Computer Vision

Multi-label Image Classification

Group 25
Riajul Islam, Andreas Calonius Kreth, Christine Midtgaard

***Abstract*—Abstract goes here.**

***Index Terms*—Multi-label learning, deep learning, computer vision, multi-label classification, deep learning for MLC.**

I. Introduction

Multi-label classification is the supervised learning problem where an instance may be associated with multiple labels. Image classification is a computer vision task that requires assigning a label or multiple labels to an image. Single-label classification, or multi-class classification, refers to the problem where an image contains only one object to be identified. However, natural images usually contain multiple objects or concepts, highlighting the importance of multi-label classification [1].

In this project we investigate methods aimed to solve two common issues that arise when training a network that assigns multiple labels to an input image: (i) the general multi-label learning issue accurately identifying multiple objects in an image under class imbalance, representing the complexity of real-world images, and (ii) the scenario where the training data underwent sparse supervision, which is often the case for data.

A. Problem Overview

a) Label Correlation:
b) Class Imbalance:
c) Sparse Supervision:

II. Related Work

Multi-label learning is a well studied problem within computer vision [2].

a) Loss Functions:

b) PU learning: Learning from positive and unlabeled data: a survey by Bekker and Davis [3].

Learning to Classify Texts Using Positive and Unlabeled Data by Li and Liu [4].

c) Partially Observed Labels:

d) Heuristics: Heuristics can be used to reduce the required annotation effort [34, 18], but this runs the risk of increasing error in the labels [2].

III. Background

This section describes relevant background theory related to the project and methods from MLSPL and Query2Label. Beginning with the definition of multi-label learning, followed by deep neural network architectures used to solve multi-label slassification tasks, and, finally, loss functions as they can handle challenges of label imbalance tasks.

A. Multi-Label Learning

Contrary to binary or multi-class classification where each instance is associated with only one label, multi-label classification allows assigning multiple labels to a single input [2]. Given K categories, an input image $x \in \mathcal{X}$ is associated with a binary vector of labels $y = [y_1, ..., y_K]$ from the label space $\mathcal{Y} = \{0, 1\}^K$, where $y_k = 1$ represents that k is present in x and $y_k = 0$ otherwise. The goal is to create a model that outputs the probability of the presence of a category $p = [p_1, ..., p_K]$ [2], [5].

B. Convolutional Neural Networks (CNNs)

In the field of computer vision, Convolutional Neural Networks (CNNs) have been a dominant approach for image analysis tasks since their introduction [6]. CNNs are designed to recognize patterns in images by applying local filters through convolutional layers, preserving the two dimensional input of an image [7]. CNNs consist of three main types of layers: convolutional layers, pooling layers, and a Fully Connected (FC) layer [8]. Convolutional layers extract spatial features through learnable filters, pooling layers reduce dimensionality and summarise information, while the fully connected layer at the end interpret the features to perform the classification.

C. Transformer Architectures

Transformers, introduced by Vaswani et al. [9], are a type of neural network architecture initially developed to model long-range dependencies in sequence data. They achieve this using attention and self-attention mechanisms, which enables the model to have a long-term memory of inputs, and dynamically weigh the importance of each element in the input sequence. Originally, transformer-based models was mainly delevoped for natural language processing tasks, but the introduction of the Vision Transformer (ViT) by Dosovitskiy et al. [10], transformers have been widely used for computer vision tasks.

a) Architecture: The original transformer consists of an encoder-decoder structure. Both encoder and decoder are made of a stack of identical layers, where encoder layers contain a multi-head self-attention mechanism followed by a position-wise feed-forward network. In addition, decoder layers consists of an encoder-decoder attention layer.

