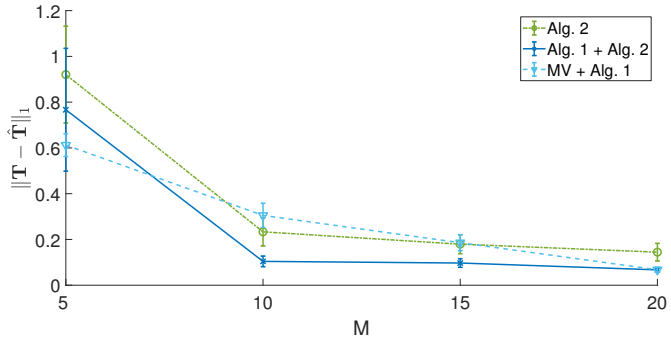
Results for this dataset are listed in Tab. 4.2. It can be seen that while majority voting achieves the best precision of all algorithms, due to its low recall, the overall F-score is low. However, Alg. 4.1 + Alg. 4.2 outperforms competing alternatives with regards to recall and F-score.

Metric	MV	DS	Alg. 4.2	Alg. 4.1 + Alg. 4.2	MV + Alg. 4.1
Precision	**1**	0.8344	0.8334	0.8750	0.8334
Recall	0.5575	0.6	0.6	**0.8**	0.6
F-score	0.7159	0.6980	0.6977	**0.8358**	0.6977

Table 4.2: Results for the Biomedical IE dataset with $N = 304,629$, $M = 91$ and $K = 2$.



(a) Confusion matrix estimation error



(b) Transition matrix estimation error

Figure 4.5: Average estimation errors of confusion matrices and prior probabilities for a synthetic dataset with $K = 2$ and $\boldsymbol{\pi} = [0.9003, 0.0997]^\top$ and $M = 10$ annotators

4.4.2 Networked data

For the networked data case Alg. 4.4 (denoted as *Alg. 4.4*) is compared to majority voting (denoted as *MV*), the Dawid and Skene model that assumes i.i.d. data described in Chap. 3 that uses moment-matching [134] as initialization (denoted as *DS*) and Alg. 4.4 initialized with majority voting (denoted as *MV + Alg. 4.3*).

The tests were conducted on five real datasets. For the Cora, Citeseer [91] and Pubmed [97] datasets the graph $\mathcal{G}$ and data features $\{x_n\}$ are provided with the dataset. In these cases, $M = 10$ classification algorithms from MATLAB's machine learning toolbox were trained on different randomly selected subsets of the datasets. Afterwards, these algorithms provided labels for all data in the dataset. For these datasets, we set $\delta_n = M$. For the Music genre and Sentence Polarity datasets [120] the features $\{x_n\}$ and annotator responses $\mathbf{F}$ are provided with the dataset. In these cases, the graphs were generated from the data features using k-nearest neighbors.

Dataset	K	M	N	MV	DS	Alg. 4.4	MV + Alg. 4.3
Cora	10	10	$2,708$	0.2785	0.4228	**0.6412**	0.336
CiteSeer	7	10	$3,312$	0.4257	0.4449	**0.5244**	0.5224
Pubmed	3	10	$19,717$	0.6968	0.7437	**0.7667**	0.7595
Music Genre	10	44	700	0.7046	0.4746	0.7649	**0.8029**
Sen. Polarity	2	203	$5,000$	0.8895	0.9129	**0.9153**	0.9139

Table 4.3: F-score for Real data experiments with Networked data.

For these datasets, $\delta_n = M_n/2$, where M_n denotes the number of annotators that provided a response for the n-th datum.

The Cora, CiteSeer and Pubmed datasets are citation networks and the versions used here are preprocessed by [11]. The Cora dataset consists of $N = 2,708$ scientific publications classified into $K = 7$ classes. The features $\{x_n\}$ of this dataset are sparse $1,433$-dimensional vectors and for this dataset each classification algorithm was trained on a random subset of 150 instances. The CiteSeer dataset consists of $N = 3,312$ scientific publications classified into one of $K = 6$ classes. The features $\{x_n\}$ of this dataset are sparse $3,703$-dimensional vectors, and each classification algorithm was trained on a subset of 100 instances. The Pubmed dataset is a citation network that consists of $N = 19,717$ scientific publications from the Pubmed database pertaining to diabetes, classified into one of $K = 3$ classes. The features $\{x_n\}$ of this dataset are 500-dimensional vectors, and each classification algorithm was trained on a subset of 300 instances. The Music genre dataset contains $N = 700$ 30-second song samples, belonging into $K = 10$ music categories, annotated by $M = 44$ annotators. The graph for this dataset was generated using $k = 3$ nearest neighbors. The sentence polarity dataset contains $N = 5,000$ sentences from movie reviews, classified into $K = 2$ categories (positive or negative), annotated by $M = 203$ annotators. The graph for this dataset was generated using $k = 1$ nearest neighbor.

The results for these datasets are tabulated in Tab. 4.3. In most datasets *Alg. 4.4* exhibits the best performance in terms of F-score followed closely by *MV+Alg. 4.3*. For the Music Genre dataset however, *MV+Alg. 4.3* outperforms *Alg. 4.4*. This is to be expected, as N is relatively small for this dataset and as such the estimated annotator moments are not very accurate, that is *MV* outperforms *DS*.

4.5 Conclusions

This chapter introduced two novel approaches to blind ensemble and crowdsourced classification in the presence of data dependencies. Two types of data dependencies were investigated: i) Sequential data; and ii) networked data, where the dependencies are encoded in a known graph. The performance of our novel schemes was evaluated on real and synthetic data.

Chapter 5

Summary and Future Directions

Backed by rigorous theoretical and extensive experimental results, the present thesis has introduced novel algorithms that help realize the goal of scalable learning for big data. The following subsections provide a summary of the work presented in this thesis, as well as possible future research directions.

5.1 Thesis Summary

In order to accelerate the task of subspace clustering, Chapter 2 aimed at devising high performance algorithms that enjoy low computational complexity. To realize this, a random projections based approach was developed that can handle both high volumes and high dimensionality of data. The proposed scheme can achieve state-of-the-art performance while requiring markedly less computational resources, and can be readily parallelized across multiple computing nodes. A rigorous performance analysis as well as extensive numerical tests on real data corroborated the potential of the proposed method.

Chapter 3 dealt with unsupervised ensemble classification. In such a setup, which arises naturally in crowdsourcing and distributed detection, the outputs of multiple, possibly heterogeneous algorithms or annotators have to be fused, in the absence of ground-truth labels at the fusion center/meta-learner. Assuming iid data and conditionally independent annotators an algorithm based on the PARAFAC structure of annotator moments was developed. Insights into the performance of the proposed algorithm and the unsupervised ensemble classification task are provided by a rigorous performance analysis and the proposed algorithm is compared to the

current state-of-the-art with extensive numerical tests on synthetic and real data. In addition, the effect of adversaries on the unsupervised ensemble classification task was briefly evaluated.

Finally, Chapter 4 builds on the results and algorithms of Chapter 3 to enable blind ensemble classification for dependent data. Two types of data dependencies were considered here: sequential data and generally dependent data, whose dependencies are captured by a graph. Expectation Maximization based approaches were developed for both cases and their performance was evaluated with extensive numerical tests on real and synthetic data.

5.2 Future Research

The promising results in this thesis open up interesting directions for a number of future research topics. The following subsections discuss a few of these directions.

5.2.1 Large-scale subspace clustering

Following the success of the results of Chap. 2, it is natural to consider online extensions, along the lines of [124, 136], that can handle truly massive and streaming data. Several technically challenging, yet pertinent research directions also emerge.

- **Kernel-based nonlinear randomized subspace clustering.** While SC thrives when data lie on a union of subspaces, many datasets might not exhibit that property. In such cases, the theory of reproducing kernel hilbert spaces (RKHS) [128] and its recent advances that have rendered it an invaluable tool for nonlinear signal processing and machine learning. As the solutions to SSC/LRR/LSR and (2.12) rely on inner products, these algorithms can be readily extended to deal with data where standard SC methods fail. Specifically, the solutions of SSC/LRR/LSR depend on the matrix $\mathbf{K} = \mathbf{X}^\top\mathbf{X}$. Extending these SC algorithms to the nonlinear case involves using $\mathbf{K} = \phi^\top(\mathbf{X})\phi(\mathbf{X})$ instead, where ϕ is a predetermined nonlinear function applied on each column of $\mathbf{X}$, and thus the approach outlined in the previous subsection can be readily applied. Performance of this scheme depends critically on the choice of ϕ. To alleviate this drawback, multi-kernel methods [50, 159] can be employed, where the kernel function is selected from a "dictionary" containing multiple predetermined kernels. In addition, the bottleneck of large-scale kernel methods is that they require the entire kernel matrix to be instantiated.

As such, it is of interest investigate whether the generation of the full kernel matrix can be avoided using recent advances in kernel learning, such as random features [115].

- **Randomized clustering for tensor data.** Current state-of-the-art approaches to SC, deal with data that are vectorized. However, many types of data, such as images or video, can be considered as slabs of a tensor $\underline{X}$. By taking advantage of this multilinear relationship between the data, the performance of the SC task can be enhanced at the price of increased computational complexity, as was shown in our recent work [137]. Thus, an extension of the randomized SC scheme presented in this thrust for tensor data is well motivated. Tensor data clustering methods include multilinear clustering [137] and tensor "self-dictionary" methods, resembling SSC/LRR/LSR [46]. Subsequently, one can investigate the clustering performance of the aforementioned tensor based clustering algorithms when the tensor is compressed randomly, i.e. when its dimensionality is reduced using JLTs. As the resulting tensor will be smaller in size, the tensor based clustering task will be accelerated significantly.

- **Large-scale randomized learning.** The proposed approach may be tailored for large-scale SC, however, it can fundamentally be applied to any machine learning algorithm that depends on inner products between data. As such, the scope of the randomized data reduction approach of Chap. 3 can be broadened to encompass classification/regression/anomaly detection tasks. To this end, algorithms are suitable for the data reduction scheme outlined in Chap. 3 have to be identified. The resulting efficient yet accurate methods could be instrumental in realizing a truly large-scale learning scheme when combined with the distributed learning paradigm proposed in the next sections, as the computational cost per node will decrease dramatically.

5.2.2 Learning with Blind Ensembles

The encouraging results of Chapters 3 and 4 prompt us to investigate several exciting research directions.

- **Learning with dependent annotators.** In the blind ensemble classification task, in many cases one cannot assume conditionally independent annotators, as classification algorithms might be trained on overlapping datasets or in the crowdsourcing case annotators might

influence each other. Correlations between annotator responses can be employed to detect dependencies between annotators [64, 133]. Highly correlated annotators can be assigned to the same "group" or cluster. Assuming that within the same group, annotators make independent decisions, the proposed algorithm in Chap. 3 can be generalized to estimate the corresponding confusion matrices.

- **Ensemble regression.** In addition to classification, it is of interest to develop tools for blind ensemble regression [34]. The approach put forth in this thrust will have to be altered to extract continuous distributions from annotator statistics, as $g_m := \Pr(f_m(X))$ is generally a continuous distribution in the regression setup. To this end, one promising idea is to consider using kernel density estimation tools [148] alongside the algorithm described earlier. Kernel density estimators, similar to the histogram and unlike parametric estimators, make minimal assumptions about the unknown pdf. For data $\{z_i\}_{i=1}^{I} \sim g$, drawn from g the kernel density estimator is given by

$$\hat{g}(z) = \frac{1}{I} \sum_{i=1}^{I} k(z, z_i) \tag{5.1}$$

  where $k(z, z_i)$ denotes a predetermined kernel function, typically chosen to be a density. Naturally, the performance of this density estimator, for finite number of data I, depends on the choice of k. Typically, kernel choice reflects some prior knowledge on the form of the pdf that is to be approximated. When such prior information is not available, multi-kernel [50, 159] approaches, where the pdf is approximated using a dictionary of appropriately weighted kernel functions, can be employed. Finally, the analytical performance of this method is worth investigating, using techniques derived from the previous subsection and tools from kernel density estimation theory.

