April 2018

Permutation-Invariant Consensus over Crowdsourced Labels

Michael Joseph Giancola
Worcester Polytechnic Institute

Permutation-Invariant Consensus over Crowdsourced Labels

Advisors:

PROFESSORS JACOB WHITEHILL, RANDY PAFFENROTH

Written By:

MICHAEL GIANCOLA

A Major Qualifying Project
WORCESTER POLYTECHNIC INSTITUTE

Submitted to the Faculty of the Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Computer Science & Mathematical Sciences.

AUGUST 24TH, 2017 – MAY 1ST, 2018

Abstract

This Major Qualifying Project introduces a novel crowdsourcing consensus model and inference algorithm – which we call PICA (Permutation-Invariant Crowdsourcing Aggregation) – that is designed to recover the ground-truth labels of a dataset while being *invariant* to the class permutations enacted by the different annotators. This is particularly useful for settings in which annotators may have systematic confusions about the meanings of different classes, as well as clustering problems (e.g., dense pixel-wise image segmentation) in which the names/numbers assigned to each cluster have no inherent meaning. The PICA model is constructed by endowing each annotator with a doubly-stochastic matrix (DSM), which models the probabilities that an annotator will perceive one class and transcribe it into another. We conduct simulations and experiments to show the advantage of PICA compared to Majority Vote for three different clustering/labeling tasks. We also explore the conditions under which PICA provides better inference accuracy compared to a simpler but related model based on right-stochastic matrices by [1]. Finally, we show that PICA can be used to crowdsource responses for dense image segmentation tasks, and provide a proof-of-concept that aggregating responses in this way could improve the accuracy of this labor-intensive task. Our work will be published in HCOMP 2018 [2], and will be presented at the conference in Zurich, Switzerland from July 5-8, 2018.

Acknowledgements

The author wishes to thank his advisors, Professor Jacob Whitehill and Professor Randy Paffenroth, for their guidance while completing this Major Qualifying Project. Their advice, collaboration, and moral support led to a publication of this work in HCOMP 2018.

Notation Dictionary

Ω is the finite set of ground-truth labels.

z_j is a scalar random variable for the ground truth label corresponding to example j.

$\mathscr{L} = \{\mathsf{L}^{(1)}, \mathsf{L}^{(2)}, \ldots, \mathsf{L}^{(m)}\}$ is an indexed set called the set of *observed labels*.

$\mathsf{L}^{(i)}$ is a vector of *observed labels*. In particular, they are the labels given by annotator i.

$\mathsf{L}^{(i)}_j$ is the label given by annotator i to example j.

$\mathsf{S}^{(i)}$ is a $n \times n$ *doubly-stochastic* matrix corresponding to annotator i.

$\mathsf{S}^{(i)}_{zl} \in (0,1)$ gives the probability that annotator i perceives an example to belong to class $z \in \Omega$ but transcribes it as $l \in \Omega$.

$\mathsf{a}^{(i)} \in (0,1)$ gives the probability that annotator i perceives the correct latent category.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

In many crowdsourcing scenarios, annotators are asked to view some kind of data (an image, video, text passage, etc.) and to label it as belonging to some specific class from a set of mutually exclusive classes.[1] For example, annotators in a facial expression labeling task might be requested to view a set of face images and to label each face as displaying one of a finite set of facial emotions – e.g., satisfaction, sensory pleasure, and amusement [3]. Based on the raw labels, a variety of existing crowdsourcing consensus algorithms could be used to try to infer simultaneously both the ground-truth labels of the face images and also each individual labeler's accuracy score.

However, what if some annotators had a *systematic confusion* in the form of a *permutation* of what the different classes meant – e.g., whenever they saw an "amused" face, they labeled it as "satisfaction", and whenever they saw a face displaying "satisfaction", they labeled it as "amused". This is plausible, because while these terms are all well-defined (i.e. there is a single correct label for each image) it may not be clear to a layperson what those defining factors are. Moreover, the annotators' labels might still carry valuable information about which face images display the same facial expression. Namely, some annotators may be able to delineate the faces into groups, but don't know which group corresponds to which facial emotion. In order to accurately infer the ground-truth, the consensus algorithm would need to "un-permute" the labels assigned to the images according to the particular – and *a priori* unknown – misunderstanding of that labeler.

In other crowdsourcing settings – e.g., asking annotators to *cluster* a dataset – there may exist multiple ground-truth labelings that are all equally valid and are equivalent modulo a permutation of the class labels. For instance, when asking multiple annotators to perform dense image segmentation and to assign each pixel a color, the "name" assigned to each cluster (red,

[1]This chapter is based on the introduction to [2], which was co-written by the author of this paper and the author's advisors.

green, blue, etc.) may not carry any inherent meaning – what is important is which pixels belong to the same class. In this case, it would be useful to be able to aggregate over multiple annotators' votes while ignoring the particular name they assign to each class.

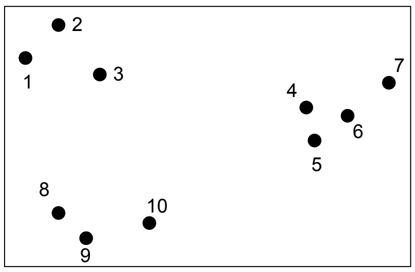
In this paper, we present a crowdsourcing consensus algorithm called PICA (Permutation-Invariant Crowdsourcing Aggregation), which is based on doubly-stochastic matrices (DSMs) and the Sinkhorn-Knopp algorithm [4–6]. PICA is designed to perform *permutation-invariant* simultaneous inference of the ground-truth labels and the individual annotators' *style* of labeling. In contrast to previous work on label aggregation for clustering settings (see section below), our method can benefit from harnessing a smaller number of degrees of freedom to reduce overfitting, and it can model class-specific accuracies.

1.1 Motivation

To convey the intuition behind the problem we are tackling, suppose three annotators were asked to cluster the points shown below into three clusters, and that the labels they assigned to the points are given in Figure 1.1. In particular, notice that, while annotators 1 and 2 agree mostly on which points are assigned to clusters 1, 2, and 3, respectively, annotator 3 has "permuted" 1,2,3 for 3,1,2 i.e. annotator 3 generally assigns points in cluster 1 to cluster 3, cluster 2 to cluster 1, and cluster 3 to cluster 2.

Motivating Example

Data



Labels

	1	2	3	4	5	6	7	8	9	10
Annotator 1	1	1	1	2	2	2	2	3	3	3
Annotator 2	1	1	1	2	2	2	1	3	3	2
Annotator 3	3	3	2	1	1	1	1	2	2	2
Majority Vote	1	1	1	2	2	2	**1**	3	3	**2**
PICA	1	1	1	2	2	2	2	3	3	3
Ground-truth	1	1	1	2	2	2	2	3	3	3

Figure 1.1: *Entries give the cluster which each annotator assigns to the given point. While the Majority Vote heuristic makes several mistakes (bolded), the proposed PICA algorithm recovers all the labels correctly.*

If we attempted to recover the true clustering simply by computing the Majority Vote for each example, we would make several inference errors due to differing – and arbitrary – permutations that the labelers might have when “naming” the different clusters (1, 2, and 3). In contrast, if we could somehow infer each annotator’s “style”, then we could recover the ground-truth with higher accuracy. This is the paramount component to our model. By simultaneously inferring a style matrix and the ground truth labels, we are able to “un-permute” the labels of each annotator to some canonical standard, at which point the most likely ground-truth label becomes clear. Indeed, when applying the PICA consensus algorithm proposed in this paper, we can recover the ground-truth labels with perfect accuracy, while Majority Vote achieves only 80% correct.

Chapter 2

Literature Review

Much of the content in this section is similar to the Related Work section in [2], which was written by the author of this paper and the author's advisors. The goal of this chapter is to provide a more in-depth analysis of the literature surrounding our work.

