Divide et Calibra: Multiclass Local Calibration via Vector Quantization

NeurIPS 2026

Cesare Barbera[1,2] Lorenzo Perini[3] Giovanni De Toni[4*]

Andrea Passerini[2] Andrea Pugnana[2]

University of Pisa[1] University of Trento[2]
Meta Platforms, Inc.[3] ETH AI Center & ETH Zurich[4]

Email: cesare.barbera@phd.unipi.it
Paper: arXiv
Code: GitHub Repo

*Work done while at Fondazione Bruno Kessler

Why Divide et Calibra?

- **The problem:** Reliable uncertainty estimates are essential for trustworthy machine learning [1, 2], yet globally calibrated models can suffer from *Proximity Bias*, *i.e.*, they remain unreliable in specific regions of the data [3]

Why Divide et Calibra?

- **The problem:** Reliable uncertainty estimates are essential for trustworthy machine learning [1, 2], yet globally calibrated models can suffer from *Proximity Bias*, *i.e.*, they remain unreliable in specific regions of the data [3]
- **The Local Calibration view:** To address this problem, predictions should match the observed class frequencies of nearby instances [4]
- **The challenge:** Estimating local calibration corrections becomes increasingly difficult in high-dimensional spaces, where data are sparse

Why Divide et Calibra?

- **The problem:** Reliable uncertainty estimates are essential for trustworthy machine learning [1, 2], yet globally calibrated models can suffer from *Proximity Bias*, *i.e.*, they remain unreliable in specific regions of the data [3]
- **The Local Calibration view:** To address this problem, predictions should match the observed class frequencies of nearby instances [4]
- **The challenge:** Estimating local calibration corrections becomes increasingly difficult in high-dimensional spaces, where data are sparse
- **Our contribution:** We introduce **Divide et Calibra**, combining Vector Quantization with compositional parameter sharing to learn region-specific calibration maps from reusable components

Key Insight

Local learning through global parameter sharing: region-specific calibration maps can be constructed compositionally from reusable latent components.

Problem Setting

- Call $\mathbf{y} = [y_1, \ldots, y_C]$ a one-hot vector for $y \in \mathcal{Y}$ with $\mid \mathcal{Y} \mid = C$
- Probabilistic classifier $f : \mathcal{X} \to \Delta^C$, $f(\mathbf{x}) = \hat{\mathbf{p}}$, with $\hat{\mathbf{p}} = [\hat{p}_1, \ldots, \hat{p}_C]$

Problem Setting

- Call $\mathbf{y} = [y_1, \ldots, y_C]$ a one-hot vector for $y \in \mathcal{Y}$ with $\mid \mathcal{Y} \mid = C$
- Probabilistic classifier $f : \mathcal{X} \to \Delta^C$, $f(\mathbf{x}) = \hat{\mathbf{p}}$, with $\hat{\mathbf{p}} = [\hat{p}_1, \ldots, \hat{p}_C]$
- Often one requires confidence calibration, i.e.,

$$\mathbb{P}\Big(Y = \hat{Y} \mid f_{\hat{Y}}(\mathbf{X}) = \hat{p}_{\hat{Y}}\Big) = \hat{p}_{\hat{Y}}, \qquad \text{with } \hat{Y} = \arg\max_{k \in \mathcal{Y}} f_k(\mathbf{X})$$

Intuition for confidence (global) calibration notion

"Among points predicted with confidence $\hat{p}_y$,
is the accuracy close to $\hat{p}_y$?"

Problem Setting

- Call $\mathbf{y} = [y_1, \ldots, y_C]$ a one-hot vector for $y \in \mathcal{Y}$ with $\mid \mathcal{Y} \mid = C$
- Probabilistic classifier $f : \mathcal{X} \to \Delta^C$, $f(\mathbf{x}) = \hat{\mathbf{p}}$, with $\hat{\mathbf{p}} = [\hat{p}_1, \ldots, \hat{p}_C]$
- Often one requires confidence calibration, i.e.,

$$\mathbb{P}\Big(Y = \hat{Y} \mid f_{\hat{Y}}(\mathbf{X}) = \hat{p}_{\hat{Y}}\Big) = \hat{p}_{\hat{Y}}, \qquad \text{with } \hat{Y} = \arg\max_{k \in \mathcal{Y}} f_k(\mathbf{X})$$