- **Ensemble clustering.** While clustering is the blind counterpart of classification, combining results from multiple clustering algorithms presents additional challenges. In particular, clustering algorithms generally do not agree in their class labeling; what is referred to as class "1" from annotator m, might be class "2" for annotator m'. This limitation, however, can be circumvented by considering an alternative representation of class labeling from the annotators. For each annotator m define an N-node graph $\mathcal{G}_m$, with corresponding adjancency matrix $\mathbf{A}_m$. Then let the (i, j)-th entry of $\mathbf{A}_m$ be $[\mathbf{A}_m]_{ij} = 1$, if $f_m(x_i) = f_m(x_j)$,

that is, if annotator m assigned x_i and x_j to the same cluster, and 0 otherwise. This new representation does not depend on the particular class labeling of an annotator. Building on the results of Chaps. 3 and 4 it is of interest to develop a probabilistic scheme similar to the one outlined for classification. The emerging challenge in this task is to find a proper decomposition of $\Pr([\mathbf{A}_m]_{ij})$ into conditional probabilities that characterize annotators. Upon developing the scheme for unsupervised ensemble clustering it will be interesting to quantify its performance analytically.

- **Semi-supervised ensemble learning.** The approach described in Chap. 3 requires no information at the meta-learner, besides annotator responses. Inclusion of prior information can be certainly beneficial to the ensemble classification task. Such prior information can be of the form of ground-truth labels for a few data. In this case, knowledge of the ground-truth labels allows one to provide initial estimates of the annotator confusion matrices and class prior probabilities, potentially enabling faster convergence of the algorithm described in Chap. 3. The proposed research further offers the potential to answer several interesting questions that arise in this context, regarding the number of required ground-truth labels and their distribution relative to the number of annotators and the number of classes. Prior information can also be provided in terms of must- and cannot-link constraints. This type of information can be encoded in a graph, and the algorithms of Chap. 4 can be readily employed. While incorporating this prior knowledge is technically more challenging, this scenario is more realistic: correlations can be derived from the data themselves, or similar data can be inserted in the dataset. In addition to enhancing the performance of the meta-learner, this prior information will also enhance the detection of adversarial annotators, even when they are numerous. Finally, the effect of prior information will be considered on other ensemble learning tasks such as clustering, regression and dimensionality reduction.

- **Adversarial attacks on ensembles and remedies.** For this task, it is of interest to investigate the effect of adversarial annotators in an ensemble learning setup. As most algorithms require the number of adversarial annotators to be less than $M/2$, development and evaluation of adversarial attack strategies is important in this scenario. In particular, the effect of adversaries on the performance of an ensemble learning scheme, when each of them acts individually and when they are cooperating, should be investigated along with adversarial attack techniques against ensembles of learners, and possible remedies. This

adversarial/ensemble learning can also be modeled as a game between the adversaries, that want to inhibit the machine learning task, and the meta-learner/fusion center, that seeks to identify the adversaries and produce reliable results. All in all, it will be exciting to complement the tools developed with game and information-theoretic analyses.

References

[1] D. Achlioptas, "Database-friendly random projections: Johnson-Lindenstrauss with binary coins," *Journal of Computer and System Sciences*, vol. 66, no. 4, pp. 671–687, 2003.

[2] P. K. Agarwal and N. H. Mustafa, "K-means projective clustering," in *Proc. 23rd ACM SIGMOD-SIGACT-SIGART Symposium*. Paris, France: ACM, June 2004, pp. 155–165.

[3] N. Ailon and B. Chazelle, "The fast Johnson–Lindenstrauss transform and approximate nearest neighbors," *SIAM Journal on Computing*, vol. 39, no. 1, pp. 302–322, 2009.

[4] N. Ailon and E. Liberty, "Fast dimension reduction using Rademacher series on dual BCH codes," *Discrete & Computational Geometry*, vol. 42, no. 4, p. 615, 2009.

[5] D. C. Anastasiu and G. Karypis, "L2knng: Fast exact k-nearest neighbor graph construction with l2-norm pruning," in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*. Melbourne, Australia: ACM, 2015, pp. 791–800.

[6] C. A. Andersson and R. Bro, "The N-way toolbox for MATLAB," *Chemometrics and Intelligent Laboratory Systems*, vol. 52, no. 1, pp. 1–4, 2000.

[7] B. W. Bader and T. G. Kolda, "Efficient matlab computations with sparse and factored tensors," *SIAM Journal on Scientific Computing*, vol. 30, no. 1, pp. 205–231, 2008.

[8] M.-F. F. Balcan, S. Ehrlich, and Y. Liang, "Distributed k-means and k-median clustering on general topologies," in *Advances in Neural Information Processing Systems*, 2013, pp. 1995–2003.

[9] M. J. Beal, "Variational algorithms for approximate Bayesian inference," Ph.D. dissertation, University of London, University College London (United Kingdom, 2003.

[10] J. L. P. Bennett A. Landman, John A. Bogovic, "Simultaneous truth and performance level estimation with incomplete, over-complete, and ancillary data," vol. 7623, 2010. [Online]. Available: https://doi.org/10.1117/12.844182

[11] D. Berberidis, A. N. Nikolakopoulos, and G. B. Giannakis, "Adaptive diffusions for scalable learning over graphs," *IEEE Trans. Sig. Proc.*, pp. 1–15, 2019.

[12] T. Berger, Z. Zhang, and H. Viswanathan, "The CEO problem," *IEEE Transactions on Information Theory*, vol. 42, no. 3, pp. 887–902, 1996.

[13] D. P. Bertsekas, *Nonlinear Programming*. Athena scientific, 1999.

[14] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society B*, vol. 48, no. 3, pp. 48–259, 1986.

[15] B. Biggio, G. Fumera, and F. Roli, "Multiple classifier systems for robust classifier design in adversarial environments," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1-4, pp. 27–41, 2010.

[16] T. Bonald and R. Combes, "A minimax optimal algorithm for crowdsourcing," 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. 4352–4360. [Online]. Available: http://papers.nips.cc/paper/7022-a-minimax-optimal-algorithm-for-crowdsourcing.pdf

[17] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *arXiv preprint arXiv:1606.04838*, 2016.

[18] C. Boutsidis, A. Zouzias, M. W. Mahoney, and P. Drineas, "Randomized dimensionality reduction for k-means clustering," *IEEE Transactions on Information Theory*, vol. 61, no. 2, pp. 1045–1062, 2015.

[19] D. C. Brabham, "Crowdsourcing as a model for problem solving: An introduction and cases," *Convergence*, vol. 14, no. 1, pp. 75–90, 2008.

[20] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[21] J. Bruna and S. Mallat, "Invariant scattering convolution networks," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1872–1886, 2013.

[22] C. Buckley, M. Lease, and M. D. Smucker, "Overview of the TREC 2010 Relevance Feedback Track (Notebook)," in *The Nineteenth Text Retrieval Conference (TREC) Notebook*, 2010.

[23] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[24] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1080/00031305.1992.10475878

[25] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 3, pp. 568–586, 2011.

[26] D. Chinavle, P. Kolari, T. Oates, and T. Finin, "Ensembles in adversarial classification for spam," in *Proc. of the ACM Conf. on Information and Knowledge Management*. Hong Kong: ACM, 2009, pp. 2015–2018.

[27] K. L. Clarkson and D. P. Woodruff, "Low rank approximation and regression in input sparsity time," in *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM, 2013, pp. 81–90.

[28] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. John Wiley & Sons, 2012.

[29] K. Cukier, "Data, data everywhere," *The Economist*, 2010. [Online]. Available: http://www.economist.com/node/15557443

[30] N. Dalvi, A. Dasgupta, R. Kumar, and V. Rastogi, "Aggregating crowdsourced binary ratings," in *Proceedings of the Intl. Conf. on World Wide Web*. Rio de Janeiro, Brazil: ACM, 2013, pp. 285–294.

[31] A. P. Dawid and A. M. Skene, "Maximum likelihood estimation of observer error-rates using the EM algorithm," *Applied Statistics*, pp. 20–28, 1979.

[32] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: Spectral clustering and normalized cuts," in *Proc. 10th ACM SIGKDD*. ACM, 2004, pp. 551–556.

[33] T. G. Dietterich, "Ensemble methods in machine learning," in *Intl. Workshop on Multiple Classifier Systems*. Springer, 2000, pp. 1–15.

[34] O. Dror, B. Nadler, E. Bilal, and Y. Kluger, "Unsupervised ensemble regression," *arXiv preprint arXiv:1703.02965*, 2017.

[35] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l1-ball for learning in high dimensions," in *Proc. of the Intl. Conf, on Machine Learning*. Helsinki, Finland: ACM, 2008, pp. 272–279.

[36] E. L. Dyer, A. C. Sankaranarayanan, and R. G. Baraniuk, "Greedy feature selection for subspace clustering." *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2487–2517, 2013.

[37] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 35, no. 11, pp. 2765–2781, 2013.

[38] Y. Fang, R. Wang, B. Dai, and X. Wu, "Graph-based learning via auto-grouped sparse regularization and kernelized extension," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 1, pp. 142–154, 2015.

[39] P. A. Forero, A. Cano, and G. B. Giannakis, "Distributed clustering using wireless sensor networks," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 707–724, 2011.

[40] G. D. Forney, "The viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, March 1973.

[41] W. N. Francis and H. Kucera, "Brown corpus manual," Department of Linguistics, Brown University, Providence, Rhode Island, US, Tech. Rep., 1979. [Online]. Available: http://icame.uib.no/brown/bcm.html

[42] Y. Freund, "Boosting a weak learning algorithm by majority," *Information and Computation*, vol. 121, no. 2, pp. 256–285, 1995.

[43] Y. Freund, R. E. Schapire *et al.*, "Experiments with a new boosting algorithm," in *Proc. of the Intl. Conf. on Machine Learning*, vol. 96, Bari, Italy, 1996, pp. 148–156.

[44] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[45] S. Fritz, L. See, C. Perger, I. McCallum, C. Schill, D. Schepaschenko, M. Duerauer, M. Karner, C. Dresel, J.-C. Laso-Bayas *et al.*, "A global dataset of crowdsourced land cover and land use reference data," *Scientific data*, vol. 4, p. 170075, 2017.

[46] Y. Fu, J. Gao, D. Tien, and Z. Lin, "Tensor LRR based subspace clustering," in *International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2014, pp. 1877–1884.

[47] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 23, no. 6, pp. 643–660, June 2001.

[48] A. Ghosh, S. Kale, and P. McAfee, "Who moderates the moderators?: Crowdsourcing abuse detection in user-generated content," in *Proceedings of the 12th ACM Conference on Electronic Commerce*. San Jose, CA: ACM, 2011, pp. 167–176.

[49] G. B. Giannakis, Q. Ling, G. Mateos, I. D. Schizas, and H. Zhu, "Decentralized learning for wireless communications and networking," in *Splitting Methods in Communication and Imaging, Science and Engineering*, R. Glowinski, S. Osher, and W. Yin, Eds. Springer, 2016.

[50] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2211–2268, 2011.

[51] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A Procrustean approach to learning binary codes for large-scale image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 12, pp. 2916–2929, 2013.

[52] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," *Intl. Conf. on Learning Representations*, San Diego, CA, USA, May 2015.

[53] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[54] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.1," http://cvxr.com/cvx, Mar. 2014.

[55] J. M. Hammersley and P. E. Clifford, "Markov random fields on finite graphs and lattices," Unpublished manuscript, 1971.

[56] R. A. Harshman and M. E. Lundy, "PARAFAC: Parallel factor analysis," *Computational Statistics & Data Analysis*, vol. 18, no. 1, pp. 39–72, 1994.

[57] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2001.

[58] R. Heckel and H. Bölcskei, "Robust subspace clustering via thresholding," *IEEE Transactions on Information Theory*, vol. 61, no. 11, pp. 6320–6342, 2015.

[59] R. Heckel, M. Tschannen, and H. Bölcskei, "Dimensionality-reduced subspace clustering," *arXiv preprint arXiv:1507.07105*, 2017.

[60] M. Honnibal and I. Montani. (2017) spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing.

[61] J. Howe, "The rise of crowdsourcing," *Wired Magazine*, vol. 14, no. 6, pp. 1–4, 2006.

[62] K. Huang, N. D. Sidiropoulos, and A. P. Liavas, "A flexible and efficient algorithmic framework for constrained matrix and tensor factorization," *IEEE Transactions on Signal Processing*, vol. 64, no. 19, pp. 5052–5065, 2016.

[63] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*. Dallas, TX: ACM, 1998, pp. 604–613.

