Object Recognition Deliverable 1: CNN Architecture Design

Agúndez, Pedro
pedro.agundez@estudiantat.upc.edu

Lang, Sheena
sheena.maria.lang@estudiantat.upc.edu

Parent, Zachary
zachary.parent@estudiantat.upc.edu

Recalde, Martí
marti.recalde@estudiantat.upc.edu

Sánchez, Bruno
bruno.sanchez@estudiantat.upc.edu

March 10, 2025

1 Introduction

This project aims to explore how to design and train Convolutional Neural Networks (CNNs) to maximize performance. In particular, we will try to build and optimize a system for tackling multi-label classification on the Pascal VOC 2012 dataset. This task consists of, given an image and a finite set of object classes, stating for each class if an object belonging to it appears in the image. In order to further explain how we built our solution to this, in the following sections we will: provide with an overview of the task, the dataset and its structure in section 2, followed by a detailed analysis of the experiments designed for the architecture design exploration in section 3 and by their results and the corresponding analysis in section 4. Finally, we will expose our conclusions in section 5.

2 Dataset

2.1 Analysis

The VOC dataset consists of various object labels distributed across numerous images. A detailed examination of the dataset reveals that the labels are not uniformly distributed. Some labels appear significantly more frequently than others, which could impact the model's ability to learn and generalize across different classes. Additionally, the area ratio of objects within images varies widely across labels, and the co-occurrence of labels also shows significant variation. These factors may contribute to the difficulty in detecting certain labels over others.

2.2 Analysis Methods

To conduct the dataset analysis, we employed several quantitative methods. We counted occurrences of each label and the count of images including each label to understand its prevalence within the dataset. The area ratio of objects was computed to assess the relative size of labeled objects within images. Additionally, we analyzed label co-occurrence by examining how often different labels appeared together in the same image. These methods provided a comprehensive overview of the dataset's characteristics, enabling us to identify potential challenges in model training.