Intuition for confidence (global) calibration notion

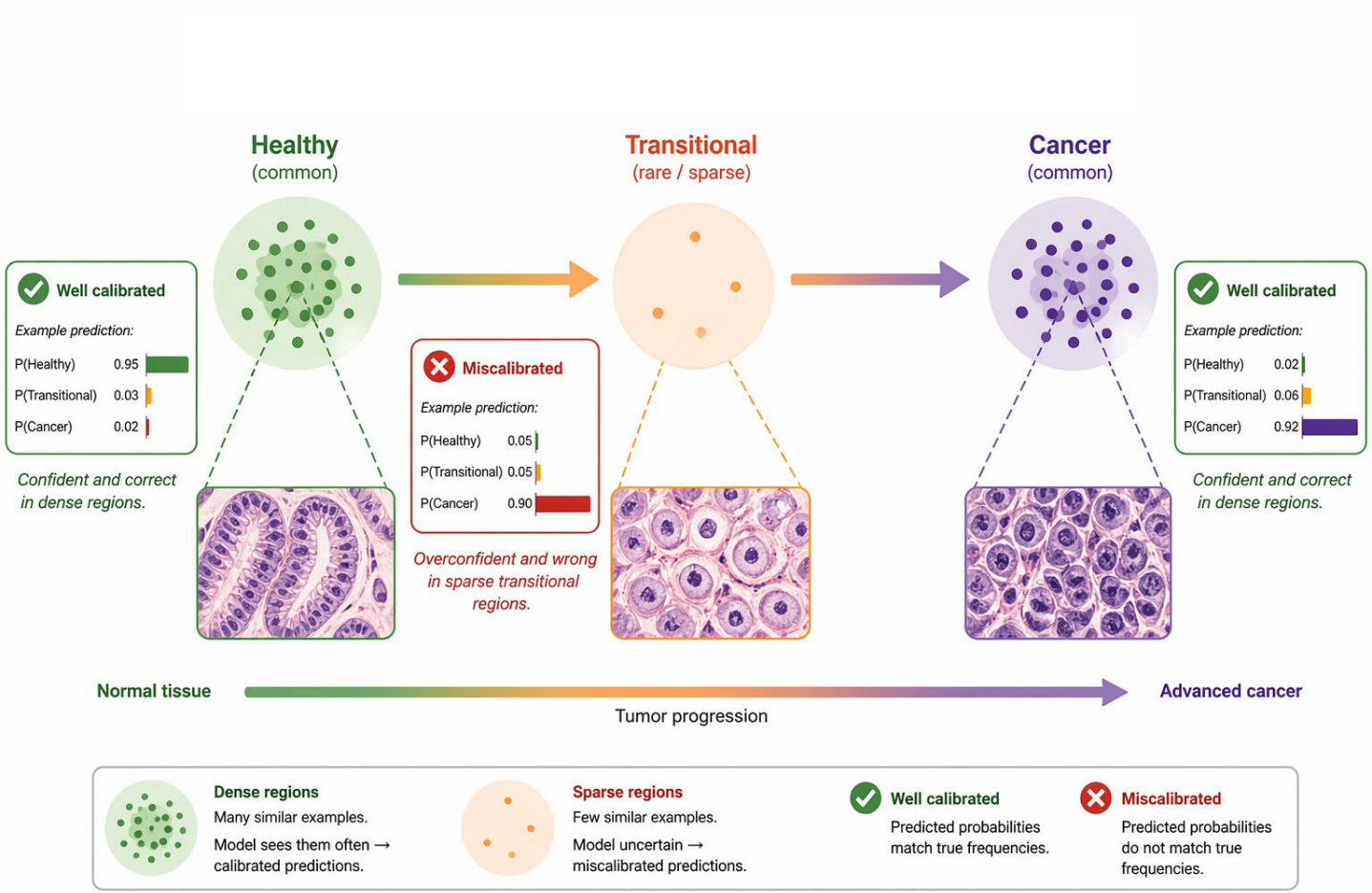
"Among points predicted with confidence $\hat{p}_y$,
is the accuracy close to $\hat{p}_y$?"

- Ideally, we want probabilistic predictions to be strongly calibrated, i.e.,

$$\mathbb{P}(Y = k|\hat{\mathbf{p}}) = \hat{p}_k \quad \forall \quad k \in \{1, \ldots, C\}$$

- Is this enough?

Motivating example: cancer cell stage detection



Calibration errors can hide in the representation space

- A model can look calibrated globally
- But it might be *unreliable in specific regions* of the representation space
- This is known as *Proximity Bias* [3]

Local calibration view

Predictions should match the class frequencies of nearby points.

$$\hat{\mathbf{p}}(x) \approx \text{local class frequencies around } x$$

$\Rightarrow$ Reliability becomes **instance- and region-aware**.

Multiclass Local Calibration Error (LCE) [5]

- **Question:** do predictions match the empirical class frequencies **around each point**?

Definition: Local Calibration Error (multiclass)

$$\mathrm{LCE} = \frac{1}{C} \sum_{b=1}^{m_B} \frac{1}{n} \sum_{i \in B_b} \left\| \frac{\sum_{j \in B_b} (\hat{p}_j - \mathbf{y}_j)\, k_\gamma(\mathbf{x}_i, \mathbf{x}_j)}{\sum_{j \in B_b} k_\gamma(\mathbf{x}_i, \mathbf{x}_j)} \right\|_1$$

- m_B is the number of bins
- B_b is a collection of indices of instance belonging to the b-th bin
- k_γ is a kernel function (it weights close points)
- γ rules locality

How to enforce local calibration for neural networks?

- Previous methods rely on kernel estimators/ KNN on representation space
- In high-dimensional spaces data becomes increasingly sparse, making local corrections difficult to estimate reliably
- To address this, we propose an approach specific for neural networks based on **Vector Quantization** (VQ) of the representation space
- We reformulate local calibration as a **compositional learning problem**: VQ identifies discrete regions (**Divide**), while reusable latent components define region-specific calibration functions (**Calibra**)
- This enables **parameter sharing across regions**, improving statistical support for local calibration

Divide et Calibra: Divide

Construction

1. **Split representation into slots**

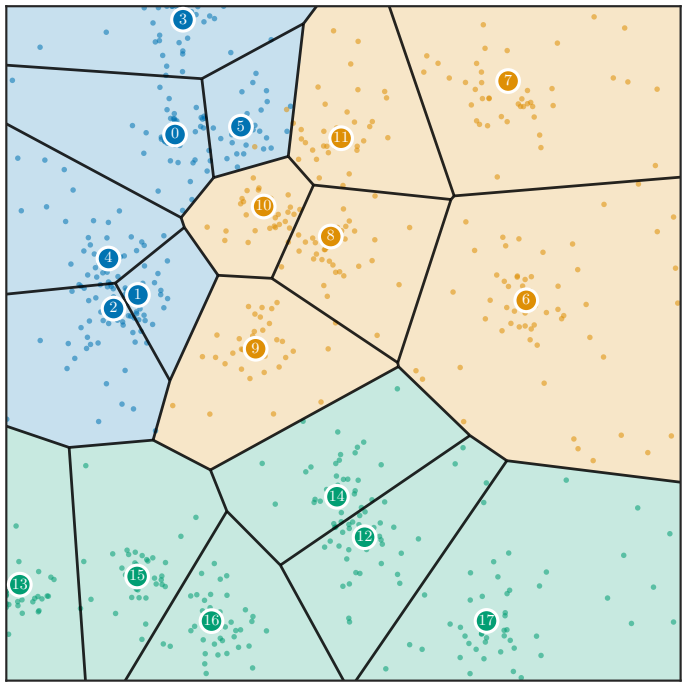
$$\mathbf{z} = E(\mathbf{x}) \in \mathbb{R}^{m'} \quad \xrightarrow{\Phi(\mathbf{z})} \quad [\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(w)}], \quad m' = w \cdot d$$

