Toloka

Toloka

ICWE 2022 hands-on tutorial

Web Engineering with Human-in-the-Loop

Dmitry Ustalov, Nikita Pavlichenko, Boris Tseytlin, Daria Baidakova and Alexey Drutsa

Part V
Results Aggregation

Nikita Pavlichenko,
Researcher

Tutorial Schedule

Part I Intro: 45 min
The Role of HITL in building Search Engines

Part II: 45 min
Ranking and Quality Metrics

Coffee Break: 30 min

Part III: 45 min
Human-in-the-Loop Essentials

Part IV: 45 min
Practice: Websites relevance

Lunch Break: 90 min

Part V: 45 min
Results aggregation and implementation into ML pipeline

Part VI: 90 min
Metric Computation

Coffee Break: 30 min

Part VII: 90 min
Results discussion and conclusion

Aggregation

- Choose a correct diagnosis from multiple doctors
- Perform better ML models bagging
- Combine humans’ opinion and ML
- Extract the true label from noisy crowdsourcing responses
- ~~Improve democracy by better voting process~~

Is aggregation necessary?

Motivation

- Each worker is a noisy “classifier”
- We know that bagging of classifiers increases accuracy
- Without overlap the annotation is not robust to fraud

Plan

- On problem of aggregation
- Baseline
- Latent Label Models (DS, GLAD, MMCE)
- Bayesian Models (BCC, Community BCC)

Notation

- Categories: $k \in \{1, \dots, K\}$
- Tasks: $j \in \{1, \dots, T\}$
- Performers: $w \in \{1, \dots, W\}$
- $W_j \subseteq \{1, \dots, W\}$ — performers labelled object j

The Problem of Aggregation

- Observe noisy labels

$$\boldsymbol{y} = \{y_j^w | j = 1, \dots, T, w = 1, \dots, W\}$$

- Recover true labels

$$\boldsymbol{z} = \{z_j | j = 1, \dots, T\}$$

Single-Coin Dawid-Skene model

- We assume that every performer has a latent parameter “skill”

$$\Pr\left(z_j = y_j^w\right) = q_w$$

- With probability q_w performer answers correctly and incorrectly with probability $(1 - q_w)/(K - 1)$ for each incorrect label

What baseline to use?

Baseline: Majority Vote

- Assume that all labels and performers are equal:

$$q_1 = q_2 = \cdots = q_W$$

- If $q_i > 1/K$ the true label will be the most probable one

$$\hat{z}_j^{MV} = \arg\max_{y=1,\dots,K} \sum_{w \in W_j} \delta(y = y_j^w),$$

where $\delta(\mathrm{A}) = 1$ if A is true and 0 otherwise

Weighted MV

- Assume that we have an estimator of the performer's skill $\hat{q}_w = \Pr(y^w = z)$ *(it could be, for example, golden set accuracy)*

- Then, we can construct more accurate aggregation

$$\hat{z}_j^{wMV} = \arg\max_{y=1,\ldots,K} \sum_{w \in W_j} \hat{q}_w \delta(y = y_j^w)$$

Or even better:

- **Theorem (Li and Yu, 2014):** *the optimal prediction under single-coin D&S model is a weighted majority vote:*

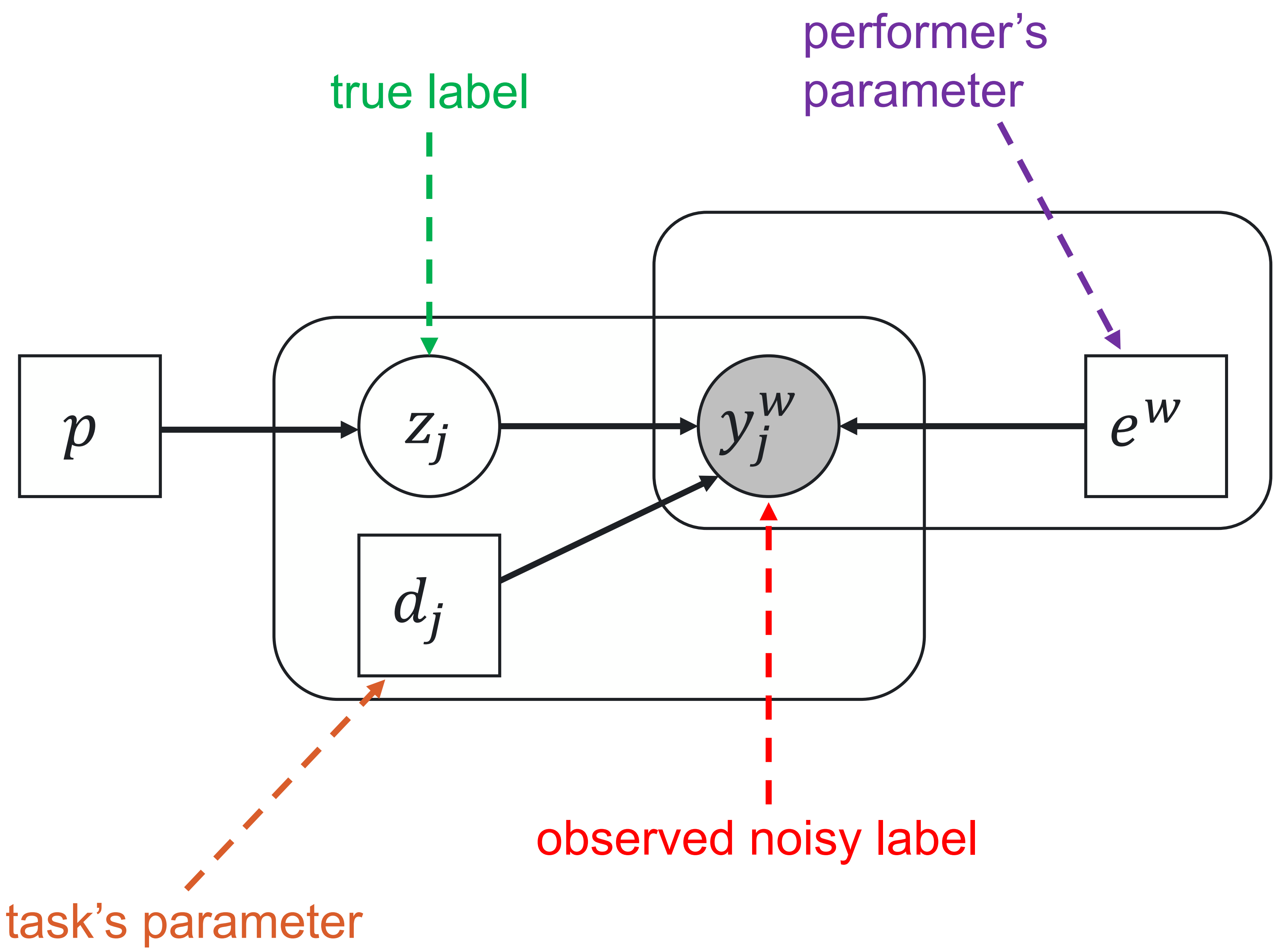
$$\hat{z}_j^{opt.wMV} = \arg\max_{y=1,\ldots,K} \sum_{w \in W_j} \log \frac{(K-1)\hat{q}_w}{\hat{q}_w} \delta(y = y_j^w)$$