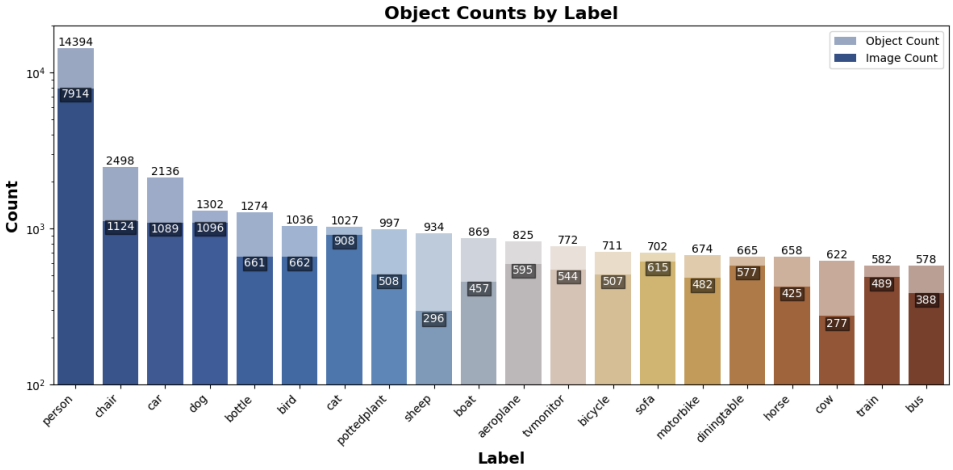

Figure 1: Object counts by label

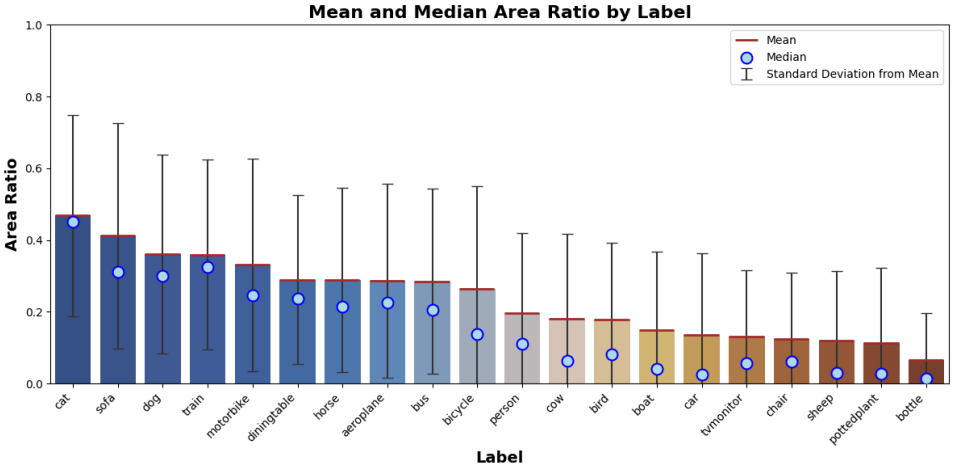

Figure 2: Object area ratio by label

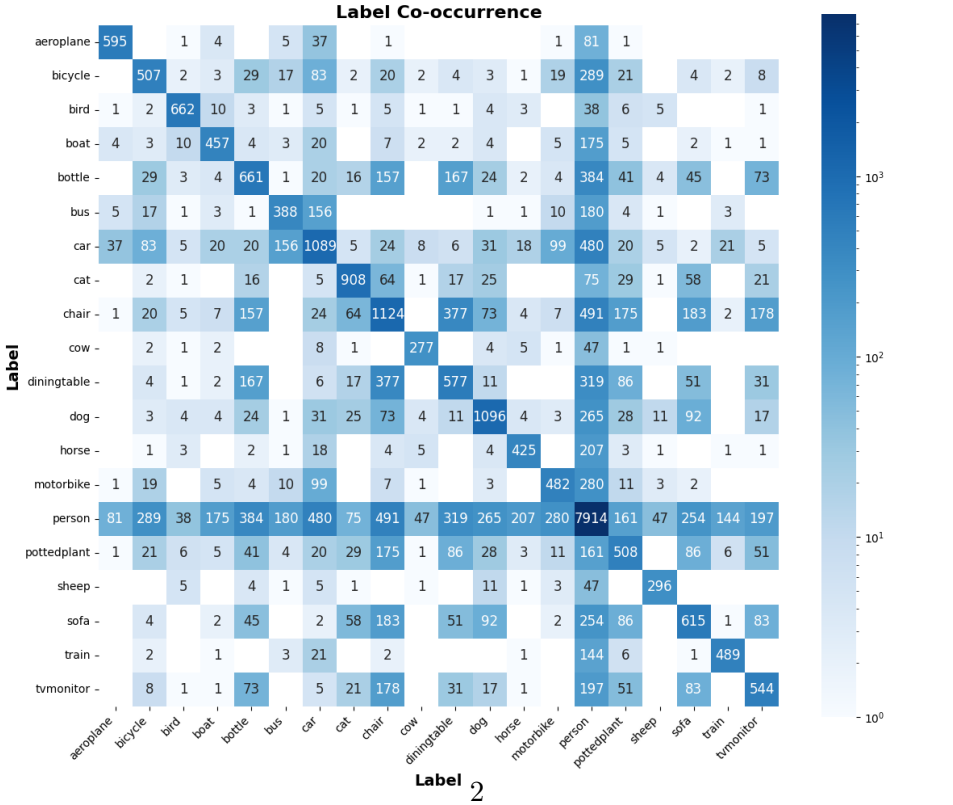

Figure 3: Label Co-occurrence

2.3 Hypotheses

Based on the initial analysis, the following hypotheses have been formulated regarding the model's learning capabilities for different labels:

1. **Person:** The 'person' label is expected to be well-learned due to its prevalence as the sole label in many images. However, it may also result in a high number of false positives, given its frequent co-occurrence with other labels.
2. **Cat and Dog:** These labels are anticipated to be well-learned because they typically have a high area ratio and relatively low co-occurrence with other labels.
3. **Aeroplane:** The 'aeroplane' label is expected to be well-learned due to its low co-occurrence with other labels.
4. **Diningtable:** This label may be challenging to learn, as very few images contain only a 'diningtable' label.
5. **Indoor Furniture Labels:** Labels such as 'chair', 'diningtable', 'pottedplant', 'sofa', and 'tvmonitor' are expected to be well-learned. However, there may be some confusion between them due to their tendency to co-occur in images.
6. **Sheep and Cow:** These labels are anticipated to be well-learned, as images containing these labels often have multiple objects of the same label.

2.4 Summary

The analysis indicates that the dataset's label distribution and co-occurrence patterns could significantly influence the model's performance. Labels that are more common or have a higher area ratio are likely to be learned more effectively. However, labels with high co-occurrence may lead to increased false positives, while those with low prevalence may be challenging to detect. These insights will guide the development of targeted strategies to improve model accuracy and robustness across all labels.

3 Experiment Design

We designed our experiments to address the following research questions:

- **RQ1:** Among the three networks (ResNet-50, Inception V3, and MobileNet V2), which one achieves the best performance? How does its performance compare with and without pre-training, and does warm-up training provide any benefits?
- **RQ2:** What impact do the learning rate and batch size have on model performance?
- **RQ3:** How do two additional data augmentation techniques impact model performance?
- **RQ4:** How can we develop two techniques to enhance label balance during training while maintaining accuracy?
- **RQ5:** What modifications to the classifier head can lead to improved performance?

