Deep Convolutional Neural Networks

Oriol Miró and Marc González

MAI, UPC
Deep Learning, Practice 1
April 25, 2025

We revisit museum–medium classification with MAMe, a 37 k-image dataset whose task is to assign each artwork to one of 29 material/technique classes. Instead of pursuing the usual high-resolution or transfer-learning angle, we ask a simpler question: how much does the architecture itself matter when everything is trained from scratch? To this end we implement MaorNet, a VGG-style "standard" CNN with 30.4 M parameters, and contrast it with an 11.2 M-parameter ("non-standard") ResNet-18. A three-stage protocol—coarse grid search, single-factor ablations, and targeted refinement—yields final test accuracies of 77.9 % for MaorNet and 81.2 % for ResNet-18. Although both lag behind the original paper's ImageNet-pre-trained counterparts by 2–3%, they recover $\approx 90\%$ of the transfer-learning bonus without external data. Ablations some some techniques are more important depending on the architecture, while either dropout or weight decay reduces the train–test gap by up to 10 %; residual connections bring a further 3 % advantage and better robustness to regularisation.

1 Introduction

Deep convolutional networks have reached near-human accuracy on benchmark datasets such as ImageNet, largely thanks to aggressive transfer learning and ever-larger backbones. Yet many specialised domains—cultural heritage, medical imaging, scientific microscopy—lack the scale or legal clearance to pretrain on millions of images. The MAMe dataset [7] exemplifies this situation: it contains 37 407 museum photographs labelled by *medium* (oil on canvas, bronze, engraving, ...), is legally redistributable, but is too small for training state-of-the-art models from scratch without great care.

Previous work on MAMe focused on input resolution and aspect ratio [7]. In contrast, we investigate the *architectural dimension*: given the same 256×256 crops and no external data, how do a classical, Standard CNNs and a modern Non-Standard CNNs compare? To that end we contribute:

1. **MaorNet**, a deliberately simple "standard" architecture that follows the canonical convolution–pooling–FC recipe but scales width,
2. a from-scratch implementation of **ResNet-18** with optional dropout hooks for controlled experiments,
3. a three-stage experimental pipeline—grid search, ablation, refinement—that exposes the impact of each training choice,
4. a quantitative comparison against the ImageNet-initialised baselines of [7].

The study shows that residual connections, even in a small network, deliver a consistent 3% advantage and allow us to recover more than 90 % of the transfer-learning benefit.

2 Dataset

We use the MAMe (Museum Artworks Medium) dataset, a carefully curated dataset introduced by Parés et al. (2020) [7]. It contains a total of 37,407 artwork images belonging to 29 classes representing artistic mediums, such as "oil on canvas", "bronze", or "engraving", among others. These images were selected from three museums: The Metropolitan Museum of Art of New York, The Los Angeles County Museum of Art, and The Cleveland Museum of Art. All images are provided under a CC0 license, thus allowing unrestricted usage for research purposes.

Figure 1: *Example images from the MAMe dataset [7].*

The original dataset consists primarily of high-resolution images with variable aspect ratios and shapes.

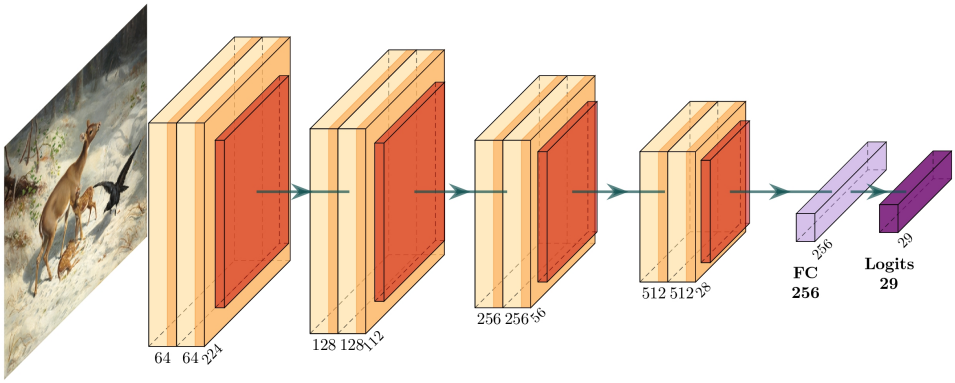

Figure 2: *Visual representation of MaorNet architecture. Four convolutional blocks (encoder) each composed of two convolutional layers with ReLU and batch normalization, followed by max pooling; and a fully-connected head (decoder) with dropout. Example photo comes from the MAMe dataset [7].*