[64] A. Jaffe, E. Fetaya, B. Nadler, T. Jiang, and Y. Kluger, "Unsupervised ensemble learning with dependent classifiers," in *Artificial Intelligence and Statistics*, 2016, pp. 351–360.

[65] A. Jaffe, B. Nadler, and Y. Kluger, "Estimating the accuracies of multiple classifiers without labeled data." in *AISTATS*, vol. 2, San Diego, CA, 2015, p. 4.

[66] P. Jain and S. Oh, "Learning mixtures of discrete product distributions using spectral decompositions." *Journal of Machine Learning Research*, vol. 35, pp. 824–856, 2014.

[67] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipschitz mappings into a Hilbert space," *Contemporary Mathematics*, vol. 26, no. 189-206, p. 1, 1984.

[68] I. Jolliffe, *Principal Component Analysis*. Wiley Online Library, 2002.

[69] V. Kalantzis, R. Li, and Y. Saad, "Spectral Schur complement techniques for symmetric eigenvalue problems," *Electronic Transactions on Numerical Analysis*, vol. 45, pp. 305–329, 2016.

[70] N. Kargas and N. D. Sidiropoulos, "Completing a joint pmf from projections: a low-rank coupled tensor factorization approach," in *Information Theory and Applications Workshop*. IEEE, 2017.

[71] D. R. Karger, S. Oh, and D. Shah, "Efficient crowdsourcing for multi-class labeling," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1, pp. 81–92, 2013.

[72] S. M. Kay, *Fundamentals of Statistical Signal Processing, volume I: Estimation Theory*. Prentice Hall, 1993.

[73] ——, *Fundamentals of Statistical Signal Processing, volume II: Detection Theory*. Prentice Hall, 1998.

[74] M. Kim, "A maximum-likelihood and moment-matching density estimator for crowdsourcing label prediction," *Applied Intelligence*, vol. 48, no. 2, pp. 381–389, Feb 2018.

[75] A. Kittur, E. H. Chi, and B. Suh, "Crowdsourcing user studies with mechanical turk," in *Proc. of SIGCHI Conf. on Human Factors in Computing Systems*. Florence, Italy: ACM, 2008, pp. 453–456.

[76] A. Kontorovich, B. Nadler, and R. Weiss, "On learning parametric-output hmms," in *Proceedings of the 30th International Conference on Machine Learning*, Atlanta,

Georgia, USA, 17–19 Jun 2013. [Online]. Available: http://proceedings.mlr.press/v28/kontorovich13.html

[77] Q. Le, T. Sarlos, and A. Smola, "Fastfood - approximating kernel expansions in loglinear time," in *30th International Conference on Machine Learning*, Atlanta, GA, 2013. [Online]. Available: http://jmlr.org/proceedings/papers/v28/le13.html

[78] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[79] Y.-h. Lee and S.-J. Kim, "Online robust subspace clustering for analyzing incomplete synchrophasor measurements," in *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE, 2016, pp. 816–820.

[80] Y. J. Lee, J. A. Arida, and H. S. Donovan, "The application of crowdsourcing approaches to cancer research: a systematic review," *Cancer Medicine*, vol. 6, no. 11, pp. 2595–2605. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cam4.1165

[81] R. B. Lehoucq, D. C. Sorensen, and C. Yang, "ARPACK users' guide: solution of large-scale eigenvalue problems with implicitly restarted arnoldi methods," vol. 6. Soc. for Industrial and Applied Math, 1998.

[82] E. Liberty and S. W. Zucker, "The mailman algorithm: A note on matrix-vector multiplication," *Information Processing Letters*, vol. 109, no. 3, pp. 179–182, 2009.

[83] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[84] Z. Lin, M. Chen, and Y. Ma, "The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices," *arXiv preprint arXiv:1009.5055*, 2010.

[85] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *Proc. ICML*, Haifa, Israel, June 2010, pp. 663–670.

[86] W. Liu, J. He, and S.-F. Chang, "Large graph construction for scalable semi-supervised learning," in *Proceedings of Intl. Conf. on Machine Learning*, 2010, pp. 679–686.

[87] S. Lloyd, "Least-squares quantization in PCM," *IEEE Trans. Info. Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[88] E. Loper and S. Bird, "Nltk: The natural language toolkit," in *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ser. ETMTNLP '02. Stroudsburg, PA, USA: Association for Computational Linguistics, 2002, pp. 63–70. [Online]. Available: https://doi.org/10.3115/1118108.1118117

[89] D. Lowd and C. Meek, "Adversarial learning," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005, pp. 641–647.

[90] C. Lu, H. Min, Z. Zhao, L. Zhu, D. Huang, and S. Yan, "Robust and efficient subspace segmentation via least-squares regression," in *European Conference on Computer Vision*. Florence, Italy: Springer, 2012, pp. 347–360.

[91] Q. Lu and L. Getoor, "Link-based classification," in *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, ser. ICML'03. AAAI Press, 2003, pp. 496–503. [Online]. Available: http://dl.acm.org/citation.cfm?id=3041838.3041901

[92] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy data coding and compression," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 29, no. 9, pp. 1546–1562, 2007.

[93] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger, "The penn treebank: Annotating predicate argument structure," in *Proceedings of the Workshop on Human Language Technology*, ser. HLT '94. Stroudsburg, PA, USA: Association for Computational Linguistics, 1994, pp. 114–119. [Online]. Available: https://doi.org/10.3115/1075812.1075835

[94] MATLAB, *version 8.6.0 (R2015b)*. Natick, Massachusetts: The MathWorks Inc., 2015.

[95] A. K. Menon, "Large-scale support vector machines: algorithms and theory," 2009.

[96] M. Micsinai, F. Parisi, F. Strino, P. Asp, B. D. Dynlacht, and Y. Kluger, "Picking chip-seq peak detectors for analyzing chromatin modification experiments," *Nucleic Acids Research*, vol. 40, no. 9, May 2012.

[97] G. Namata, B. London, L. Getoor, and B. Huang, "Query-driven active surveying for collective classification," 2012.

[98] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-100)," CUCS-006-96, Tech. Rep., 1996.

[99] A. T. Nguyen, B. C. Wallace, J. J. Li, A. Nenkova, and M. Lease, "Aggregating and Predicting Sequence Labels from Crowd Annotations," *Proc Conf Assoc Comput Linguist Meet*, vol. 2017, pp. 299–309, 2017.

[100] N. Nguyen and Y. Guo, "Comparisons of sequence labeling algorithms and extensions," in *Proceedings of the 24th International Conference on Machine Learning*, ser. ICML '07. New York, NY, USA: ACM, 2007, pp. 681–688. [Online]. Available: http://doi.acm.org/10.1145/1273496.1273582

[101] G. O' Connor. (2014) Moore's law gives way to Bezos' law. [Online]. Available: https://gigaom.com/2014/04/19/moores-law-gives-way-to-bezoss-law/

[102] J. Orasanu and E. Salas, "Team decision making in complex environments." 1993.

[103] Y. Panagakis and C. Kotropoulos, "Elastic net subspace clustering applied to pop/rock music structure analysis," *Pattern Recognition Letters*, vol. 38, pp. 46–53, 2014.

[104] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against deep learning systems using adversarial examples," *arXiv preprint arXiv:1602.02697*, 2016.

[105] N. Parikh, S. Boyd *et al.*, "Proximal algorithms," *Foundations and Trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[106] F. Parisi, F. Strino, B. Nadler, and Y. Kluger, "Ranking and combining multiple predictors without labeled data," *Proc. of the Ntl. Academy of Sciences*, vol. 111, no. 4, pp. 1253–1258, 2014.

[107] Y. Park, S. Park, S.-g. Lee, and W. Jung, "Greedy filtering: A scalable algorithm for k-nearest neighbor graph construction," in *International Conference on Database Systems for Advanced Applications*. Bali, Indonesia: Springer, 2014, pp. 327–341.

[108] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," *ACM SIGKDD Explorations Newsletter*, vol. 6, no. 1, pp. 90–105, 2004.

[109] X. Peng, H. Tang, L. Zhang, Z. Yi, and S. Xiao, "A unified framework for representation-based subspace clustering of out-of-sample and large-scale data." *IEEE Trans. on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2499–2512, 2016.

[110] D. Pimentel-Alarcón, L. Balzano, and R. Nowak, "Necessary and sufficient conditions for sketched subspace clustering," in *54th Annual Allerton Conference on Communication, Control, and Computing*. Champaign, IL: IEEE, 2016, pp. 1335–1343.

[111] H. V. Poor, *An Introduction to Signal Detection and Estimation*. Springer Science & Business Media, 2013.

[112] F. Pourkamali-Anaraki and S. Becker, "Preconditioned data sparsification for big data with applications to PCA and K-means," *IEEE Transactions on Information Theory*, vol. 63, no. 5, pp. 2954–2974, 2017.

[113] F. Pourkamali-Anaraki and S. Hughes, "Memory and computation efficient pca via very sparse random projections," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014, pp. 1341–1349.

[114] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb 1989.

[115] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in neural information processing systems*, 2008, pp. 1177–1184.

[116] M. Rahmani and G. Atia, "Innovation pursuit: A new approach to subspace clustering," *arXiv preprint arXiv:1512.00907*, 2015.

[117] M. Razaviyayn, M. Hong, and Z.-Q. Luo, "A unified convergence analysis of block successive minimization methods for nonsmooth optimization," *SIAM Journal on Optimization*, vol. 23, no. 2, pp. 1126–1153, 2013.

[118] F. Rodrigues, M. Lourenço, B. Ribeiro, and F. C. Pereira, "Learning supervised topic models for classification and regression from crowds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2409–2422, Dec 2017.

[119] F. Rodrigues, F. Pereira, and B. Ribeiro, "Sequence labeling with multiple annotators," *Machine Learning*, vol. 95, no. 2, pp. 165–181, May 2014. [Online]. Available: https://doi.org/10.1007/s10994-013-5411-2

[120] ——, "Gaussian process classification and active learning with multiple annotators," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 2. Bejing, China: PMLR, 22–24 Jun 2014, pp. 433–441. [Online]. Available: http://proceedings.mlr.press/v32/rodrigues14.html

[121] P. Ruiz, P. Morales-Álvarez, R. Molina, and A. Katsaggelos, "Learning from crowds with variational gaussian processes," *Pattern Recognition*, vol. 88, pp. 298–311, April 2019. [Online]. Available: http://decsai.ugr.es/vip/files/journals/1-s2.0-S0031320318304060-main.pdf

[122] T. Sarlos, "Improved approximation algorithms for large matrices via random projections," in *47th Annual IEEE Symposium on Foundations of Computer Science*, Berkeley, CA, 2006, pp. 143–152.

[123] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, 2009.

[124] J. Shen, P. Li, and H. Xu, "Online low-rank subspace clustering by basis dictionary pursuit," in *Proceedings of The 33rd International Conference on Machine Learning*, New York, NY, 2016, pp. 622–631.

[125] V. S. Sheng, F. Provost, and P. G. Ipeirotis, "Get another label? improving data quality and data mining using multiple, noisy labelers," in *Proceedings of the 14th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*. Las Vegas, NV: ACM, 2008, pp. 614–622.

[126] N. D. Sidiropoulos, L. De Lathauwer, X. Fu, K. Huang, E. E. Papalexakis, and C. Faloutsos, "Tensor decomposition for signal processing and machine learning," *IEEE Transactions on Signal Processing*, vol. 65, no. 13, pp. 3551–3582, 2017.

[127] M. Slaney and M. Casey, "Locality-sensitive hashing for finding nearest neighbors [lecture notes]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 128–131, 2008.

[128] A. J. Smola and B. Schölkopf, *Learning with kernels*. GMD-Forschungszentrum Informationstechnik, 1998.

[129] M. Sørensen and L. De Lathauwer, "Coupled canonical polyadic decompositions and (coupled) decompositions in multilinear rank-$(L_{r,n}, L_{r,n}, 1)$ terms—part I: Uniqueness," *SIAM Journal on Matrix Analysis and Applications*, vol. 36, no. 2, pp. 496–522, 2015.

[130] A. Timmermann, "Forecast combinations," *Handbook of Economic Forecasting*, vol. 1, pp. 135–196, 2006.

[131] M. Tipping and C. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural Computation*, vol. 11, no. 2, pp. 443–482, 1999.

[132] P. A. Traganitis and G. B. Giannakis, "Efficient subspace clustering of large-scale data streams with misses," in *Annual Conference on Information Science and Systems*. Princeton, NJ: IEEE, March 2016.