What else can we add?

More Complex Methods: Latent Label Models

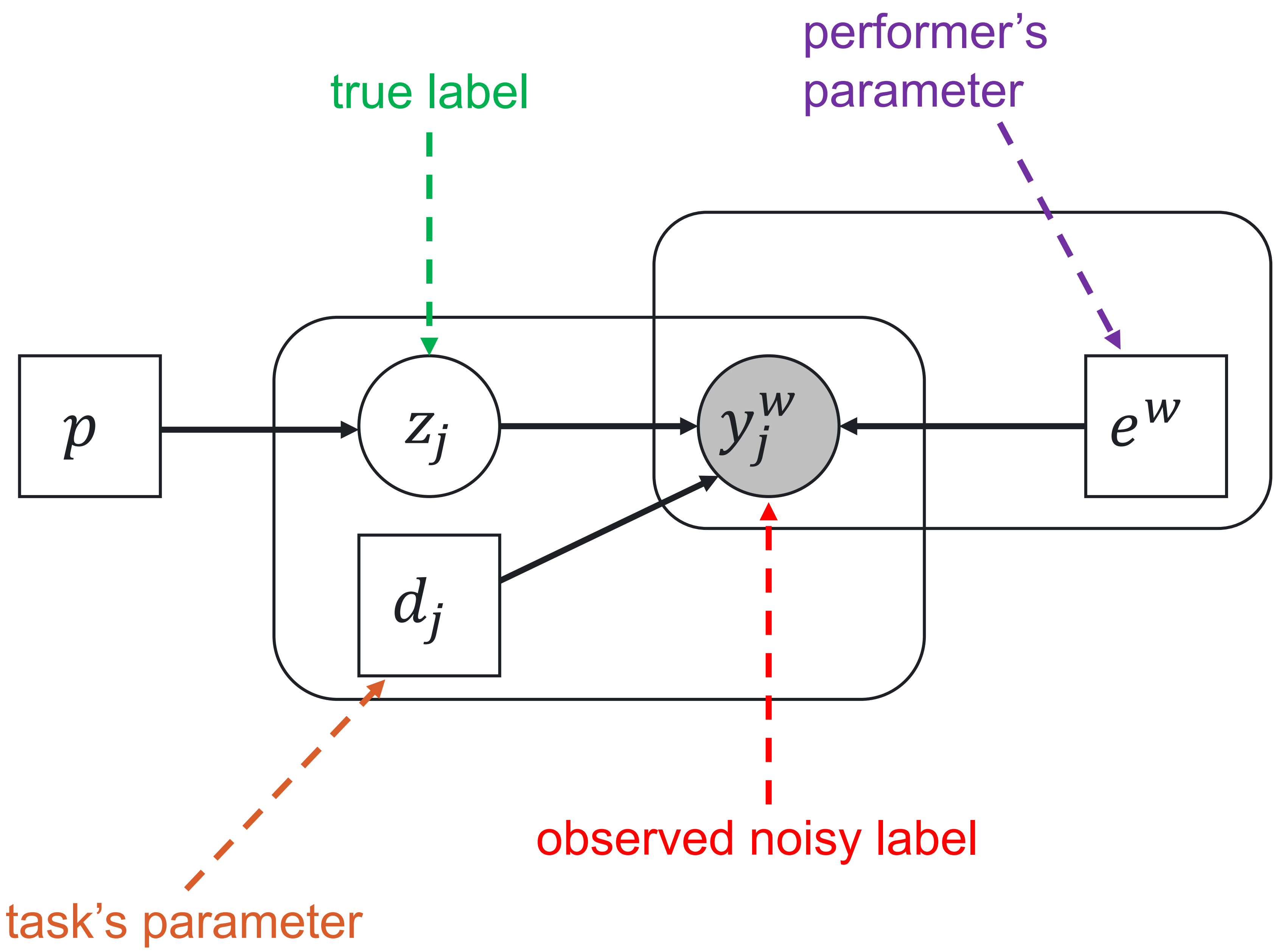
- Parametrize performers by e^w(e.g. skills)
- Parametrize tasks by d_j (e.g. difficulties)
- Each task has a **unique** true label
- Observed labels are corrupted versions of this true label

Latent Label Models: Noisy Label Model



- A noisy label model $M_j^w = M(e^w, d_j)$ is a matrix of size $K \times K$ with elements $M_j^w[c,k] = \Pr\left(y_j^w = k \middle| z_j = c\right)$

Latent Label Models: Generation Process



- A noisy label model assumes that the annotation process can be modelled as follows:

1. Sample z_j from prior distribution $P_z(p)$
2. Sample y_j^w from a distribution $P_Y(M_j^w[z_j,\cdot])$

Latent Label Models: Parameters Optimization

- Assumption: y_j^w is cond. Independent of everything else given z_j, d_j, e^w
- The likelihood of y and z under the latent label model:

$$\underbrace{L(\{z_j\}_{j=1}^T}_{\text{latent true label}}, \underbrace{p, \{d_j\}_{j=1}^T, \{e^w\}_{w=1}^W}_{\text{latent parameters}}) = \prod_{j \le T} \underbrace{\sum_{z_j \in \{1,\dots,K\}} \Pr(z_j | p) \prod_{w \in W_j} \Pr(y_j^w | z_j, d_j, e^w)}_{\text{likelihood of noisy and true labels for task } j}$$

- Estimate parameters and true labels by maximizing L

Latent Label Models: EM algorithm

- Maximization of the expectation of log-likelihood (LL)

$$\mathbb{E}_z \log \Pr(y, z) = \sum_{j \le T} \sum_{z_j \in \{1,\ldots,K\}} \Pr(z_j|p) \log \prod_{w \in W_j} \Pr\left(z_j \middle| p\right) \Pr(y_j^w|z_j, d_j, e^w)$$

- **E-step:** Use Bayes' theorem for posterior distribution of $\hat{z}$ given p, d, e:

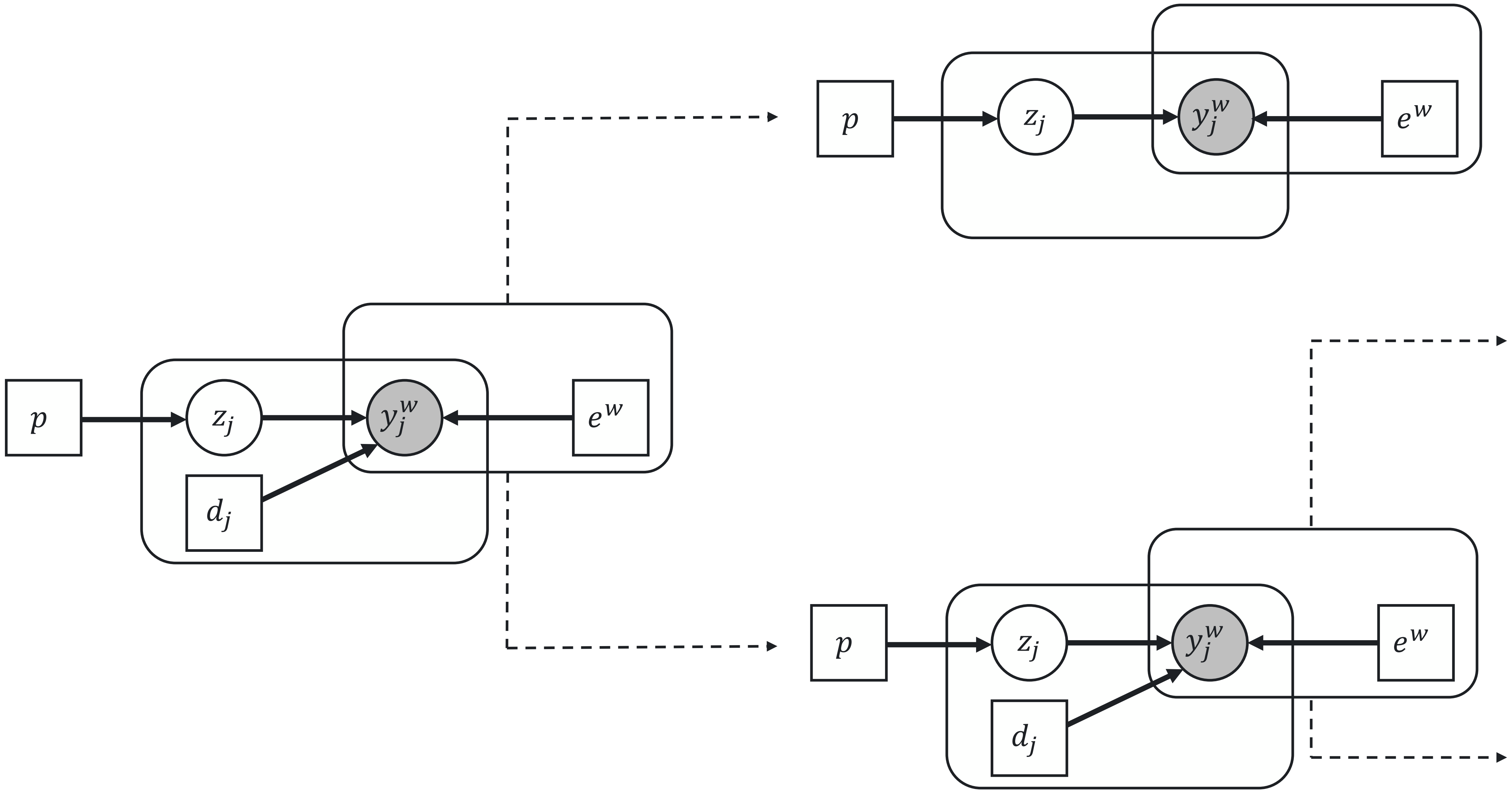
$$\hat{z}_j[c] = \Pr\left(z_j = c \middle| y, p, d, e\right) \propto \Pr\left(z_j = c \middle| p\right) \prod_{w \in W_j} \Pr\left(y_j^w \middle| z_j = c, d_j, e^w\right)$$

- **M-step:** Maximize the expectation of LL with respect to the posterior distribution of $\hat{z}$:

$$(p, d, e) = \arg\max \mathbb{E}_{\hat{z}} \log \Pr(z_j|p) \prod_{w \in W_j} \Pr(y_i^w|z_j, d_j, e^w)$$

We can use an analytical solution if exists or optimization methods such as gradient descend (with Autograd)

Latent Label Models: Special Cases



Dawid and Skene (DS):