2.1 Overview

The work of this report is mainly concerned with crowdsourcing over clusters. We first discuss previous work in crowdsourcing consensus in general. In particular, we discuss several methods which utilize an Expectation-Maximization-based scheme (of which PICA is one). We then discuss works that focus specifically on crowdsourcing over clusters, and highlight the differences between their approaches and ours.

2.2 EM for Crowdsourcing Consensus

To the author's knowledge, the first use of Expectation-Maximization (EM) for crowdsourcing consensus was [7]. They were interested in modeling the error rate of clinicians' reports based on the questions they ask patients and the responses they receive.

Given some clinician k, and for each pair of responses $j \neq l$ in a finite set, they compute the number of times clinician k reports l when j is correct over the number of times j was correct. This serves as their estimator, which they use in an EM routine to iteratively determine the error rates of the clinicians and the true responses.

Many crowdsourcing consensus algorithms – including our model PICA – are based on the Expectation-Maximization scheme presented in [7].

2.3 Consensus over Complex Data

While early work on crowdsourcing consensus focused on binary and multiple-choice labeling tasks [1, 8, 9], consensus algorithms now exist for more complex label types. For instance, [10] crowdsourced bounding boxes in order to perform object localization. Some consensus algorithms were developed to label hierarchical data, including [11] for taxonomies and [12] for trees. Also, [13] developed a crowdsourcing method for labeling open-ended text responses. For example, they found that their model could solve SAT math questions better than majority vote.

2.4 Improving Performance

Some algorithms are sometimes able to achieve higher label inference accuracy by exploiting more detailed information about the annotators and the task itself. For instance, [8] modeled the difficulty of each example in addition to the accuracy of each labeler and ground truth label. Similarly, [14] modeled task clarity and worker reliability to discover schools of thought among labelers. Also, [15] modeled task features and worker biases to correct for task-dependent biases.

2.5 Ensemble of Clusterings

There exist several algorithms for computing a clustering based on an ensemble of clusterings. The work of [16] presents three algorithms, all of which use an EM-based approach to iteratively determine the cluster centers and then assign points to those clusters based upon some distance metric. [17] also utilizes EM - they infer the ground truth labels based on the observed labels, the prior probabilities of the clusters, and a set of scalar parameters. [18] represents clusters as hypergraphs and presents several heuristics for solving the cluster ensemble problem over hypergraphs.

2.6 Crowdsourcing over Clusterings

To date, the issue of how to conduct crowdsourcing consensus for *clusterings* has received relatively little research attention, despite the prevalent use of clustering for data visualization and data pre-processing. Several works [19–23] frame the clustering problem as a binary classification task, whereby each example is compared to every other example to assess whether or not they belong to the same cluster. By estimating the ground-truth binary decision value for each of the $\frac{d(d-1)}{2}$ unordered comparisons (for d examples), and by taking such annotator parameters into account as overall skill [19] and granularity [20], the ground-truth cluster labels of every example can be inferred. In both of these papers, the authors applied their technique to a dense image segmentation task, whereby the annotator partitions the set of pixels in an image into disjoint sets that represent different objects. Compared to other segmentation algorithms, their consensus-based approach yielded improved accuracy.

The work of [21] was concerned with uncovering a “master” clustering which captures the partitions of each annotator i.e. one annotator may cluster objects by size while another clusters by color; these “atomic” clusters can be subdivided in the master clustering. They discuss the use of an inference model based on the Variational Bayes method.

In [22], the authors introduced the concept of *semi-crowdsourced clustering*. The goal is to reconstruct the ideal pairwise similarities of all pairs of examples when only a subset of examples have crowdsourced labels. They propose an algorithm for learning a distance metric from the subset of crowdsourced pairwise similarities which they show to be robust to noise in the labels.

[23] considers pairwise comparisons as edges in a graph (where the nodes are the data examples). Since the complete graph is large, they compare two methods for partially observing the graph: random edge queries - where a random edge in the graph is observed - and random triangle queries - where a triplet is observed. They provide models which utilize both types of queries and show their ability to reconstruct a complete adjacency matrix for the graph of examples. However, when when edges are queried multiple times, [23] picks one. Conversely, our work considers *all* observed labels, and aggregates them all in determining the optimal clustering.

2.7 Our Approach

In our work, we frame the clustering problem differently: rather than conduct inference over the $d(d-1)/2$ unordered comparisons, we directly optimize the cluster labels assigned to the individual examples as n-way classification problem (for n clusters). Our model endows each annotator with his/her own *style* matrix that specifies how the ground-truth label assignment is mapped to the annotator’s own labels. This allows our model to capture *cluster-specific* accuracy characteristics – e.g., when performing dense image segmentation of a sky, some annotators might be better at distinguishing certain kinds of clouds than others. Moreover, our algorithm generalizes beyond consensus over clusterings: it can be used for inferring ground-truth for multiple-choice questions in which each annotator may permute his/her labels according to some fixed transformation matrix. In this sense, our algorithm offers a way of conducting *permutation-invariant* crowdsourcing consensus.

Finally, we note that our proposed PICA method is similar to the much older method by [1]; however, they did not explicitly test their model for clustering applications or permutation-invariant labeling settings. We discuss differences between and empirically compare the two models in greater detail later in the report.

Chapter 3

Doubly-Stochastic Matrices

3.1 Definition

For the purposes of this report, we define a *doubly-stochastic matrix* (DSM) to be a square matrix with values in $[0,1]$ in which each row and each column sums to 1. Equivalently, a matrix which is both *left-* and *right-* stochastic.

3.2 Motivation

We are interested in a structure for representing probability distributions over permutations, and DSMs are a natural choice. Following the work of [4], we utilize DSMs as a differentiable relaxation of permutation matrices, and this enables us to use continuous optimization methods such as gradient descent and Expectation-Maximization. However, in order to do so, some additional methods are necessary.

3.3 Methods

3.3.1 Sinkhorn Normalization

Sinkhorn Normalization is a method of transforming a square matrix A (which we call the *parameterizing matrix*) to a DSM. The algorithm iteratively performs row and column normalizations on A. [5] showed that this iterative normalization necessarily converges to a DSM if all the entries of A are strictly positive. The row and column normalization functions are given below:

$$R_{i,j}(M) = \frac{M_{i,j}}{\sum_{k=1}^{n} M_{i,k}} \qquad\qquad C_{i,j}(M) = \frac{M_{i,j}}{\sum_{k=1}^{n} M_{k,j}}$$

3.3.2 Sinkhorn Propagation

Sinkhorn Propagation (SinkProp) [4] utilizes Sinkhorn Normalization to optimize functions of DSMs. Starting with a strictly positive parameterizing matrix A, we perform alternating rounds (row, column, row, etc.) of Sinkhorn Normalization to arrive at a DSM T. In particular, we define the SinkProp function SP as:

$$\text{SP}^s(A) = \begin{cases} A & \text{if } s = 0 \\ C(R(\text{SP}^{s-1}(A))) & \text{otherwise} \end{cases} \tag{3.1}$$

where s is the number of Sinkhorn iterations and R and C are the row and column normalization functions applied element-wise to the matrix A. (For the experiments in this paper we found that $s = 25$ rounds was sufficient.)