To systematically structure our experiments, we divided them into four distinct categories: **Network experiments** (3.1), addressing RQ1; **Hyperparameter experiments** (3.2), answering RQ2; **Augmentation experiments** (3.3), covering RQ3; **Imbalance handling experiments** (3.4), focusing on RQ4; and **Classifier head experiments** (3.5), providing insights into RQ5.

It is important to note that the different experiment categories are performed sequentially, as the results of a prior experiment influence the subsequent ones. The **Hyperparameter experiments** are conducted using the best model identified in the **Network experiments**, while both the **Augmentation experiments** and **Imbalance handling experiments** and the **Classifier head experiments** are carried out with the best model selected from the **Hyperparameter experiments**.

To determine the 'best' model, we selected a primary evaluation metric. After reviewing several recent and widely recognized image classification tasks performed on the Pascal VOC 2012 dataset, we found that mean Average Precision (mAP) is a commonly used metric [1][2][3]. mAP measures the mean precision across different recall levels, providing a comprehensive assessment of a model's ability to correctly rank relevant predictions while minimizing false positives. This makes it particularly suitable for multi-label image classification, where multiple objects may be present in a single image, requiring a metric that evaluates both precision and recall effectively.

3.1 Model Experiments

To address **RQ1**, we conducted nine different experimental runs. For each of the three models (ResNet-50, Inception V3, and MobileNet V2) we performed three configurations: training from scratch, fine-tuning a pretrained model without warmup, and fine-tuning a pretrained model with warmup.

For the warmup strategy, we initialized training with three warmup epochs, as we considered this a reasonable middle ground based on the varying depths of the three models. Since the number of warmup epochs should be proportional to model depth, this choice aimed to balance training stability and efficiency.

All experiments were conducted using a batch size of 32 and a learning rate of 0.001. Additionally, we employed a sigmoid activation function in the final layer and used binary cross-entropy as the loss function.

3.2 Hyperparameter Experiments

Building on the best model from the previous experiment, Inception V3 fine-tuned from a pretrained model without warmup, we proceeded to evaluate **RQ2**. To maintain consistency, we kept all parameters from the best-performing model unchanged, except for the hyperparameters under investigation.

In this experiment, we assessed a hyperparameter grid comprising three learning rates $(0.0001, 0.001, 0.01)$ and three batch sizes $(16, 32, 64)$, leading to a total of nine experimental runs.

3.3 Augmentation Experiments

With this experiments we aim to answer **RQ3**. It is relevant to understand how different data can enrich our models. In order to do so, we have implemented two different algorithms to artificially generate new data from the available one. We have designed four experiments to test the impact of data augmentation in our models' performance. With these augmentation experiments we want to test wether augmenting the data will make the trained model more resilient to variations in the images. We also want to see if augmentation will help to reduce overfitting and the overall generalization capability of the model. In order to assess this we will analyze how the mean average precision is affected by these methods.

3.3.1 Simple augmentation

In this experiment we generate augmentation data by resorting to the code already provided as part of this task. This is a simple augmentation method that generates new images to train on

by applying rotation, translation, zoom and vertical/horizontal flips to the available data.

3.3.2 Color Augmentation

In this second experiment we generate new images by applying *Random Color Jittering*. This method helps models become robust to variations in lighting conditionos, camera settings and color distributions, as can be seen in [4], [5] and [6] among others. This type of data augmentation works by adding images with slight adjustments in their color, saturation, illumination and hue. These modifications were done using the already existing methods from tensorflow.image package. We carefully selected the ranges for each of the random variations in order to avoid generating unrealistic images.

3.3.3 Occlusion Augmentation

For this third experiment we have implemented *Random Erasing*, also known as *Random Cutout*. This methos is particularly pertinent for our task for Pascal VOC contains many objects with occlusions and partial visibility. Random erasing makes models robust to partially visible objects by generating images which have parts of them covered [5], [7]. We implemented it by adding images that had rectangles of random dimensions and location erased. The expectation with this approach is that the model will learn to identify objects even when they are not fully visible.

3.3.4 Augmentation Combination

Finally, we designed an experiment to generate data by applying all the prior methods to the current images. Despite being complementary methods, the fact that the synthetic data has undergone so many variations could make it less valuable for our models' training. We will see this further in the results section.

3.4 Imbalance Handling Experiments

The Pascal VOC dataset (and the fraction of it we are using to train our models) is unbalanced. In order to address this, we have developed two algorithms to have a better balance among labels during training. Reducing imbalance during training should translate in a less biased model during test phase. There are two experiments to assess this, answering **RQ4**.

