Fine-Grained Classification for Poisonous Fungi Identification with Transfer Learning

Notebook for the LifeCLEF Lab at CLEF 2024

Christopher Chiu[1,*,†], Maximilian Heil[1,*,†], Teresa Kim[1,*,†] and Anthony Miyaguchi[1,*,†]

[1]*Georgia Institute of Technology, North Ave NW, Atlanta, GA 30332, USA*

Abstract

FungiCLEF 2024 addresses the fine-grained visual categorization (FGVC) of fungi species, with a focus on identifying poisonous species. This task is challenging due to the size and class imbalance of the dataset, subtle inter-class variations, and significant intra-class variability amongst samples. In this paper, we document our approach in tackling this challenge through the use of ensemble classifier heads on pre-computed image embeddings. Our team (DS@GT) demonstrate that state-of-the-art self-supervised vision models can be utilized as robust feature extractors for downstream application of computer vision tasks without the need for task-specific fine-tuning on the vision backbone. Our approach achieved the best Track 3 score (0.345), accuracy (78.4%) and macro-F1 (0.577) on the private test set in post competition evaluation. Our code is available at https://github.com/dsgt-kaggle-clef/fungiclef-2024.

Keywords

Fine-Grained Visual Categorization (FGVC), Poisonous Fungi Identification, Transfer Learning, Vision Transformers, CEUR-WS

1. Introduction

Classifying a fungi's species and toxicity accurately and efficiently goes beyond simple image classification; it requires discerning subtle differences between species within the intricate context of fine-grained visual categorization (FGVC). FGVC is more challenging than regular image classification tasks due to small inter-class variations - high similarities between genetically related fungi, and high intra-class variances at the same time as the image observation depends on several factors, such as genotype, age, time of year, and local conditions. This paper is part of FungiCLEF 2024 [1] competition, part of the LifeCLEF [2] lab series.

Figure 1: Sample images from the dataset. The images of fungi have high degree of variability across lighting, substrate, focus, subject, and other image features. This poses additional challenge in effectively training a classifier model.
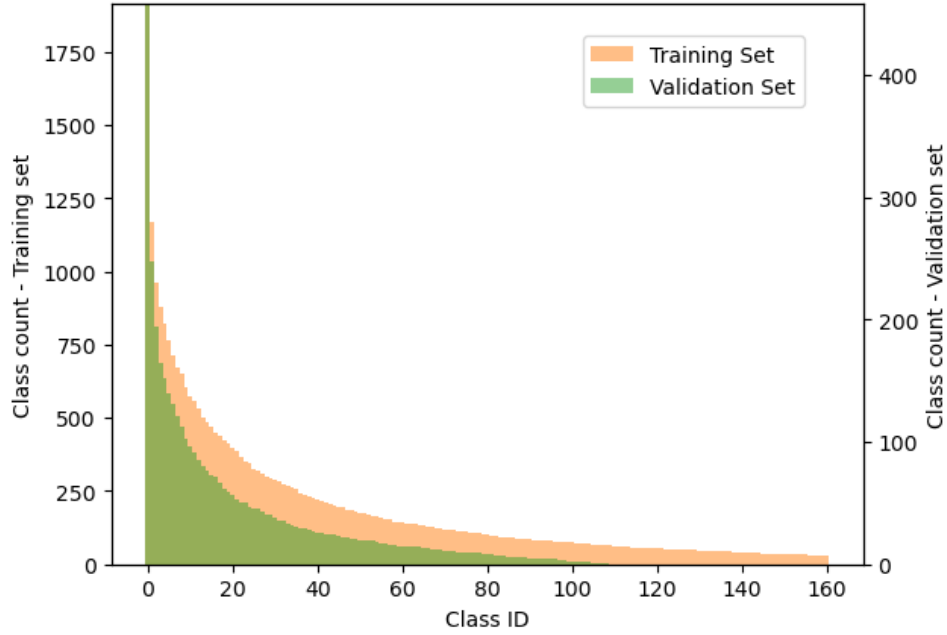

Figure 2: Distribution of classes in DF20 (training set) and DF21 (validation set). We observe that the two datasets have different class distributions. Both datasets have significant class imbalance, as signified by the long tail in classes with low counts.