The transfomer encoder is of interest, as the authors of [5] use them to extract features and automatically learn label embedding, as described in section IV-B.

b) Attention: The core principle of the transformer architecture is the attention mechanism, which allows the model to attend to all posistions in an input sequence when processing each element. Thus alloowing the model to weigh the relevance of different posistions in a sequence. Given a query and a set of key-value pairs, the attention function computes a weighted sum of the values, where the weights are determined by similarities between the query and the keys.

c) Self-Attention: The transformer model uses self-attention by relating every element in the input sequence to every other element. The attention function is a function that maps a query and a set of key-value pairs to an output. The scaled dot-product self-attention is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

where Q, K, V, are the query, key, and value vectors, and d_k is the dimension of the key vectors and serves as a scaling factor [9].

d) Multi-Head Attention: The transformer applies multiple attention functions in parallel, allowing the model to attend to information from different parts of the sequence. The embedding is split into multiple heads, perform attention for each, and then concatenate them back together, defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O \tag{2}$$

where each head is computed as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \tag{3}$$

with projection matrices W_i^Q, W_i^K, W_i^V, and W_i^O.

e) Cross-Attention: Contrary to self-attention, where queries, keys, and values are generated from the same input, cross-attention is a mechanisn where queries come from one source, and the keys and values come fron another. This mechanism allows the model to relate elements from one input to relevant parts of another input. The authours of Query2Label [5] make use of this mechanism, where label embeddings (queries) attend to spatial image features (key-value pairs).

f) Feed-Forward Networks and Positional Encoding: Each layer in the Transformer also includes a fully connected feed-forward network applied to each position. To compensate for the lack of order in the input sequence, sinusoidal positional encodings are added to the input embeddings. These encodings allow the model to distinguish the order of elements in the sequence using functions of varying frequencies.

g) Vision Transformers (ViTs): The Vision Transformer (ViT), introduced by Dosovitskiy et al. [10], is a model for image classification that leverages the transformer architecture: the self-attention mechanism of transformers that allows the model to selectively weigh the significance of each part of the input, and positional embeddings that represent the order of tokens in a sequence. In ViTs, images are represented as sequences, where the label function as a learnable token for classification. The input image is divided into a sequence of patches that are flattened and linearly embedded into a vector. The spatial information is preserved by adding positional encodings to the embeddings. The resulting sequence is fed into a transformer decoder identical to that introduced in [9] to model global relations for classification.

D. Loss Functions

Loss functions are a fundamental component in deep learning as they serve as a measure of how much the model predictions deviate from the ground truth, guiding optimization of the network [7]. This section describes some of the common loss functions used in multi-label learning, beginning with the Binary Cross-Entropy (BCE) loss.

a) Binary Cross-Entropy (BCE): The BCE loss is a common choice for multi-label classification for a fully observed data point $(\mathbf{x}_n, \mathbf{y}_n)$ [2], as it's purpose is to classify data into one of two possible categories, negative or positive, represented as 0 and 1.

b) Focal Loss:

c) Assymetric Loss:

d) Online Estimation of Unobserved Labels (ROLE):

$$\mathcal{L}_{ROLE}(\mathbf{F}_B, \tilde{\mathbf{Y}}_B) = \frac{\mathcal{L}'(\mathbf{F}_B)|\tilde{\mathbf{Y}} + \mathcal{L}'(\tilde{\mathbf{Y}}|\mathbf{F}_B)}{2} \tag{4}$$

for a batch B, where $\tilde{\mathbf{Y}} \in [0,1]^{N\times L}$ represent the estimated labels, and $\mathbf{F} \in [0,1]^{N\times L}$ is the matrix of classifier predictions.

The goal is to train the label estimator $g(\cdot;\phi)$ and the image classifier $f(\cdot;\theta)$

IV. Method

In real-world datasets, obtaining full label annotations is practiaclly impossible. This project focuses on finding solutions to the multi-label learning problems:

- The challenge of multi-label learning in scenarios where only a single positive label per image is available durin training (MLSPL).
- Label imbalance (Query2Label).
- Feature localization (Query2Label).

A. Overview of Approach

A brief description of what we did to solve the multi-label classification problem.

B. Query2Label: A Simple Transformer Way to Multi-Label Classification

Query2Label: A Simple Transformer Way to Multi-Label Classification (Query2Label) by Liu et al. [5] is a two-stage framework for multi-label classification. It uses transformer decoders to extract features with multi-head attentions focusing on different parts of an object category and learn label embeddings from data automatically.

C. Multi-Label Learning from Single Positive Labels

Multi-Label Learning from Single Positive Labels (MLSPL) by Cole et al. [2]. How it uses weak supervision and contrastive learning.

V. Experiments

A description of our setup versus the author's setups.

All experiments were run on a single NVIDIA GeForce RTX 3060 GPU (12 GB VRAM), with Python 3.11.8, NVIDIA driver 550.90.07 and CUDA 12.4.