3.4.1 Class-Weighted Loss Function

In this experiment we train our model with a modified loss function that penalizes more heavily mistakes on minority classes. This has been shown to be an effective approach to class imbalance in [8] and [9] among others. With this experiment we aim to see our models overcome the "person" class outnumbering of all others in the dataset. We implemented this by tracing the relevance each class has in the dataset in terms of presence and creating a weighting function based on that that enables us to modify our current crossentropy loss function.

3.4.2 Class-Balanced Batch Sampling

This algorithm ensures that minority classes are well represented in each training batch. It does so while preserving the dataset's natural distribution despite improving the learning signal for rare classes [10], [11]. This was done by using, in a similar fashion as before, the relevance in terms of presence of each class in the dataset. This information was used when structuring the training batches to make sure the representation of each class was balanced.

3.5 Classifier Head Experiments

The previous experiments have explored the difference in multi-label classification performance between models, training approaches, hyperparameter configurations, and augmentation strategies. Throughout all of these experiments, we have always used the same classifier head (which we will refer to as "`default`") at the top of the neural networks, consisting of:

- A global average pooling (GAP) layer.
- A hidden dense layer with 1024 units and ReLU activation.
- An output dense layer with 20 units (equal to the number of classes) and sigmoid activation.

Now, in order to answer to **RQ5**, we tested two additional classifier heads that differ significantly from `default` in architecture. It is relevant to mention that all other aspects of the model and training configuration are taken from the best model obtained from the Hyperparameter Experiments described in Section 3.2, according to the mAP scores. The details of the new classifier heads to be tested, called "`ensemble`" and "`attention`", will be discussed in Sections 3.5.1 and 3.5.2, respectively.

3.5.1 `Ensemble` Classifier Head

The first of the new classifier heads consists of an *ensemble of classifiers* formed by three parallel sub-heads placed on top of the CNN backbone. Ensemble methods have been widely studied and have proven capable of obtaining better predictive performance than could be obtained from any of the constituent learning algorithms alone [12]. Furthermore, having multiple independent classifiers from which to extract predictions makes the whole learning system more robust against noise and less prone to overfitting [13], which is also reinforced by the use of an intermediate dropout layer.

Formally, the architecture of the `ensemble` classifier head consists of:

- A GAP layer.
- Three parallel sub-heads, each with the same architecture:
  - A hidden dense layer with 512 units and ReLU activation.
  - A dropout layer with a dropout rate of 0.2.
  - An output dense layer with 20 units and sigmoid activation.
- An average layer.

We decided to make the architecture of each sub-head similar to that of the `default` head so that the results are more comparable, while using a smaller hidden dense layer to maintain the complexity of the model within approximately the same range.

3.5.2 `Attention` Classifier Head

The second classifier head features a *Transformer-based attention mechanism*, which have demonstrated remarkable success in computer vision tasks by effectively modeling global relationships between features [14]. The use of attention mechanisms in multi-label classification tasks can potentially enhance the detection of co-occurrences of classes [15, 16].

Formally, the architecture of the `attention` classifier head consists of:

- A GAP layer.
- A dense embedding layer with 256 units and ReLU activation.

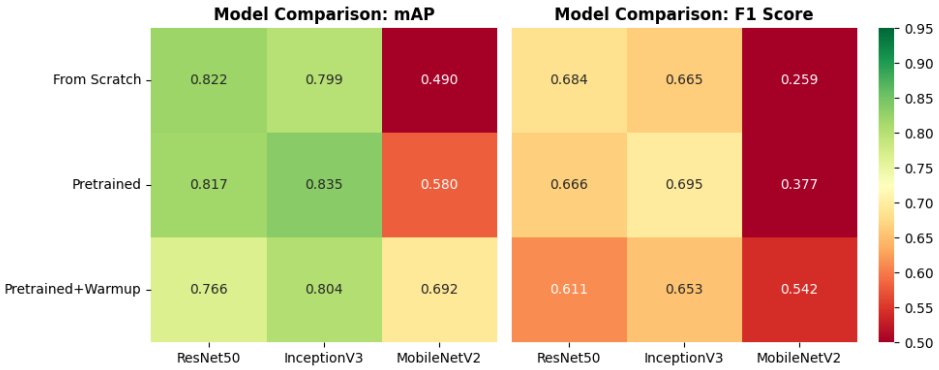

Figure 4: Model Comparisons: mAP and F1 Score

- A Transformer encoder layer composed of:
  - A multi-head self-attention mechanism with 4 attention heads.
  - A residual connection followed by layer normalization.
  - A feedforward network with:
    - A hidden dense layer with 512 units and ReLU activation.
    - An output dense layer with 256 units.
    - A dropout layer with a dropout rate of 0.2.
  - A final residual connection with layer normalization.
- An output dense layer with 20 units and sigmoid activation.