After computing $T = \text{SP}^s(A)$ for some s, we can compute the gradient of f with respect to the parameterizing matrix A by back-propagating the gradient through the row and column normalizations. Since R and C are matrix-valued functions, we vectorize them so that the Jacobian of each function can be represented by a 2-D matrix (rather than a tensor). We thus apply the chain rule as:

$$\frac{\partial f}{\partial A} = \frac{\partial f}{\partial \text{vec}[C]} \frac{\partial \text{vec}[C]}{\partial \text{vec}[R]} \frac{\partial \text{vec}[R]}{\partial \text{vec}[C]} \cdots \frac{\partial \text{vec}[R]}{\partial \text{vec}[A]} \tag{3.2}$$

where

$$\frac{\partial f}{\partial \text{vec}[C]} = \begin{bmatrix} \frac{\partial f}{\partial C_{11}} & \cdots & \frac{\partial f}{\partial C_{1n}} & \frac{\partial f}{\partial C_{21}} & \cdots & \frac{\partial f}{\partial C_{nn}} \end{bmatrix}$$

$$\frac{\partial \text{vec}[C]}{\partial \text{vec}[R]} = \begin{bmatrix} \frac{\partial C_{11}}{\partial R_{11}} & \cdots & \frac{\partial C_{1n}}{\partial R_{11}} & \frac{\partial C_{21}}{\partial R_{11}} & \cdots & \frac{\partial C_{nn}}{\partial R_{11}} \\ \vdots & & & & & \\ \frac{\partial C_{11}}{\partial R_{1n}} & & & & & \\ \frac{\partial C_{11}}{\partial R_{21}} & & & & \ddots & \vdots \\ \vdots & & & & & \\ \frac{\partial C_{11}}{\partial R_{2n}} & & & & & \\ \vdots & & & & & \\ \frac{\partial C_{11}}{\partial R_{nn}} & & & & \cdots & \frac{\partial C_{nn}}{\partial R_{nn}} \end{bmatrix}$$

$$\frac{\partial C_{ij}}{\partial R_{xy}} = \delta(i,x)\left(\frac{\delta(j,y)}{\sum_{k=1}^{n} R_{ky}} - \frac{R_{xy}}{(\sum_{k=1}^{n} R_{ky})^2}\right)$$

Here, $\delta(\cdot)$ is the Kronecker delta function. The matrix $\frac{\partial R}{\partial C}$ has a similar structure to $\frac{\partial C}{\partial R}$. The gradient expression is the same as well, with indices transposed appropriately.

For an illustrative example of a SinkProp computation, see Appendix B.

3.3.3 Optimization in Log-Space

When using Sinkhorn Propagation within our inference algorithm, we set the parameterizing matrix $A = I$, where I is the identity matrix. Although A is required to be strictly positive to guarantee convergence of Sinkhorn Normalization [5], the SinkProp backpropagation doesn't guarantee that A will stay strictly positive after the gradient update. Hence, to maintain positivity, we conduct the optimization in log-space, and set $A' = \exp(A)$ (element-wise exponentiation) as the starting value for each $S^{(i)}$.

CHAPTER 4

MODEL DESCRIPTION

4.1 Notation

We use Futura font to denote random variables and Roman font to denote random draws of these variables. For instance, S and z are random variables, and S and z are random draws from these variables, respectively.

We use upper-case letters to denote matrices/vectors and lower-case letters to denote scalars. For example, S (a matrix) and z (a scalar) represent draws from random variables S and z, respectively.

4.2 Problem Description

Consider a dataset of d examples. Each example has a ground-truth label in the finite set Ω. Let $n = |\Omega|$. Our goal is to use crowdsourcing to determine the ground-truth label of each example j – which we represent with random variable $\mathsf{z}_j \in \Omega$ – for every example in our dataset. To do this, we collect a vector $\mathsf{L}^{(i)} \in (\Omega \cup \{\epsilon\})^d$ of labels from each of m different annotators, where $\mathsf{L}_j^{(i)}$ is the label given by annotator i to example j. If annotator i did not label example j, then we define $\mathsf{L}_j^{(i)} = \epsilon$. Together, the annotators' label vectors define an indexed set $\mathscr{L} = \{\mathsf{L}^{(1)}, \mathsf{L}^{(2)}, \ldots, \mathsf{L}^{(m)}\}$. $\mathscr{L}$ is called the set of *observed labels*.

We wish to model how different annotators who are labeling the same example may *perceive* the same latent category but *transcribe* the perceived category into different labels. For example, in clustering tasks, different annotators may use different names, numbers, or colors to denote the same set of clusters. We can model the process by which each annotator transcribes his/her perception into a label via a doubly-stochastic matrix, which we call the *style* of the annotator. In particular, our model endows each annotator i with an $n \times n$ matrix $\mathsf{S}^{(i)}$, where entry $\mathsf{S}^{(i)}_{zl} \in (0,1)$ gives the probability that the annotator perceives an example to belong to class $z \in \Omega$ but transcribes it as $l \in \Omega$. In addition to style, we also model each annotator's *accuracy*. In particular, we define random variable $\mathsf{a}^{(i)} \in (0,1)$ as the probability that annotator i *perceives* the correct category of some example. When conducting crowdsourcing consensus over all the annotators to infer the ground-truth labels, it is important to take both style and accuracy into account.

The dependence of the variables introduced in this section is shown graphically in Figure 4.1.

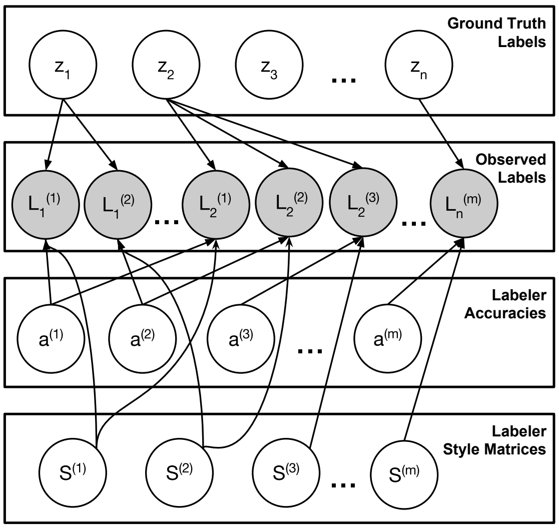


Figure 4.1: *Graphical model of ground-truth, observed labels, styles, and accuracies. Only the observed labels (shaded) are observed. The accuracies* $\mathsf{a}^{(1)},\ldots,\mathsf{a}^{(m)}$ *can be "folded" into the style matrices due to Lemma 4.1.*

4.3 Likelihood Model

We define the probability that annotator i assigns label $l \in \Omega$ to example j, given the ground-truth label z, style matrix S, and accuracy a as:

$$\begin{aligned}
& p(\mathsf{L}^{(i)}_j = l \mid \mathsf{z}_j = z, \mathsf{S}^{(i)} = S, \mathsf{a}^{(i)} = a) \\
\doteq\ & a \times S_{zl} + (1-a) \times \frac{\sum_{z' \neq z} S_{z'l}}{n-1} \\
=\ & \left(S \begin{bmatrix} a & \frac{1-a}{n-1} & \cdots & \\ \frac{1-a}{n-1} & a & \frac{1-a}{n-1} & \cdots \\ \vdots & & \ddots & \\ \frac{1-a}{n-1} & & & a \end{bmatrix} \right)_{zl}
\end{aligned}$$

(4.1)

The intuition is that there are two situations in which an annotator i would respond with some particular label l: either (i) they correctly perceived the example as belonging to class z but transcribed it into l (i.e., $a \times S_{zl}$), or (ii) they incorrectly perceived some other class z' but still transcribed it into l (i.e., $(1-a) \times \sum_{z' \neq z} S_{z'l}$). Notice that, in either case, it is possible that $l = z$. We assume that the probability of an incorrect perception $z' \neq z$ is distributed uniformly over all $(n-1)$ incorrect classes.

4.4 Simplifying the Likelihood Model

Notice that, in Equation 4.1, the style matrix S is right-multiplied by an accuracy matrix containing a on the diagonal and $\frac{1-a}{n-1}$ everywhere else. The accuracy matrix can easily be verified to be a DSM. Moreover, by making use of a simple lemma, we can simplify our model:

Lemma 4.1. *Let A and B be two arbitrary DSMs. Then AB is also a DSM.*

Proof. See Appendix A. ■

We can therefore "fold" the accuracy matrix into the style matrix, so that the latter expresses both the permutation and accuracy of the annotator. This enables us to simplify the likelihood model to be:

$$p(\mathsf{L}_j^{(i)} = l \mid \mathsf{z}_j = z, \mathsf{S}^{(i)} = S) = S_{zl} \tag{4.2}$$

This simplified model has an intuitive form. As opposed to a permutation matrix, which can only model discrete permutations, our model, utilizing a single DSM, can model a "soft" style whereby an annotator "usually" transcribes z into l but may sometimes transcribe it into l', etc.