2. **Quantize each slot independently**

$$\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_{|\mathcal{C}|}\}, \qquad s(i) = \arg\min_j \|\mathbf{z}^{(i)} - \mathbf{c}_j\|_2.$$

3. **Build the discrete representation**

$$\mathbf{q}_\mathbf{s} = [\mathbf{c}_{s(1)}, \ldots, \mathbf{c}_{s(w)}].$$



Divide et Calibra: Divide

Construction

1. **Split representation into slots**

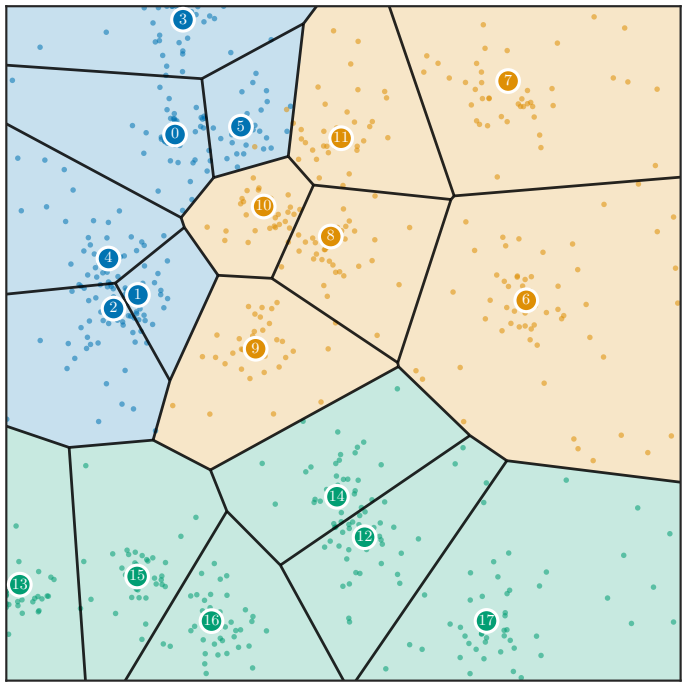
$$\mathbf{z} = E(\mathbf{x}) \in \mathbb{R}^{m'} \quad \xrightarrow{\Phi(\mathbf{z})} \quad [\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(w)}], \quad m' = w \cdot d$$

2. **Quantize each slot independently**

$$\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_{|\mathcal{C}|}\}, \qquad s(i) = \arg\min_j \|\mathbf{z}^{(i)} - \mathbf{c}_j\|_2.$$

3. **Build the discrete representation**

$$\mathbf{q}_\mathbf{s} = [\mathbf{c}_{s(1)}, \ldots, \mathbf{c}_{s(w)}].$$



Result

Each index sequence $\mathbf{s} = (s(1), \ldots, s(w))$ identifies a Voronoi cell. Up to $|\mathcal{C}|^w$ compositional regions are built.

Divide et Calibra: Calibra

Divide gives Voronoi regions. **Calibra** assigns each region its own calibration map.

Proposition 2 — Region-Conditional Calibration Maps

Assume that, conditioned on a cell $\mathcal{V}$ and the true label $y = j$, the predicted probability vector $\hat{\mathbf{p}}$ follows a Dirichlet distribution, i.e., $\hat{\mathbf{p}} \mid (y = j,\, \mathcal{V}) \;\sim\; \mathrm{Dir}\big(\boldsymbol{\alpha}^{(j,\mathcal{V})}\big)$. The calibration bias and weight vector for j and $\mathcal{V}$ are:

$$\boldsymbol{\beta}_{j,\mathcal{V}} = \log \pi_{j|\mathcal{V}} - \log \mathrm{Beta}\big(\boldsymbol{\alpha}^{(j,\mathcal{V})}\big) \qquad \boldsymbol{\omega}_{j,\mathcal{V}} = \boldsymbol{\alpha}^{(j,\mathcal{V})} - \mathbf{1}$$

and the logarithm of posterior probabilities admits the form:

$$\log \mathbb{P}(y = j \mid \widehat{\mathbf{p}}, \mathcal{V}) = \boldsymbol{\beta}_{j,\mathcal{V}} + \boldsymbol{\omega}_{j,\mathcal{V}}^{\top} \log \widehat{\mathbf{p}} \;+\; \mathrm{const}$$

Where $\mathrm{Beta}(\cdot)$ denotes the multivariate Beta function, $\pi_{j|\mathcal{V}}$ a possibly cell-dependent prior probability and $\mathbf{1} \in \mathbb{R}^{|\mathcal{Y}|}$ is the all ones vector.

Takeaway

Instead of a global map, each Voronoi region gets a log-linear Dirichlet calibration map.

Too many regions!

Problem

Each instance is assigned an index sequence

$$\mathbf{s} = (s(1), \ldots, s(w)), \quad s(i) \in \{1, \ldots, |\mathcal{C}|\}$$

But too many regions ($|\mathcal{C}|^w$) to learn independent calibration maps!

Too many regions!

Problem

Each instance is assigned an index sequence

$$\mathbf{s} = (s(1), \ldots, s(w)), \quad s(i) \in \{1, \ldots, |\mathcal{C}|\}$$

But too many regions ($|\mathcal{C}|^w$) to learn independent calibration maps!