This architecture introduces an attention-based inductive bias, allowing the model to weigh feature importance dynamically. By maintaining a single Transformer encoder layer, we balance expressiveness and computational efficiency, keeping the overall model complexity comparable to the other classifier heads.

4 Results and Analysis

4.1 RQ1

The Inception V3 model, when pretrained without warmup, achieves the highest mAP score among the nine experiments, as shown in Figure 4. ResNet50 also performs well, but its best performance is observed when trained from scratch, indicating that pretraining does not significantly benefit this model. In contrast, MobileNetV2 achieves its best performance when pretrained with warmup, yet it still lags behind the other models.

When evaluating performance using the F1 score, the models exhibit a similar trend—ResNet50 and InceptionV3 perform well, while MobileNetV2 struggles, especially when trained from scratch. This suggests that MobileNetV2 lacks the capacity to learn effectively from scratch and benefits more from gradual fine-tuning. The overall results indicate that pretraining without warmup is the optimal approach for Inception V3, while ResNet50 can achieve strong performance without requiring pretraining. MobileNetV2, despite benefiting from warmup, remains the weakest model, suggesting it may not be the best choice for this task.

4.2 RQ2

We observed that the highest mAP score of 0.912 was achieved with a learning rate of 0.0001 for both batch sizes of 32 and 64. Our experimental results highlight that learning rate has a

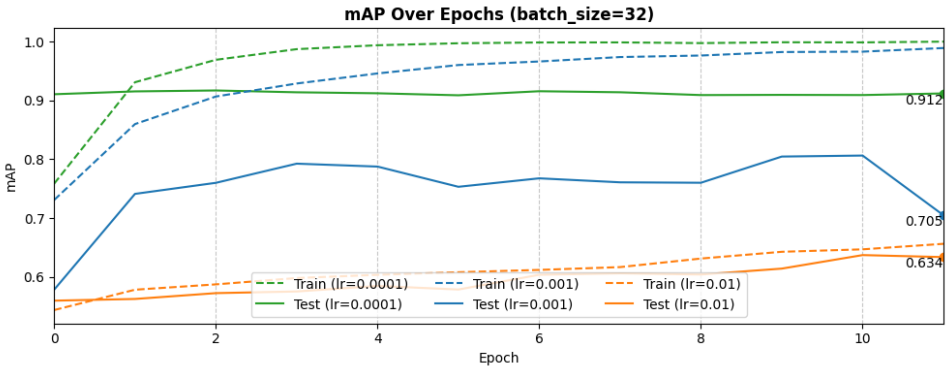

Figure 5: mAP Over Epochs (batch_size=32)

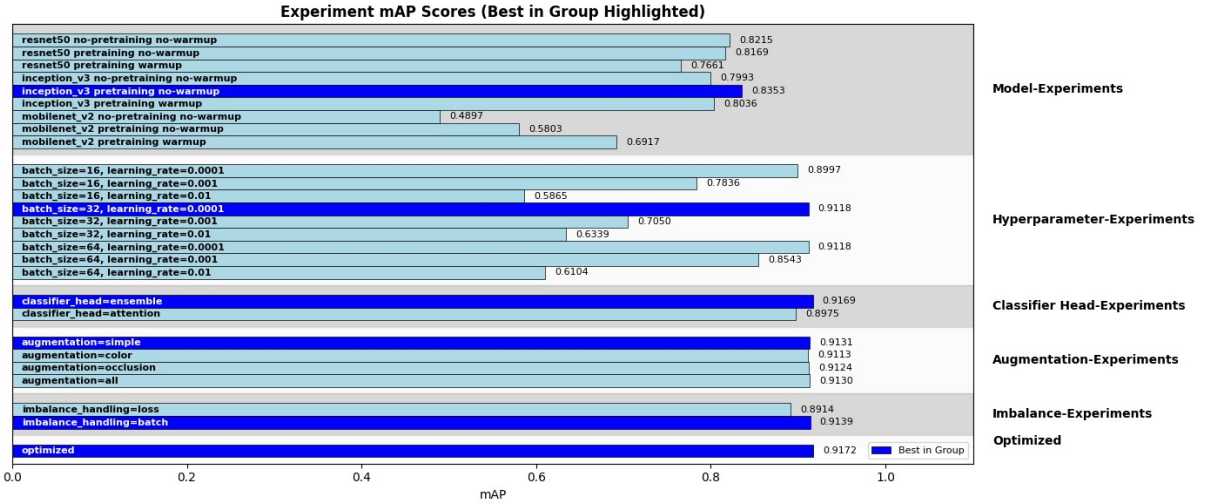

Figure 6: mAP score for the different experiments

significant impact on performance, whereas batch size plays a comparatively smaller role. Both batch sizes of 32 and 64 performed well, while a batch size of 16 was less effective.