Nevertheless, to facilitate efficient and standardized training processes, the authors provided a downsampled version of the dataset called "R65k-FS" (Resolution 65k pixels - Fixed Shape). Every image in the "R65k-FS" dataset is downscaled to a unified spatial resolution of 256×256 pixels, resulting in $65,536$ pixels per image. This reshaping introduces a mild image distortion, but standardises input dimensions and allowing easier computational operations with balanced batches.

The paper introducing the MAMe dataset [7] tested various hypotheses regarding factors affecting performance, such as fixed versus variable input sizes and high versus low resolutions. We are, however, not interested in performance *w.r.t.* the image characteristics but rather *w.r.t.* the architectures. For this reason, stick to the "R65k-FS" version to provide a baseline scenario compatible with standard deep CNN training. Figure 1 shows some examples from the dataset.

3 Architectures

We will evaluate an *Standard* architecture and a *non-Standard* architecture. For the former, we introduce Maornet (Marc-Oriol-Net), implemented by us *from scratch* following the "General Scheme of Standard CNNs Architectures" described in the course; we expand on this architecture in Subsection 3.1. For the latter, we chose ResNet-18, a popular non-standard CNN. We chose this architecture because of pure personal interest, as we thought it to be very interesting, as well as having a similar number of layers to MaorNet; such architecture is already tested in the MAMe paper [7]—albeit they start off a version pre-trained on ImageNet, while we will start from scratch—and we believe comparing their performances will be interesting. Despite it being a well-known architecture, we briefly describe it in Subsection 3.2.

Note that although the original images are of size 256×256 (as discussed in Section 2), in the following section we refer to the input as 224×224. This discrepancy is due to the data augmentation we apply, which will be further explained in Section 4.

3.1 MaorNet (Standard Architecture)

We propose MaorNet as a standard CNN architecture that adheres to the guidelines and structural conventions introduced in the course and common CNN literature (we revised the papers introduced in class: AlexNet [6], ZfNet [10], and VGG [9]). As such, MaorNet follows an encoder-decoder pattern: multiple blocks of 3×3 (as [9] found them so useful) convolutions, each block ending with a 2×2 Max Pooling, followed by fully-connected classification layers. Unlike the earliest "pure" VGG implementations, we allow ourselves the liberty of including Batch Normalization [4] (as it accelerates convergence and stabilizes deep networks) and Dropout for regularization (although we disable these for ablation studies later on).

The encoder consists of four convolutional blocks, each having exactly two convolutional layers and each convolution layer is formally:

$$x_{l+1} = \mathrm{ReLU}(\mathrm{BatchNorm}(\mathrm{Conv2D}(x_l))),$$

where kernel size 3×3 and "same" padding to preserve spatial dimensions. After every block, a 2×2 Max Pool reduces the spatial dimensions by half. Concretely, assuming 224×224 inputs, at the end of the four blocks we have a $\frac{224 \times 224}{2^4} = 14 \times 14$ output. The first block uses 64 filters, and we double the number for each block, ending up with $64 \times 2^{4-1} = 512$ filters

After the encoder, the decoder (fully-connected head) flattens the feature maps and processes them via two linear layers. The first layer has 256 neurons with ReLU activation and optional Dropout, and the second layer is a linear projection to the logits for the 29 classes:

$$y = \mathrm{FC}_2\left(\mathrm{Dropout}\left(\mathrm{ReLU}\left(\mathrm{FC}_1\left(\mathrm{Flatten}(x)\right)\right)\right)\right).$$