1.1. Dataset Overview

The featured dataset for the FungiCLEF competition [1] is the Danish Fungi dataset [3]. This dataset includes a training set (DF20), which includes 356,770 images over 1,604 different classes of fungi, and a validation / testing dataset (DF21), consisting of 60,832 images over 2,713 species of fungi, covering a year's worth of observation. For species within the validation dataset that were not in the training dataset, they were marked as an "unknown" class. The dataset provides both full sized images (110GB) and downsized images (300px max dimension, 5.6GB). It also provides metadata for the fungi images including date, location, substrate and metasubstrate of the fungi growth, and the full taxonomical ranks of the classified fungi species, including phylum, class, order, family, and genus.

These two datasets do not have the same distribution of classes (Figure 2). Moreover, there was significant class imbalance in both datasets, with the most common class having 1,913 images, and the least common class only ~30 images in DF20, and down to only one observation for some species in DF21. There were also significant variations in terms of lighting, background, and clarity, due to the real-world conditions under which fungi were photographed (Figure 1). This adds another level of complexity on this task - Fungi classes are not only hard to distinguish due to inter-class variations or intra-class variance, but also due to varying image quality and image features. This highlights the need for a robust model in effectively performing fine-grained classification on this rich and varied dataset.

1.2. Related Work

State-of-the-art work on this dataset primarily utilizes models such as Swin Transformer [4] and MetaFormer [5]. However, results from FungiCLEF 2023 [6] underscore limitations in current research, where the best accuracy from participants have not improved significantly since the competition's inception in 2022 [7]. Last year's winner incorporated metadata into the model with MetaFormer as the vision model [8], and utilized Seesaw Loss [9] to handle class imbalance. This led to a macro F1 of 0.571, with a poisonous and edible species confusion rate of 5.31% and 2.05% respectively [8]. To handle unknown classes, the team also introduced an entropy based approach to identify unknown, out-of-distribution species [10].

Beyond FungiCLEF, Wei et al. [11] provides a comprehensive examination of fine-grained visual categorization (FGVC) challenges, such as accurately localizing object parts, selecting informative features under varied conditions, and integrating segmentation with classification. It emphasizes the need for the model to generalize across species, maintain efficiency, and handle real-world issues like occlusions. Other directions that demonstrated promise on FGVC datasets such as CUB-200-2011 [12] include Mask-CNN [13] which outperformed other methods by better capturing subtle differences between species, and SR-GNNs [14] which extracted context-aware features from relevant image regions to discriminate between object classes.

2. Methodology

Our overall approach to this challenge of fine-grained classifying of fungi species was to:

1. Incorporate metadata as additional input / prediction targets for the model.
2. Learn the concept of unknown classes by incorporating the validation dataset into training.
3. Experiment with objective functions to induce model capability in fine-grained classification task.
4. Train only on metadata and image embeddings for rapid prototyping and model optimization.

Cloud computing resources were supplied by *Data Science @ Georgia Tech*. Data was hosted on Google Cloud Storage, and models developed on virtual instances with NVIDIA L4. For more memory intensive experiments, GPUs used in model development include NVIDIA RTX 4090 and a distributed cluster with 2x NVIDIA V100.