- categories are different
- objects are similar
- workers are different

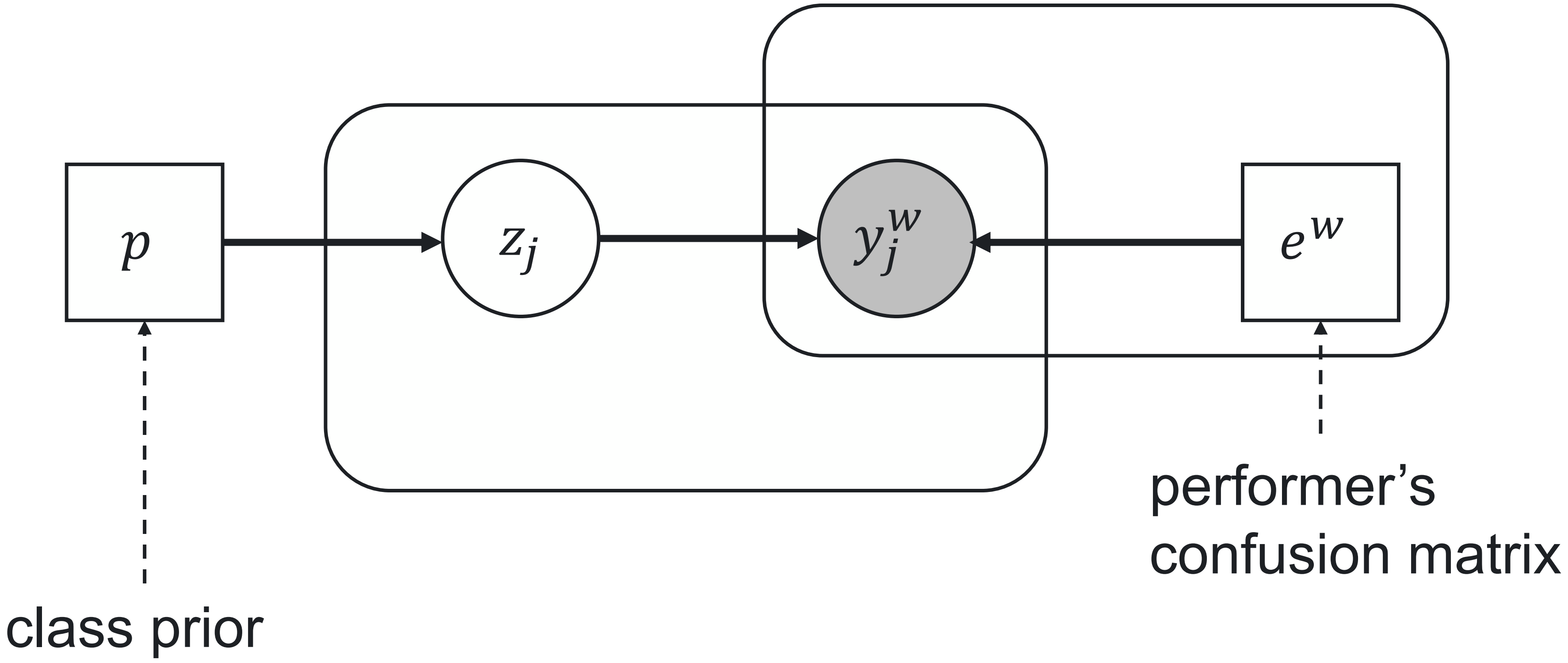
Generative model of labels, abilities, and difficulties (GLAD):

- categories are similar
- objects are different
- workers are different

Minimax conditional entropy model (MMCE):

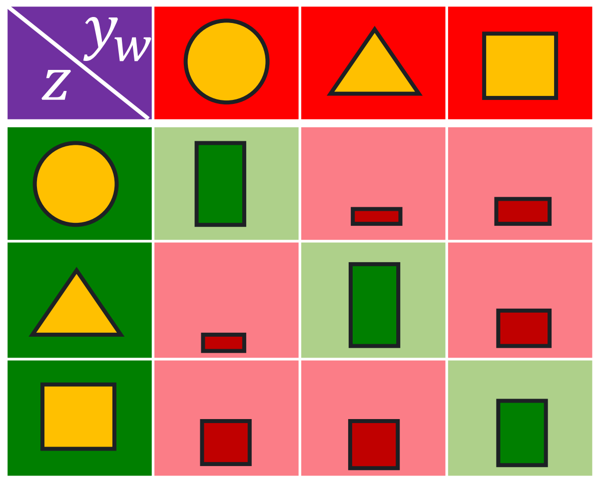
- categories are different
- objects are different
- workers are different

Dawid and Skene Model (DS)



LLM with parameters:

- p — vector of length K: $p[i] = \Pr(z = c)$
- e^w — matrix of size $K \times K$:
  $e^w[c, k] = \Pr(y^w = k | z = c)$