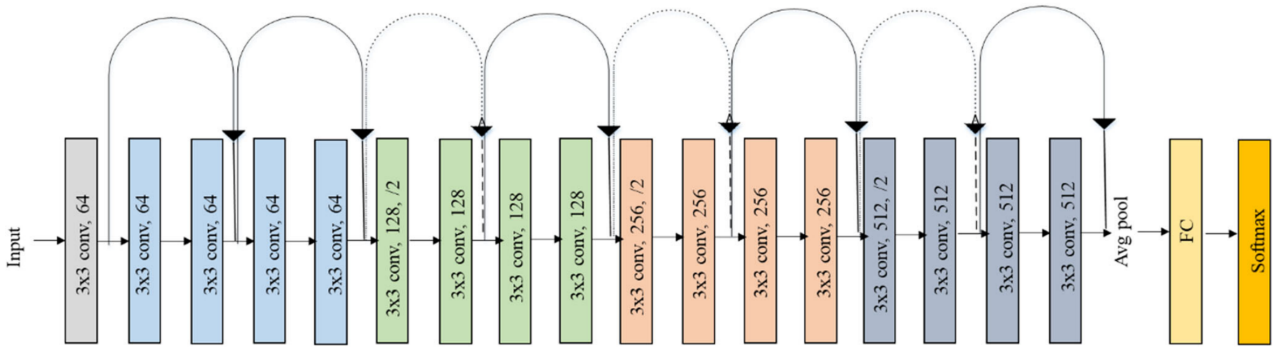

Figure 3: *Original ResNet-18 Architecture [extracted from [8]]*

In total, for 224×224 inputs, MaorNet has around **30.4 million** parameters (all trainable). The majority (around **25.7 million**) lie in the fully connected layers. Despite most modern architectures moving more parameter mass onto the encoder, we keep a FC head to study explicit capacity vs. residual capacity. Figure 2 illustrates the model[1]

3.2 ResNet-18 (Non-Standard Architecture)

As a representative of non-standard architectures, we selected ResNet-18 [3], one of the most influential and widely adopted CNNs to date. Although extensively studied, we implemented it from scratch to thoroughly understand its internal structure and to experiment with certain design choices (*e.g.*, weight initialization strategies, incorporation of Dropout, *etc.*). We followed the canonical ResNet-18 structure but additionally explored inserting dropout layers as suggested by [5], with the option to toggle them on or off for experimentation.

Each stage of ResNet-18 consists of several *basic blocks*, each containing two 3×3 convolutional layers. Batch Normalization and ReLU activations are applied after each convolution. These blocks employ residual (identity) connections to facilitate effective gradient propagation and ease optimization. When the dimensions of input and output differ, a 1×1 projection shortcut (with stride 2) is used to match them. The mathematical formulation of the basic block can be expressed as:

$$y = \mathrm{BN}_2\left(\mathrm{Conv}_2\left(\mathrm{ReLU}\left(\mathrm{BN}_1\left(\mathrm{Conv}_1(x)\right)\right)\right)\right)$$

with the final block output computed as $\mathrm{ReLU}(y + x)$.

Specifically, the ResNet-18 architecture comprises:

- A stem consisting of a 7×7 convolution (stride 2), Batch Normalization, ReLU, followed by a 3×3 Max Pooling (stride 2),
- Four residual stages with **2, 2, 2, 2** basic blocks respectively, featuring channel dimensions **64, 128, 256, 512**,
- Adaptive average pooling and a fully connected layer that projects the feature maps onto the 29 MAMe classes.

For inputs of size 224×224, our ResNet-18 implementation has approximately **11.2 million trainable parameters**—about a third of MaorNet's. Unlike MaorNet, where the majority of parameters (over 80%) reside in the fully connected layers, ResNet-18 distributes its capacity more evenly across the convolutional blocks. This leads to a more parameter-efficient representation and potentially better generalization, especially for image recognition tasks.

In terms of structural complexity, ResNet-18 about as deep as MaorNet, so it will be interesting to compare the effect of the residual connections. Figure 3 provides a visual overview of ResNet-18.

4 Methodology

We want to both find suitable configurations for each architecture, and study the effects of specific hyperparameters on overfitting and underfitting. Some of these hyperparameters—such as dropout or weight decay—have already been studied in the literature, and we were particularly interested in validating some of those hypotheses under our own controlled setting.