Libraries used include pandas [15], PaCMAP [16], scikit-learn [17] for data exploration; PySpark [18], PyArrow [19], Luigi [20] for data processing; PyTorch [21], timm [22], Lightning [23], and transformers [24] for model development. Evaluation functions for the FungiCLEF competition were referenced in the development of internal model benchmarks [25].
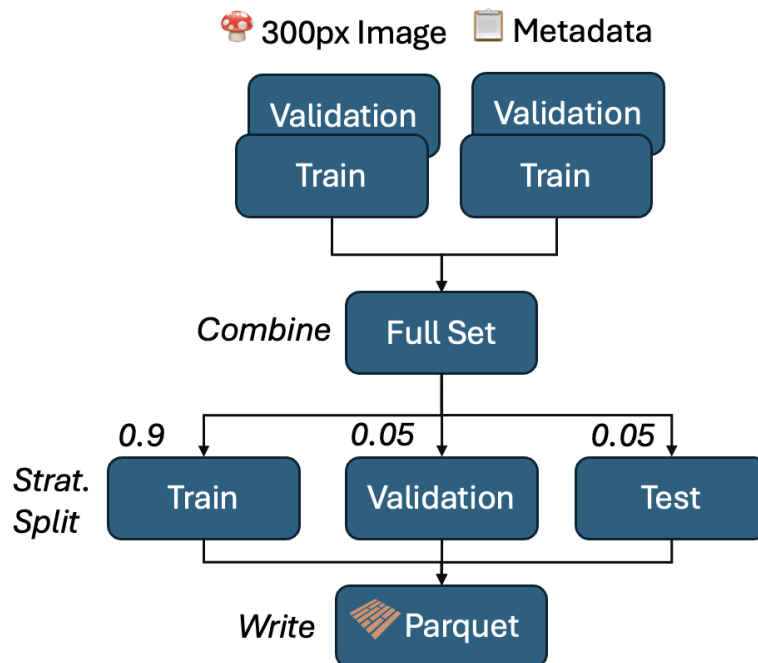

Figure 3: Dataset preparation pipeline: In order to include the unknown classes of fungi from the validation dataset into model training, we mixed the training and validation datasets into a full dataset for further dataset splitting.

2.1. Dataset Preparation

To improve the efficiency of experiments, we built a data preprocessing pipeline with PySpark (Figure 3). We appended the 300 pixel and full versions of image data with associated metadata, and stored them as parquet files for faster I/O. Embeddings were also precomputed and stored as parquet files separately. A custom PyTorch dataset object was created to serve the image and embedding data alongside metadata.

The metadata columns were grouped based on their potential use as either model inputs or prediction targets. For the validation set / public test set, only substrate, metasubstrate, habitat, date, and location were provided [25]. As such, these columns were used as additional inputs to the model. Categorical columns were expanded into one-hot vectors. For date information, we converted the month and day into cyclical encoding using sine / cosine transformation [26]. For location data, we converted longitude and latitude into Geohash, which preserves spatial ordinality and unifying location inputs using a Z-order curve [27]. Levels 2-5 of the resultant Geohash were extracted and converted from base-32 to normalized base-10 integers. The toxicity and one-hot vectors of the taxonomical levels of fungi classes were included as additional prediction targets. Other metadata columns were excluded.

Given that unknown classes were only present in the validation dataset, we divided DF21 into three equal sections of 20,000 cases, stratified by species. One of the sections was designated as the held out test set, with the remaining two sections utilized as validation set / addition to the training set in a two-fold cross validation training. By percentage, this gives us a ratio of 90.4%, 4.8%, 4.8% for training, validation, and testing over the entire dataset. Significantly, the test and validation sets in each validation fold have the same class distribution.

2.2. Embeddings for Transfer Learning

Embeddings are the learned intermediate representation of deep learning models that capture structure about the input domain. We experimented with two models as the vision model backbone to generate embeddings - DINOv2 [28] and ResNet [29]. DINOv2 was chosen as it is state-of-the-art in terms of vision model and its richness and robustness as a visual feature extractor [28]. ResNet was chosen due to its widespread application and downstream [29], and serves as a representative for the CNN family in contrast to the transformer family where DINOv2 originated from. For ResNet18, we generated embeddings by extracting the output features from the last hidden state before the classification head. This resulted in embeddings of shape (1000,) per image. For DINOv2, we utilized the [CLS] token from the last hidden state of model output. In initial experiments and ablation studies, *dinov2-small* [28] was used, which results in embedding shape of (768,). In our optimized model used for competition submission, *dinov2-large* with register [28] was used, which had embedding shape of (1024,).

For training, the image embeddings were precomputed. For the testing set and for our competition submission, the vision backbone model was frozen, and embeddings were generated during inference and fed into our trained classifier heads.

