Supervised Learning: classification on FoodX-251

Alessandro Dubini 885957
a.dubini@campus.unimib.it

Elia Leonardo Martin 886366
e.martin4@campus.unimib.it

Abstract

This project explores the problem of fine-grained food classification using the FoodX-251 dataset [3], which contains 251 categories and over 158,000 web-sourced images. A subset of the dataset was employed due to computational constraints, and different data augmentation techniques were applied to improve model generalization. A custom convolutional neural network architecture was developed, employing residual connections, squeeze-and-excitation (SE) blocks, and the SiLU activation function, while accommodating the constraint of staying under 5 million parameters. The model achieved 51.04% accuracy and demonstrated balanced performance across classes. In addition to the supervised baseline, a self-supervised learning (SSL) approach was explored using jigsaw puzzle reconstruction and geometric transformation recognition tasks as pretext training stages. Although the SSL approach yielded slightly lower accuracy (45.34%), it confirmed the model's ability to learn transferable features without direct supervision. These results highlight both the potential of lightweight architectures for visual recognition tasks and the impact of limited computational resources on performance.

1 Introduction

Fine-grained food classification is a challenging task due to high inter-class similarity and intra-class variability. This project addresses this problem using the FoodX-251 dataset, which contains 251 food categories collected from the web.
To manage computational limitations, a lightweight convolutional neural network was designed using residual connections, squeeze-and-excitation blocks, and the SiLU activation function. The model was trained on a subset of the dataset and evaluated through standard classification metrics.
In parallel, a self-supervised approach was explored using jigsaw puzzle and geometric transformation tasks to pretrain the model without labels. This enabled a comparison between supervised and self-supervised performance under consistent training conditions.

2 Dataset Analysis

The dataset employed in this project is FoodX-251 [3], consisting of 251 fine-grained food categories with 158'000 images collected from the web. It is originally partitioned into training, validation, and test sets. However, due to the absence of labels in the test set, an 80/20 split was applied to the original training set to create new training and validation subsets. The provided validation set was subsequently used as an external test set.
Several characteristics of the dataset were observed during analysis. Notably, a significant degree of inter-class similarity exists, where distinct classes share visual attributes such as color, shape, or texture (e.g., different types of pasta can share the same color and the same texture but belong in different classes).

Similarly, intra-class variability is present, as individual samples within the same category may differ substantially in appearance (e.g., pasta served on a plate versus packaged pasta boxes in the "farfalle" class).
Since the images have been collected from the web using the label itself as keyword, some classes include irrelevant or misleading images. For instance, the "nachos" class contains images of the television character Nacho Varga from Better Call Saul. The "POI" class is particularly problematic, as it includes no actual food-related images.

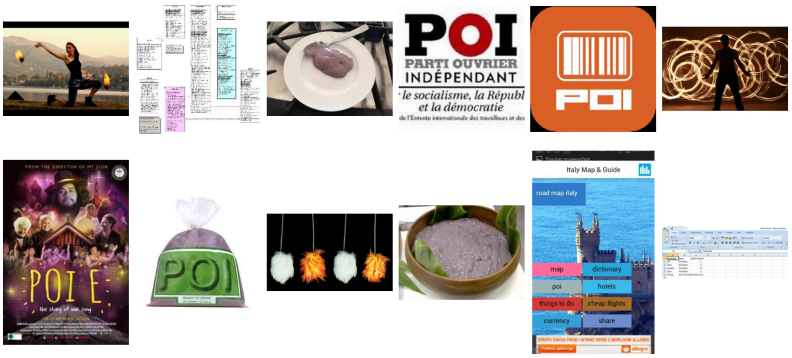

Figure 1: Samples extracted from the poi class

As stated in the original dataset paper:

> "*For each class, we use web image search to download the corresponding images. Due to the nature of images on these search engines, these images often include images of processed and packaged food items and their ingredients resulting in cross-domain noise*"[3].

This cross-domain noise significantly impacts classification performance, especially in classes with a lower proportion of relevant food images, leading to a noticeable decrease in accuracy.

2.1 Dataset management

Due to computational constraints, the number of training images per class was limited. Various limits were tested (e.g., 100, 200, and 350 images), with 350 determined to offer the best balance between training time and classification accuracy. In cases where fewer than 350 images were available for a given class, all available images were used. To enhance generalization performance, several data augmentation techniques were applied. These included:

- **Resolution standardization**: all images were resized to 128×128 pixels.
- **Geometric transformations**: such as *RandomResizedCrop* and *RandomHorizontalFlip*.
- **Color transformations**: including *ColorJitter*, *RandomGrayscale*, *RandomErasing*, and *Normalize*.

For validation, a maximum of 30 images per class was used to ensure consistency.