4.5 Comparison to RSM-based Model

PICA is based on doubly-stochastic matrices (DSMs), which are a special case of right-stochastic matrices (RSMs) – non-negative matrices in which the rows, but not necessarily the columns, sum to 1. RSMs are therefore also capable of modeling the same kinds of "styles" that our proposed DSM-based model can model. RSM-based models for crowdsourcing consensus algorithms were first developed by [1]. Compared to RSM-based models, DSMs directly encode the fact that each annotator may permute his/her perception of the class into a different label. DSMs can have an advantage over RSMs due to a smaller number of degrees of freedom: where an $n \times n$ RSM has $n(n-1)$ free parameters, DSMs have only $(n-1)^2$, which for small n (i.e., a small number of classes or clusters) can be substantial in relative terms. When collecting only a modest number of labels per annotator, this reduction in free parameters can lead to better estimation of each annotator's style matrix and more accurate inference of the ground-truth labels.

4.6 Comparison to Unpermutation Heuristic

A simple way to perform permutation-invariant consensus is to “un-permute” the labels based on similarities between annotators. For example, if for some subset of examples Annotator 1 uses label A and Annotator 2 uses label B, it is likely A and B refer to the same class. Based on this idea, we constructed the following algorithm – which we refer to as the *Unpermutation Heuristic* – and for which we provide comparative results in the following sections.

Algorithm:

- Randomly pick an annotator i as the “leader”.
- Using i’s labels,
- For each annotator $j \neq i$,
    - For each class $k = 1, ..., n$
        - Find the symbol used most frequently by j to express class k according to i’s labels
        - Un-permute j’s labels based on the inferred permutation
- Conduct majority vote over the unpermuted labels of all annotators

Recall the toy example presented in Figure 1.1 with the new set of observed labels given in Table 4.1. Annotator 1 is the same as in the previous example (perfect accuracy, identity style). Annotator 2 and 3 also have an identity style, but have low accuracy. PICA is able to identify this and again recovers the ground-truth labels with perfect accuracy. Conversely, the Unpermutation Heuristic misinterprets the low accuracy of Annotators 2 and 3 as a style transformation and makes several mistakes.

It is important to note that the performance of the Unpermutation Heuristic is highly sensitive to the choice of the “leader”. This isn’t surprising; unpermuting the other annotators’ labels based on the labels of an inaccurate leader will propagate the error. The results of the Unpermutation Heuristic for each possible choice of leader are given in Table 4.1. The accuracy of the Unpermutation Heuristic for each leader is 80%, 90%, and 60% respectively, while PICA scores 100%.

	Labels									
	1	2	3	4	5	6	7	8	9	10
Annotator 1	1	1	1	2	2	2	2	3	3	3
Annotator 2	1	3	3	2	2	2	2	3	3	1
Annotator 3	1	2	1	2	1	3	1	1	2	3
Heuristic (L=1)	1	1	**3**	2	2	2	2	3	3	**1**
Heuristic (L=2)	1	1	1	2	2	2	2	**1**	3	3
Heuristic (L=3)	**2**	1	1	2	2	**3**	2	**2**	**1**	3
PICA	1	1	1	2	2	2	2	3	3	3
Ground-truth	1	1	1	2	2	2	2	3	3	3

Table 4.1: *With labels from inaccurate annotators, PICA perfectly recovers the ground-truth labels while the Unpermutation Heuristic makes mistakes regardless of the choice of leader.*

Chapter 5

Inference

Given the observed labels $\mathscr{L}$ and our simplified likelihood model (Equation 4.2), we can use Expectation-Maximization to optimize over the style matrices of all the annotators and infer the ground-truth of each example. However, the constraint that each style matrix be doubly-stochastic requires special handling, which we describe in the M-Step section below.

The high-level intuition of our procedure is the following: in the E-Step, we compute our predictions (i.e. probabilities) for the label of each data example based on our current estimation of the annotators' style matrices. Then in the M-Step, we modify our estimates of the style matrices to make them a more accurate reflection of the observed labels and our current predictions of the ground-truth labels. These steps are performed iteratively until accurate predictions of the style matrices and ground-truth labels have been reached.

Our derivations below make use of the conditional independence encoded in the graphical model in Figure 4.1.

5.1 E-Step

In this step, we compute the posterior probability distributions of $\mathsf{z}_j \in \Omega\ \forall j \in \{1,\dots,n\}$ given $\mathsf{S}^{(1)},\dots,\mathsf{S}^{(m)}$ from the last M-Step and the observed labels $\mathscr{L}$.

$$\begin{aligned} & p(\mathsf{z}_j = z_j \mid \mathsf{L}^{(1)} = L^{(1)},\dots,\mathsf{L}^{(m)} = L^{(m)}, \\ & \qquad \mathsf{S}^{(1)} = S^{(1)},\dots,\mathsf{S}^{(m)} = S^{(m)}) \\ = \quad & p(z_j \mid L^{(1)},\dots,L^{(m)},S^{(1)},\dots,S^{(m)}) \end{aligned}$$

$$\begin{aligned}
&\propto \quad p(z_j)p(L^{(1)},\ldots,L^{(m)} \mid S^{(1)},\ldots,S^{(m)},z_j)\\
&= \quad p(z_j) \prod_{i:L_j^{(i)}\neq\epsilon} p(L_j^{(i)} \mid z_j, S^{(i)})\\
&= \quad p(z_j) \prod_{i:L_j^{(i)}\neq\epsilon} S^{(i)}_{z_j,L_j^{(i)}}
\end{aligned}$$

where we note that $p(z_j|S^{(1)},\ldots,S^{(m)}) = p(z_j)$ by conditional independence assumptions from the graphical model. Also, note that $S^{(i)}_{z_j,L_j^{(i)}}$ is the z_jth row and $L_j^{(i)}$th column of $S^{(i)}$.

5.2 M-Step

In this step, we maximize the auxiliary function Q, defined as the expectation of the joint log-likelihood of the observed labels and ground-truth labels, with respect to the posterior distribution of each z_j computed in the last E-Step, denoted $\widetilde{p}$.

$$\begin{aligned}
&Q(S^{(1)},\ldots,S^{(m)})\\
&\quad= E\left[\,\log p(\mathscr{L}=L,\mathrm{z}=z \mid \mathrm{S}=S)\,\right]\\
&\quad= E\left[\,\log p(L,z \mid S)\,\right]\\
&\quad= E\left[\log \prod_j \left(p(z_j)\prod_i p(L_j^{(i)} \mid z_j,S^{(i)})\right)\right]\\
&\qquad \text{since } L_j^{(i)} \text{ are cond. indep.}\\
&\quad= \sum_j E\left[\,\log p(z_j)\right] + \sum_{ij} E\left[\,\log\left(L_j^{(i)} \mid z_j,S^{(i)}\right)\right]\\
&\quad= \sum_{ij} E\left[\log\left(L_j^{(i)} \mid z_j,S^{(i)}\right)\right] + C\\
&\qquad \text{since } \sum_j E[\log p(z_j)] \text{ is const. w.r.t. each } S^{(i)}\\
&\quad= \sum_{ij}\sum_{z_1}\cdots\sum_{z_d} \log\left(p(L_j^{(i)}|z_j,S^{(i)})\right)\widetilde{p}(z_1,\ldots,z_d)\\
&\quad= \sum_{ij}\sum_{z_j} \log\left(p(L_j^{(i)}|z_j,S^{(i)})\right)\widetilde{p}(z_j)\\
&\qquad \sum_{z_1}\cdots\sum_{z_{j-1}}\sum_{z_{j+1}}\cdots\sum_{z_d}\widetilde{p}(z_1,\ldots,z_{j-1},z_{j+1},\ldots,z_d)\\
&\quad= \sum_{ij}\sum_{z_j} \log\left(p(L_j^{(i)}|z_j,S^{(i)})\right)\widetilde{p}(z_j)\\
&\quad= \sum_{ij}\sum_{z_j} \log\left(S^{(i)}_{z,L_j^{(i)}}\right)\widetilde{p}(z_j)
\end{aligned}$$