A.P. Dawid and A.M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. 1979

DS: Parameters Optimization

- **E-step:**

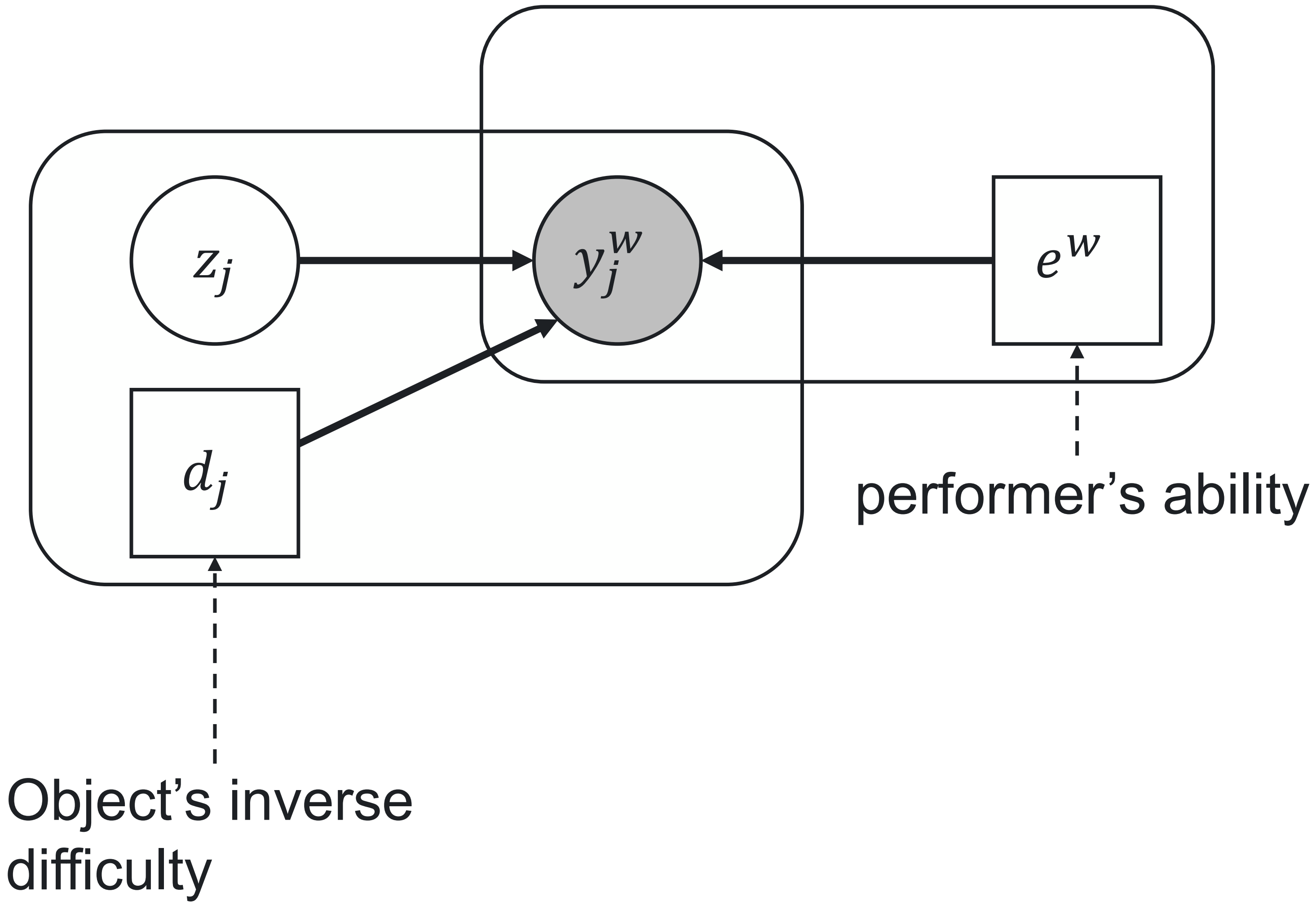
$$\hat{z}_j[c] = \frac{p[c]\prod_{w \in W_j} e^w[c, y_j^w]}{\sum_k p[k]\prod_{w \in W_j} e^w[k, y_j^w]}, \qquad c = 1, \dots, K$$

- **M-step:** Analytical solution

$$e^w[c, k] = \frac{\sum_{j \le J} \hat{z}_j[c]\delta(y_j^w = k)}{\sum_{q=1}^{K} \sum_{j \le J} \hat{z}_j[c]\delta(y_j^w = q)}, \qquad k, c = 1, \dots, K$$

$$p[c] = \frac{\sum_{j \le J} \hat{z}_j[c]}{J}, \qquad c = 1, \dots, K$$

Generative Model of Labels, Abilities, and Difficulties (GLAD)



LLM with parameters:

- scalar $d_j \in (0, \infty)$
- scalar $e^w \in (-\infty, \infty)$
- Model:

$$\Pr\left(y_j^w = k \middle| z_j = c\right) = \begin{cases} a(w,j), & c = k \\ \dfrac{1 - a(w,j)}{K - 1}, & c \neq k \end{cases}$$

$$\text{where } a(w,j) = \frac{1}{1 + \exp(-e^w d_j)}$$

J. Whitehill et al, Whose vote should count more: Optimal integration of labels from labelers of unknown expertise, 2009

GLAD: Parameters Optimization

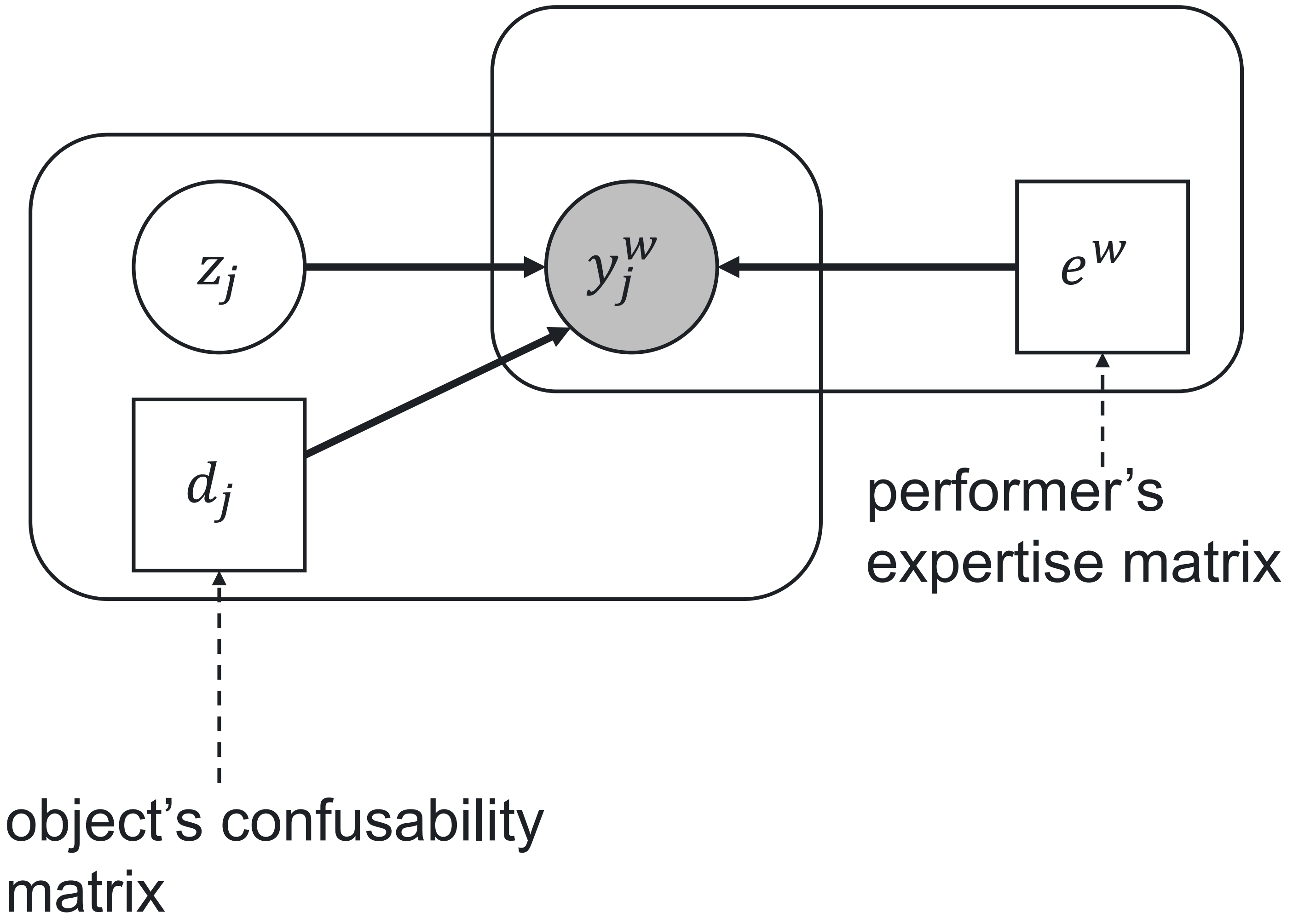
- Let $a(w,j) = \frac{1}{1+\exp(-e^{w} d_{j})}$ and $P(z_j)$ be a predefined prior (e.g., $P(z_j) = {}^1/_K$)
- **E-step:**