3 Network architecture

To successfully address the classification task, we explored different network architectures with different key ideas, focusing on simplicity and efficiency in order to keep the parameter count under 5 millions.
Our first attempt was a deep convolutional neural network with approximately 4 million parameters. However, this model suffered from severe gradient vanishing issues, which led to poor performance. In our second approach, we incorporated squeeze-and-excitation (SE) blocks to help the network focus on the most informative channels, enabling the model to emphasize relevant features and

suppress less useful ones. These blocks were implemented following the design proposed by Hu et al [2]. Despite obtaining an increase in performance, gradient vanishing remained a challenge.
Our final and most successful architecture was designed considering the residual connection strategy explained by He et al [1]. The model we built combined three key elements: skip connections to alleviate gradient vanishing, SE blocks, and the SiLU activation function in place of ReLU for smoother gradient flow. This architecture was used as the backbone for both our supervised and self-supervised learning experiments, with a parameter count of 2.2 millions.

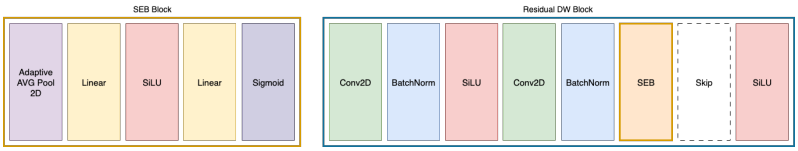

Figure 2: Squeeze and excitation block and residual depthwise block

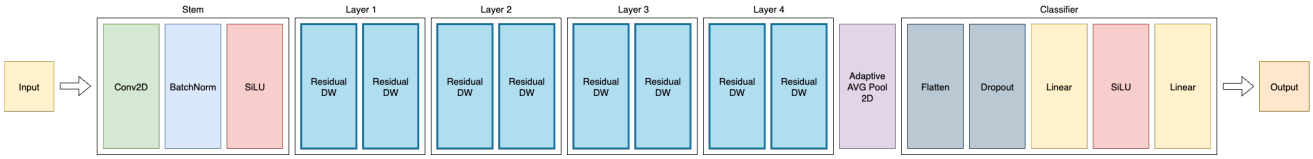

Figure 3: Residual depthwise block

4 Supervised Experiment

The following experiments were conducted on different devices to ensure the best results by parallelizing the workflow, employing the following setups:

- Google Colab: Tesla T4/P100, RAM 16GB
- i5 9600k, 2070 Super, RAM 32GB
- i7 11800H, 3070 Laptop, RAM 16GB

This allowed us to test different model configurations in a reasonable time and evaluating their performance.

4.1 Training hyperparameters

To ensure efficient training, we carefully selected a set of hyperparameters that balanced convergence speed and generalization. The following configuration was employed:

- **Epochs**: 50 and **Learning rate**: 0.001 to obtain a good balance between efficiency and performance.
- **Optimizer**: AdamW in order to take advantage of the weight decay feature (0.0001).
- **Scheduler**: ReduceLROnPlateau to dynamically adapt the learning rate in response to validation accuracy stall.
- **Criterion**: CrossEntropyLoss selected as standard choice for multiclass classification tasks.

4.2 Training results

As shown in the curves below, the model starts to learn meaningful features early in training. After the 20th epoch, we can notice a separation between the training and validation curves, indicating a sign of overfitting. This is behavior is due to the limited size of the training dataset, limitation imposed by the restricted computational resources available.

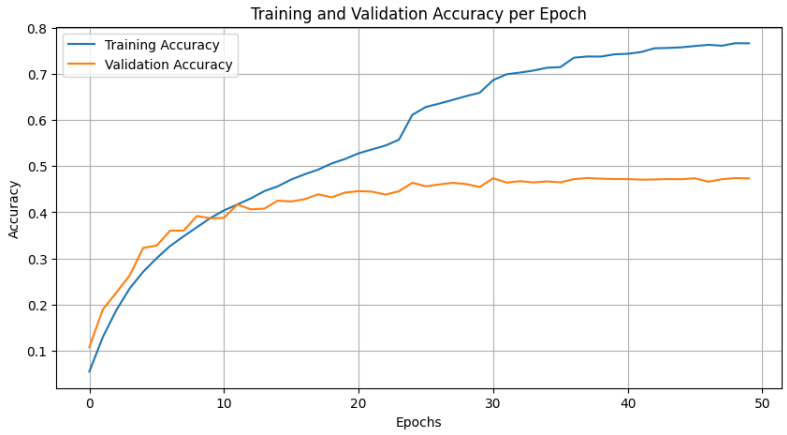

Figure 4: Training and validation accuracies (350 images per class)

In fact, as shown below, increasing the number of available samples always led to better results, suggesting the validity of our model design and highlighting the impact of limited computational power.