As shown in Figure 5, the training and test curves for a learning rate of 0.0001 remain closely aligned, indicating strong generalization. In contrast, with a learning rate of 0.001, the training mAP continues to improve while the test mAP fluctuates, suggesting potential overfitting. The poor performance of the 0.01 learning rate indicates that the model struggles to learn effectively, likely due to large updates preventing convergence to an optimal solution.

4.3 RQ3

When comparing the results obtained by augmenting the data to those obtained in the rest of the experiments, we can conclude that for this particular dataset and augmentation techniques there is no significant improvement in model performance. We observe this through the mAP scores obtained for each experiment during testing (see figure 6).

As for the particular data augmentation techniques, we see practically no difference in mean average precision between them (see 6).

These results are likely due to the already large training dataset, which serves to achieve high mean average precision without the need of augmenting the data.

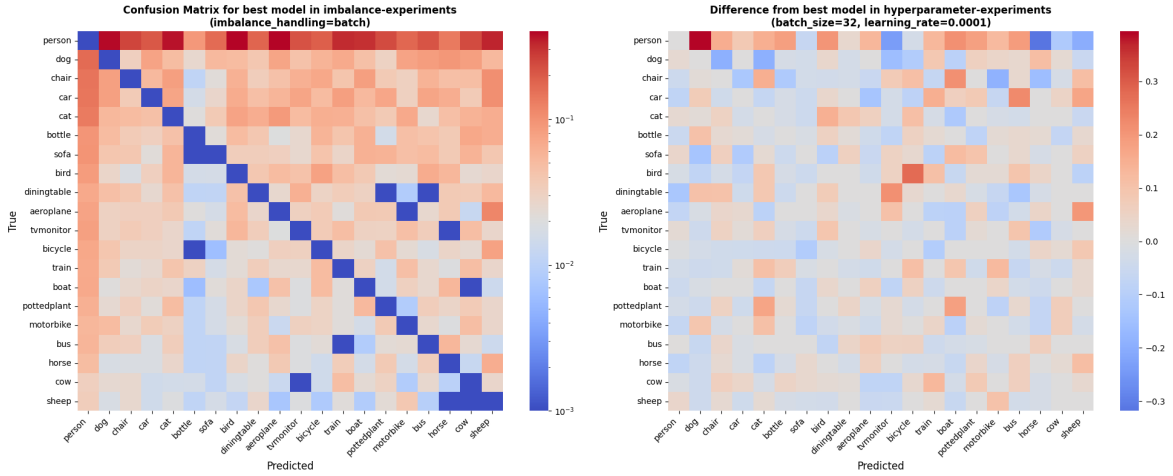

Figure 7: Comparison between the confusion matrices of the best balancing experiment and the best hyperparameter experiment

4.4 RQ4

As with the augmentation experiments, we have observed no significant improvement in the model's mean average precision after applying balancing compared to the results of the previous experiments. With the Class-Balanced Batch Sampling performing better than the Class-Weighted Loss Function (see 6)

When analyzing the confusion matrix (see 7), we can see that the best performing model from the data balancing experiments does not improve the detection of the minority classes when compared to the original data set.

4.5 RQ5

Our experiments on the classifier head (introduced in Section 3.5), show that performance is improved when using the `ensemble` head, while it does not seem to be altered when using `attention` as compared to using `default`. In Figure 8, we display the train and test curves of the loss and the mAP scores for the three classifier heads.

When we look at the mAP curves, we see that `ensemble` finished with a slightly better test score than the other two heads. However, this difference does not seem to be significant and is more likely due to stochasticity. In fact, none of the three the mAP scores seem to be improved significantly over the entire training process, which indicates that the pre-trained weights of the Inception V3 network already reach the level of performance that the network can achieve on this task, and the training process only produces overfitting. This hypothesis is reinforced when we study the loss curves, where we see that the testing losses achieve their lowest point at the first or second epoch, and then start slowly rising, while the training losses keep decreasing over epochs. In that same plot, we can also observe how the `ensemble` testing loss is consistently lower than the other two heads', while its training loss is slightly higher (although it follows a very similar trajectory to the others). This indicates that `ensemble` suffered less overfitting than the other two classifier heads.