$$\hat{z}_j[c] \propto \Pr(z_j = c) \prod_{w \in W_j} a(w,j)^{\delta\left(y_j^w = c\right)} \left(\frac{1 - a(w,j)}{K-1}\right)^{\delta\left(y_j^w = c\right)}, \qquad \mathrm{c} = 1, \ldots, \mathrm{K}$$

- **M-step:** estimate (d, e) for given $\hat{z}$ using gradient optimization

$$(d, e) = \arg\max \sum_{j \le J} \left[\mathbb{E}_{\hat{z}_j} \log P(z_j) + \sum_{w \in W_j} \mathbb{E}_{\hat{z}_j} \log \Pr(y_j^w | z_j) \right]$$

MiniMax Conditional Entropy Model (MMCE)



Find parameters that minimize the maximum conditional entropy of observed labels:

$$\min_{Q} \max_{P} \sum_{\substack{j \le J \\ c \in \{1,\ldots,K\}}} Q(z_j = c) \sum_{\substack{w \in W \\ k \in \{1,\ldots,K\}}} P\left(y_j^w = k \middle| z_j = c\right) \log P(y_j^w = k|z_j = c)$$

LLM with parameters:

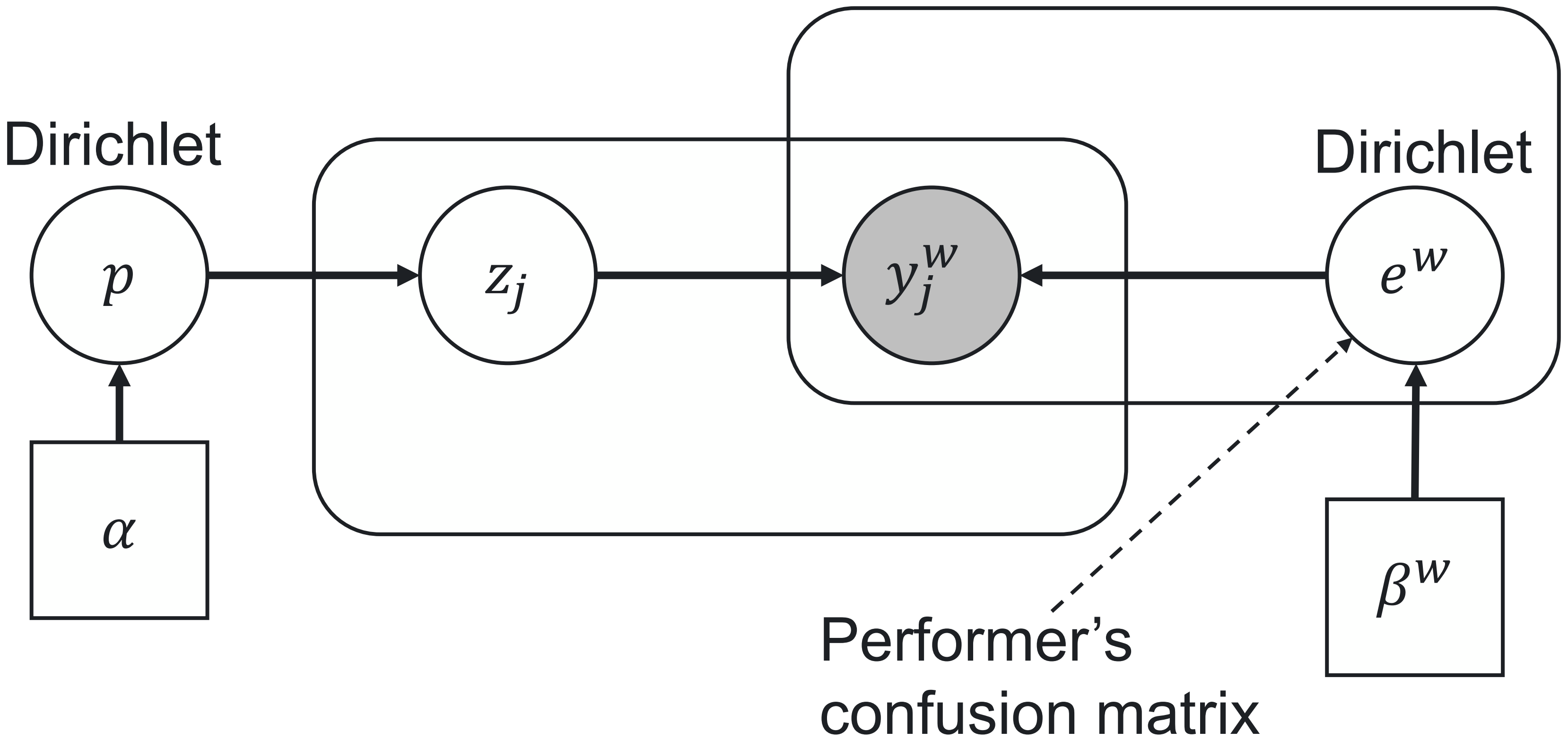
- d_j — matrix of size $K \times K$
- e^w — matrix of size $K \times K$
- Noisy label model:

$$\Pr\left(y_j^w = k \middle| z_j = c\right) = \exp(d_j[c,k] + e^w[c,k])$$

Zhou et al, Learning from the wisdom of crowds by minimax entropy, 2012

Going Deeper into Bayesian Models

Bayesian Classifier Combination (BCC)



- Object's category is generated from categorical distribution with parameter $\boldsymbol{p}$

$$z_j|\boldsymbol{p} \sim \mathrm{Cat}(z_j|\boldsymbol{p})$$

- Observed label is generated from a categorical distribution with parameters $e_{z_j}^w$:

$$y_j^w|e^w, z_j \sim \mathrm{Cat}\left(y_j^w \middle| e_{z_j}^w\right),$$

where $e_{z_j}^w$ is a row of a confusion matrix

- For p and e^w we assume prior distributions

$$p \sim \mathrm{Dir}(p|\alpha)$$
$$e_c^w \sim \mathrm{Dir}(e_c^w|\beta_c^w)$$

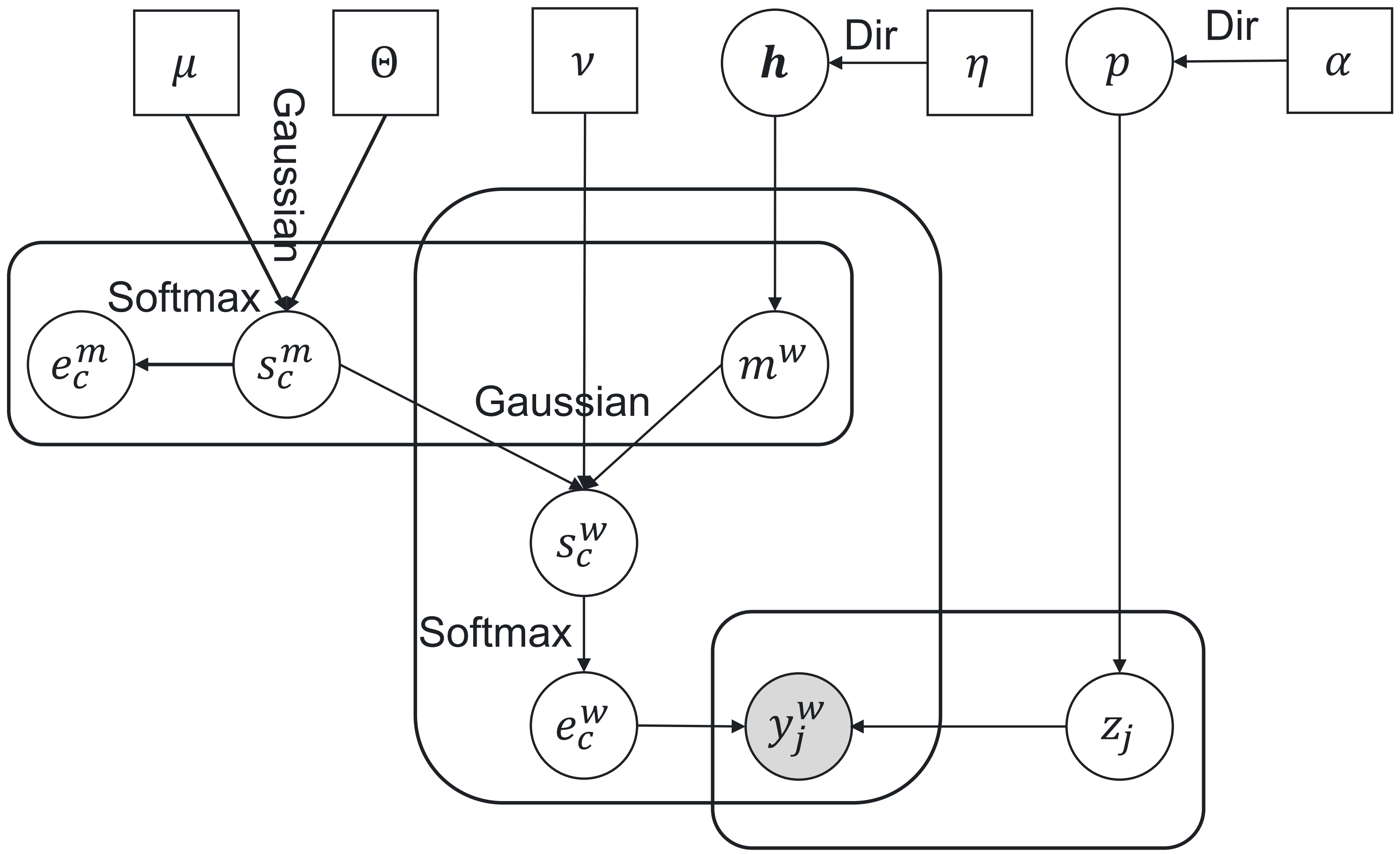
- Then the posterior distribution over model parameters, given the observed noisy labels, can be written as

$$\Pr(\boldsymbol{e}, \boldsymbol{z}, \boldsymbol{p}|\boldsymbol{Y}) \sim \mathrm{Dir}(\boldsymbol{p}|\boldsymbol{\alpha}) \prod_{j=1}^{J} \left[\mathrm{Cat}(z_j|p) \prod_{w \in W} \mathrm{Cat}\left(y_j^w \middle| e_{z_j}^w\right) \mathrm{Dir}(e_y^w|\boldsymbol{\beta})\right]$$