[133] ——, "Blind multi-class ensemble learning with dependent classifiers," in *Proc. of the 26th European Signal Processing Conference (EUSIPCO)*, Rome, Italy, Sep 2018.

[134] P. A. Traganitis, A. Pagès-Zamora, and G. B. Giannakis, "Blind multiclass ensemble classification," *IEEE Transactions on Signal Processing*, vol. 66, no. 18, pp. 4737–4752, Sept 2018.

[135] P. A. Traganitis, K. Slavakis, and G. B. Giannakis, "Sketch and validate for big data clustering," *IEEE J. Selected Topics Signal Processing*, vol. 9, no. 4, pp. 678–690, June 2015.

[136] P. A. Traganitis and G. B. Giannakis, "Efficient subspace clustering of large-scale data streams with misses," in *Annual Conference on Information Science and Systems*. Princeton, NJ: IEEE, 2016, pp. 590–595.

[137] ——, "PARAFAC-based multilinear subspace clustering for tensor data," in *IEEE Global Conference on Signal and Information Processing*. Washington DC: IEEE, 2016, pp. 1280–1284.

[138] P. A. Traganitis, A. Pagès-Zamora, and G. B. Giannakis, "Learning from unequally reliable blind ensembles of classifiers," in *Proc. of the 5th IEEE Global Conference on Signal and Information Processing*. Montreal, CA: IEEE, 2017.

[139] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv:1704.03453*, 2017.

[140] R. Tron and R. Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, 2007, pp. 1–8.

[141] J. N. Tsitsiklis, "Decentralized detection," *Advances in Statistical Signal Processing*, vol. 2, no. 2, pp. 297–344, 1993.

[142] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer Science and Business Media, 2013.

[143] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008.

[144] N. Vervliet, O. Debals, L. Sorber, M. Van Barel, and L. De Lathauwer, "Tensorlab 3.0," *available online, URL: www. tensorlab. net*, 2016.

[145] R. Vidal, "A tutorial on subspace clustering," *IEEE Signal Process. Magazine*, vol. 28, no. 2, pp. 52–68, 2010.

[146] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (GPCA)," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 27, no. 12, pp. 1945–1959, 2005.

[147] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, Aug. 2007.

[148] M. P. Wand and M. C. Jones, *Kernel Smoothing*. CRC Press, 1994.

[149] Y. Wang, Y.-X. Wang, and A. Singh, "A theoretical analysis of noisy sparse subspace clustering on dimensionality-reduced data," *arXiv preprint arXiv:1610.07650*, 2016.

[150] P. Welinder, S. Branson, P. Perona, and S. J. Belongie, "The multidimensional wisdom of crowds," in *Advances in Neural Information Processing Systems 23*, J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, Eds. Curran Associates, Inc., 2010, pp. 2424–2432. [Online]. Available: http://papers.nips.cc/paper/4074-the-multidimensional-wisdom-of-crowds.pdf

[151] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Foundations and Trends in Theoretical Computer Science*, vol. 10, no. 1–2, pp. 1–157, 2014.

[152] F. Wright, C. De Vito, B. Langer, A. Hunter *et al.*, "Multidisciplinary cancer conferences: A systematic review and development of practice standards," *European Journal of Cancer*, vol. 43, pp. 1002–1010, 2007.

[153] F. Xia, L. T. Yang, L. Wang, and A. Vinel, "Internet of things," *International Journal of Communication Systems*, vol. 25, no. 9, p. 1101, 2012.

[154] Y. Yan, R. Rosales, G. Fung, R. Subramanian, and J. Dy, "Learning from multiple annotators with varying expertise," *Machine Learning*, vol. 95, no. 3, pp. 291–327, Jun 2014. [Online]. Available: https://doi.org/10.1007/s10994-013-5412-1

[155] Y. Yang, M. Pilanci, and M. J. Wainwright, "Randomized sketches for kernels: Fast and optimal non-parametric regression," *arXiv preprint arXiv:1501.06195*, 2015.

[156] C. You, D. Robinson, and R. Vidal, "Scalable sparse subspace clustering by orthogonal matching pursuit," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 2016.

[157] C. You, C.-G. Li, D. P. Robinson, and R. Vidal, "Oracle based active set algorithm for scalable elastic net subspace clustering," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition*, Las Vegas, NV, June 2016.

[158] J. Zhang, "The mean field theory in em procedures for markov random fields," *IEEE Transactions on Signal Processing*, vol. 40, no. 10, pp. 2570–2583, Oct 1992.

[159] L. Zhang, D. Romero, and G. B. Giannakis, "Fast convergent algorithms for multi-kernel regression," in *2016 IEEE Statistical Signal Processing Workshop*. IEEE, 2016, pp. 1–4.

[160] T. Zhang, A. Szlam, and G. Lerman, "Median k-flats for hybrid linear modeling with many outliers," in *Proc. of ICCV*. Kyoto, Japan: IEEE, September 2009, pp. 234–241.

[161] T. Zhang, A. Szlam, Y. Wang, and G. Lerman, "Hybrid linear modeling via local best-fit flats," *Intern. J. Computer Vision*, vol. 100, no. 3, pp. 217–240, 2012.

[162] Y. Zhang, X. Chen, D. Zhou, and M. I. Jordan, "Spectral methods meet EM: A provably optimal algorithm for crowdsourcing," in *Advances in Neural Information Processing Systems*, 2014, pp. 1260–1268.

Appendix A

Proofs for Chapter 2

A.1 Supporting Lemmata

The following lemmata will be used to assist in the proofs of the propositions and theorems.

Lemma A.1. *[122, Corollary 11]* Consider an $N \times k$ orthonormal matrix $\mathbf{V}$ with $N \geq k$, and a JLT(ε, δ, k) matrix $\mathbf{R}$ of size $N \times n$. If $n = \mathcal{O}(k\frac{\log(k/\varepsilon)}{\varepsilon^2} f(\delta))$, then the following holds w.p. at least $1 - \delta$

$$1 - \varepsilon \leq \sigma_i^2(\mathbf{V}^\top \mathbf{R}) \leq 1 + \varepsilon \quad \text{for } i = 1, \ldots, k \tag{A.1}$$

where $\sigma_i(\mathbf{V}^\top \mathbf{R})$ denotes the i-th singular value of $\mathbf{V}^\top \mathbf{R}$.

Lemma A.2. *[18, Lemma 8]* Let $\varepsilon > 0$, and consider the $n \times k$ orthonormal matrix $\mathbf{V}$ with $n > k$, as well as the $n \times r$ matrix $\mathbf{R}$, with $r > k$ satisfying $1 - \varepsilon \leq \sigma_i^2(\mathbf{V}^\top \mathbf{R}) \leq 1 + \varepsilon$ for $i = 1, \ldots, k$. It then holds deterministically that

$$\|(\mathbf{V}^\top \mathbf{R})^\dagger - (\mathbf{V}^\top \mathbf{R})^\top\|_2 \leq \frac{\varepsilon}{\sqrt{1 - \varepsilon}}. \tag{A.2}$$

A.2 Main proofs

Proposition 2.1. Let $\mathbf{X}$ be a $D \times N$ matrix such that $\text{rank}(\mathbf{X}) = \rho$, and define the $D \times n$ matrix $\mathbf{B} := \mathbf{X}\mathbf{R}$, where $\mathbf{R}$ is a JLT(ε, δ, D) of size $N \times n$. If $n = \mathcal{O}(\rho\frac{\log(\rho/\varepsilon)}{\varepsilon^2} f(\delta))$ then w.p. at least $1 - \delta$, it holds that

$$\text{range}(\mathbf{X}) = \text{range}(\mathbf{B}).$$

Proof. Let $\mathbf{X} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top$ be the SVD of $\mathbf{X}$. Since $\mathbf{V}_\rho$ is invertible and $\mathbf{\Sigma}_\rho$ is diagonal, it holds that

$$\text{range}(\mathbf{X}) = \text{range}(\mathbf{U}_\rho) \tag{2.3}$$

i.e., the columns of $\mathbf{X}$ can be written as linear combinations of the columns of $\mathbf{U}_\rho$ and vice versa. Now consider $\mathbf{B} = \mathbf{XR} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top \mathbf{R} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \tilde{\mathbf{V}}_\rho^\top$, where $\tilde{\mathbf{V}}_\rho := \mathbf{R}^\top \mathbf{V}_\rho$, which implies $\text{range}(\mathbf{B}) \subseteq \text{range}(\mathbf{U}_\rho)$. By Lemma A.1 $\tilde{\mathbf{V}}_\rho^\top := \mathbf{V}_\rho^\top \mathbf{R}$ is full row rank w.p. at least $1 - \delta$ and thus

$$\mathbf{B}(\tilde{\mathbf{V}}^\top)^\dagger = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \tag{2.4}$$

which implies that $\text{range}(\mathbf{U}_\rho) = \text{range}(\mathbf{B}) = \text{range}(\mathbf{X})$, where the last equality is due to (2.3). □

Proposition 2.2. Let $\mathbf{X}$ be a $D \times N$ matrix such that $\text{rank}(\mathbf{X}) = \rho$, and define the $D \times n$ matrix $\mathbf{B} := \mathbf{XR}$, where $\mathbf{R}$ is a JLT(ε, δ, D) of size $N \times n$. If $n = \mathcal{O}(r \frac{\log(r/\varepsilon)}{\varepsilon^2} f(\delta))$, then w.p. at least $1 - 2\delta$ it holds that

$$\|\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \mathbf{U}_r \mathbf{\Sigma}_r\|_F \leq (\varepsilon \frac{\sqrt{1+\varepsilon}}{\sqrt{1-\varepsilon}} + 1 + \varepsilon) \|\bar{\mathbf{X}}_r\|_F.$$

Proof. From the first part of the proof of Prop. 2.1 we have that $\text{range}(\mathbf{B}) \subseteq \text{range}(\mathbf{U}_\rho)$. Now consider

$$\mathbf{B} = \mathbf{XR} = \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^\top \mathbf{R} + \bar{\mathbf{U}}_r \bar{\mathbf{\Sigma}}_r \bar{\mathbf{V}}_r^\top \mathbf{R} \tag{2.5}$$

By Lemma A.1 $\mathbf{V}_r^\top \mathbf{R}$ is full row rank w.p. at least $1 - \delta$; thus, right multiplying (2.5) with $(\mathbf{V}_r^\top \mathbf{R})^\dagger$ yields $\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger = \mathbf{U}_r \mathbf{\Sigma}_r + \bar{\mathbf{U}}_r \bar{\mathbf{\Sigma}}_r \bar{\mathbf{V}}_r^\top \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^\dagger$, or

$$\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \mathbf{U}_r \mathbf{\Sigma}_r = \bar{\mathbf{U}}_r \bar{\mathbf{\Sigma}}_r \bar{\mathbf{V}}_r^\top \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^\dagger$$

which upon substituting $\bar{\mathbf{X}}_r$ boils down to

$$\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \mathbf{U}_r \mathbf{\Sigma}_r = \bar{\mathbf{X}}_r \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \bar{\mathbf{X}}_r \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^\top + \bar{\mathbf{X}}_r \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^{\top \cdot} \tag{2.6}$$

Using the triangle inequality, and the spectral submultiplicativity of the Frobenius norm, yields

$$\|\mathbf{B}(\mathbf{V}_r^\top \mathbf{R})^\dagger - \mathbf{U}_r \mathbf{\Sigma}_r\|_F = \|\bar{\mathbf{X}}_r \mathbf{R} \left((\mathbf{V}_r^\top \mathbf{R})^\dagger - (\mathbf{V}_r^\top \mathbf{R})^\top \right) \|_F + \|\bar{\mathbf{X}}_r \mathbf{R}(\mathbf{V}_r^\top \mathbf{R})^\top\|_F \tag{2.7}$$

$$\leq \|\bar{\mathbf{X}}_r\mathbf{R}\|_F \, \|(\mathbf{V}_r^\top\mathbf{R})^\dagger - (\mathbf{V}_r^\top\mathbf{R})^\top\|_2 + \|\bar{\mathbf{X}}_r\mathbf{R}\|_F \, \|(\mathbf{V}_r^\top\mathbf{R})^\top\|_2.$$