Shared Parameters

Create local maps compositionally from shared parameters:

$$\mathcal{A} = \{\mathbf{a}_1, \ldots, \mathbf{a}_{|\mathcal{C}|}\}, \qquad \mathcal{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_{|\mathcal{C}|}\},$$

with both $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{|\mathcal{Y}|} \times \mathbb{R}^{|\mathcal{C}|}$

Indexed Parametrization Trick

Idea: Indexed Parameterization Trick

For region $\mathcal{V}$ with index sequence $\mathbf{s}$, we build two matrices:

$$\mathbf{A}^{(\mathcal{V})} = [\mathbf{a}_{s(1)}, \ldots, \mathbf{a}_{s(w)}]^\top, \qquad \mathbf{B}^{(\mathcal{V})} = [\mathbf{b}_{s(1)}, \ldots, \mathbf{b}_{s(w)}]^\top$$

with $\mathbf{A}^{(\mathcal{V})}, \mathbf{B}^{(\mathcal{V})} \in \mathbb{R}^w \times \mathbb{R}^{|\mathcal{Y}|}$

Indexed Parametrization Trick

Idea: Indexed Parameterization Trick

For region $\mathcal{V}$ with index sequence $\mathbf{s}$, we build two matrices:

$$\mathbf{A}^{(\mathcal{V})} = [\mathbf{a}_{s(1)}, \ldots, \mathbf{a}_{s(w)}]^\top, \qquad \mathbf{B}^{(\mathcal{V})} = [\mathbf{b}_{s(1)}, \ldots, \mathbf{b}_{s(w)}]^\top$$

with $\mathbf{A}^{(\mathcal{V})}, \mathbf{B}^{(\mathcal{V})} \in \mathbb{R}^w \times \mathbb{R}^{|\mathcal{Y}|}$

Compositional Map

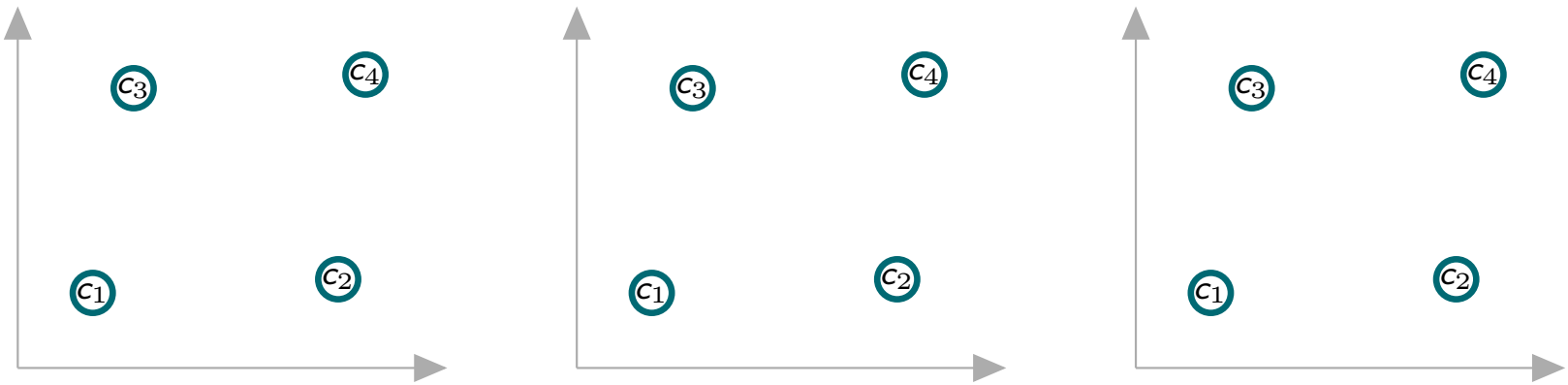
Then compose each region map as:

$$\boldsymbol{\alpha}^{(\mathcal{V})} \;=\; \phi\big(\mathbf{A}^{(\mathcal{V})\top} \mathrm{diag}(\boldsymbol{\sigma}^2)\, \mathbf{B}^{(\mathcal{V})}\big) + \mathbf{I}, \tag{1}$$

where $\boldsymbol{\alpha}^{\mathcal{V}} \in \mathbb{R}^{|\mathcal{Y}|} \times \mathbb{R}^{|\mathcal{Y}|}$ is a matrix storing the local map Dirichlet parameters, ϕ is a positive monotone link function (*e.g.*, the softplus), $\boldsymbol{\sigma}$ is a slot-dependent scaling and $\mathbf{I}$ is an identity shift to ensure $\alpha_{jj}^{(\mathcal{V})} > 1$.

Practical Example: From Slots to Shared Parameters

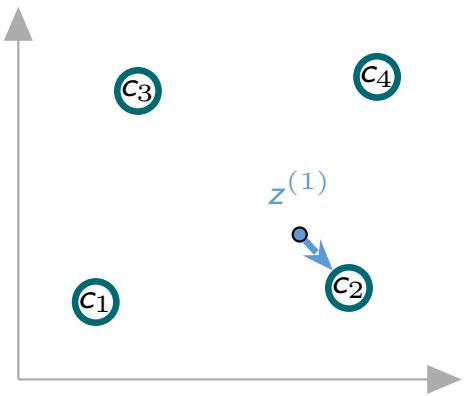
For illustration, take $\mathbf{z} \in \mathbb{R}^6$ ($d = 2$, $w = 3$), shared codebooks $\mathcal{C} = \{c_1, c_2, c_3, c_4\} \subset \mathbb{R}^2$ and $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$, $\mathcal{B} = \{b_1, b_2, b_3, b_4\} \subset \mathbb{R}^{|\mathcal{Y}|}$



