Computer Vision

Multi-Label Image Classification
Query2Label & MLSPL

Group 25
Riajul Islam, Andreas C. Kreth, Christine Midtgaard

June 19, 2025

Aarhus University

Overview

- Problem & Motivation
- Related works
- Background concepts
- Method 1: Query2Label (Q2L)
- Method 2: Multi-label Learning from Single Positive Labels (MLSPL)
- Experiments
- Results & Discussion
- Conclusion

Problem & Motivation

What is Multi-label Classification?

- Assigns multiple labels to a single instance (e.g., an image with both “dog” and “cat”).
- Unlike single-label (one label per image) or multi-class (choose one among many), multi-label allows overlapping labels.

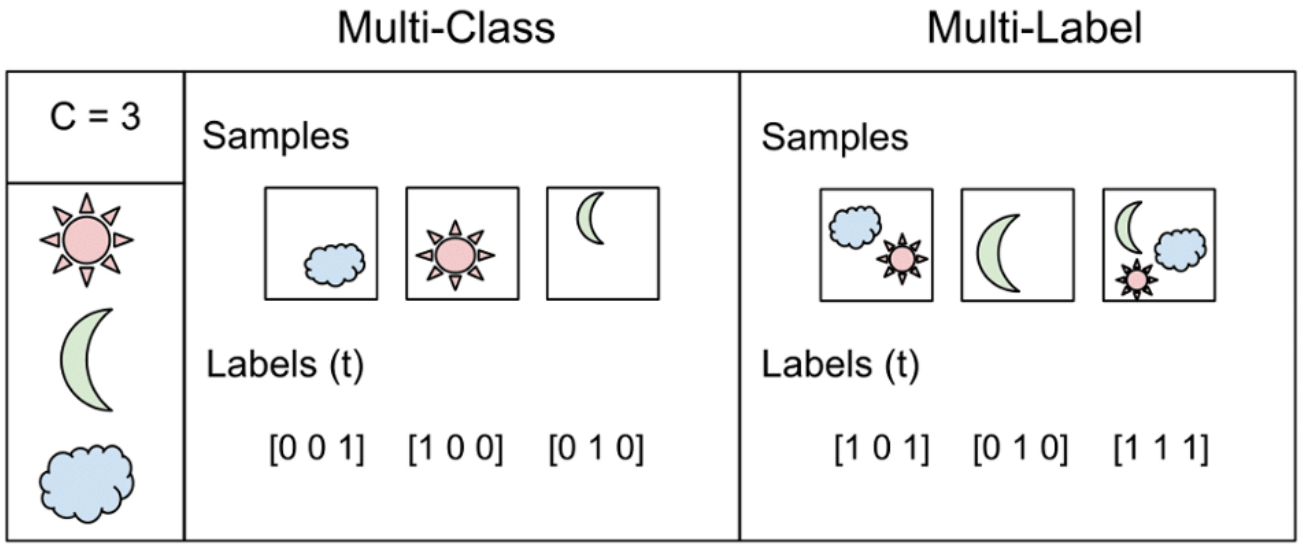


Illustration of single-label vs multi-class vs multi-label classification. From `https://medium.com/@wongsirikuln/fire-alert-system-with-multi-label-classification-model-explained-by-gradcam-bc18affe178c`.

Problem & Motivation

Applications

- Image tagging
- Medical imaging
- Autonomous driving
- Etc.

Problem & Motivation

Challenges

- Small objects and background clutter (feature recognition).
- Severe class imbalance.
- Growing annotation cost.
- Sparse annotations.

Problem & Motivation

Project goals

- Implemented two approaches:
  Q2L and **MLSPL**.
- Reproduce results on MS-COCO 2014.
- Analyze resource trade-offs and reproducibility.

Related work

- Loss functions
- Locating areas of interest
- PU Learning
- Partially observed labels
- Attention and Transformer architectures

Background

Concepts

- Multi-label Classification (MLC)
- Convolutional Neural Networks (CNNs)
- Transformers
- Loss functions

Background

Multi-label classification:

The goal is to create a model that outputs the probability $p = [p_1, ..., p_K]$ of the presence of a category K in an input image $x \in \mathcal{X}$.

Labels $y = [y_1, ..., y_K]$ from the label space $\mathcal{Y} = \{0, 1\}^K$.

- $y_k = 1$: k is present.
- $y_k = 0$: otherwise.