We have from Def. 2.1 $\|\bar{\mathbf{X}}_r\mathbf{R}\|_F \leq \sqrt{1+\varepsilon}\|\bar{\mathbf{X}}_r\|_F$ w.p. at least $1-\delta$, while Lemma A.1 ensures $\|(\mathbf{V}_r^\top\mathbf{R})^\top\|_2 \leq \sqrt{1+\varepsilon}$ w.p. at least $1-\delta$. Since Lemma A.2 also implies that $\|(\mathbf{V}_r^\top\mathbf{R})^\dagger - (\mathbf{V}_r^\top\mathbf{R})^\top\|_2 \leq \frac{\varepsilon}{\sqrt{1-\varepsilon}}$ we arrive at [cf. 2.7]

$$\|\mathbf{B}(\mathbf{V}_r^\top\mathbf{R})^\dagger - \mathbf{U}_r\mathbf{\Sigma}_r\|_F \leq (\varepsilon\frac{\sqrt{1+\varepsilon}}{\sqrt{1-\varepsilon}} + 1 + \varepsilon)\|\bar{\mathbf{X}}_r\|_F. \tag{2.8}$$

□

Theorem 2.1. Consider noise-free and normalized data vectors obeying (2.3) with $\boldsymbol{v}_i \equiv 0$, to form columns of a $D \times N$ data matrix $\mathbf{X}$, with unit ℓ_2 norm per column, and rank$(\mathbf{X}) = \rho$. Let also $\mathbf{R}$ denote JLT(ε, δ, D) of size $N \times n$. Let $\boldsymbol{g}^*(\boldsymbol{x}) := \mathbf{X}\boldsymbol{a}^* = \boldsymbol{x}$ denote the ground-truth representation of $\boldsymbol{x}$, and $\hat{\boldsymbol{g}}(\boldsymbol{x}) := \mathbf{X}\mathbf{R}\hat{\boldsymbol{a}}$ the representation given by Sketch-LSR. If $n = \mathcal{O}(r\frac{\log(r/\varepsilon)}{\varepsilon^2}f(\delta))$, then the following bound holds w.p. at least $1-2\delta$

$$\|\boldsymbol{g}^*(\boldsymbol{x}) - \hat{\boldsymbol{g}}(\boldsymbol{x})\|_2 \leq \lambda \, (1 + \sqrt{\frac{1+\varepsilon}{1-\varepsilon}} \, \sqrt{\rho - r} \, \sigma_{r+1}^2) + \frac{1}{\sqrt{1-\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denotes the $(r+1)$st singular value of $\mathbf{X}$.

Proof. The proof will follow the steps in [155]. Consider the Sketch-LSR objective for $\boldsymbol{x}$, namely

$$\frac{\lambda}{2}\|\boldsymbol{x} - \mathbf{X}\mathbf{R}\boldsymbol{a}\|_2^2 + \|\boldsymbol{a}\|_2^2 \tag{2.9}$$

and the SVD $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$. As $\mathbf{U}$ is unitary, minimizing (2.9) is equivalent to minimizing

$$\frac{\lambda}{2}\|\mathbf{U}^\top\boldsymbol{x} - \mathbf{\Sigma}\tilde{\mathbf{V}}^\top\boldsymbol{a}\|_2^2 + \|\boldsymbol{a}\|_2^2 \tag{2.10}$$

where $\tilde{\mathbf{V}}^\top := \mathbf{V}^\top\mathbf{R}$. Now, decompose the dataset as

$$\mathbf{X} = \mathbf{X}_r + \bar{\mathbf{X}}_r \tag{2.11}$$

where $\mathbf{X}_r := \mathbf{U}_r\mathbf{\Sigma}_r\mathbf{V}_r^\top$ and $\bar{\mathbf{X}}_r := \bar{\mathbf{U}}_r\bar{\mathbf{\Sigma}}_r\bar{\mathbf{V}}_r^\top$. Using (2.11), we can rewrite (2.10) as

$$\frac{\lambda}{2}\underbrace{\|\boldsymbol{\chi}_r - \mathbf{\Sigma}_r\tilde{\mathbf{V}}_r^\top\boldsymbol{a}\|_2^2}_{:=T_1^2} + \frac{\lambda}{2}\underbrace{\|\bar{\boldsymbol{\chi}}_r - \bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\boldsymbol{a}\|_2^2}_{:=T_2^2} + \underbrace{\|\boldsymbol{a}\|_2^2}_{:=T_3^2} \tag{2.12}$$

where $\boldsymbol{\chi}_r := \mathbf{U}_r^\top\boldsymbol{x}$, and $\bar{\boldsymbol{\chi}}_r := \bar{\mathbf{U}}_r^\top\boldsymbol{x}$. Selecting $\boldsymbol{a}$ as

$$\boldsymbol{a} = \tilde{\mathbf{V}}_r(\tilde{\mathbf{V}}_r^\top\tilde{\mathbf{V}}_r)^{-1}\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r$$

T_1^2 vanishes, and T_2 reduces to

$$T_2 = \|\bar{\boldsymbol{\chi}}_r - \bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\tilde{\mathbf{V}}_r(\tilde{\mathbf{V}}_r^\top\tilde{\mathbf{V}}_r)^{-1}\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2. \tag{2.13}$$

The triangle inequality and the submultiplicativity of the ℓ_2 norm, allows us to bound T_2 as

$$T_2 \leq \|\bar{\boldsymbol{\chi}}_r\|_2 + \|\bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\|_2\,\|\tilde{\mathbf{V}}_r(\tilde{\mathbf{V}}_r^\top\tilde{\mathbf{V}}_r)^{-1}\|_2\|\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2. \tag{2.14}$$

Now note that $\|\bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\|_2 \leq \|\bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\|_F = \|\bar{\mathbf{U}}_r\bar{\mathbf{\Sigma}}_r\bar{\tilde{\mathbf{V}}}_r^\top\|_F = \|\bar{\mathbf{X}}_r\mathbf{R}\|_F$ and recall from Def. 2.1 that $\|\bar{\mathbf{X}}_r\mathbf{R}\|_F \leq \sqrt{1+\varepsilon}\|\bar{\mathbf{X}}_r\|_F \leq \sqrt{1+\varepsilon}\sqrt{\rho-r}\|\bar{\mathbf{X}}_r\|_2 \leq \sqrt{1+\varepsilon}\sqrt{\rho-r}\;\sigma_{r+1}^2$ w.p. at least $1-\delta$. By Lemma A.1 $\tilde{\mathbf{V}}_r^\top = \mathbf{V}_r^\top\mathbf{R}$ is full row rank w.p. at least $1-\delta$; thus, $\tilde{\mathbf{V}}_r(\tilde{\mathbf{V}}_r^\top\tilde{\mathbf{V}}_r)^{-1} = \tilde{\mathbf{V}}_r^\dagger$, and $\|\tilde{\mathbf{V}}_r^\dagger\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}}$. Furthermore, $\|\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2 = \|\mathbf{V}_r\mathbf{\Sigma}_r^{-1}\mathbf{U}_r^\top\boldsymbol{x}\|_2 \leq 1$, and $\|\bar{\boldsymbol{\chi}}_r\|_2 = \|\bar{\mathbf{U}}_r^\top\boldsymbol{x}\|_2 = 1$. Similarly, T_3 in (2.12) can be bounded w.p. at least $1-\delta$ due to Lemma A.1 as

$$T_3 = \|\tilde{\mathbf{V}}_r(\tilde{\mathbf{V}}_r^\top\tilde{\mathbf{V}}_r)^{-1}\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2 = \|\tilde{\mathbf{V}}_r^\dagger\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2 \leq \|\tilde{\mathbf{V}}_r^\dagger\|_2\,\|\mathbf{\Sigma}_r^{-1}\boldsymbol{\chi}_r\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}} \tag{2.15}$$

Finally, since the chosen $\boldsymbol{a}$ in (2.12) satisfies (2.14) and (2.15), so will do any minimizer $\hat{\boldsymbol{a}}$ of (2.9). □

Corollary 2.1. Consider the setting of Thm. 2.1, and let $\hat{\boldsymbol{g}}(\boldsymbol{x}) := \mathbf{X}\mathbf{R}\hat{\boldsymbol{a}}$ be the representation of a datum given by Sketch-SSC. The following bound holds w.p. at least $1-2\delta$

$$\|\boldsymbol{g}^*(\boldsymbol{x}) - \hat{\boldsymbol{g}}(\boldsymbol{x})\|_2 \leq \lambda\,(1 + \sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\;\sqrt{\rho-r}\;\sigma_{r+1}^2) + \sqrt{\frac{n}{1-\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denotes the $(r+1)$st singular value of $\mathbf{X}$.

Proof. Consider the Sketch-SSC objective for $\boldsymbol{x}$, namely

$$\frac{\lambda}{2}\underbrace{\|\boldsymbol{x}-\mathbf{X}\mathbf{R}\boldsymbol{a}\|_2^2}_{:=T_1^2}+\underbrace{\|\boldsymbol{a}\|_1}_{:=T_2}. \tag{2.16}$$

From Thm. 2.1 we have $T_1 \leq \lambda\,(1+\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\,\sqrt{\rho-r}\,\sigma_{r+1}^2)$, and $\|\boldsymbol{a}\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}}$. Since for any $n\times 1$ vector $\boldsymbol{z}$ it holds that $\|\boldsymbol{z}\|_1 \leq \sqrt{n}\|\boldsymbol{z}\|_2$, we have $T_2 \leq \sqrt{n}\|\boldsymbol{a}\|_2 \leq \sqrt{\frac{n}{1-\varepsilon}}$ yielding the claim of the corollary. □

Corollary 2.2. Consider the setting of Thm. 2.1, and let $\boldsymbol{g}^*(\mathbf{X}) := \mathbf{X}\mathbf{Z}$ and $\hat{\boldsymbol{g}}(\mathbf{X}) := \mathbf{X}\mathbf{R}\hat{\mathbf{A}}$ be the representations of all the data given by LRR and Sketch-LRR respectively. The following bound holds w.p. at least $1-2\delta$

$$\|\boldsymbol{g}^*(\mathbf{X})-\hat{\boldsymbol{g}}(\mathbf{X})\|_F \leq \lambda\,(\sqrt{N}+\sqrt{\frac{1+\varepsilon}{1-\varepsilon}}\,\sqrt{\rho-r}\,\sigma_{r+1}^2)+\sqrt{\frac{n}{1-\varepsilon}}$$

with λ as in (2.12), and σ_{r+1} denoting the $(r+1)$st singular value of $\mathbf{X}$.

Proof. Consider the Sketch-LRR objective for $\mathbf{X}$, namely

$$\frac{\lambda}{2}\underbrace{\|\mathbf{X}-\mathbf{X}\mathbf{R}\mathbf{A}\|_F^2}_{:=T_1^2}+\underbrace{\|\mathbf{A}\|_*}_{:=T_2}. \tag{2.17}$$

As with Corr. 2.1, T_1 can be bounded using the results of Thm. 2.1, and $\|\mathbf{A}\|_F \leq \frac{1}{\sqrt{1-\varepsilon}}$. Since for any rank n matrix $\mathbf{Z}$ it holds that $\|\mathbf{Z}\|_* \leq \sqrt{n}\|\mathbf{Z}\|_F$ we have $T_2 \leq \sqrt{n}\|\mathbf{A}\|_F \leq \sqrt{\frac{n}{1-\varepsilon}}$, yielding the claim of the corollary. □

Proposition 2.3. Consider $\boldsymbol{x}_i = \mathbf{X}\boldsymbol{z}_i$ and $\boldsymbol{x}_j = \mathbf{X}\boldsymbol{z}_j$, and their representation provided by SSC, LRR or LSR $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$, respectively. Let $\rho = \text{rank}(\mathbf{X})$ and $\boldsymbol{a}_i$, $\boldsymbol{a}_j$ be the representation obtained by the corresponding Sketch algorithm of Section 3.2; that is, $\boldsymbol{x}_i = \mathbf{X}\mathbf{R}\boldsymbol{a}_i$, where the $N\times n$ matrix $\mathbf{R}$ is a JLT(ε,δ,D). If $n = \mathcal{O}(\rho\frac{\log(\rho/\varepsilon)}{\varepsilon^2}f(\delta))$, then w.p. at least $1-\delta$ it holds that

$$\frac{1}{\sqrt{1+\varepsilon}}\|\boldsymbol{z}_i-\boldsymbol{z}_j\|_2 \leq \|\boldsymbol{a}_i-\boldsymbol{a}_j\|_2 \leq \frac{1}{\sqrt{1-\varepsilon}}\|\boldsymbol{z}_i-\boldsymbol{z}_j\|_2.$$