- Then you can obtain marginal distribution of individual parameters by integrating out all the remaining joint parameters.
- It's not possible to do it analytically, so we need to do it numerically with, for instance, Expectation Propagation (EP) algorithm

Kim and Ghahramani, Bayesian classifier combination, 2012

Community BCC



- Usually, in crowdsourcing, performers conform to a few different types, so we can represent the performers from one community through a single confusion matrix
- This allows us to encode correlations between performers' responses

- Assume that community membership variable m^w is generated from a categorical distribution with parameters h:
$$m^w|\boldsymbol{h} \sim \mathrm{Cat}(m^w|\boldsymbol{h})$$
- Each community has a probability score s_c^m representing the log probability vector of the c-th row of the confusion matrix e^m
- So, the performer's score vector is a noisy version of the community's vector:
$$s_c^w|s_c^{m^w} \sim \mathcal{N}(s_c^w|s_c^{m^w}, \nu^{-1}\boldsymbol{I})$$
- Let's also write a pretty technical thing:
$$\Pr(e_c^w|s_c^w) = \delta(e_c^w - \mathrm{softmax}(s_c^w))$$
- Then, taking into account all the priors, the joint posterior distribution:

$$\Pr(\boldsymbol{\Theta}|\boldsymbol{Y}) \propto \mathrm{Dir}(\boldsymbol{p}|\boldsymbol{\alpha}) \prod_{j=1}^{J} \left[\mathrm{Cat}(z_j|\boldsymbol{p}) \prod_{k=1}^{K} \left(\mathrm{Cat}\left(y_j^w \middle| e_{z_j}^w\right) \delta\left(e_{z_j}^{m^w} - \mathrm{softmax}\left(s_{z_j}^{m^w}\right)\right) \mathrm{Dir}(\boldsymbol{h}|\boldsymbol{\alpha}) \mathcal{N}(s_{z_j}^w|s_{z_j}^w, \boldsymbol{\nu}^{-1}) \mathcal{N}\left(s_{z_j}^{m^w} \middle| \boldsymbol{\mu}, \boldsymbol{\theta}^{-1}\right) \mathrm{Cat}(m^w|\boldsymbol{h}) \right) \right]$$

Finally, we need to find the optimal number of communities through a simple linear search on some discrete grid:

$$M^* = \arg\max_M \int_\Theta \Pr(Y|\Theta, M) \Pr(\Theta)\, d\Theta$$

Venanzi et al. Community-Based Bayesian Aggregation Models for Crowdsourcing, 2014

Which one is better?

Methods Comparison

Table 6: The Quality and Running Time of Different Methods with Complete Data (Section 6.3.1).

Method	D_Product			D_PosSent			S_Rel		S_Adult		N_Emotion		
	Accuracy	*F1-score*	*Time*	*Accuracy*	*F1-score*	*Time*	*Accuracy*	*Time*	*Accuracy*	*Time*	*MAE*	*RMSE*	*Time*
MV	89.66%	59.05%	0.13s	93.31%	92.85%	0.08s	54.19%	0.49s	36.04%	0.40s	×	×	×
ZC [16]	92.80%	63.59%	1.04s	95.10%	94.60%	0.55s	48.21%	7.39s	35.34%	6.42s	×	×	×
GLAD [53]	92.20%	60.17%	907.11s	95.20%	94.71%	407.66s	53.59%	5850.39s	36.47%	4194.50s	×	×	×
D&S [15]	93.66%	**71.59%**	1.46s	**96.00%**	**95.66%**	0.80s	61.30%	10.67s	36.05%	9.18s	×	×	×
Minimax [61]	84.09%	55.26%	272.05s	95.80%	95.43%	35.71s	57.59%	1728.09s	36.03%	1223.75s	×	×	×
BCC [27]	**93.78%**	70.10%	9.82s	**96.00%**	**95.66%**	6.06s	60.72%	153.50s	36.34%	137.92s	×	×	×
CBCC [46]	93.72%	70.87%	5.53s	**96.00%**	**95.66%**	4.12s	56.05%	44.69s	36.28%	42.52s	×	×	×
LFC [41]	93.73%	71.48%	1.42s	**96.00%**	**95.66%**	0.83s	**61.64%**	10.75s	36.29%	9.26s	×	×	×
CATD [30]	92.66%	65.92%	2.97s	95.50%	95.07%	1.32s	45.32%	16.13s	36.23%	12.96s	16.36	25.94	2.15s
PM [5, 31]	89.81%	59.34%	0.56s	95.04%	94.53%	0.33s	59.02%	2.60s	**36.50%**	2.09s	13.91	21.96	0.36s
Multi [51]	88.67%	58.32%	15.48s	95.70%	95.44%	4.98s	×	×	×	×	×	×	×
KOS [26]	89.55%	50.31%	24.06s	93.80%	93.06%	10.14s	×	×	×	×	×	×	×
VI-BP [33]	64.64%	37.43%	306.23s	**96.00%**	**95.66%**	58.52s	×	×	×	×	×	×	×
VI-MF [33]	83.91%	55.31%	38.96s	**96.00%**	**95.66%**	6.71s	×	×	×	×	×	×	×
LFC_N [41]	×	×	×	×	×	×	×	×	×	×	12.20	18.97	0.23s
Mean	×	×	×	×	×	×	×	×	×	×	**12.02**	**17.84**	0.09s
Median	×	×	×	×	×	×	×	×	×	×	13.53	21.26	0.11s

Zheng et al. Truth Inference in Crowdsourcing: Is the Problem Solved?, 2017

Conclusion

Conclusion

- Majority vote is not that bad
- We don’t have SOTA for every dataset — choose the most appropriate method for your data
- Categorical aggregation is quite overresearched problem — lots of methods but still no significant improvements since DS

Join our Slack: icwe_tutorial channel

Nikita Pavlichenko

Researcher

pavlichenko@toloka.ai

https://toloka.ai/events/icwe-2022/