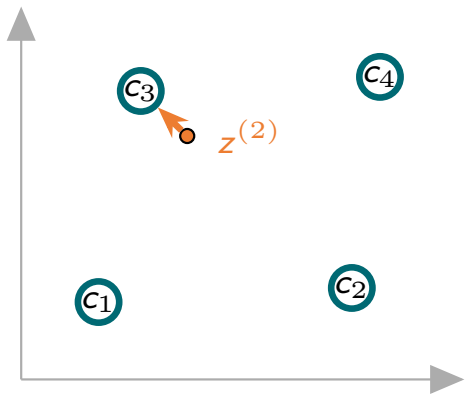
Practical Example: From Slots to Shared Parameters

For illustration, take $\mathbf{z} \in \mathbb{R}^6$ ($d = 2$, $w = 3$), shared codebooks $\mathcal{C} = \{c_1, c_2, c_3, c_4\} \subset \mathbb{R}^2$ and $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$, $\mathcal{B} = \{b_1, b_2, b_3, b_4\} \subset \mathbb{R}^{|\mathcal{Y}|}$

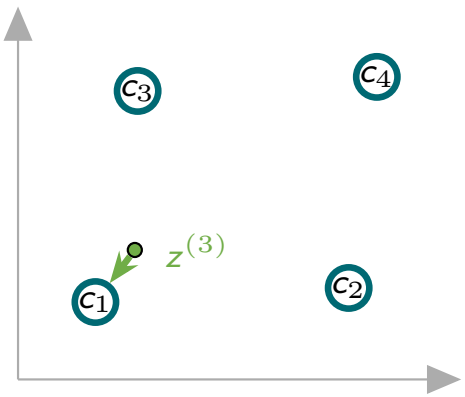
① **Codeword assignment**



slot 1: $s^{(1)} = 2$



slot 2: $s^{(2)} = 3$

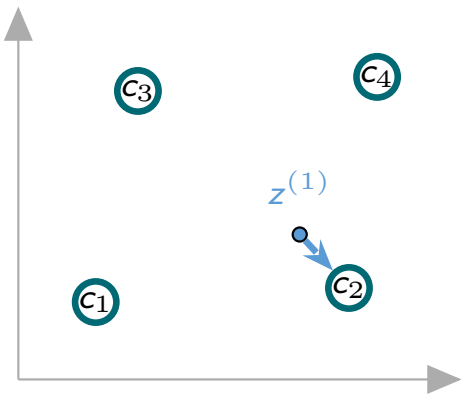


slot 3: $s^{(3)} = 1$

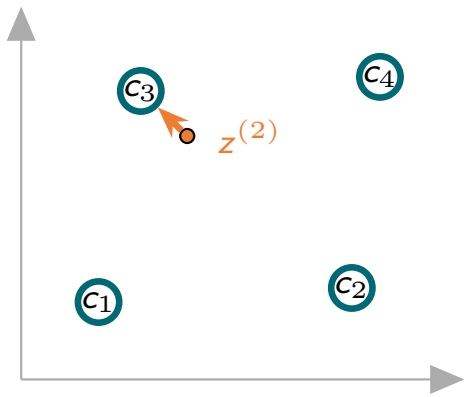
Practical Example: From Slots to Shared Parameters

For illustration, take $\mathbf{z} \in \mathbb{R}^6$ ($d = 2$, $w = 3$), shared codebooks $\mathcal{C} = \{c_1, c_2, c_3, c_4\} \subset \mathbb{R}^2$ and $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$, $\mathcal{B} = \{b_1, b_2, b_3, b_4\} \subset \mathbb{R}^{|\mathcal{Y}|}$

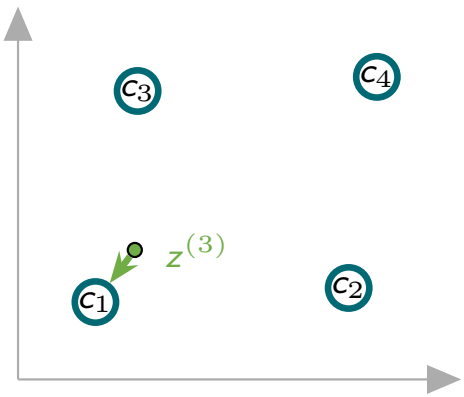
① Codeword assignment



slot 1: $s^{(1)} = 2$



slot 2: $s^{(2)} = 3$



slot 3: $s^{(3)} = 1$

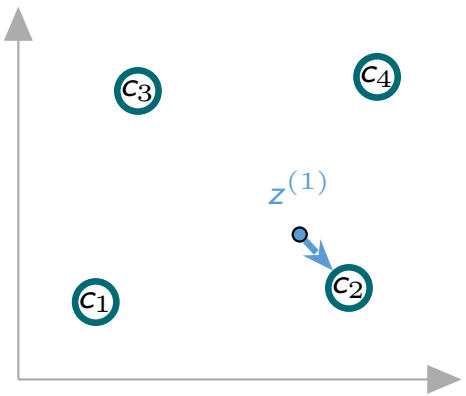
② Calibration parameter selection

compositional region: $\mathbf{s} = (2, 3, 1)$

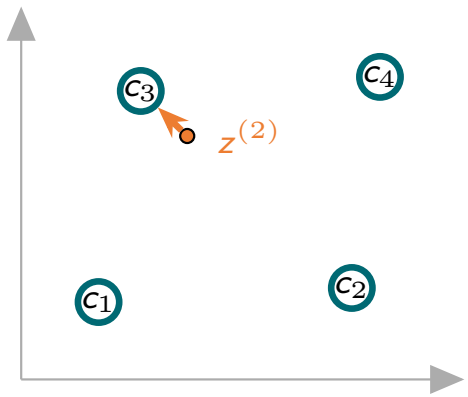
Practical Example: From Slots to Shared Parameters