Proof. By definition, we have $\boldsymbol{x}_i = \mathbf{X}\boldsymbol{z}_i = \mathbf{X}\mathbf{R}\boldsymbol{a}_i$, and thus

$$\mathbf{X}(\boldsymbol{z}_i-\boldsymbol{z}_j) = \mathbf{X}\mathbf{R}(\boldsymbol{a}_i-\boldsymbol{a}_j) = \boldsymbol{x}_i-\boldsymbol{x}_j. \tag{2.18}$$

Let $\mathbf{X} = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top$, and rewrite (2.18) as

$$\mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top (\boldsymbol{z}_i - \boldsymbol{z}_j) = \mathbf{U}_\rho \mathbf{\Sigma}_\rho \mathbf{V}_\rho^\top \mathbf{R}(\boldsymbol{a}_i - \boldsymbol{a}_j). \tag{2.19}$$

Left-multiplying by $\mathbf{\Sigma}_\rho^{-1} \mathbf{U}_\rho^\top$ reduces (2.19) to

$$\mathbf{V}_\rho^\top (\boldsymbol{z}_i - \boldsymbol{z}_j) = \mathbf{V}_\rho^\top \mathbf{R}(\boldsymbol{a}_i - \boldsymbol{a}_j). \tag{2.20}$$

Taking the norm of both sides, and noting that $\mathbf{V}$ is an orthonormal matrix implies that

$$\|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 = \|\mathbf{R}(\boldsymbol{a}_i - \boldsymbol{a}_j)\|_2 \tag{2.21}$$

which upon recalling Def. 2.1 yields

$$\begin{aligned} \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 &\leq \sqrt{1+\varepsilon} \|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2, \\ \sqrt{1-\varepsilon} \|\boldsymbol{a}_i - \boldsymbol{a}_j\|_2 &\leq \|\boldsymbol{z}_i - \boldsymbol{z}_j\|_2 \end{aligned} \tag{2.22}$$

w.p. at least $1 - \delta$. □

Appendix B

Algorithmic details for Chapter 2

B.1 ADMM algorithm for (2.15)

Consider the Sketch-SSC for a single datum $\boldsymbol{x}$

$$\min_{\boldsymbol{a}} \frac{\lambda}{2}\|\boldsymbol{x} - \mathbf{B}\boldsymbol{a}\|_2^2 + \|\boldsymbol{a}\|_1 \tag{B.1}$$

The optimization problem of (B.1) will be solved using the alternating direction method of multipliers [49]. Define a new $n \times 1$ vector of auxiliary variables $\boldsymbol{c}$, and consider the following optimization problem that is equivalent to (B.1)

$$\begin{aligned} &\min_{\boldsymbol{a},\boldsymbol{c}} \frac{\lambda}{2}\|\boldsymbol{x} - \mathbf{B}\boldsymbol{a}\|_2^2 + \|\boldsymbol{c}\|_1 \\ &\text{s. to. } \boldsymbol{a} = \boldsymbol{c}. \end{aligned} \tag{B.2}$$

The augmented Lagrangian of (B.2) is

$$\mathcal{L} = \frac{\lambda}{2}\|\boldsymbol{x} - \mathbf{B}\boldsymbol{a}\|_2^2 + \|\boldsymbol{c}\|_1 + \frac{\nu}{2}\|\boldsymbol{a} - \boldsymbol{c} + \boldsymbol{\delta}\|_2^2 \tag{B.3}$$

where $\boldsymbol{\delta}$ is a $n \times 1$ vector of dual variables and $\nu > 0$ is a penalty parameter. At each ADMM iteration the variables $\boldsymbol{a}, \boldsymbol{c}$ are updated by setting the gradient of $\mathcal{L}$ w.r.t. $\boldsymbol{a}$ and $\boldsymbol{c}$ respectively to $\mathbf{0}$. Furthermore, the dual variables $\boldsymbol{\delta}$ are updated using a gradient ascent step at each iteration.

Algorithm B.1 ADMM solver of Sketch-SSC [cf. (2.15)]

Input: $D \times N$ data matrix $\mathbf{X}$; $D \times n$ basis $\mathbf{B}$; regularization parameter λ;
Output: Model matrix $\mathbf{A}$;

for Each datum $\boldsymbol{x}_j$ to $\boldsymbol{x}_N$ **do**
  Initialize $\boldsymbol{a}_j[0], \boldsymbol{c}[0], \boldsymbol{\delta}[0]$
  repeat
    Compute $\boldsymbol{a}_j[i+1]$ using (B.4)
    Compute $\boldsymbol{c}[i+1]$ using (B.5)
    Compute $\boldsymbol{\delta}[i+1]$ using (B.7)
    Update iteration counter $i \leftarrow i+1$
  until convergence
end for
$\mathbf{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_N]$.

The update of $\boldsymbol{a}$ at the i-th iteration is given by

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \boldsymbol{a}} = -\lambda \mathbf{B}^\top (\boldsymbol{x} - \mathbf{B}\boldsymbol{a}) + \nu(\boldsymbol{a} - \boldsymbol{c} + \boldsymbol{\delta}) = \mathbf{0} \Rightarrow \\ &\boldsymbol{a}[i+1] = (\lambda \mathbf{B}^\top \mathbf{B} + \nu \mathbf{I})^{-1} (\lambda \mathbf{B}^\top \boldsymbol{x} + \nu(\boldsymbol{c}[i] - \boldsymbol{\delta}[i])) \end{aligned} \tag{B.4}$$

where brackets indicate ADMM iteration indices. Accordingly, the update for $\boldsymbol{c}$ is given by

$$\boldsymbol{c}[i+1] = \mathcal{T}_{1/\nu}(\boldsymbol{a}[i+1] + \boldsymbol{\delta}[i]) \tag{B.5}$$

where $\mathcal{T}_\sigma(\cdot)$ denotes the element-wise soft-thresholding operator

$$\mathcal{T}_\sigma(z) := \begin{cases} z - \sigma & \text{if } z > \sigma \\ 0 & \text{if } |z| \leq \sigma \\ z + \sigma & \text{if } z < -\sigma \end{cases} . \tag{B.6}$$

Finally, $\boldsymbol{\delta}$ is updated as

$$\boldsymbol{\delta}[i+1] = \boldsymbol{\delta}[i] + \boldsymbol{a}[i+1] - \boldsymbol{c}[i+1]. \tag{B.7}$$

The entire process is listed in Alg. B.1.

B.2 ALM algorithm for (2.16)

Consider the Sketch-LRR

$$\min_{\mathbf{A}} \frac{\lambda}{2}\|\mathbf{X}-\mathbf{B}\mathbf{A}\|_F^2 + \|\mathbf{A}\|_* \tag{B.8}$$

The optimization problem of (B.8) will be solved using the augmented Lagrangian method (ALM) [84, 85]. Define a new $n \times N$ matrix of auxiliary variables $\mathbf{C}$, and consider the following optimization task that is equivalent to (B.8)

$$\begin{aligned} &\min_{\mathbf{A},\mathbf{C}} \frac{\lambda}{2}\|\mathbf{X}-\mathbf{B}\mathbf{A}\|_F^2 + \|\mathbf{C}\|_* \\ &\text{s. to. } \mathbf{A}=\mathbf{C} \end{aligned} \tag{B.9}$$

The augmented Lagrangian of (B.9) is

$$\mathcal{L} = \frac{\lambda}{2}\|\mathbf{X}-\mathbf{B}\mathbf{A}\|_F^2 + \|\mathbf{C}\|_* + \frac{\nu}{2}\|\mathbf{A}-\mathbf{C}+\mathbf{\Delta}\|_F^2 \tag{B.10}$$

where $\mathbf{\Delta}$ is a $n \times N$ matrix of dual variables and $\nu > 0$ is a penalty parameter. At each ALM iteration the variables $\mathbf{A}, \mathbf{C}$ are updated by setting the gradient of $\mathcal{L}$ w.r.t. $\mathbf{A}$ and $\mathbf{C}$ respectively to $\mathbf{0}$. Furthermore, the dual variables $\mathbf{\Delta}$ are updated using a gradient ascent step per iteration. The update of $\mathbf{A}$ at the i-th iteration is given by

$$\begin{aligned} &\frac{\partial \mathcal{L}}{\partial \mathbf{A}} = \mathbf{0} \Rightarrow \\ &\mathbf{A}[i+1] = (\lambda \mathbf{B}^\top \mathbf{B} + \nu \mathbf{I})^{-1}(\lambda \mathbf{B}^\top \mathbf{X} - \nu(\mathbf{C}[i] - \mathbf{\Delta}[i])) \end{aligned} \tag{B.11}$$

where brackets indicate ALM iteration indices. Accordingly, the update for $\mathbf{C}$ is given by

$$\mathbf{C}[i+1] = \arg\min_{\mathbf{C}} \frac{1}{\nu}\|\mathbf{C}\|_* + \frac{1}{2}\|\mathbf{C} - (\mathbf{A}[i+1] + \mathbf{\Delta}[i])\|_F^2. \tag{B.12}$$

Note that the update (B.12) can be performed using the Singular Value Thresholding algorithm [23]. Finally $\mathbf{\Delta}$ is updated as

$$\mathbf{\Delta}[i+1] = \mathbf{\Delta}[i] + \mathbf{A}[i+1] - \mathbf{C}[i+1] \tag{B.13}$$

Algorithm B.2 ALM solver of Sketch-LRR [cf. (2.16)]

Input: $D \times N$ data matrix $\mathbf{X}$; $D \times n$ basis $\mathbf{B}$; regularization parameter λ;
Output: Model matrix $\mathbf{A}$;

1: Initialize $\mathbf{A}, \mathbf{C}, \boldsymbol{\Delta}$
2: **repeat**
3: Compute $\mathbf{A}[i+1]$ using (B.11)
4: Compute $\mathbf{C}[i+1]$ using (B.12)
5: Compute $\boldsymbol{\delta}[i+1]$ using (B.13)
6: Update ν using (B.14)
7: Update iteration counter $i \leftarrow i+1$
8: **until** convergence

and the penalty parameter is also updated as

$$\nu = \min(p\nu, \nu_{\max}) \tag{B.14}$$

where $p > 1$ is a prescribed constant, and $\nu_{\max}$ is a predefined maximum limit for ν.

Appendix C

Proofs for Chapter 3

Proof of Lemma 3.1. Suppose that $\text{rank}(\mathbf{\Gamma}_m) = \text{rank}(\mathbf{\Gamma}_{m'}) = \text{rank}(\mathbf{\Gamma}_{m''}) = K$, for some $m \neq m', m''$ and $m' \neq m''$. Then by [126, Thm. 2] the decomposition of $\underline{\Psi}_{mm'm''}$ is *essentially unique*. Invoking [129, Prop 4.10] the joint tensor decomposition of (3.25) is *essentially unique*, meaning the solutions of (3.25) will be of the form

$$\hat{\mathbf{\Gamma}}_m = \mathbf{\Gamma}_m^* \mathbf{P} \mathbf{\Lambda}_m, \quad m = 1, \ldots, M, \quad \hat{\boldsymbol{\pi}} = \mathbf{\Lambda} \mathbf{P}^\top \boldsymbol{\pi}^*$$

where $\mathbf{P}$ is a permutation matrix, and $\{\mathbf{\Lambda}_m\}_{m=1}^M$, $\mathbf{\Lambda}$ are diagonal scaling matrices such that $\mathbf{\Lambda}_m \mathbf{\Lambda}_{m'} \mathbf{\Lambda}_{m''} = \mathbf{\Lambda}^{-1}$, for $m \neq m', m'', m' \neq m''$. Since $\{\hat{\mathbf{\Gamma}}_m\}$ and $\hat{\boldsymbol{\pi}}$ are the solutions to (3.25), they must satisfy the constraints of the optimization problem; that is $\hat{\mathbf{\Gamma}}_m \in \mathcal{C} \quad m = 1, \ldots, M$ and $\hat{\boldsymbol{\pi}} \in \mathcal{C}_p$. Since ${\mathbf{\Gamma}_m^*}^\top \mathbf{1} = \mathbf{1}$ for all m, and $\mathbf{P}^\top \mathbf{1} = \mathbf{1}$, we have

$$\hat{\mathbf{\Gamma}}_m^\top \mathbf{1} = \mathbf{1} \Rightarrow \mathbf{\Lambda}_m \mathbf{P}^\top {\mathbf{\Gamma}_m^*}^\top \mathbf{1} = \mathbf{1} \Rightarrow \mathbf{\Lambda}_m \mathbf{1} = \mathbf{1} \;\; m = 1, \ldots, M$$

which implies that $\mathbf{\Lambda}_m = \mathbf{I}$ for $m = 1, \ldots, M$. Since $\mathbf{\Lambda}_m \mathbf{\Lambda}_{m'} \mathbf{\Lambda}_{m''} = \mathbf{\Lambda}^{-1}$, for $m \neq m', m''$, $m' \neq m''$, we arrive at $\mathbf{\Lambda} = \mathbf{I}$. Thus, the constraints of (3.25) solve the possible scaling ambiguities. Letting $\hat{\mathbf{P}} = \mathbf{P}^\top = \mathbf{P}^{-1}$, we arrive at the statement of the lemma. □