Given the relatively high training time per configuration (approximately two hours), a full exhaustive grid search was not viable. Instead, we followed a three-stage process: we began with a broad grid search to explore the space and find viable candidates; we then performed ablation studies to understand the effect of individual choices (*e.g.*, enabling or disabling Dropout, or batch normalization); and finally, we refined the best configurations to further improve their performance. This process was repeated independently for each architecture.

In the first stage, we fixed some hyperparameters that are widely accepted in CNN training pipelines. Specifically, we used **SGD** (as noted in the course slides as the optimizer for Standard CNN Architectures) with momentum (0.9) and weight decay (0.0005) for MaorNet, following classical practice; and **AMSGrad**, a vari-

[1] generated using `https://github.com/HarisIqbal88/PlotNeuralNet`

ant of Adam, for ResNet-18, as it was used in [7] for the same architecture. We then explored combinations of learning rates $\{10^{-2}, 10^{-3}, 10^{-4}\}$, batch sizes $\{128, 256, 512\}$, and initialization methods (Gaussian vs. He). The tested ranges are inspired by the literature, as most successful architectures in image recognition tasks tend to use values in that ballpark. If we encounter overfitting, we diagnose the situation and either adjust the aforementioned or introduce some other control techniques.

To account for the stochastic nature of training, we ran each configuration three times with different seeds. We then averaged the validation accuracy across runs at each epoch. For selection among configurations, we took the maximum of this averaged curve minus its standard deviation; this favors configurations that are both effective and consistently performant, reducing the influence of lucky outliers.

Once a good configuration was found, we moved on to ablation studies. These were not meant to improve performance, but to observe how individual components influenced training behaviour. For instance, we tested what happened when Dropout was removed, or when Batch Normalization was disabled. Finally, "refinement stage" is very situation-dependant. We always increase the number of training epochs (trying first with early stopping, patience$= 10$) and introduce techniques such as learning rate scheduling; however, depending on the situation and results from the two previous experiments, we tune different parameters (*e.g.*, weight decay) or introduce different techniques (*e.g.*, learning rate warmup).

All models were trained using cross-entropy loss. We guide our decisions by model accuracy, consistent with the methodology employed in the original MAMe study [7]. While the training and validation splits are *balanced*, as noted in their work, the test set is not, which may affect the representativeness of test performance results.

Data Augmentation and Preprocessing. To help with generalization, we followed the augmentation protocol proposed in the original paper [7]. During training, each image was randomly:

1. rotated within $[-30°, 30°]$,
2. random crop to 224×224 from the original 256×256 image,
3. and flipped horizontally with 50% probability.

At validation and test time, we simply performed a center crop to 224×224, avoiding any randomness.

For normalization, we computed the mean and standard deviation over the entire raw training set (before applying augmentation), and used these values for all splits. This differs slightly from [7], where fixed values were used, but we found that dataset-specific statistics helped improve convergence and stability.

All experiments ran on a BSC cluster.

5 Experiments

We organize our experiments into three stages, as described in Section 4. In this section, we present results for **MaorNet**, followed by those for **ResNet-18**.

5.1 MaorNet

5.1.1 Stage 1: Initial Grid Search

We evaluated 18 configurations combining learning rates $\{10^{-4}, 10^{-3}, 10^{-2}\}$, batch sizes $\{128, 256, 512\}$, and initialization methods (Gaussian vs. He), using fixed momentum (0.9), dropout (0.5) and weight decay (0.0005). Each configuration was run three times with different seeds. Results were averaged.

We found that initialisation did not have much effect, but learning rate and batch size did. Figure 4 shows aggregated accuracy and loss curves grouped by learning rate and batch size (we do not show by initialisation as it provided no extra information). Learning rate 10^{-3} consistently yielded faster convergence and higher accuracy, while 10^{-4} underfit and 10^{-2} often became unstable. Smaller batch sizes (128 and 256) generalized better than 512, showing smaller train-validation gaps and lower final loss. We also observed signs of overfitting, for which we will adjust regularization parameters in Stage 3. Additionally, the network had not yet plateaued, so we plan to later increase the number of training epochs.