Remark 5.1. *Notice we drop the constant C after introducing it. Since this constant is the same for all values of S, it doesn't need to be computed.*

To take the first derivative of Q, we vectorize each $S^{(i)}$:

$$\frac{\partial Q}{\partial \mathrm{vec}[S^{(i)}]} = \begin{bmatrix} \frac{\partial Q}{\partial S^{(i)}_{11}} & \cdots & \frac{\partial Q}{\partial S^{(i)}_{1n}} & \frac{\partial Q}{\partial S^{(i)}_{21}} & \cdots & \frac{\partial Q}{\partial S^{(i)}_{nn}} \end{bmatrix}$$

where

$$\begin{aligned}
\frac{\partial Q}{\partial S^{(i)}_{xy}} &= \sum_{j:L^{(i)}_j \neq \epsilon} \sum_{z_j} \frac{\partial}{\partial S^{(i)}_{xy}} [\log\left(S^{(i)}_{z_j, L^{(i)}_j}\right)] \widetilde{p}(z_j) \\
&= \sum_{j:L^{(i)}_j \neq \epsilon} \frac{\partial}{\partial S^{(i)}_{xy}} [\log\left(S^{(i)}_{x, L^{(i)}_j}\right)] \widetilde{p}(x) \\
&= \sum_{j:L^{(i)}_j \neq \epsilon} \delta(L^{(i)}_j, y) \frac{\partial}{\partial S^{(i)}_{xy}} [\log\left(S^{(i)}_{xy}\right)] \widetilde{p}(x) \\
&= \sum_{j:L^{(i)}_j \neq \epsilon} \delta(L^{(i)}_j, y) \frac{\widetilde{p}(x)}{S^{(i)}_{xy}}
\end{aligned}$$

If S were a real-valued matrix, we could simply use a gradient ascent method (stochastic, conjugate, etc.) to find values of $S^{(1)}, \ldots, S^{(m)}$ that maximize Q. However, the constraint that each style matrix is doubly-stochastic requires a more specialized optimization method to ensure that the result of each gradient update remains on the Birkhoff polytope (i.e. the manifold whose points are DSMs). As discussed in Chapter 3, we can utilize Sinkhorn Propagation to satisfy the DSM constraints.

After conducting Expectation-Maximization, we take the final probability distribution, from the last E-Step, of each z_j as the estimated label for each example j.

5.3 Prior on Style

In some settings, we may also wish to add a regularization term to "push" the style matrices towards the identity permutation; this can be useful if most labelers are assumed to use some "default" permutation (identity). To do so, we can subtract from Q an additional term $\frac{\gamma}{n^2} \|S^{(i)} - I\|^2_{\mathrm{Fr}}$ where γ specifies the strength of the regularization, I is the identity matrix with the same dimensions as $S^{(i)}$ (i.e. $n \times n$), and $\| \cdot \|^2_{\mathrm{Fr}}$ is the squared Frobenius norm. Similarly, for the derivative $\frac{\partial Q}{\partial S^{(i)}_{xy}}$, we subtract the term $\frac{2\gamma}{n^2}(S_{xy} - I_{xy})$.

Chapter 6

Experiments

6.1 Overview

We developed several experiments to evaluate our model. Our experiments fall under two categories: the first are simulations which were designed to highlight specific benefits of our model over other models. The second demonstrate the efficacy of our model in useful applications i.e. text passage clustering and dense image segmentation. In all of the experiments, we compute percent correct and cross-entropy as our performance metrics.

6.1.1 Permutation-invariant accuracy measurement

Since the cluster labels in our experiments have no inherent meaning (e.g., we could swap the labels "1" and "2" without changing any semantics of the clusters), we computed accuracy of the estimated cluster labels as the *maximum* accuracy, over all $n!$ permutations, of the estimates with respect to ground-truth cluster labels. As we focus on applications where n is small, computing over all $n!$ permutations is computationally feasible.

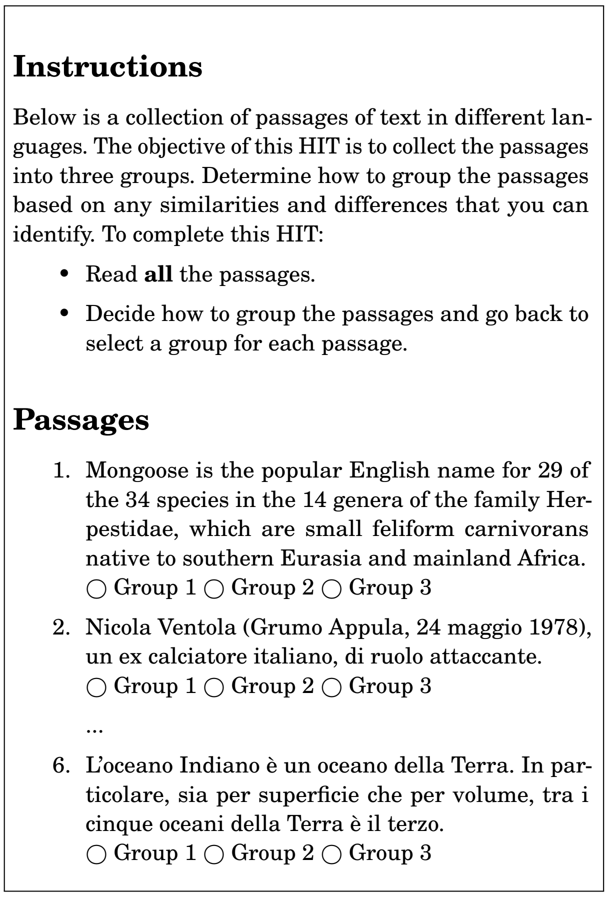
6.1.2 Optimization details

For all but the image segmentation experiment, we conducted Expectation-Maximization until $Q^{(k)} - Q^{(k-1)} < 10^{-4}$, where $Q^{(k)}$ is the value of Q at the kth iteration. For the image segmentation experiment, we used a tolerance of 10^{-5}. The code for the PICA model implementation as well as the experimental analyses is available at `https://github.com/mjgiancola/MQP`.

6.2 Text Passage Clustering

6.2.1 Description

We designed a text passage clustering task which required annotators to cluster 6 passages into three groups (see Figure 6.1). This experiment was conducted on real annotators from Amazon Mechanical Turk. The workers were not told how many passages to put into each group, or by what characteristics to cluster them. They were told simply to "Determine how to group the passages based on any similarities and differences that you can identify". Of the six text passages, two were in English, two were in Italian, and two were in Russian. We expected that most people would cluster by language, as the content of the passages were selected to be totally unrelated. (The passages were selected from Wikipedia articles on disparate topics such as mongoose, jet pack, Italian soccer players, etc.) See Figure 6.1 for the task description that we posed on Mechanical Turk.



Instructions

Below is a collection of passages of text in different languages. The objective of this HIT is to collect the passages into three groups. Determine how to group the passages based on any similarities and differences that you can identify. To complete this HIT:

- Read **all** the passages.
- Decide how to group the passages and go back to select a group for each passage.

Passages

1. Mongoose is the popular English name for 29 of the 34 species in the 14 genera of the family Herpestidae, which are small feliform carnivorans native to southern Eurasia and mainland Africa.
   ◯ Group 1 ◯ Group 2 ◯ Group 3
2. Nicola Ventola (Grumo Appula, 24 maggio 1978), un ex calciatore italiano, di ruolo attaccante.
   ◯ Group 1 ◯ Group 2 ◯ Group 3

   ...