Proof of Theorem 3.1. For notational convenience, collect all optimization variables in $\boldsymbol{\theta}$, and denote the aggregated constraint set as $\bar{\mathcal{C}}$. Note that $\bar{\mathcal{C}}$ is a compact set, since the probability simplex is compact and $\bar{\mathcal{C}}$ is an intersection of simplexes. Since $h_N(\boldsymbol{\theta})$ is continuous and $\bar{\mathcal{C}}$ is compact, $h_N(\boldsymbol{\theta})$ is uniformly continuous on $\bar{\mathcal{C}}$, that is, $\forall \varepsilon > 0$ there exists a neighborhood $\mathcal{V}$ of

$\tilde{\boldsymbol{\theta}}$ such that

$$\sup_{\boldsymbol{\theta}\in\mathcal{V}\cap\bar{\mathcal{C}}} |h_N(\boldsymbol{\theta}) - h_N(\tilde{\boldsymbol{\theta}})| < \varepsilon/2. \tag{C.1}$$

Due to the compactness of $\bar{\mathcal{C}}$ there exist a finite number of points $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_L \in \bar{\mathcal{C}}$, with corresponding neighborhoods $\mathcal{V}_1, \ldots, \mathcal{V}_L$ that cover $\bar{\mathcal{C}}$, that is

$$\sup_{\boldsymbol{\theta}\in\mathcal{V}_\ell\cap\bar{\mathcal{C}}} |h_N(\boldsymbol{\theta}) - h_N(\boldsymbol{\theta}_\ell)| < \varepsilon/2, \quad \text{for } \ell = 1, \ldots, L. \tag{C.2}$$

Invoking the LLN, it is straightforward to show that, for sufficiently large N, w.p. 1

$$|h_N(\boldsymbol{\theta}_\ell) - h_\infty(\boldsymbol{\theta}_\ell)| < \varepsilon/2, \quad \text{for } \ell = 1, \ldots, L. \tag{C.3}$$

Using the triangle inequality along with (C.2), and (C.3) we have

$$\sup_{\boldsymbol{\theta}\in\bar{\mathcal{C}}} |h_N(\boldsymbol{\theta}) - h_\infty(\boldsymbol{\theta})| < \varepsilon, \tag{C.4}$$

that is, for sufficiently large N, h_N converges uniformly to h_∞ on $\bar{\mathcal{C}}$. Then, by [123, Thm. 5.3] we have that $D(\mathcal{S}_N, \mathcal{S}_*) \to 0$ as $N \to \infty$. □

Proof of Theorem 3.2. Let $\bar{L}(x|k) = L(x|k)\pi_k$, with $L(x|k)$ as defined in Sec. 3.2.1. Then the average probability of error of the MAP detector can be expressed as

$$\mathrm{P_e} = \sum_{k=1}^{K} \mathrm{P_{e,k}}\pi_k \tag{C.5}$$

where $\mathrm{P_{e,k}} = \Pr(\bar{L}(x|k) < \bar{L}(x|k'), k' \neq k | Y = k)$. By applying a union bound on $\mathrm{P_{e,k}}$ it is easy to show that

$$\mathrm{P_{e,k}} \leq \sum_{k'\neq k} \Pr(\bar{L}(x|k) < \bar{L}(x|k') | Y = k). \tag{C.6}$$

Defining $\mathrm{P}_{\bar{L}}(k, k') := \Pr(\bar{L}(x|k) < \bar{L}(x|k') | Y = k)$, substituting (C.6) in (C.5) and grouping terms we have

$$\mathrm{P_e} \leq \sum_{k=1}^{K} \sum_{k'>k}^{K} \pi_k \mathrm{P}_{\bar{L}}(k, k') + \pi_{k'} \mathrm{P}_{\bar{L}}(k', k). \tag{C.7}$$

Consider now the binary hypothesis testing problem between classes k and $k' \neq k$. The average probability of error of a MAP detector for the binary problem is

$$\mathrm{P_e}(k,k') = \frac{\pi_k}{\pi_k + \pi_{k'}} \mathrm{P}_{\bar{L}}(k,k') + \frac{\pi_{k'}}{\pi_k + \pi_{k'}} \mathrm{P}_{\bar{L}}(k',k). \tag{C.8}$$

Then

$$\pi_k \mathrm{P}_{\bar{L}}(k,k') + \pi_{k'} \mathrm{P}_{\bar{L}}(k',k) = (\pi_k + \pi_{k'}) \mathrm{P_e}(k,k') \leq \mathrm{P_e}(k,k') \tag{C.9}$$

where the inequality is due to $\pi_k + \pi_{k'} \leq 1$. Combining (C.9) with (C.7) yields

$$\mathrm{P_e} \leq \sum_{k=1}^{K} \sum_{k'>k}^{K} \mathrm{P_e}(k,k'). \tag{C.10}$$

Therefore, we have upper bounded the average probability of error of our M-class hypothesis testing problem by the average error probabilities of binary hypothesis testing problems. For the binary hypothesis testing problem between classes k and $k' \neq k$, collect all annotator responses in an $M \times 1$ vector $\tilde{\boldsymbol{f}}$ and define two complementary regions $\mathcal{R}$ and $\mathcal{R}^C$ as

$$\mathcal{R} = \{\tilde{\boldsymbol{f}} : \bar{L}(x|k) < \bar{L}(x|k')\} \tag{C.11a}$$

$$\mathcal{R}^C = \{\tilde{\boldsymbol{f}} : \bar{L}(x|k') < \bar{L}(x|k)\}. \tag{C.11b}$$

Upon defining $\tilde{\pi}_{k,k'} = \frac{\pi_k}{\pi_k + \pi_{k'}}$ and using (C.11), (C.8) can be rewritten as

$$\begin{aligned} \mathrm{P_e}(k,k') &= \Pr(\tilde{\boldsymbol{f}} \in \mathcal{R} | Y = k) \tilde{\pi}_{k,k'} + \Pr(\tilde{\boldsymbol{f}} \in \mathcal{R}^C | Y = k') \tilde{\pi}_{k',k} \\ &= \prod_{m=1}^{M} \Pr([\tilde{\boldsymbol{f}}]_m \in \mathcal{R}_m | Y = k) \tilde{\pi}_{k,k'} + \prod_{m=1}^{M} \Pr([\tilde{\boldsymbol{f}}]_m \in \mathcal{R}_m^C | Y = k') \tilde{\pi}_{k',k} \end{aligned} \tag{C.12}$$

where the second equality follows from As. 1 and $\mathcal{R}_m, \mathcal{R}_m^C$ denote the subsets of $\mathcal{R}, \mathcal{R}^C$ corresponding to the m-th entry of $\tilde{\boldsymbol{f}}$, respectively. Now let

$$m^* = \arg\max_m \Pr([\tilde{\boldsymbol{f}}]_m \in \mathcal{R}_m | Y = k)^M \tilde{\pi}_{k,k'} + \Pr([\tilde{\boldsymbol{f}}]_m \in \mathcal{R}_m^C | Y = k')^M \tilde{\pi}_{k',k} \tag{C.13}$$

and define

$$\bar{\mathrm{P}}_e(k,k') = \Pr([\tilde{\boldsymbol{f}}]_{m^*} \in \mathcal{R}_{m^*} | Y=k)^M \tilde{\pi}_{k,k'} + \Pr([\tilde{\boldsymbol{f}}]_{m^*} \in \mathcal{R}_{m^*}^C | Y=k')^M \tilde{\pi}_{k',k}. \quad \text{(C.14)}$$

Clearly $\mathrm{P_e}(k,k') \leq \bar{\mathrm{P}}_e(k,k')$. From standard results in detection theory (C.14) can be bounded as [28, 111]

$$\bar{\mathrm{P}}_e(k,k') \leq \exp(-Md(p||q)) \quad \text{(C.15)}$$

where $p := \Pr([\tilde{\boldsymbol{f}}]_{m^*} \in \mathcal{R}_{m^*} | Y=k)$, $q := \Pr([\tilde{\boldsymbol{f}}]_{m^*} \in \mathcal{R}_{m^*}^C | Y=k')$, and $d(p||q)$ denotes the Chernoff information between pdfs p and q. Combining (C.15) with (C.10) yields the claim of the theorem. □

Appendix D

Algorithmic details for Chapter 3

D.1 ADMM subproblem for prior probabilities

Consider the following problem that is equivalent to (3.26)

$$\begin{aligned} \min_{\boldsymbol{\pi},\boldsymbol{\phi}} \quad & g_{N,\boldsymbol{\pi}}(\boldsymbol{\phi}) + \rho_{\mathcal{C}_p}(\boldsymbol{\pi}) \\ \text{s.to} \quad & \boldsymbol{\pi} = \boldsymbol{\phi} \end{aligned} \tag{D.1}$$

where $\boldsymbol{\phi}$ is an auxiliary variable used to capture the smooth part of the optimization problem, and $\rho_{\mathcal{C}_p}$ is an indicator function for the constraints of (3.26), namely

$$\rho_{\mathcal{C}_p}(\boldsymbol{u}) := \begin{cases} 0 & \text{if } \boldsymbol{u} \in \mathcal{C}_p \\ \infty & \text{otherwise.} \end{cases} \tag{D.2}$$

The augmented Lagrangian of (D.1) is then

$$\ell = g_{N,\boldsymbol{\pi}}(\boldsymbol{\phi}) + \rho_{\mathcal{C}_p}(\boldsymbol{\pi}) + \frac{\lambda}{2}\|\boldsymbol{\pi} - \boldsymbol{\phi} + \boldsymbol{\delta}\|_2^2 \tag{D.3}$$

where the $K \times 1$ vector $\boldsymbol{\delta}$ contains the scaled Lagrange multipliers for subproblem (3.26). Per ADMM iteration, (D.3) is minimized w.r.t. $\boldsymbol{\phi}$ and $\boldsymbol{\pi}$ before performing a gradient ascent step for $\boldsymbol{\delta}$. Specifically, the update for $\boldsymbol{\phi}$ at iteration $i+1$ is obtained by setting the gradient of ℓ w.r.t. $\boldsymbol{\phi}$

to $\mathbf{0}$, and solving for $\boldsymbol{\phi}$; that is,

$$\begin{aligned}
&\Bigg((\lambda+\nu)\mathbf{I}+\sum_{m=1}^{M}\boldsymbol{\Gamma}_m^\top\boldsymbol{\Gamma}_m+\sum_{\substack{m=1\\m'>m}}^{M}\mathbf{K}_{m'm}^\top\mathbf{K}_{m'm}\\
&+\sum_{\substack{m=1\\m'>m\\m''>m'}}^{M}(\boldsymbol{\Gamma}_{m''}\odot\mathbf{K}_{m'm})^\top(\boldsymbol{\Gamma}_{m''}\odot\mathbf{K}_{m'm})\Bigg)\boldsymbol{\phi}[i+1]\\
&=\sum_{m=1}^{M}\boldsymbol{\Gamma}_m^\top\boldsymbol{\mu}_m+\sum_{\substack{m=1\\m'>m}}^{M}\mathbf{K}_{m'm}^\top\boldsymbol{s}_{mm'}+\nu\boldsymbol{\pi}^{(\mathrm{prev})}\\
&+\lambda(\boldsymbol{\pi}[i]+\boldsymbol{\delta}[i])+\sum_{\substack{m=1\\m'>m\\m''>m'}}(\boldsymbol{\Gamma}_{m''}\odot\mathbf{K}_{m'm})^\top\boldsymbol{t}_{mm'm''},
\end{aligned}\tag{D.4}$$

where $\mathbf{K}_{mm'}:=\boldsymbol{\Gamma}_m\odot\boldsymbol{\Gamma}_{m'}$. Brackets here indicate ADMM iteration indices. Accordingly, the update for $\boldsymbol{\pi}$ is given by

$$\boldsymbol{\pi}[i+1]=P_{\mathcal{C}_p}\big(\boldsymbol{\phi}[i+1]-\boldsymbol{\delta}[i]\big)\tag{D.5}$$

where $P_{\mathcal{C}_p}$ is the projection operator onto the convex set $\mathcal{C}_p$; that is, $\boldsymbol{\phi}[i+1]-\boldsymbol{\delta}[i]$ is projected onto the probability simplex. This projection can be performed using efficient methods [35]. Finally, a gradient ascent step is performed for $\boldsymbol{\delta}$ as

$$\boldsymbol{\delta}[i+1]=\boldsymbol{\delta}[i]+\boldsymbol{\pi}[i+1]-\boldsymbol{\phi}[i+1].\tag{D.6}$$