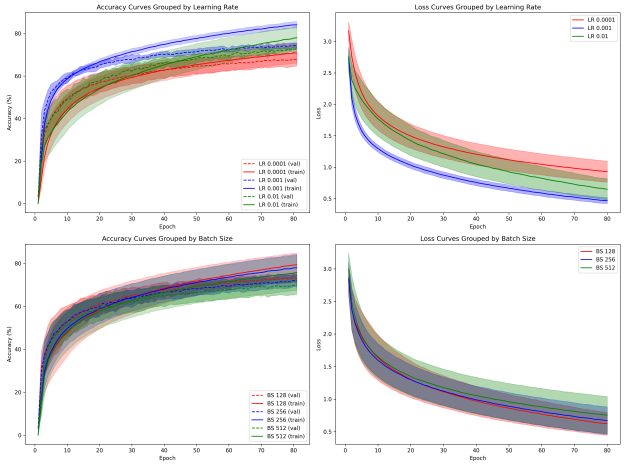

Figure 4: *Top: Accuracy/loss curves grouped by learning rate. Bottom: Accuracy/loss curves grouped by batch size. Solid lines: validation; dashed: training. Shaded areas indicate standard deviation across seeds.*

Table 1 summarizes the top five configurations (per validation accuracy). The best validation accuracy (75.77%) was achieved with learning rate 10^{-3}, batch size 128, and He weight initialisation,. Although some configurations had higher training accuracy, they showed larger generalization gaps; all of these show slight overfit, which we will try to control on Stage 3.

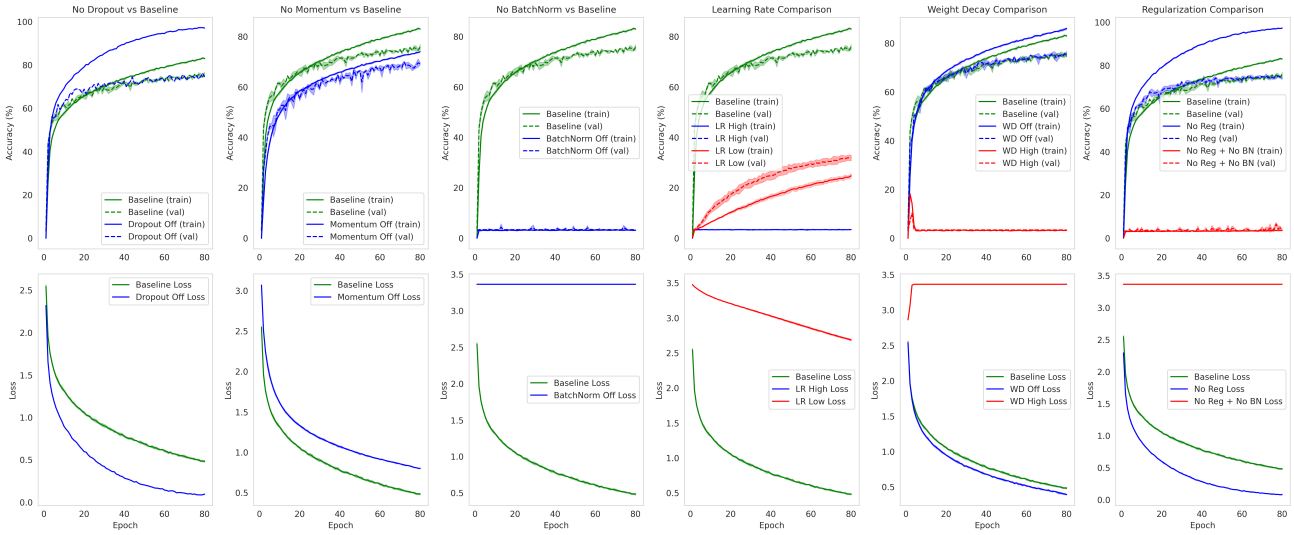

Figure 5: *Ablation study results for MaorNet. Each column compares the baseline (green) to modified configurations in terms of training/validation accuracy (top) and loss (bottom).*