A. Dataset

The MS-COCO 2014 [11] dataset is used as a benchmark for evaluation both Query2Label and MLSPL. MS-COCO (Microsoft Common Objects in Context) is a large-scale dataset commonly used for object detection, segmentation, and multi-label image classification. COCO consists of 82,081 training images and 80 classes, and a validation set of 40,137 images.

B. Query2Label

All Query2Label experiments were conducted on environment versions cuda==12.4, torch==2.1.0+cu121, torchvision==0.16.0+cu121, python==3.11.8. We evaluated the Query2Label model using the best performing backbone model, the CvT-w24 backbone, with a 384 input resolution, pretrained on the ImageNet-22k dataset, as described by Liu et al. [5]. For our experiments, we used a pretrained model checkpoint released by the authors, which was trained on the MS-COCO 2024 dataset. The model was tested on the MS-COCO 2024 validation set with a batchsize of 16 and otherwise no fine-tuning or modification.

C. Multi-Label Learning from Single Positive Labels

All MLSPL experiments were conducted on environment versions cuda==12.4, torch==2.2.1+cu121, torchvision==0.17.1+cu121, python==3.11.8.

a) Data preparation: The dataset preparation for the MS-COCO dataset in MLSPL relies on converting the standard multi-label MS-COCO dataset to a single positive label format, simulating a weakly supervised setting. Cole et al. does this beginning with the fully annotated multi-label image dataset and corrupt it by discarding annotations, simulating single positive training data by randomly selecting one positive label for each training example [2].

To convert into a single positive label format, the instructions provided by Cole et al. are followed: firsly, both images and annotations for training and validation are downloaded. Secondly, pre-extracted features for COCO, provided by the authors are downloaded. Lastly, the script `format_coco.py` is used to produce uniformly formatted image lists and labels.

b) Hyperparameters:

D. Evaluation Metrics

To asses model performance, we adopt mean Average Precision (mAP), a standard metric widely reported in multi-label classification tasks as it is used to analyze the performance object detection and segmentation. Both Query2Label and MLSPL report results in terms of mAP.

VI. Results and Discussion

This section compares results from the experiments to the those of the papers. Discuss why they might not be the same, or describe the similarities. Discuss whether the methods are able to solve the problems.

A. Results from Query2Label

The results from our experiments are compared to those of the paper in table I.

B. Results from Multi-Label Learning from Single Positive Labels

VII. Conclusion

References

[1] T. Ridnik, G. Sharir, A. Ben-Cohen, E. Ben-Baruch, and A. Noy, "Ml-decoder: Scalable and versatile classification head," 2021. [Online]. Available: https://arxiv.org/abs/2111.12933

[2] E. Cole, O. M. Aodha, T. Lorieul, P. Perona, D. Morris, and N. Jojic, "Multi-label learning from single positive labels," 2021. [Online]. Available: https://arxiv.org/abs/2106.09708

[3] J. Bekker and J. Davis, "Learning from positive and unlabeled data: a survey," *Machine Learning*, vol. 109, no. 4, p. 719–760, Apr. 2020. [Online]. Available: http://dx.doi.org/10.1007/s10994-020-05877-5

[4] X. Li and B. Liu, "Learning to classify texts using positive and unlabeled data." 01 2003, pp. 587–594.

[5] S. Liu, L. Zhang, X. Yang, H. Su, and J. Zhu, "Query2label: A simple transformer way to multi-label classification," 2021. [Online]. Available: https://arxiv.org/abs/2107.10834

[6] Y. Lecun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, P. Simard, and V. Vapnik, "Comparison of learning algorithms for handwritten digit recognition," 01 1995.

[7] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive into Deep Learning*. Cambridge University Press, 2023. [Online]. Available: https://d2l.ai

[8] X. L. C. Asawa, *CS231n: Convolutional Neural Networks for Visual Recognition*. Stanford University, 2023. [Online]. Available: http://cs231n.github.io/

[9] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021. [Online]. Available: https://arxiv.org/abs/2010.11929

[11] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2015. [Online]. Available: https://arxiv.org/abs/1405.0312

TABLE I
COMPARISON OF MAP RESULTS RETWEEN OUR EXPERIMENTS AND REPORTED MAP RESULTS ON THE MS-COCO 2014 DATASET.

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