2.3. Model Development

We explored two separate approaches in model development (1) Training a computer vision model from end-to-end, and (2) Training a classifier head only on precomputed embeddings. While approach (1) is the more traditional method for computer vision tasks, it is much more compute intensive due to the number of parameters to be trained [30]. In comparison, approach (2) had significantly less memory requirements and faster training time (Table 1). While using precomputed image embeddings imply that training data could not undergo traditional augmentation techniques in computer vision such as flipping and random cropping, we hypothesize that modern vision models had sufficient amount of information in the feature representation that the downstream model can be robust and generalisable.

Table 1
Memory and compute requirements for a fine-tuned vision backbone model (represented by MetaFormer) v.s. transfer learning model (our approach). Overall, the transfer learning model had less memory overhead and faster training time.

	Trainable Parameters	Batch Size	Training Time per Epoch (s)	VRAM Requirement
Vision Backbone Model (MetaFormer)	69M	64	2580	60GB
Transfer Learning Model (DinoV2)	47M	512	67	8GB

2.3.1. Model Training

For transfer learning with the embedding model, we use a traditional MLP classifier head with a hidden dimension of 4096, with metadata directly concatenated to the embedding. Inspired by Diao et al. [31], we also experiment with using a transformer block for better integration of metadata into the classifier. We transformed metadata into the same dimensions as the embedding with a separate MLP layer, and added them to the image embeddings before streaming all the data into a transformer block for image classification. To leverage the benefits of cross fold validation, we utilized an ensemble model approach [32]. Output logits of our model are averaged over all the classifier heads.

All models were first trained on a smaller, exploratory development set, before undergoing training runs on the full dataset. For experiments that appeared promising in its initial outcomes, training parameters were further tuned using Optuna to generate a full model for benchmark. Training performance were logged on Weights & Biases, with the top 2 performing models saved as checkpoints. A two-fold cross-validation was used, where each fold had 1/3rd of DF21 dataset as the validation set, and another 1/3rd incorporated into the training data. Our experiment logs can be viewed at https://wandb.ai/chiu/FungiClef.

All experiments were trained 20 to 50 epochs each, with batch sizes of 64 to 512. Initial learning rates ranged from $1 \cdot 10^{-5}$ to $1 \cdot 10^{-3}$, with AdamW [33] as optimizer. Learning rate schedulers experimented with include cosine scheduler with restarts [34], and ReduceLROnPlateau [35].

Metrics recorded during training include training / validation loss, top-1, top-3 accuracy, macro F1 score, and accuracy for correct identification of poisonous species. Calculation for specific track scores were adapted from the FungiCLEF competition [1] for model benchmark. This includes classification error (Track 1), cost for poisonousness confusion (Track 2), and user specific cost (Track 3) [25].

2.3.2. Loss Function

The baseline loss function for model development was unweighted, multi-class cross entropy loss. We also explored incorporating class weights in cross-entropy loss, and other loss functions such as focal loss [36] and seesaw loss [9], which was used by last year's winner [8] to overcome class imbalance. Additionally, we experimented with using various metadata such as the higher level taxonomy of the fungi class and the toxicity of the fungi class as additional prediction targets.

For our benchmark model, the model was trained with a custom loss function:

$$L_{\text{composite}} = L_{\text{seesaw}} + \alpha \cdot L_{\text{poison}}$$

Where L_{seesaw} is the seesaw loss of the class prediction, L_{poison} is the binary cross entropy loss of the model's prediction in whether the fungi is poisonous, and α an adjustable weighting factor for the composite loss function.

2.3.3. Weighted Sampling

While weighted sampler is usually utilized to overcome class imbalance [37], we utilized this technique in our data loader to ameliorate the difference in class distribution between the training and validation

set. Instead of adjusting class weights such that each class is evenly represented, we derived the per-sample weight by dividing the class frequency of the validation set over the training set:

$$W_{sampling} = \frac{\mathcal{D}_{training}}{\mathcal{D}_{validation}}$$

3. Results

3.1. Training Results

Our best performing model was an ensemble model on DINOv2 embeddings consisting of two classifier heads (180MB each) from the two-folds of cross-validation training. The model was trained on image embeddings precomputed from DINOv2-large. The weighting for poison loss α was 0.1. The initial learning rate was $1 \cdot 10^{-4}$, with AdamW [33] as optimizer, and cosine learning rate scheduler with warm restarts [34].