For illustration, take $\mathbf{z} \in \mathbb{R}^6$ ($d = 2$, $w = 3$), shared codebooks $\mathcal{C} = \{c_1, c_2, c_3, c_4\} \subset \mathbb{R}^2$ and $\mathcal{A} = \{a_1, a_2, a_3, a_4\}$, $\mathcal{B} = \{b_1, b_2, b_3, b_4\} \subset \mathbb{R}^{|\mathcal{V}|}$

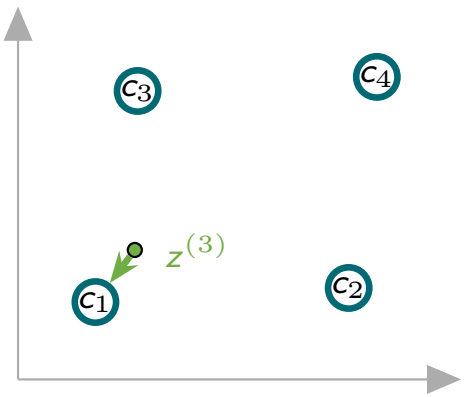
① Codeword assignment



slot 1: $s^{(1)} = 2$



slot 2: $s^{(2)} = 3$



slot 3: $s^{(3)} = 1$

② Calibration parameter selection

compositional region: $\mathbf{s} = (2, 3, 1)$ — index lookup →

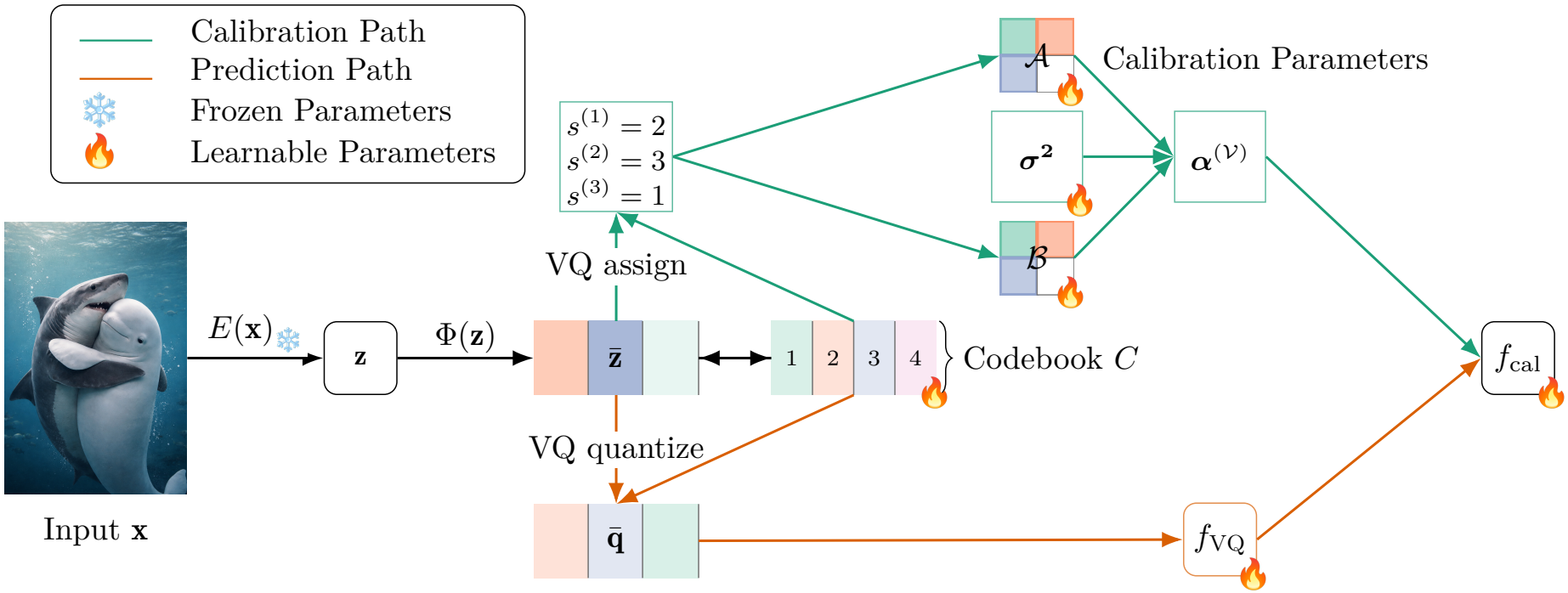
$$A^{(\mathcal{V})} = [a_2, a_3, a_1]^\top$$

$$B^{(\mathcal{V})} = [b_2, b_3, b_1]^\top$$

Shared factors define the region-specific calibration function:

$$\boldsymbol{\alpha}^{(\mathcal{V})} = \phi\left(\mathbf{A}^{(\mathcal{V})\top} \operatorname{diag}(\boldsymbol{\sigma}^2)\mathbf{B}^{(\mathcal{V})}\right) + \mathbf{I}$$

Divide et Calibra: Pipeline



Pipeline with $w = 3$ slots and $|\mathcal{C}| = 4$ codewords.

Parameter sharing

Region-specific maps are built compositionally from shared factors. This is parameter efficient:
$(|\mathcal{Y}|^2 + |\mathcal{Y}|) \times |\mathcal{C}|^w \to 2 \times |\mathcal{C}| \times |\mathcal{Y}| + |\mathcal{Y}| + w$

Statistical Stability of Calibration Codebook

Why VQ helps

A Voronoi cell may be rare but each codeword $s(i)$ can appear in many other regions.

Parameter sharing turns region sparsity into codeword frequency.