The most relevant conclusions we can extract from these results about the two new classifier heads are that `ensemble` did in fact perform as expected and mitigated overfitting, with its ensemble of classifiers and use of dropout layers, while the `attention` did not show the improvements that were expected of it. The latter is likely due to insufficient complexity in the Transformer-based classifier head, since the embedding depth and number of hidden units were reduced from their standard values so that the total number of parameters was in a range simi-

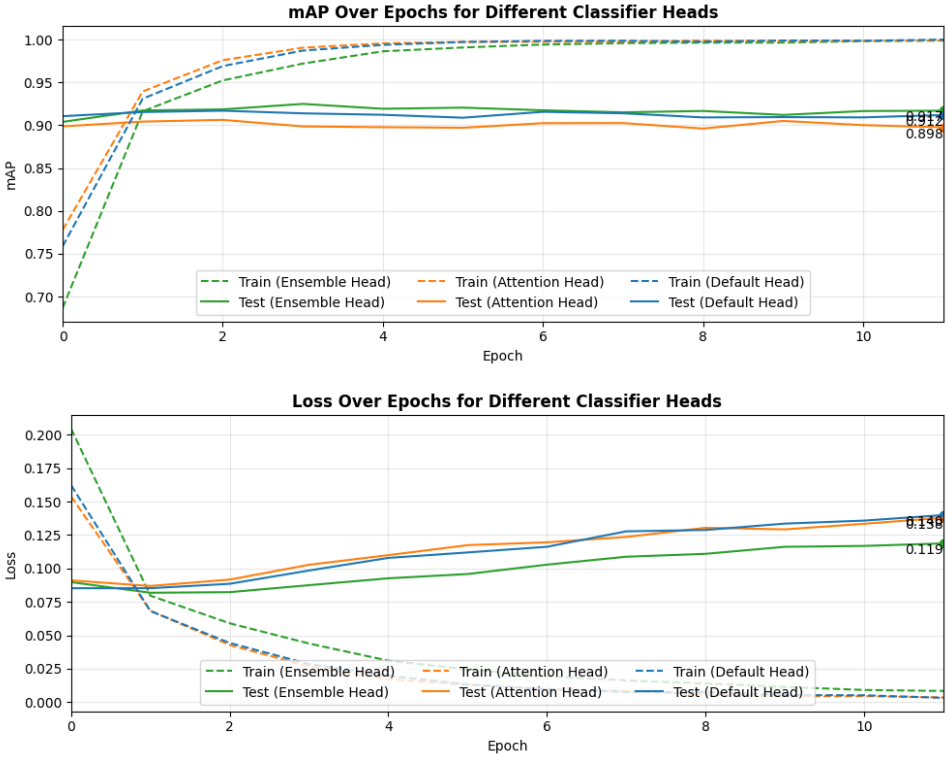

Figure 8: Train and test learning curves of loss and mAP scores for the three classifier heads

lar to the other two heads. Due to time and resource constraints, we were not able to perform further experiments with the `attention` head, but testing a deeper head architecture, possibly with more than one encoder layer, would be an interesting future line of work

5 Conclusions

Our experiments revealed several key insights for multi-label classification on Pascal VOC. We can state that the hypotheses derived from the initial dataset analysis were fairly correct. Despite the imbalance in the dataset makes classes like "cow" harder to learn than others like "person", we were able to build models that successfully fulfill this multi-label classification task. Inception V3 pretrained without warmup delivered the best performance among the tested architectures (mAP 0.835), while ResNet50 performed well even when trained from scratch. Learning rate had a more significant impact on model performance than batch size, with 0.0001 yielding optimal results regardless of batch size.

Interestingly, neither data augmentation techniques nor class imbalance handling methods significantly improved performance, suggesting the dataset was already sufficiently robust. For classifier heads, our ensemble approach showed modest improvements by reducing overfitting, while the attention-based mechanism would likely benefit from increased complexity in future implementations.

Our optimal configuration (Inception V3, learning rate 0.0001, batch size 64, ensemble classifier, with simple augmentation and batch imbalance handling) achieved an mAP of 0.917, demonstrating strong multi-label classification capabilities. These findings provide practical guidance for similar tasks, emphasizing the importance of architecture selection, learning rate optimization, and effective classifier head design.

References

[1] Baoyuan Wu et al. "Tencent ML-Images: A Large-Scale Multi-Label Image Database for Visual Representation Learning". In: *IEEE Access* 7 (2019), pp. 172683–172693. ISSN: 2169-3536. DOI: 10.1109/access.2019.2956775. URL: http://dx.doi.org/10.1109/ACCESS.2019.2956775.

[2] Fan Lyu et al. *Coarse to Fine: Multi-label Image Classification with Global/Local Attention*. 2020. arXiv: 2012.13662 [cs.CV]. URL: https://arxiv.org/abs/2012.13662.

[3] Xing Cheng et al. *MlTr: Multi-label Classification with Transformer*. 2021. arXiv: 2106.06195 [cs.CV]. URL: https://arxiv.org/abs/2106.06195.

[4] Ekin D Cubuk et al. "Autoaugment: Learning augmentation strategies from data". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 113–123.

[5] Terrance DeVries and Graham W Taylor. "Improved regularization of convolutional neural networks with cutout". In: *arXiv preprint arXiv:1708.04552* (2017).