Note that products of the form $\mathbf{K}_{m'm}^\top\mathbf{K}_{m'm}=(\boldsymbol{\Gamma}_m\odot\boldsymbol{\Gamma}_{m'})^\top(\boldsymbol{\Gamma}_m\odot\boldsymbol{\Gamma}_{m'})$ can be efficiently computed by using the following observation: $(\boldsymbol{\Gamma}_m\odot\boldsymbol{\Gamma}_{m'})^\top(\boldsymbol{\Gamma}_m\odot\boldsymbol{\Gamma}_{m'})=(\boldsymbol{\Gamma}_m^\top\boldsymbol{\Gamma}_m)*(\boldsymbol{\Gamma}_{m'}^\top\boldsymbol{\Gamma}_{m'})$, where $*$ denotes the elementwise matrix product [126]. In addition, the products $\boldsymbol{\Gamma}_m^\top\boldsymbol{\Gamma}_m$ do not have to be explicitly computed each time (D.1) is solved, as they can be cached every time (D.7) is solved. As suggested in [62], the maximum number of ADMM iterations, I, for each subproblem can be set to be small, e.g. $I=10$.

D.2 ADMM subproblem for confusion matrices

Proceeding along similar lines with the previous subsection, consider the following problem which is equivalent to (3.27)

$$\begin{aligned} \min_{\boldsymbol{\Gamma}_m, \boldsymbol{\Phi}} \quad & \bar{g}_{N,m}(\boldsymbol{\Gamma}_m, \boldsymbol{\Phi}) \\ \text{s.to} \quad & \boldsymbol{\Gamma}_m = \boldsymbol{\Phi}^\top \end{aligned} \tag{D.7}$$

where $\boldsymbol{\Phi}$ is an auxiliary variable used to capture the smooth part of the optimization problem in (3.27), and

$$\bar{g}_{N,m}(\boldsymbol{\Gamma}_m, \boldsymbol{\Phi}) = g_{N,m}(\boldsymbol{\Phi}^\top) + \rho_{\mathcal{C}}(\boldsymbol{\Gamma}_m).$$

The augmented Lagrangian of (D.7) is then

$$\ell' = \bar{g}_{N,m}(\boldsymbol{\Gamma}_m, \boldsymbol{\Phi}) + \frac{\lambda}{2}\|\boldsymbol{\Gamma}_m - \boldsymbol{\Phi}^\top + \boldsymbol{\Delta}_m\|_F^2 \tag{D.8}$$

where the $K \times K$ matrix $\boldsymbol{\Delta}_m$ contains the scaled Lagrange multipliers for subproblem (3.27), and λ is a positive scalar. As in the previous section, per ADMM iteration, (D.8) is minimized with respect to (w.r.t.) $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}_m$ before performing a gradient ascent step for $\boldsymbol{\Delta}_m$. Specifically, the update for $\boldsymbol{\Phi}$ at iteration $i+1$ is obtained by setting the gradient of ℓ' w.r.t. $\boldsymbol{\Phi}$ to $\mathbf{0}$, and solving for $\boldsymbol{\Phi}$. Since $\mathbf{S}_{m'm} = \mathbf{S}_{mm'}^\top$ and $\boldsymbol{\Pi} = \boldsymbol{\Pi}^\top$, it is easy to see that the update w.r.t. $\boldsymbol{\Phi}$ can be expressed as

$$\begin{aligned} &\Bigg((\lambda+\nu)\mathbf{I} + \boldsymbol{\pi}\boldsymbol{\pi}^\top + \sum_{m' \neq m}^{M} \boldsymbol{\Pi}\boldsymbol{\Gamma}_{m'}^\top \boldsymbol{\Gamma}_{m'}\boldsymbol{\Pi} \\ &+ \sum_{\substack{m'>m \\ m''>m'}} \boldsymbol{\Pi}\mathbf{K}_{m''m'}^\top \mathbf{K}_{m''m'}\boldsymbol{\Pi}\Bigg)\boldsymbol{\Phi}[i+1] \\ &= \boldsymbol{\pi}\boldsymbol{\mu}_m^\top + \sum_{m' \neq m}^{M} \boldsymbol{\Pi}\boldsymbol{\Gamma}_{m'}^\top \mathbf{S}_{m'm} + \sum_{\substack{m'>m \\ m''>m'}} \boldsymbol{\Pi}\mathbf{K}_{m''m'}^\top \mathbf{T}_{mm'm''}^{(1)} \\ &+ \nu\boldsymbol{\Gamma}_m^{(\text{prev})\top} + \lambda(\boldsymbol{\Gamma}_m[i] + \boldsymbol{\Delta}_m[i])^\top. \end{aligned} \tag{D.9}$$

Accordingly, the update for $\mathbf{\Gamma}_m$ is given by

$$\mathbf{\Gamma}_m[i+1] = P_{\mathcal{C}}\big(\mathbf{\Phi}^\top[i+1] - \mathbf{\Delta}_m[i]\big) \tag{D.10}$$

where $P_{\mathcal{C}}$ is the projection operator onto the convex set $\mathcal{C}$ with each column of $\mathbf{\Phi}^\top[i+1]-\mathbf{\Delta}_m[i]$ projected onto the probability simplex. Finally, a gradient ascent step is performed per $\mathbf{\Delta}_m$, as follows

$$\mathbf{\Delta}_m[i+1] = \mathbf{\Delta}_m[i] + \mathbf{\Gamma}_m[i+1] - \mathbf{\Phi}^\top[i+1]. \tag{D.11}$$

D.3 Algorithm complexity

For the ADMM subproblems of Apps. D.1 and D.2 the complexity per iteration is dominated by the matrix inversions required in (D.4) and (D.9) respectively, that is $\mathcal{O}(K^3)$. However, in order to instantiate the left- and right-hand sides of (D.4), $\mathcal{O}(M^3K^2)$ and $\mathcal{O}(M^3K^4)$ operations are required respectively. These operations have to be performed only once and cached to be used in each iteration. The increased complexity of the right-hand side is due to the matricized tensor times Khatri-Rao product (MTTKRP) $(\mathbf{\Gamma}_{m''} \odot \mathbf{K}_{m'm})^\top \boldsymbol{t}_{mm'm''}$. These MTTKRPs however, can be computed efficiently due to the Khatri-Rao structure, and are easily parallelizable, see e.g. [7]. This brings the overall complexity of App. D.1 to $\mathcal{O}(M^3K^4+IK^3)$, with I denoting the number of ADMM iterations. Accordingly, the operations required to instantiate the left- and right-hand sides of (D.9) are $\mathcal{O}(M^2K^2)$ and $\mathcal{O}(M^2K^4)$ respectively. This brings the total complexity of App. D.2 to $\mathcal{O}(M^2K^4 + IK^3)$. As the number of iterations for the ADMM algorithms of Apps. D.1 and D.2 is set to be small the overall computational complexity of Alg. 3.1 is $\mathcal{O}(I_T M^3K^4)$, where I_T is the number of AO-ADMM iterations required until convergence.

Furthermore, the number of tensors $\underline{T}_{mm'm''}$ required to solve (21) is $\binom{M}{3}$, while the number of matrices $\mathbf{S}_{mm'}$ required is $\binom{M}{2}$, and the number of vectors $\boldsymbol{\mu}_m$ is M. Thus, for K classes, the memory needed for storing all the tensors, matrices and vectors involved is $\mathcal{O}\left(\binom{M}{3}K^3 + \binom{M}{2}K^2 + MK\right)$. Finally, computing the cross-correlation tensors, matrices and mean vectors of annotators incurs a complexity of $\mathcal{O}(M^3KN)$ as each of the annotator response matrices $\{\mathbf{F}_m\}_{m=1}^M$ is of size $K \times N$ and has N nonzero entries.

Appendix E

The forward-backward algorithm

Let $b_{n,k}$ denote the probability of observing $\{f_m(x_n)\}_{m=1}^M$ given that $y_n = k$, that is

$$b_{n,k} = \prod_{m=1}^{M} \Pr(f_m(x_n)|y_n = k) = \prod_{m=1}^{M} \Gamma_m(f_m(x_n), k). \tag{E.1}$$

The forward-backward algorithm [114] seeks to efficiently calculate the probability of the observed variable sequence $\{f_m(x_n)\}_{n=1,m=1}^{N,M}$, given current HMM parameter estimates $\boldsymbol{\theta}$. The forward backward algorithm takes advantage of the fact that the past and future states of a Markov chain are independent given the current state. Specifically in our ensemble HMM case we can write

$$\Pr(\mathbf{F}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \Pr(\mathbf{F}_{1:n}, y_n = k; \boldsymbol{\theta}) \Pr(\mathbf{F}_{n+1:N}|y_n = k; \boldsymbol{\theta}), \tag{E.2}$$

where $\mathbf{F}_{1:n}$ is a matrix collecting all annotator responses for $n' = 1, \ldots, n$, and $\mathbf{F}_{n+1:N}$ is a matrix collecting annotator responses for $n' = n+1, \ldots, N$.

The forward backward algorithm computes the probability of the observed sequence iteratively using so-called forward and backward variables. Define the forward variable as

$$\alpha_{n,k} = \Pr(\mathbf{F}_{1:n}, y_n = k; \boldsymbol{\theta}) \tag{E.3}$$

Then let

$$\alpha_{1,k} = \Pr(y_1 = k) b_{1,k} \quad \text{for} \quad k = 1, \ldots, K. \tag{E.4}$$

and for $n = 1, \ldots, N$

$$\alpha_{n+1,k} = b_{n+1,k} \sum_{k'=1}^{K} \alpha_{n,k'} T(k, k') \tag{E.5}$$

The backward variables are defined as

$$\beta_{n,k} = \Pr(\mathbf{F}_{n+1:N} | y_n = k; \boldsymbol{\theta}) \tag{E.6}$$

$$\beta_{N,k} = 1 \quad \text{for } k = 1, \ldots, K \tag{E.7}$$

then for $n = N - 1, \ldots, 1$

$$\beta_{n,k} = \sum_{k'=1}^{K} T(k, k') \beta_{n+1,k'} b_{n+1,k'}. \tag{E.8}$$

All forward and backward variables can be computed iteratively using (E.5), (E.8). Having computed all forward and backward variables the probability of the observed variable sequence is given by

$$\Pr(\mathbf{F}|\boldsymbol{\theta}) = \sum_{k=1}^{K} \alpha_{n,k} \beta_{n,k}. \tag{E.9}$$

which holds for any $n \in \{1, \ldots, N\}$. Then the variables of interest $q_{nk}, \xi_n(k, k')$ can be computed as follows:

$$q_{nk} = \Pr(y_n = k|\mathbf{F}, \boldsymbol{\theta}) = \frac{\alpha_{n,k}\beta_{n,k}}{\sum_{k'=1}^{K} \alpha_{n,k'}\beta_{n,k'}}, \tag{E.10}$$

$$\xi_n(k, k') = \Pr(y_n = k, y_{n+1} = k'|\mathbf{F}, \boldsymbol{\theta}) = \frac{\alpha_{n,k} T(k, k') b_{n+1,k'} \beta_{n+1,k'}}{\sum_{k',k''=1}^{K} \alpha_{n,k'} T(k', k'') b_{n+1,k''} \beta_{n+1,k''}}. \tag{E.11}$$