Corollary (Cell-Level Estimation Error)

Under regularity assumptions, assume that ϕ is 1-Lipschitz. Since the codebook is fixed and finite, for every fixed cell $\mathcal{V}$ with index sequence $\mathbf{s}$ composed of active codewords,

$$\left\|\hat{\boldsymbol{\alpha}}^{(\mathcal{V})} - \boldsymbol{\alpha}^{(\mathcal{V})\star}\right\|_F = O_p\left(\sum_{i=1}^{w} \sigma_i^2 \left[\frac{1}{\sqrt{N p_{s(i)}}} + \frac{1}{\sqrt{N}} + \frac{\varepsilon_N}{p_{s(i)}}\right]\right),$$

where N is the number of samples and $p_k = \sum_{i=1}^{w} \mathbb{P}(s(i) = k)$ denotes the expected number of occurrences of codeword k per sample.

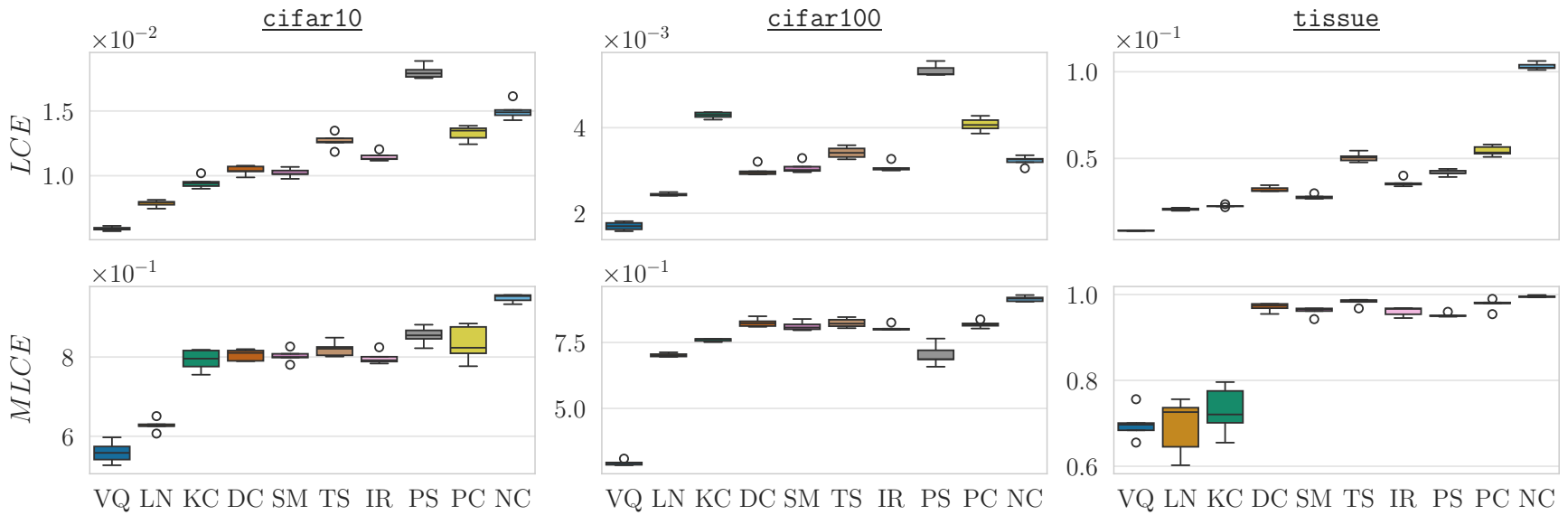
Takeaway

The estimation rate depends on **codeword reuse**, not on the density of the Voronoi cell.

Results: Local Calibration Metrics

Setup

Comparison of local calibration performance over five runs (`ResNet` backbone), measured by **LCE** and **MLCE**.



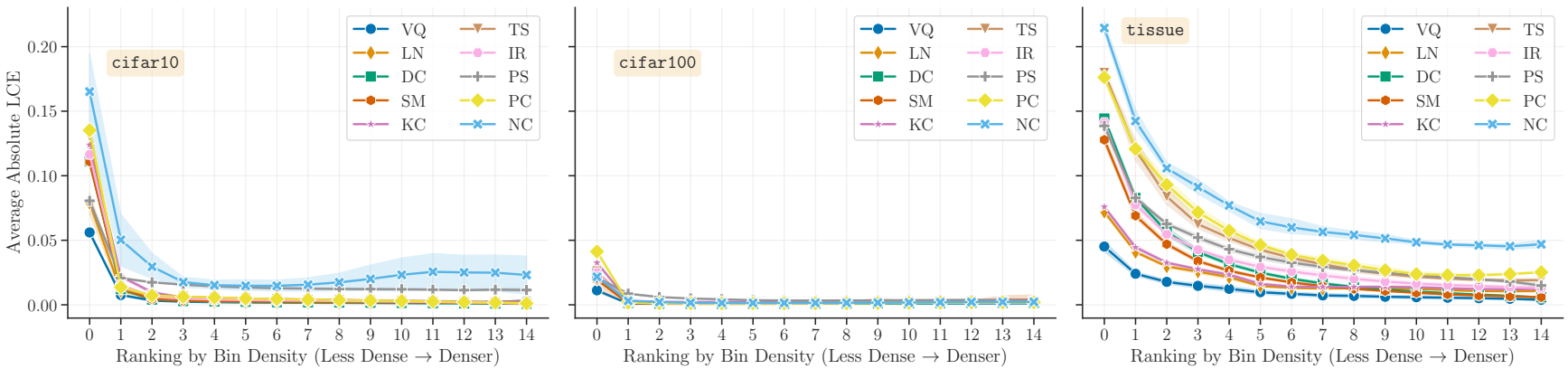
Takeaway

VQ achieves the best **LCE** on all datasets and the best or second-best maximum-**LCE**.

Results: Local Calibration under Data Sparsity

Setup

We group samples by the **Effective Sample Size (ESS)** and report the average absolute **LCE** in each bin.