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[7] Zhun Zhong et al. "Random erasing data augmentation". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 07. 2020, pp. 13001–13008.

[8] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988.

[9] Yin Cui et al. "Class-balanced loss based on effective number of samples". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 9268–9277.

[10] Li Shen, Zhouchen Lin, and Qingming Huang. "Relay backpropagation for effective learning of deep convolutional neural networks". In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*. Springer. 2016, pp. 467–482.

[11] Bingyi Kang et al. "Decoupling representation and classifier for long-tailed recognition". In: *arXiv preprint arXiv:1910.09217* (2019).

[12] Robi Polikar. "Ensemble based systems in decision making". In: *IEEE Circuits and Systems Magazine* 6.3 (2006), pp. 21–45. DOI: 10.1109/MCAS.2006.1688199.

[13] Lior Rokach. "Ensemble-based classifiers". In: *Artificial Intelligence Review* 33.1–2 (2010), pp. 1–39. DOI: 10.1007/s10462-009-9124-7.

[14] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *arXiv preprint arXiv:2010.11929* (2020).

[15] Shilong Liu et al. "Query2Label: A Simple Transformer Way to Multi-Label Classification". In: *CoRR* abs/2107.10834 (2021). arXiv: 2107.10834. URL: https://arxiv.org/abs/2107.10834.

[16] Jack Lanchantin et al. "General Multi-label Image Classification with Transformers". In: *CoRR* abs/2011.14027 (2020). arXiv: 2011.14027. URL: https://arxiv.org/abs/2011.14027.

6 Appendix

	Experiment Set	Model	Test Loss	Test Accuracy	Test F1 Score	Test mAP
0	model-experiments	resnet50 no-pretraining no-warmup	0.2417	0.8806	0.6842	0.8215
1	model-experiments	resnet50 pretraining no-warmup	0.1988	0.9018	0.6658	0.8169
2	model-experiments	resnet50 pretraining warmup	0.2253	0.8721	0.6115	0.7661
3	model-experiments	inception_v3 no-pretraining no-warmup	0.2269	0.8698	0.6650	0.7993
4	model-experiments	**inception_v3 pretraining no-warmup**	0.1727	0.9137	0.6954	**0.8353**
5	model-experiments	inception_v3 pretraining warmup	0.1889	0.8985	0.6532	0.8036
6	model-experiments	mobilenet_v2 no-pretraining no-warmup	0.4132	0.6947	0.2586	0.4897
7	model-experiments	mobilenet_v2 pretraining no-warmup	0.3973	0.7572	0.3768	0.5803
8	model-experiments	mobilenet_v2 pretraining warmup	0.2647	0.8256	0.5424	0.6917
9	hyperparameter-experiments	batch_size=16, learning_rate=0.0001	0.1652	0.9240	0.7779	0.8997
10	hyperparameter-experiments	batch_size=16, learning_rate=0.001	0.1787	0.8994	0.6376	0.7836
11	hyperparameter-experiments	batch_size=16, learning_rate=0.01	0.2010	0.7925	0.3092	0.5865
12	hyperparameter-experiments	**batch_size=32, learning_rate=0.0001**	0.1399	0.9294	0.7905	**0.9118**
13	hyperparameter-experiments	batch_size=32, learning_rate=0.001	0.2596	0.8531	0.5439	0.7050
14	hyperparameter-experiments	batch_size=32, learning_rate=0.01	0.2534	0.8491	0.4387	0.6339
15	hyperparameter-experiments	batch_size=64, learning_rate=0.0001	0.1251	0.9315	0.7977	0.9118
16	hyperparameter-experiments	batch_size=64, learning_rate=0.001	0.2009	0.8926	0.7222	0.8543
17	hyperparameter-experiments	batch_size=64, learning_rate=0.01	0.1912	0.8308	0.4596	0.6104
18	classifier_head-experiments	**classifier_head=ensemble**	0.1187	0.9388	0.7973	0.9169
19	classifier_head-experiments	classifier_head=attention	0.1379	0.9261	0.7824	0.8975
20	augmentation-experiments	**augmentation=simple**	0.1381	0.9279	0.7972	**0.9131**
21	augmentation-experiments	augmentation=color	0.1337	0.9314	0.7985	0.9113
22	augmentation-experiments	augmentation=occlusion	0.1341	0.9307	0.7951	0.9124
23	augmentation-experiments	augmentation=all	0.1356	0.9299	0.7950	0.9130
24	imbalance-experiments	imbalance_handling=loss	4.1940	0.9043	0.7772	0.8914
25	imbalance-experiments	**imbalance_handling=batch**	0.1403	0.9264	0.7965	**0.9139**
26	optimized	**optimized**	0.1212	0.9379	0.8010	**0.9172**

Table 1: Results for all experiments. Best model from each experiment set shown in bold.