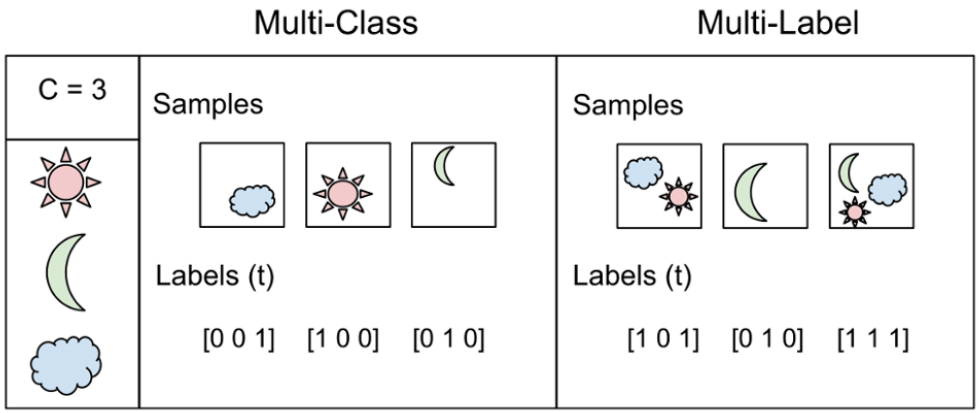


Illustration of single-label vs multi-class vs multi-label classification. From `https://medium.com/@wongsirikuln/fire-alert-system-with-multi-label-classification-model-explained-by-gradcam-bc18affe178c`.

Background

CNNs: Three main types of layers:

- Convolutional layers: extract spatial features.
- Pooling layers: reduce dimensionality and summarise information.
- Fully Connected (FC) layer: interpret features for classification.

Background

Transformers:

Vision Transformers (ViTs):

Loss functions

- Measures distance between prediction (model output) and ground truth.
- Guides parameter optimization.
- Multi-label learning adds complexity: an image may trigger multiple simultaneous decisions.
- Choosing an appropriate loss is therefore important.

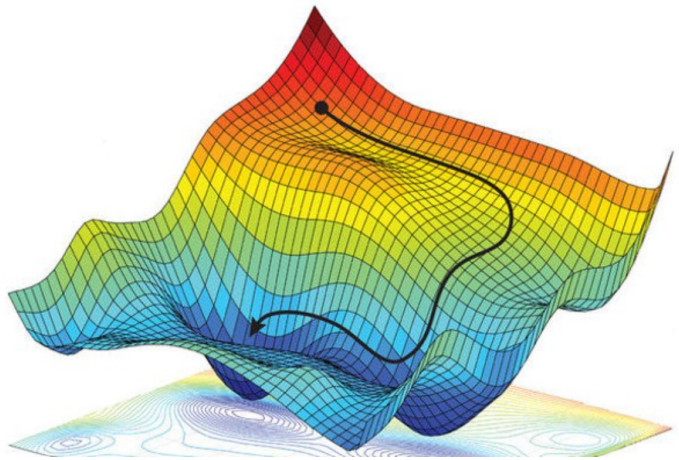


Illustration of a loss landscape.

Background

Loss functions covered:

- Binary Cross-Entropy (BCE)
- Expected Positive Regularisation (EPR)
- Regularised Online Label Estimation (ROLE)
- Focal and Asymmetric Focal Losses

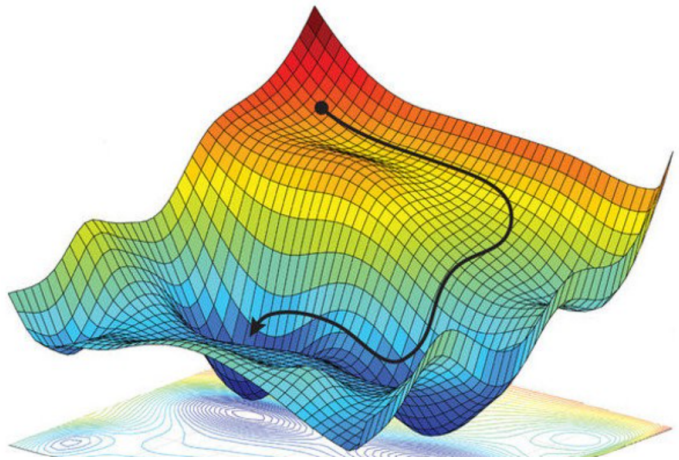


Illustration of a loss landscape.

Binary Cross-Entropy (BCE):

- Independent positive/negative prediction per class: $y_i \in \{0, 1\}$
- Model outputs probability $p_i \in [0, 1]$ for each class $i \in \{1, \ldots, K\}$.