6. L'oceano Indiano è un oceano della Terra. In particolare, sia per superficie che per volume, tra i cinque oceani della Terra è il terzo.
   ◯ Group 1 ◯ Group 2 ◯ Group 3

Figure 6.1: *The text passage clustering task we posted on Amazon Mechanical Turk. The annotators' job was to group the different text passages into clusters.*

We had 25 workers complete our task, one of whom didn't label one of the passages, resulting in 149 labels. Qualitatively, we observed that, while the majority of workers clustered by language, there was some noise in the data due to people clustering with some other reasoning in mind.

6.2.2 Results – Experiment 1

We assessed accuracy based on a randomly chosen subset (without replacement) of just 3 (out of 25) annotators, and then repeated the experiment 100 times to obtain an average performance estimate. Hence, the accuracy statistics we report reflect an annotation scenario in which only a small number of annotators provide labels, and the job of the aggregator is to infer the ground-truth labels despite the annotators' differing styles.

The proposed PICA model (based on DSMs) achieved an average (over all 100 samplings) of 93% accuracy, with average cross-entropy (computed between the probability distributions of the z_j with respect to ground-truth labels) of 2.54. In contrast, the Unpermutation Heuristic achieved an accuracy of only 90.5%, and simple Majority Vote scored 89%. This provides a simple proof-of-concept on labels from real human annotators that permutation-invariant consensus algorithms can be useful. The best performing model, however, was actually the RSM approach [1], which achieved an accuracy of 96% and cross-entropy of 1.65; this shows that the lower number of free parameters in DSMs may not always be decisive.

6.2.3 Results – Experiment 2

We again assessed accuracy based on a subset of the data, except this time we randomly sampled (without replacement) 2 labels from each of the 25 annotators, for a total of 50 observed labels in each experiment. In this set of experiments, there were many more observed labels but less per annotator, so being able to accurately infer the style of each annotator becomes more difficult and more important for accurately recovering the ground-truth labels.

Over 100 independent trials, PICA scored 94% with 6.31 cross-entropy, RSM scored 97% with 1.46 cross-entropy, the Unpermutation Heuristic scored 59%, and Majority Vote scored 91.67%. We find that RSM is slightly better than PICA yet again, although we see a significant performance difference from the Unpermutation Heuristic, which did not have enough data per annotator to make accurate inferences.

6.3 Rare Class Simulation

6.3.1 Description

This simulation was designed to highlight the difference between our DSM-based model and a simpler RSM-based model (e.g., [1]). Recall that, once $n-1$ rows of an $n \times n$ DSM have been identified, then the last row can be inferred unambiguously. With RSMs, on the other hand, all of the rows must be inferred from data independently (since there is no constraint that each column sums to 1). This difference can be decisive in a setting in which one of the classes (from $\{1,\dots,n\}$) occurs only rarely in the dataset, so that few of the observed labels provide any information on the values of each $S^{(i)}$.

For this simulation, we generated 100 examples, each of which was assigned a label $z_j \in \Omega = \{$ 'a', 'b', 'c' $\}$, with p('a') = 0.5, p('b') = 0.45, p('c') = 0.05. We then simulated 100 annotators who each have a random accuracy $a^{(i)}$ sampled uniformly from $[0.75, 1)$. Each annotator had a random permutation of the identity for their style matrix. For each annotator, we collected 10 labels, sampled randomly (without replacement) from the 100 total examples, giving a total of 1000 observed labels. As in the Text Passage Clustering experiments, results were averaged over 100 independent trials.

6.3.2 Results

The DSM-based PICA algorithm achieved a percent correct of 88% and 4.01 cross entropy, while the RSM-based model scored 79% with 6.32 cross entropy. As a baseline, the Unpermutation Heuristic scored 62% and Majority Vote scored 47%. These results show an example of how the more constrained DSM-based PICA algorithm can yield higher accuracy than the simpler (and less constrained) RSM-based approach.

6.4 Dense Image Segmentation

6.4.1 Description

In this experiment we explored whether PICA could reconstruct a dense (pixel-wise) image segmentation from multiple noisy segmentations. In our experiment, we used an image and ground-truth segmentation (Figure 6.2 (a-b)) from the ADE20K dataset [24], which in turn included images from the LabelMe dataset [25]. Based on the ground-truth segmentation, we then generated segmentations for 10 simulated annotators by permuting the class labels and adding noise (see Figure 6.2 (c) for an example). The input to our model thus consisted of $10 \times$ `numPixels` labels, where each of the 10 segmentations corresponded to the labels of a single annotator, and each pixel corresponded to an individual example (ie. pixel j in image i is label $L_j^{(i)}$). We generated a segmentation dataset for four different images: "couple", "flag", "light', and "people"; the "couple" image is shown in Figure 6.2 (a). Together, these data enable us to assess whether PICA can infer each annotator's different "style" in assigning pixels to colors and combine them across annotators to yield aggregated segmentations that are more accurate.

6.4.2 Results

Accuracy in inferring the ground-truth clustering labels with respect to the ground-truth of the four approaches – PICA (based on DSMs), RSM, the Unpermutation Heuristic, and Majority

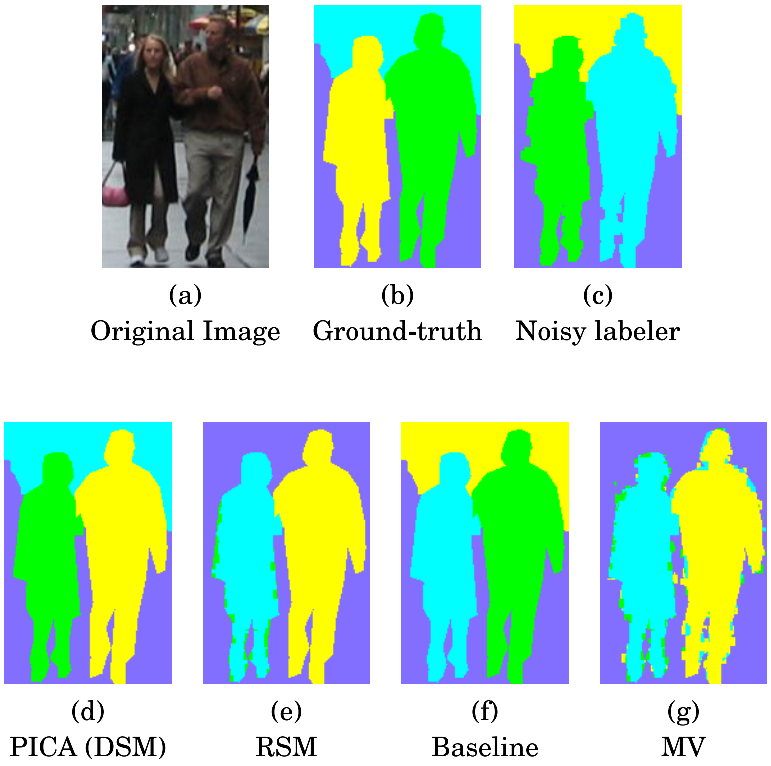


Figure 6.2: *A dense (pixel-wise) image segmentation task on one of the images ("couple") from the ADE20K dataset [24]. Sub-figures a-f show (in order) the original image, the ground-truth segmentation, the segmentation collected from one of the simulated noisy labelers, and then the inferred ground-truth (over all 10 noisy labelers) using the proposed DSM-based PICA algorithm, the RSM-based algorithm [1], the baseline algorithm, and simple Majority Vote.*

Vote – are shown in Table 6.1. Graphical results just for the "couple" image are also shown in Figure 6.2 (d)-(g). As with the previous experiments, we compute the %-correct accuracy and cross-entropy as the *maximum* over all $n!$ permutations, where n is the number of class labels. Hence, the accuracies of the models are not penalized due to just the arbitrary coloring assigned to each cluster in the image.