Local calibration in density-based sub-bins over five runs (`ResNet` backbone).

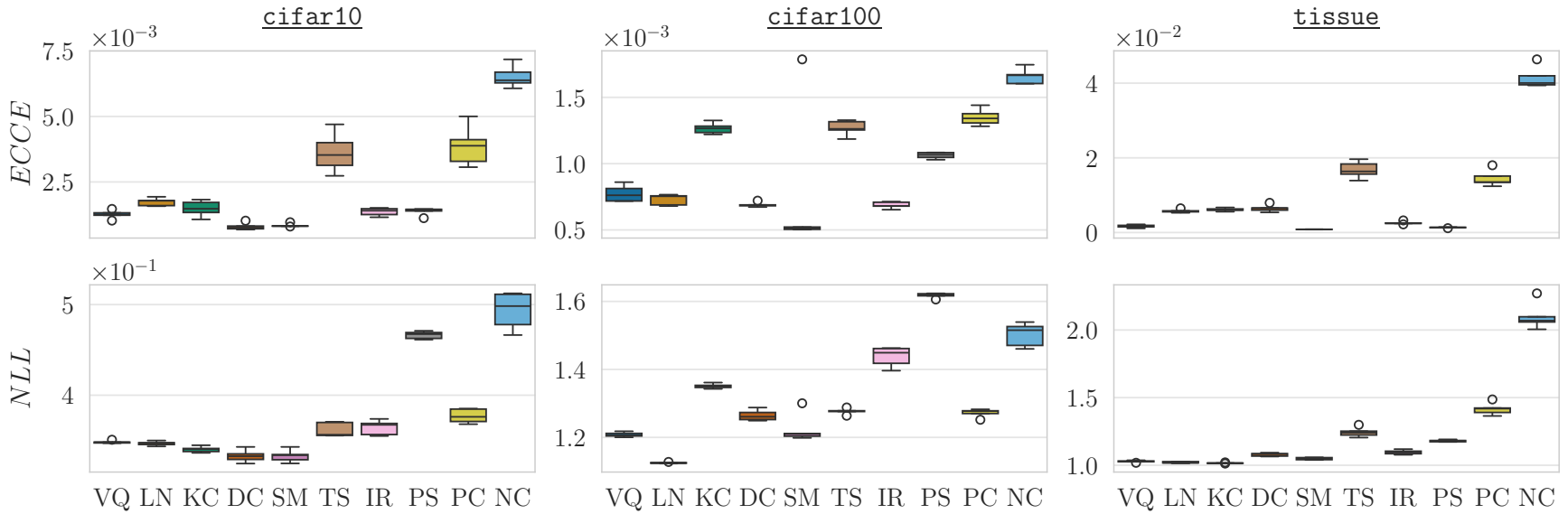
Main observations

- Local calibration error decreases as *ESS* increases; low-*ESS* regions are harder to calibrate.
- **VQ** achieves lower *LCE* across the spectrum, with the largest gains in **low-support regions**.

Results: Global Calibration and Predictive Metrics

Setup

Comparison of global calibration and predictive performance over five runs (`ResNet` backbone).



Takeaway

- : VQ achieves substantial improvements in local calibration while maintaining competitive global and predictive performance measured in terms of **ECCE** [6] and **NLL** [7].

Wrapping Up

- We introduced **Divide et Calibra**, a Vector Quantization-based framework for multiclass local calibration.
- Through **compositional parameter sharing**, we enable region-specific calibration while mitigating data sparsity.
- **Open questions:**
  - **Local specialization vs. parameter sharing:** How can we optimally balance flexible *local* corrections with reliable *global* statistical estimation?
  - **Bridging local calibration and multicalibration:** Can subpopulations identified over the representation space provide meaningful calibration guarantees for input-space groups [8]?

Be in touch

- Email: **cesare.barbera@phd.unipi.it**
- X: **@cesbar99**
- Paper: **arXiv**
- Code: **GitHub Repo**



References I

[1] Telmo de Menezes Silva Filho, Hao Song, Miquel Perelló-Nieto, Raúl Santos-Rodríguez, Meelis Kull, and Peter A. Flach. Classifier calibration: a survey on how to assess and improve predicted class probabilities. *Mach. Learn.*, 112(9): 3211–3260, 2023.

[2] Abhishek Singh Sambyal, Usma Niyaz, Narayanan C. Krishnan, and Deepti R. Bathula. Understanding calibration of deep neural networks for medical image classification. *Comput. Methods Programs Biomed.*, 242:107816, 2023.

[3] Miao Xiong, Ailin Deng, Pang Wei Koh, Jiaying Wu, Shen Li, Jianqing Xu, and Bryan Hooi. Proximity-informed calibration for deep neural networks. In *NeurIPS*, 2023.

[4] Rachel Luo, Aadyot Bhatnagar, Yu Bai, Shengjia Zhao, Huan Wang, Caiming Xiong, Silvio Savarese, Stefano Ermon, Edward Schmerling, and Marco Pavone. Local calibration: metrics and recalibration. In *UAI*, volume 180 of *Proceedings of Machine Learning Research*, pages 1286–1295. PMLR, 2022.

[5] Cesare Barbera, Lorenzo Perini, Giovanni De Toni, Andrea Passerini, and Andrea Pugnana. Multiclass local calibration with the jensen-shannon distance. In *AISTATS*, 2026.

References II

[6] Imanol Arrieta Ibarra, Paman Gujral, Jonathan Tannen, Mark Tygert, and Cherie Xu. Metrics of calibration for probabilistic predictions. *J. Mach. Learn. Res.*, 23: 351:1–351:54, 2022.

[7] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

[8] Úrsula Hébert-Johnson, Michael P. Kim, Omer Reingold, and Guy N. Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *ICML*, volume 80 of *Proceedings of Machine Learning Research*, pages 1944–1953. PMLR, 2018.