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{K} \sum_{i=1}^{K} [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

Background

BCE limitations:

- **Class imbalance:** gradients dominated by frequent classes $\rightarrow$ rare classes under-represented.
- **Missing labels:** datasets rarely fully annotated. BCE treats unobserved labels as negatives, introducing false-negative bias.

Positive-only BCE used when only the observed positive label $z_{ni} = 1$ is known:

$$\mathcal{L}^{+}_{\text{BCE}}(\mathbf{f}_n, \mathbf{z}_n) = -\sum_{i=1}^{K} \mathbf{1}[z_{ni} = 1] \log f_{ni}$$

Expected Positive Regularisation (EPR)

$$\mathcal{L}_{EPR}(\mathbf{F}_B, \mathbf{Z}_B) = \frac{1}{|B|} \sum_{n \in B} \mathcal{L}^{+}_{BCE}(\mathbf{f}_n, \mathbf{z}_n) + \lambda R_\kappa(\mathbf{F}_B),$$

- κ: expected positives per image (domain prior).
- $\hat{\kappa}$: average predicted positives in the batch.
- Batch penalty: $R_\kappa(\mathbf{F}_B) = \left(\frac{\hat{\kappa}(\mathbf{F}_B) - \kappa}{K}\right)^2$.

Regularised Online Label Estimation (ROLE)

- Jointly trains classifier $f(\cdot;\theta)$ and label-estimator $g(\cdot;\phi)$.
- Alternating updates:

$$\mathcal{L}'(\mathbf{F}_B, \tilde{\mathbf{Y}}_B) = \frac{1}{|B|} \sum_{n \in B} \mathcal{L}_{BCE}(\mathbf{f}_n, \text{sg}(\tilde{\mathbf{y}}_n)) + \mathcal{L}_{EPR}(\mathbf{F}_B, \mathbf{Z}_B)$$

- Softly imputes missing labels ($0 < y_{ni} < 1$) and reduces false negatives.

$$\mathcal{L}_{ROLE}(\mathbf{F}_B, \tilde{\mathbf{Y}}_B) = \frac{\mathcal{L}'(\mathbf{F}_B|\tilde{\mathbf{Y}}) + \mathcal{L}'(\tilde{\mathbf{Y}}|\mathbf{F}_B)}{2}$$

Focal loss

$$\mathcal{L}_{\mathrm{FL}} = -\frac{1}{K}\sum_{i=1}^{K}\alpha_i(1-p_i)^{\gamma}y_i\log p_i$$

- Down-weights easy examples; focuses on hard, misclassified ones.
- Hyperparameters: focusing γ and class weight α.

Assymetric focal loss

- Variant tailored for imbalance in multi-label settings.
- Uses different focusing parameters for positive vs. negative terms.

$$\mathcal{L} = \frac{1}{K} \sum_{k=1}^{K} \begin{cases} (1 - p_k)^{\gamma^+} \log(p_k), & \text{if } y_k = 1 \\ (p_k)^{\gamma^-} \log(1 - p_k), & \text{if } y_k = 0 \end{cases}$$

Method

- **Feature localization**: Query2Label: A Simple Transformer Way to Multi-Label Classification (Q2L)
- **Sparse label annotation**: Multi-Label Learning from Single Positive Labels (MLSPL)

Query2Label: Core Idea

- Treat each label as a *learnable query* in a Transformer decoder.
- Decoder cross-attends to backbone feature map.
- Produces class-specific feature vectors.

Q2L Architecture

- Stage 1: CNN/ViT backbone extracts spatial features.
- Stage 2: Multi-layer Transformer decoder with K queries.
- Linear head $\rightarrow$ K sigmoid outputs (one per label).
- Loss: Asymmetric Focal (AF) to mitigate imbalance.

MLSPL: Learning from One Positive Label

- Extreme weak supervision: exactly **one** positive per image.
- Objective: recover missing positives during training.

Method

ROLE: Regularised Online Label Estimation

- Maintain soft estimates $\hat{y}$ for unobserved labels.
- Jointly optimise classifier & label estimator.
- Batch-level regulariser keeps expected positives $\approx \kappa$.

Experiments

- **Hardware:** 2 × NVIDIA RTX 3060 GPUs (12 GB VRAM) on Ubuntu 22.04.5 LTS.
- **Q2L Environment:**
  - Python 3.7.3, PyTorch 1.9.0, Torchvision 0.10.0, CUDA 11.1.
  - `inplace_abn` rebuilt from earlier release for compatibility.
  - **RandAugment dependency missing:**
    - Added local `augmentations.py` from *pytorch-randaugment*.
    - Changed import in `get_dataset.py`: `from .augmentations import RandAugment`.
    - Replaced default call with `RandAugment(n=2, m=9)` to set parameters explicitly.
- **MLSPL Environment:**
  - Python 3.11.8, PyTorch 2.2.1, Torchvision 0.17.1, CUDA 12.4.
  - Single-GPU training to ensure reproducibility across runs.