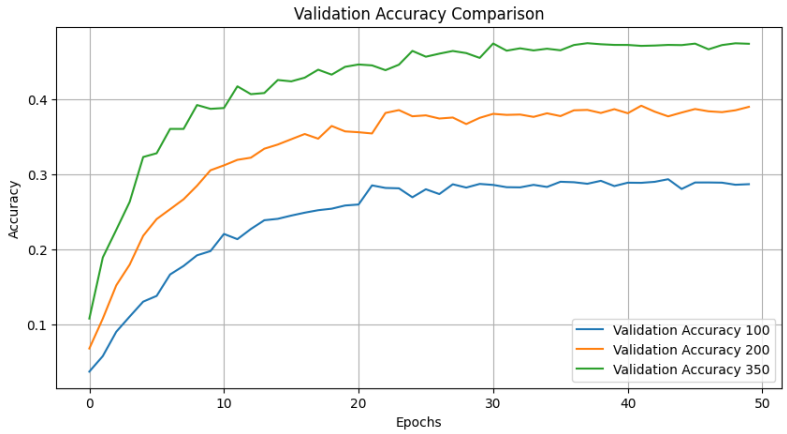

Figure 5: Comparison of validation results for different sample sizes

At the end of the training routine, the model reached a training accuracy of 76.67% and a validation accuracy of 47.36%

4.3 Metrics on external test

To further asses the performance of our model, we evaluated it on the external set through different classification metrics. This allowed us to gain a more meaningful understanding of its performance.

These results show that the model maintains balanced performance across classes, as explained by the close values of macro and weighted scores.
The macro scores, while slightly lower than the weighted ones, confirm that the model is not completely biased toward the majority classes, which is particularly relevant given the imbalanced nature of the dataset.

Metric	Score (%)
Accuracy	51.04
F1 Score (Macro)	49.17
F1 Score (Weighted)	50.94
Precision (Macro)	49.83
Precision (Weighted)	51.78
Recall (Macro)	49.55

Table 1: Evaluation metrics on the test set.

5 Self-Supervised Experiment

The core idea behind our self-supervised approach is to encourage the model to learn meaningful visual feature such as shape, position, texture and color to perform predictions. Hence, we selected two distinct pretext tasks in order to help the model capture spatial context structures:

- Jigsaw puzzle resolution.
- Geometric transformation recognition.

5.1 Jigsaw puzzle

This pretext task has the purpose of help the model to learn spatial relationships while solving a Jigsaw puzzle by predicting the correct permutation of shuffled image patches. The implementation of this approach followed the design proposed by Noroozi et al.[4], with the puzzle visualization function adapted from the code provided by Park [5].

Figure 6: Image example extracted from the jigsaw dataset

5.2 Geometric transformation recognition

Once the network has been trained on the first pretext task, we further stimulate the model to learn new spatial features by recognizing which, among the following geometric transformations, has been applied to the image.

- Rotation 0°
- Rotation 90°
- Rotation 180°
- Rotation 270°
- Center crop 64x64

Figure 7: Image example extracted from the geometric permutations dataset

5.3 Training

The self-supervised training routine follows this pipeline:

1. **Training on the Jigsaw Puzzle pretext task**

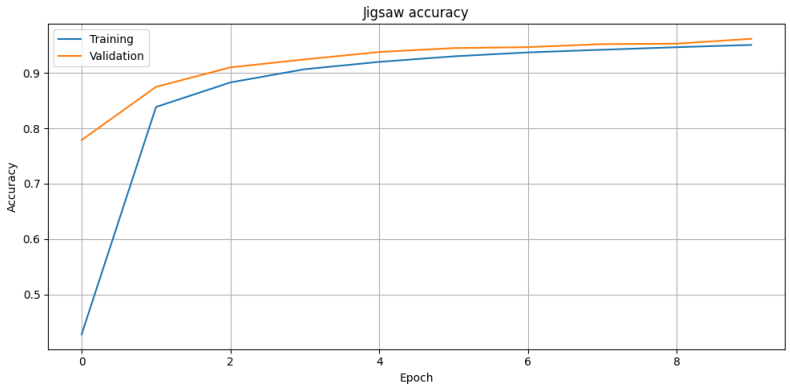

Figure 8: Training results for the Jigsaw task

2. **Training on the Geometric Transformation Recognition task**

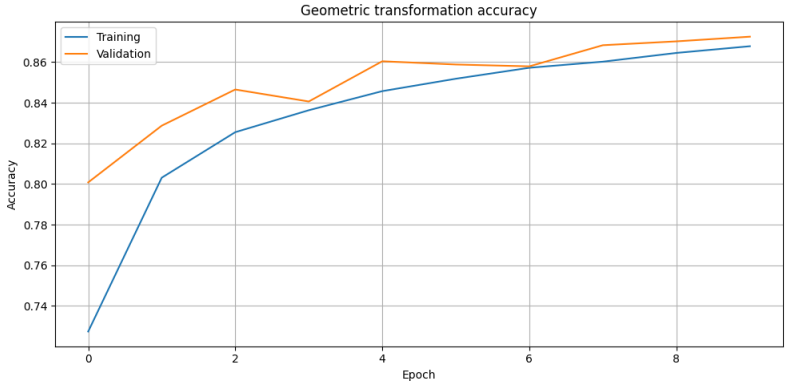

Figure 9: Training results for the Geometric task

3. **Training on the Food Classification task**

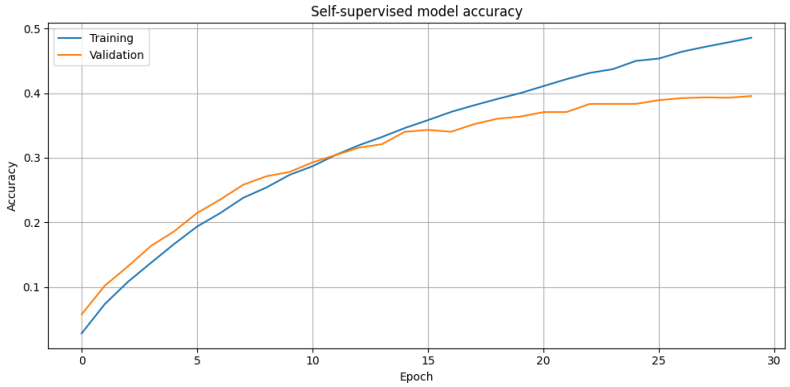

Figure 10: Training results for the Classification task

Both the pretext tasks were trained for 10 epochs with a learning rate of 0.001. To ensure a fair comparison between the supervised and self-supervised approaches, identical training routines were employed. Hence we make reference to the one described before in Section 4.1. The dataset structures was kept consistent, with 350 images per class, modified according to each pretext task.

5.4 Metrics

We computed the Section 4.3 metrics to perform a comparison between the two approaches. The

Metric	**Score (%)**
Accuracy	45.34
F1 Score (Macro)	43.24
F1 Score (Weighted)	44.85
Precision (Macro)	45.43
Precision (Weighted)	46.95
Recall (Macro)	43.67

Table 2: Evaluation metrics on the test set after self-supervised pretraining.

results obtained shows slightly lower performance compared to the supervised setting. The small gap between the Macro and Weighted metrics confirms that the learned features generalize across both frequent and less represented classes, despite the absence of direct supervision during pretraining.

6 Conclusion and future developments

The overall model performance, both in supervised and self-supervised approaches, prove that the architecture employed is suitable for the task. The primary limitation faced is represented by the unsuitable computational power available, which restricted the amount of training data we could use across the entire pipeline (including both supervised and self-supervised training), in order to keep the total training time under 15 hours.
Future developments may involve retraining the network on a more powerful system using all the available training images, to better assess the network's potential. In table3, we can see how the self-supervised approach performs slightly worse than the supervised model, despite the more complicated training approach.

Metric	Supervised (%)	Self-Supervised (%)
Accuracy	51.04	45.34
F1 Score (Macro)	49.17	43.24
F1 Score (Weighted)	50.94	44.85
Precision (Macro)	49.83	45.43
Precision (Weighted)	51.78	46.95
Recall (Macro)	49.55	43.67

Table 3: Comparison between supervised and self-supervised performance on the test set.

This may be caused by a suboptimal choice of the pretext task or the architecture itself not being suitable for self-supervised learning. In future, it would be interesting to explore different SSL approaches based on contrastive learning methods such as SimCLR, where the model may be able to learn better features early in training and provide a concrete boost in performances.

References

[1] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[2] Jie Hu et al. *Squeeze-and-Excitation Networks*. 2019. arXiv: 1709.01507 [cs.CV]. URL: https://arxiv.org/abs/1709.01507.

[3] Parneet Kaur et al. *FoodX-251: A Dataset for Fine-grained Food Classification*. 2019. arXiv: 1907.06167 [cs.CV]. URL: https://arxiv.org/abs/1907.06167.

[4] Mehdi Noroozi and Paolo Favaro. *Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles*. 2017. arXiv: 1603.09246 [cs.CV]. URL: https://arxiv.org/abs/1603.09246.

[5] Kalel Park. *FG-SSL: Self-Supervised Learning for Fine-Grained Visual Recognition*. https://github.com/kalelpark/FG-SSL. Accessed: 2025-07-15. 2023.

Declaration

This paper is the result of original work. All external sources of information or ideas were appropriately cited. The analysis, research and conclusions presented are the result of our independent work.