Table 2
Performance of the top 3 models from each training fold and ensemble model across various FungiCLEF metrics.

Fold	Acc.	Track 1	Track 2	Track 3	F1
1	74.9%	0.251	0.414	0.665	0.409
1	74.2%	0.258	0.418	0.675	0.406
1	74.1%	0.259	0.430	0.689	0.378
2	73.0%	0.270	0.287	**0.557**	0.374
2	72.2%	0.278	**0.282**	0.560	0.405
2	75.0%	0.250	0.322	0.571	0.391
Ensemble	**78.0%**	**0.221**	0.385	0.606	**0.451**

Table 3
Ablation study of DINOv2 embedding classifier with various configurations of class weighting and incorporation of metadata as input or additional prediction targets.

	Val. Acc. (%)	Difference (%)
DINOv2 baseline	49.3	-
w/ class weighting	65.5	+16.2
w/ metadata	69.1	+3.6
incl. toxicity	68.9	-0.1
incl. taxonomy	69.5	+0.5
w/ weighted taxonomy	69.7	+0.2

Table 2 outlined the three top performing models from each fold. Overall, Fold 1 had worse performance on the held out test set than Fold 2. The ensemble model with the best performing model from each fold yielded better accuracy than either folds, but suffered partially with poisonous performance. We also experimented with incorporating more models in the ensemble model, but the model performance was worse, likely due to the model negatively biased by inferior models.

To explore the impact of various factors in model performance, we performed an ablation study with results summarized in Table 3. The ablation experiments were done using smaller embeddings and a MLP head instead of a transformer head. We find that class-weighting increased the scores by the largest margin, followed by the inclusion of metadata.

3.2. Leaderboard Results

The results of our team's experiments are outlined in Table 4. For our first submission during competition, we used a pre-trained MetaFormer model from the previous year's competition as a baseline. In post-competition evaluation, our best model achieved an accuracy of 78.4% and a macro F1 score of 0.577 in

the private test set. Our model's performance was comparable to previous years' winners [8], and was the best performing model in this year's competition in terms of Track 1, Track 3, and accuracy. Our Track 2 and F1 score was ranked 2nd compared to the rest of the competitors [1]. The inference time across the full public test set (40,216 images) was 25:26 minutes, and 0.126s per image on average on a RTX 4090.

Table 4
Public and private test set scores on the official leaderboard and post-competition evaluation.[2]

	Name	Track 1 ↓	Track 2 ↓	Track 3 ↓	F1 ↑	Acc. ↑
Private	MetaFormer (Competition)	0.391	1.604	2.044	30.0	60.9
Private	DINOv2 (Post Competition)	0.216	0.129	0.345	57.7	78.4
Public	MetaFormer (Competition)	0.395	1.649	2.044	27.6	60.5
Public	DINOv2 (Post Competition)	**0.211**	0.165	0.375	49.8	**79.0**
Public	Rank 1 - IES	0.2922	**0.0699**	**0.3621**	**54.99**	70.78
Public	Rank 2 - jack-etheredge	0.2394	0.1681	0.4075	49.81	76.06
Public	Rank 6 - Baseline with EfficientNet-B1	0.4926	0.6599	1.1526	32.99	50.74

4. Discussion

We intially experimented with vision models including EfficientNet [38], VisionTransformer [39], and MetaFormer [5]. Due to training time and memory overhead, we opted to focus our efforts into developing a lightweight classifier on precomputed embeddings instead.

4.1. ResNet v.s. DINOv2 as Vision Backbone for Embedding Generation

Overall, while DINOv2 embeddings proved to be good input for image classification, our embedding model using ResNet embeddings did not perform well, with best validation accuracy at 25%. This was likely due to DINOv2 being a class agnostic, self-supervised model, whereas ResNet was trained on ImageNet with specific classification targets. As such, the features extracted from ResNet would be more tailored to the dataset, whereas DINOv2 features were more representative of the underlying image [28]. To further investigate this, we visualized the embeddings with UMAP [40] in Figure 4, which showed that ResNET embeddings did not separate well, whereas there was a clear separation in DINOv2 embeddings.