For the "couple" image, PICA was able to reconstruct the original segmentation with 100% accuracy. In contrast, the RSM model achieved 80% accuracy relative to the original segmentation, as it was unable to distinguish the foreground and background of the image. A simple majority vote still achieved 77% accuracy, but generates a segmentation with a lot of noise. For this example, the Unpermutation Heuristic achieved the same level of performance as PICA – this was expected as the simulated annotators were highly accurate.

The results for all four images we tested on are shown in Table 6.1. For three of the four images, PICA performed the best in terms of %-correct clustering accuracy (fraction of pixels assigned to the correct cluster, after taking the maximum over all $n!$ permutations), tying with

Model	**Image**							
	"couple"		"flag"		"light"		"people"	
	%	C.E.	%	C.E.	%	C.E.	%	C.E.
PICA	**100**	65.5	**99.4**	**.077**	**99.8**	**.020**	**99.0**	48.2
RSM	80.4	**13.9**	98.7	.139	99.1	.040	**99.0**	**39.6**
Heuristic	**100**	–	99.2	–	**99.8**	–	98.9	–
MV	77.5	–	81.5	–	88.9	–	86.0	–

Table 6.1: *Accuracy of the proposed DSM-based PICA model compared to the RSM-based model [1] for aggregating over multiple annotators' dense (pixel-wise) image segmentations. We applied both algorithms to four different images ("couple", "flag", "light", and "people"). Accuracy was measured using %-correct accuracy in assigning pixels to clusters, as well as cross-entropy.*

the Unpermutation Heuristic twice and with RSM once. In terms of cross-entropy, the results were more mixed, and both RSM and PICA achieved the best cross-entropy in two of the four images.

While this is simply a proof-of-concept, it illustrates how crowdsourcing – and principled algorithms for aggregating over crowdsourced labels – can be used for image segmentation tasks. After investigating annotation consistency between segmentations, [24] discussed the important sources of error. The first was varying levels of segmentation quality from different annotators. The second was ambiguities in object naming (ie. one annotator labeled a truck as "car"). As shown empirically, the proposed PICA model can construct a segmentation which eliminates the noise in the given data. The object naming issue would be eliminated entirely, as PICA is invariant to each annotator's style.

Chapter 7

Conclusion

This paper presented a crowdsourcing consensus algorithm called PICA (Permutation-Invariant Crowdsourcing Aggregation) which performs *permutation-invariant* simultaneous inference of the ground-truth labels and the individual annotators' *style* of labeling. This allows our model to excel in scenarios when annotators have *systematic confusions* of what the different classes mean, or when the particular class labels don't matter (e.g. clustering, image segmentation).

We modeled style using a doubly-stochastic matrix (DSM) for each annotator, whose entries gave the probability that the annotator perceived some latent class, but transcribed another. We then utilized Sinkhorn Normalization [5] and Sinkhorn Propagation [4] to find the optimal style matrix for each annotator, within an Expectation-Maximization-based optimization scheme.

In contrast to [1], our model can benefit from harnessing a smaller number of degrees of freedom to reduce overfitting. Namely, PICA outperformed the RSM-based model [1] in a simulated experiment where there were a relatively small number of examples from one class.

We evaluated our model on simulated image segmentation data [24] and found that PICA was able to perfectly reconstruct an image segmentation based on several noisy segmentations.

7.1 Future Work

An interesting area of future inquiry is exploring the relationship between the number of annotators and examples in some labeling task. One would expect that the number of annotators and examples have an inverse relationship in the sense that having more annotators can compensate for having fewer labels per annotator. But is there a limit to the extent that more examples can make up for a lack of annotators (and vice-versa)? For the particular case of PICA, it would be interesting to investigate how this issue interacts with the relative benefits/drawbacks of using DSM-based versus RSM-based models.

As our work on image segmentation was simulated, it would be interesting to see how PICA performs on real data i.e. aggregating the labels from several annotators to create a consensus segmentation. One challenge will be to quantify the results. In our work, we were able to compare the models' performances to the "ground-truth" i.e. the original segmentation from which the noisy segmentations were generated. However on real data, there would be no ground-truth to compare to, so the results of such experiments would likely be more qualitative.

Appendices

Appendix A

DSMs Closed Under Dot Product

Lemma 4.1: *Let A and B be two arbitrary DSMs. Then AB is also a DSM.*

Proof. Let A and B be two arbitrary DSMs.

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ & & \vdots & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} B = \begin{bmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ & & \vdots & \\ b_{n1} & b_{n2} & \dots & b_{nn} \end{bmatrix}$$

Column j of AB has the form:

$$AB_{*j} = \begin{bmatrix} a_{11}b_{1j} + a_{12}b_{2j} + \dots + a_{1n}b_{nj} \\ a_{21}b_{1j} + a_{22}b_{2j} + \dots + a_{2n}b_{nj} \\ \vdots \\ a_{n1}b_{1j} + a_{n2}b_{2j} + \dots + a_{nn}b_{nj} \end{bmatrix}$$

Taking the sum of the elements in column j, we can reorder the terms and find:

$$\begin{aligned} & b_{1j}(a_{11} + \dots + a_{n1}) \\ & +b_{2j}(a_{12} + \dots + a_{n2}) \\ & +\dots \\ & +b_{nj}(a_{1n} + \dots + a_{nn}) \\ & = b_{1j} + b_{2j} + \dots + b_{nj} \\ & = 1 \end{aligned}$$

A similar argument shows that each row of AB sums to 1. ■

APPENDIX B

EXAMPLE OF ONE SINKPROP ITERATION

This appendix contains a simple, illustrative example of how Sinkhorn Propagation is performed. We start with a strictly positive matrix A[1], perform one Sinkhorn iteration (ie. one row normalization and one column normalization), and compute the value of a function f (sum of squares) of our (almost) DSM. Then we compute the gradient $\frac{\partial f}{\partial A}$ using Sinkhorn Propagation.

Let $A = \begin{bmatrix} 2 & 5 \\ 3 & 4 \end{bmatrix}$ and $f = \sum_{i=1}^{2}\sum_{j=1}^{2} A_{ij}^2$.

Then $R(A) = \begin{bmatrix} \frac{2}{7} & \frac{5}{7} \\ \frac{3}{7} & \frac{4}{7} \end{bmatrix}$, $C(R(A)) = \begin{bmatrix} \frac{2}{5} & \frac{5}{9} \\ \frac{3}{5} & \frac{4}{9} \end{bmatrix}$, and $f(C(R(A))) = 1\frac{53}{2025} \approx 1.026$

Next, we compute $\frac{\partial f}{\partial A}$ by "back-propagating" the gradient through the Sinkhorn iterations. Notice we have vectorized the matrices so that $\frac{\partial f}{\partial \text{vec}[C]}$ has shape $1 \times n^2$, $\frac{\partial C}{\partial \text{vec}[R]}$ has shape $n^2 \times n^2$, and $\frac{\partial R}{\partial \text{vec}[A]}$ has shape $n^2 \times n^2$.

$$\frac{\partial f}{\partial \text{vec}[C]} = 2 * \text{vec}[C] = \begin{bmatrix} \frac{4}{5} & \frac{10}{9} & \frac{6}{5} & \frac{8}{9} \end{bmatrix}$$

$$\frac{\partial C}{\partial \text{vec}[R]} = \begin{bmatrix} \frac{\partial C_{11}}{\partial R_{11}} & \frac{\partial C_{12}}{\partial R_{11}} & \frac{\partial C_{21}}{\partial R_{11}} & \frac{\partial C_{22}}{\partial R_{11}} \\ \frac{\partial C_{11}}{\partial R_{12}} & \frac{\partial C_{12}}{\partial R_{12}} & \frac{\partial C_{21}}{\partial R_{12}} & \frac{\partial C_{22}}{\partial R_{12}} \\ \frac{\partial C_{11}}{\partial R_{21}} & \frac{\partial C_{12}}{\partial R_{21}} & \frac{\partial C_{21}}{\partial R_{21}} & \frac{\partial C_{22}}{\partial R_{21}} \\ \frac{\partial C_{11}}{\partial R_{22}} & \frac{\partial C_{12}}{\partial R_{22}} & \frac{\partial C_{21}}{\partial R_{22}} & \frac{\partial C_{22}}{\partial R_{22}} \end{bmatrix} = \begin{bmatrix} \frac{21}{25} & 0 & \frac{-14}{25} & 0 \\ 0 & \frac{28}{81} & 0 & \frac{-35}{81} \\ \frac{-21}{25} & 0 & \frac{14}{25} & 0 \\ 0 & \frac{-28}{81} & 0 & \frac{35}{81} \end{bmatrix}$$