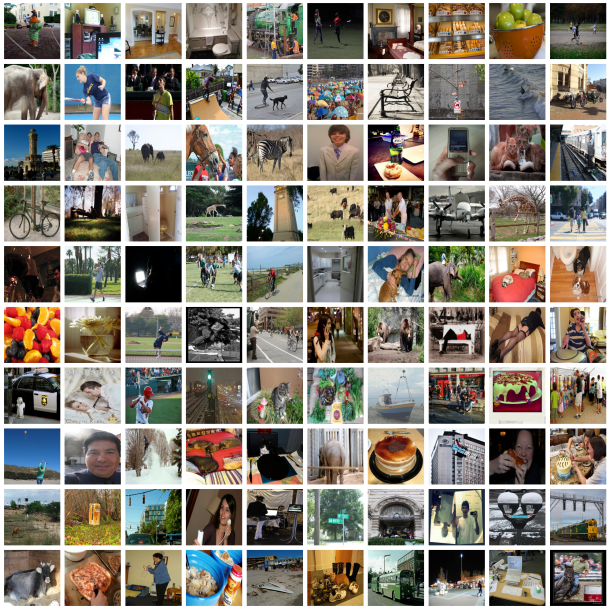
Experiments

Dataset: MS-COCO 2014

- 82,081 training images
- 40,137 validation images.
- 80 categories - multiple objects per image.

Metric: mean Average Precision (mAP)

- Widely reported metric in MLC.
- Used in both Q2L and MLSPL.



Example COCO images (448×448)

Experiments

Q2L Backbones evaluated

- ResNet-101: 448 and 576
- TResNetL 448
- TresNetL (22k) 448
- Swin-L (22k) 384
- CvT-w24 (22k) 384

Training protocol

- Pre-trained weights from authors - batch size 16.
- Scratch training attempted with all backbones.
- Memory overflow with SwinL and CvT-w24 during training.

Experiments

MLSPL Experiments

- Convert each image to exactly one positive label.
- *Linear* classifier on fixed features (25 epochs).
- End-to-end *fine-tuning* (10 epochs).

Hyperparameter search

- Learning rate $\in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.
- Batch size $\in \{8, 16\}$.

Selected configuration

- **Linear**: lr $= 10^{-3}$, batchsize 16.
- **Fine-tuned**: lr $= 10^{-5}$, batchsize 16.

Q2L

Table 1: Mean Average Precision (mAP) comparison between our reproduced results and the Q2L paper on MS-COCO 2014. All values in percentage.

Method	Backbone	Resolution	mAP (Ours)	mAP (Paper)
Q2L-R101	ResNet-101	448×448	84.9	84.9
Q2L-R101	ResNet-101	576×576	86.5	86.5
Q2L-TresL	TResNetL	448×448	87.3	87.3
Q2L-Tres	TResNetL(22k)	448×448	89.2	89.2
Q2L-SwinL	Swin-L(22k)	384×384	90.5	90.5
Q2L-CvT	CvT-w24(22k)	384×384	91.3	91.3

Result

MLSPL

Table 2: Comparison of mean Average Precision (mAP) results between our implementation and the originally reported values for the MLSPL $\mathcal{L}_{ROLE}$ method on the MS-COCO 2014 dataset. Results are shown for both linear and fine-tuned variants. All mAP values are reported in percentage.

Loss	Method	mAP(Ours)	mAP (Paper)
$\mathcal{L}_{ROLE}$	Linear	66.3	66.3
$\mathcal{L}_{ROLE}$	Fine-Tuned	66.9	66.3

Discussion

- Q2L excels at localisation but is memory-hungry.
- MLSPL robust under weak labels, lightweight.
- Complementary strengths $\rightarrow$ potential hybrid model.

Limitations

- Q2L on ViTs exceeds 12 GB VRAM.
- MLSPL relies on accurate κ estimation.
- Experiments limited to COCO – needs broader validation.

Conclusions & Future Work

- Both methods reproduced successfully.
- Transformer label queries push SOTA accuracy.
- ROLE shows promise for cost-efficient annotation.
- Next steps: larger datasets, unify approaches.