4.2. Incorporation of Metadata

We experimented with using metadata as additional prediction targets as seen in our ablation in Table 3. However, this did not yield additional performance, but the additional overhead required to tune the weighting of various targets was not worth the complexity. As such, we did not utilize metadata in our final model. The inclusion of metadata appeared to provide some marginal benefits in validation accuracy and F1 score. This echoes the finding from previous research on the dataset, where the incorporation of metadata as input had a positive contribution to model performance.

---

[1]Due to numerous issues with the HuggingFace platform, our best results were not recorded in the official competition. Our post competition evaluation was performed under the same constraints as the official competition. Post-competition results were provided and verified by the organiser of FungiCLEF.

[2]Official competition results are from test submissions with an under-tuned vision model. These results are included for completeness.
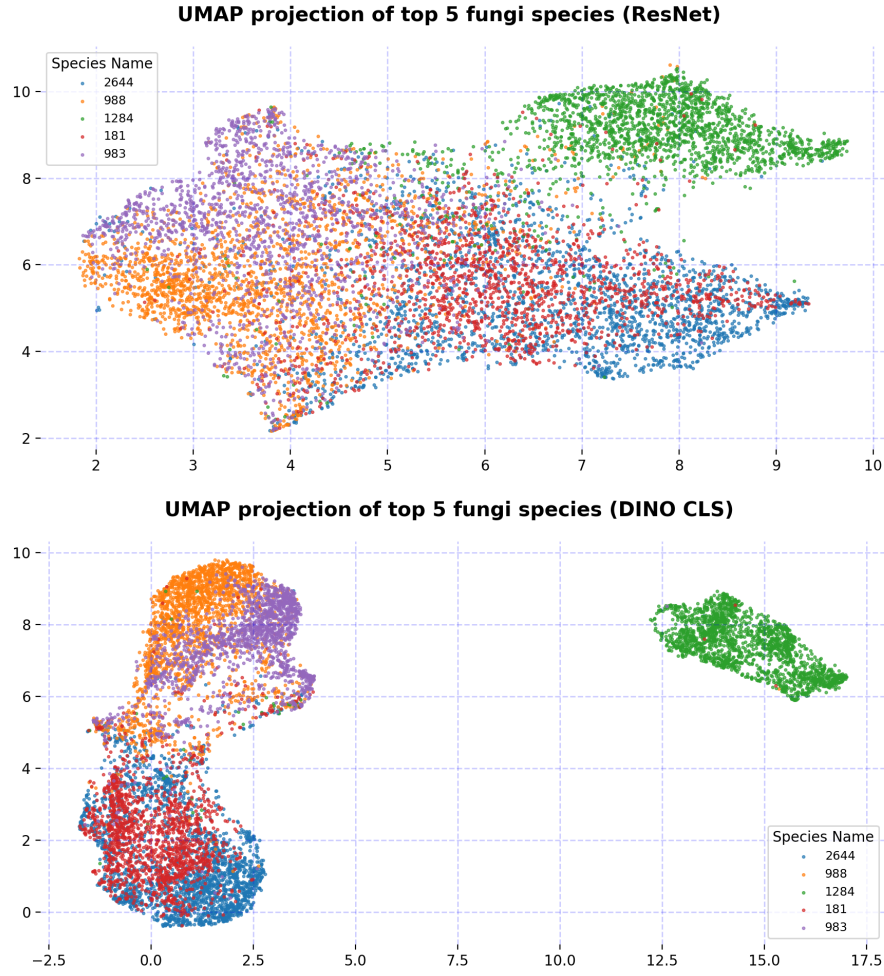
Figure 4: Clustering of top 5 fungi species on ResNet and DINOv2 with UMAP. We observe that ResNet embeddings do not separate well, but there were clear separation of clusters in DINOv2.

5. Future Work

Whilst using embeddings allowed for much faster model development time, there is still an additional gap in the performance of the embedding classifier compared to traditional image-based models. It is likely that the information loss in the transformation of image to embeddings was too significant for the simple classifier architecture to overcome. It would be interesting to further fine-tune DINOv2 on the DanishFungi dataset, and repeat our experiments. Moreover, a more rigorous incorporation of metadata into our models could provide a more holistic understanding of the data, leading to more accurate and reliable classification systems.