[1]To make the computations simpler, we exclude the exp(A) step in this example.

$$\frac{\partial R}{\partial \text{vec}[A]} = \begin{bmatrix} \frac{\partial R_{11}}{\partial A_{11}} & \frac{\partial R_{12}}{\partial A_{11}} & \frac{\partial R_{21}}{\partial A_{11}} & \frac{\partial R_{22}}{\partial A_{11}} \\ \frac{\partial R_{11}}{\partial A_{12}} & \frac{\partial R_{12}}{\partial A_{12}} & \frac{\partial R_{21}}{\partial A_{12}} & \frac{\partial R_{22}}{\partial A_{12}} \\ \frac{\partial R_{11}}{\partial A_{21}} & \frac{\partial R_{12}}{\partial A_{21}} & \frac{\partial R_{21}}{\partial A_{21}} & \frac{\partial R_{22}}{\partial A_{21}} \\ \frac{\partial R_{11}}{\partial A_{22}} & \frac{\partial R_{12}}{\partial A_{22}} & \frac{\partial R_{21}}{\partial A_{22}} & \frac{\partial R_{22}}{\partial A_{22}} \end{bmatrix} = \begin{bmatrix} \frac{5}{49} & \frac{-2}{49} & 0 & 0 \\ \frac{-5}{49} & \frac{2}{49} & 0 & 0 \\ 0 & 0 & \frac{4}{49} & \frac{-3}{49} \\ 0 & 0 & \frac{-4}{49} & \frac{3}{49} \end{bmatrix}$$

Using Equation (3.1),

$$\frac{\partial f}{\partial A} \approx \begin{bmatrix} -0.042 & 0.017 \\ 0.026 & -0.020 \end{bmatrix}$$

Notice if we apply this gradient to A, we get $A' \approx \begin{bmatrix} 1.958 & 5.017 \\ 3.026 & 3.98 \end{bmatrix}$ and $f(C(R(A'))) \approx 1.028$, which is a slight improvement from our original A.[2]

[2]Notice, the optimal DSM matrix for f is the identity (or some transformation of it), which will produce the value 2 in the 2x2 case.

Appendix C

Data Collection

C.1 Text Passage Clustering Data

We collected passages on Wikipedia for our text clustering experiment. Since we wanted passages in multiple languages, we used English, Russian, and Italian Wikipedia articles. The passages we used came from the following articles: Mongoose and Jet Pack (English), Nicola Ventola and Indian Ocean (Italian), and Ludwig van Beethoven and G-Eazy (Russian).

C.2 Image Segmentation Data

We used an image and segmentation from the ADE20K dataset [24]. The image was originally from the LabelMe dataset [25]. To reduce time costs, we cropped sections of the image to use. Also, to simplify the results, we regenerated our own segmentations which ignored a few class labels (ie. small objects in the background).

Bibliography

[1] Padhraic Smyth, Usama M Fayyad, Michael C Burl, Pietro Perona, and Pierre Baldi.
Inferring ground truth from subjective labelling of venus images.
In *Advances in neural information processing systems*, pages 1085–1092, 1995.

[2] Michael Giancola, Randy Paffenroth, and Jacob Whitehill.
Permutation-invariant consensus over crowdsourced labels.
In *Sixth AAAI Conference on Human Computation and Crowdsourcing*, 2018.

[3] Paul Ekman.
All emotions are basic.
The nature of emotion: Fundamental questions, pages 15–19, 1994.

[4] Ryan Prescott Adams and Richard S Zemel.
Ranking via sinkhorn propagation.
arXiv preprint arXiv:1106.1925, 2011.

[5] Richard Sinkhorn.
A relationship between arbitrary positive matrices and doubly stochastic matrices.
The annals of mathematical statistics, 35(2):876–879, 1964.

[6] Richard Sinkhorn and Paul Knopp.
Concerning nonnegative matrices and doubly stochastic matrices.
Pacific Journal of Mathematics, 21(2):343–348, 1967.

[7] Alexander Philip Dawid and Allan M Skene.
Maximum likelihood estimation of observer error-rates using the em algorithm.
Applied statistics, pages 20–28, 1979.

[8] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo.
Whose vote should count more: Optimal integration of labels from labelers of unknown expertise.
In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[9] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy.

Learning from crowds.
Journal of Machine Learning Research, 11(Apr):1297–1322, 2010.

[10] Mahyar Salek, Yoram Bachrach, and Peter Key.
Hotspotting-a probabilistic graphical model for image object localization through crowdsourcing.
In *AAAI*, 2013.

[11] Jonathan Bragg, Daniel S Weld, et al.
Crowdsourcing multi-label classification for taxonomy creation.
In *First AAAI conference on human computation and crowdsourcing*, 2013.

[12] Yuyin Sun, Adish Singla, Dieter Fox, and Andreas Krause.
Building hierarchies of concepts via crowdsourcing.
In *IJCAI*, pages 844–853, 2015.

[13] Christopher H Lin, Daniel Weld, et al.
Crowdsourcing control: Moving beyond multiple choice.
arXiv preprint arXiv:1210.4870, 2012.

[14] Yuandong Tian and Jun Zhu.
Learning from crowds in the presence of schools of thought.
In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–234. ACM, 2012.

[15] Ece Kamar, Ashish Kapoor, and Eric Horvitz.
Identifying and accounting for task-dependent bias in crowdsourcing.
In *Third AAAI Conference on Human Computation and Crowdsourcing*, 2015.

[16] Nam Nguyen and Rich Caruana.
Consensus clusterings.
In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 607–612. IEEE, 2007.

[17] Alexander Topchy, Anil K Jain, and William Punch.
Clustering ensembles: Models of consensus and weak partitions.
IEEE transactions on pattern analysis and machine intelligence, 27(12):1866–1881, 2005.

[18] Alexander Strehl and Joydeep Ghosh.
Cluster ensembles—a knowledge reuse framework for combining multiple partitions.
Journal of machine learning research, 3(Dec):583–617, 2002.

[19] Amir Alush and Jacob Goldberger.

Ensemble segmentation using efficient integer linear programming.
IEEE transactions on pattern analysis and machine intelligence, 34(10):1966–1977, 2012.

[20] Oded Kaminsky and Jacob Goldberger.
Combining clusterings with different detail levels.
In *Machine Learning for Signal Processing (MLSP), 2016 IEEE 26th International Workshop on*, pages 1–6. IEEE, 2016.

[21] Ryan G Gomes, Peter Welinder, Andreas Krause, and Pietro Perona.
Crowdclustering.
In *Advances in neural information processing systems*, pages 558–566, 2011.

[22] Jinfeng Yi, Rong Jin, Shaili Jain, Tianbao Yang, and Anil K Jain.
Semi-crowdsourced clustering: Generalizing crowd labeling by robust distance metric learning.
In *Advances in neural information processing systems*, pages 1772–1780, 2012.

[23] Ramya Korlakai Vinayak and Babak Hassibi.
Crowdsourced clustering: Querying edges vs triangles.
In *Advances in Neural Information Processing Systems*, pages 1316–1324, 2016.

[24] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba.
Scene parsing through ade20k dataset.
In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[25] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman.
Labelme: a database and web-based tool for image annotation.
International journal of computer vision, 77(1-3):157–173, 2008.