6. Conclusion

In summary, we addressed the complex task of fine-grained visual categorization (FGVC) for identifying poisonous fungi using transfer learning and advanced deep learning methodologies. The Danish Fungi 2020 dataset presented significant challenges such as class imbalance, subtle inter-class variations, and high intra-class variability, necessitating a comprehensive data preprocessing and augmentation pipeline.

Our experiments with various deep learning models, including vision transformers, convolutional neural networks, and linear classifiers with embeddings, highlighted the potential of DINOv2 embed-

dings combined with a multi-layer perceptron. Integrating multimodal metadata further enhanced classification performance, emphasizing the value of auxiliary information. Despite promising results, embedding-based classifiers faced limitations due to potential information loss, suggesting the need for fine-tuning self-supervised models on domain-specific datasets and improved metadata incorporation. Overall, our research advances FGVC technical capabilities, providing valuable methodologies for mycological safety and educational applications, and contributes to the broader field of fine-grained classification tasks.

Acknowledgements

References

[1] L. Picek, M. Sulc, J. Matas, Overview of FungiCLEF 2024: Revisiting fungi species recognition beyond 0-1 cost, in: Working Notes of CLEF 2024 - Conference and Labs of the Evaluation Forum, 2024.

[2] A. Joly, L. Picek, S. Kahl, H. Goëau, V. Espitalier, C. Botella, B. Deneu, D. Marcos, J. Estopinan, C. Leblanc, T. Larcher, M. Šulc, M. Hrúz, M. Servajean, et al., Overview of lifeclef 2024: Challenges on species distribution prediction and identification, in: International Conference of the Cross-Language Evaluation Forum for European Languages, Springer, 2024.

[3] L. Picek, M. Šulc, J. Matas, T. S. Jeppesen, J. Heilmann-Clausen, T. Læssøe, T. Frøslev, Danish fungi 2020 - not just another image recognition dataset, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2022, pp. 1525–1535.

[4] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: Hierarchical vision transformer using shifted windows (2021). arXiv:2103.14030.

[5] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, S. Yan, Metaformer is actually what you need for vision (2021). arXiv:2111.11418.

[6] L. Picek, M. Šulc, R. Chamidullin, J. Matas, Overview of fungiclef 2023: Fungi recognition beyond 1/0 cost, Proceedings of the Working Notes of CLEF 2023 - Conference and Labs of the Evaluation Forum (CLEF) (2023).

[7] J. Heilmann-Clausen, M. Sulc, L. Picek, Overview of fungiclef 2022: Fungi recognition as an open set classification problem, in: CLEF 2022 Conference and Labs of the Evaluation Forum, volume 3180, 2022, pp. 1–10.

[8] H. Ren, H. Jiang, W. Luo, M. Meng, T. Zhang, Entropy-guided open-set fine-grained fungi recognition, Proceedings of the Working Notes of CLEF 2023 - Conference and Labs of the Evaluation Forum (CLEF) (2023).

[9] J. Wang, W. Zhang, Y. Zang, Y. Cao, J. Pang, T. Gong, K. Chen, Z. Liu, C. C. Loy, D. Lin, Seesaw loss for long-tailed instance segmentation (2020). arXiv:2008.10032.

[10] D. Macêdo, T. I. Ren, C. Zanchettin, A. L. I. Oliveira, T. Ludermir, Entropic out-of-distribution detection: Seamless detection of unknown examples (2020). arXiv:2006.04005.

[11] X.-S. Wei, C.-W. Xie, J. Wu, Mask-cnn: Localizing parts and selecting descriptors for fine-grained bird species categorization (2016). arXiv:1605.06878.

[12] C. Wah, S. Branson, P. Welinder, P. Perona, S. Belongie, The caltech-ucsd birds-200-2011 dataset, 2011.

[13] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask r-cnn (2017). arXiv:1703.06870.

[14] A. Bera, Z. Wharton, Y. Liu, N. Bessis, A. Behera, Sr-gnn: Spatial relation-aware graph neural network for fine-grained image categorization (2022). arXiv:2209.02109v1.

[15] W. McKinney, Data structures for statistical computing in python, in: 9th Python in Science Conference (SciPy 2010), 2010, pp. 51–56.

[16] Y. Wang, H. Huang, C. Rudin, Y. Shaposhnik, Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization, 2021.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, Édouard Duchesnay, Scikit-learn: Machine learning in python, 2011.

[18] A. S. Developers, Pyspark: Python api for apache spark, 2024. URL: https://spark.apache.org/docs/latest/api/python/index.html.

[19] W. McKinney, Pyarrow: Python api for apache arrow, 2024. URL: https://arrow.apache.org/docs/latest/api/python/index.html.

[20] E. Bernhardsson, E. Freider, Luigi: A python package for building complex pipelines of batch jobs, 2024. URL: https://luigi.readthedocs.io.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library (2019). arXiv:1912.01703.

[22] R. Wightman, timm: Pytorch image models, 2019. URL: https://github.com/rwightman/pytorch-image-models.

[23] W. Falcon, T. P. L. team, Pytorch lightning, 2024. URL: https://lightning.ai/docs/pytorch/stable/.

[24] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, A. M. Rush, Transformers: State-of-the-art natural language processing, in: 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 38–45.

[25] L. Picek, M. Šulc, J. Matas, J. Heilmann-Clausen, T. S. Jeppesen, T. Læssøe, T. Frøslev, Danish fungi 2020 - not just another image recognition dataset, 2021. arXiv:2103.10107.

[26] I. London, Encoding cyclical continuous features — 24-hour time, 2016. URL: https://ianlondon.github.io/blog/encoding-cyclical-features-24hour-time/.

[27] I. S. Suwardi, D. Dharma, D. P. Satya, D. P. Lestari, Geohash index based spatial data model for corporate, in: 2015 International Conference on Electrical Engineering and Informatics (ICEEI), 2015, pp. 478–483. doi:10.1109/ICEEI.2015.7352548.

[28] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, P. Bojanowski, Dinov2: Learning robust visual features without supervision, 2024. arXiv:2304.07193.

[29] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition (2015). arXiv:1512.03385.

[30] Z.-Y. Dou, Y. Xu, Z. Gan, J. Wang, S. Wang, L. Wang, C. Zhu, P. Zhang, L. Yuan, N. Peng, Z. Liu, M. Zeng, An empirical study of training end-to-end vision-and-language transformers (2021). arXiv:2111.02387.

[31] Q. Diao, Y. Jiang, B. Wen, J. Sun, Z. Yuan, Metaformer: A unified meta framework for fine-grained recognition (2022). arXiv:2203.02751.

[32] M. A. Ganaie, M. Hu, A. K. Malik, M. Tanveer, P. N. Suganthan, Ensemble deep learning: A review (2021). arXiv:2104.02395.

[33] I. Loshchilov, F. Hutter, Decoupled weight decay regularization (2017). arXiv:1711.05101.

[34] I. Loshchilov, F. Hutter, Sgdr: Stochastic gradient descent with warm restarts (2017). arXiv:1608.03983.

[35] A. Al-Kababji, F. Bensaali, S. P. Dakua, Scheduling techniques for liver segmentation: Reducelronplateau vs onecyclelr (2022). arXiv:2202.06373.

[36] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection (2017). arXiv:1708.02002.

[37] Y. Zhang, B. Dong, J. Wu, Q. Zheng, A new weighted sampling method to handle class imbalance

problem, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 875–884.

[38] M. Tan, Q. V. Le, Efficientnetv2: Smaller models and faster training, CoRR abs/2104.00298 (2021). URL: https://arxiv.org/abs/2104.00298. `arXiv:2104.00298`.

[39] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale, CoRR abs/2010.11929 (2020). URL: https://arxiv.org/abs/2010.11929. `arXiv:2010.11929`.

[40] L. McInnes, J. Healy, J. Melville, Umap: Uniform manifold approximation and projection for dimension reduction (2018). `arXiv:1802.03426`.