Table 1: *Top MaorNet configurations from the grid search (mean ± std over 3 runs). η stands for Learning Rate, BS for Batch Size. "Gap" is defined as Train Acc minus Val Acc, quantifying the extent of overfitting.*

η	BS	Init	Val (%)	Train (%)	Gap (%)
10^{-3}	128	He	$\mathbf{75.77 \pm 0.67}$	83.13 ± 0.39	7.36
10^{-3}	128	Gaussian	75.47 ± 0.94	86.08 ± 0.24	10.61
10^{-3}	256	He	74.69 ± 0.52	84.95 ± 0.19	10.26
10^{-2}	256	Gaussian	74.44 ± 0.97	82.24 ± 0.54	7.80
10^{-3}	256	Gaussian	73.98 ± 0.55	84.50 ± 0.05	10.52

5.1.2 Stage 2: Ablation Studies

To assess the contribution of specific training components, we performed a series of ablation experiments using the best MaorNet configuration from Experiment 1. Figure 5 summarizes the results across six controlled scenarios.

Removing dropout resulted in a sharper increase in training accuracy, but validation accuracy plateaued earlier, leading to a larger generalization gap. Disabling momentum produced slower convergence and consistently lower accuracy on both training and validation sets, confirming its role in stabilizing updates.

The absence of batch normalization had the most dramatic effect: the model failed to learn meaningful representations, with both accuracy curves flattening near random chance, indicating strong underfitting. Learning rate sensitivity was also evident—using a very low learning rate led to minimal progress, while an excessively high one caused instability and diverging behavior, especially on the validation set.

Turning off weight decay increased training performance but harmed validation, again showing signs of overfitting. In contrast, using a very high weight decay prevented the model from fitting the data altogether. Finally, when all regularization (dropout and weight decay) was removed, validation accuracy degraded noticeably despite strong training performance. Removing batch normalization further worsened results, pushing the model into strongly underfitting.

These results confirm that dropout and weight decay are important for controlling overfitting in MaorNet, while batch normalization is essential for enabling learning. Momentum accelerates convergence and improves final performance, and proper learning rate selection is critical—extreme values can lead to either underfitting or divergence. Overall, the ablation studies validate the importance of regularization and training stability mechanisms in ensuring generalizable performance.

5.1.3 Stage 3: Refinement

Building on the Stage-1 winner (SGD, $\eta = 10^{-3}$, batch 128), we tightened the training recipe as follows. First, the time budget was stretched from 80 to *at most* 120 epochs while early-stopping (patience $= 10$ on the validation loss) guards against wasteful over-training. Second, whenever validation accuracy stalls for five consecutive epochs we now drop the learning rate by a factor of ten, yielding a coarse cosine-like decay without additional hyper-parameters. Third, regularisation was explored through a 4×3 grid: drop-out $p \in \{0.30, 0.40, 0.50, 0.60\}$ crossed with weight decay $w \in \{10^{-4}, 5 \times 10^{-4}, 10^{-3}\}$.
All other hyper-parameters were left untouched. Table 2 reports the most relevant runs (mean over three seeds).

Across the grid we observe that *moderate* regularisation on a single axis works best: raising p beyond 0.4 mainly cuts capacity, while pushing w to 10^{-3} lowers both train and validation curves; the sweet-spot appears when one knob is medium and the other mild.

We select $w = 10^{-4}$, $p = 0.40$ as the final MaorNet configuration. Although its 77.14% validation accuracy

Table 2: *Stage 3 results (validation metrics only). "Gap"=Train − Val. A standard deviation (Std) of 0.0 indicates that the value was obtained from an epoch where early stopping ended the other runs, and only one seed remained active.*

w	p	Val (%)	Train (%)	Gap (%)	Epochs
5×10^{-4}	0.40	$\mathbf{78.00 \pm 0.00}$	89.76 ± 0.00	11.76	72 ± 6.18
10^{-3}	0.30	77 ± 0.00	92.76 ± 0.00	15.11	76 ± 9.10
10^{-4}	0.40	77.14 ± 0.59	85.89 ± 0.20	**8.75**	71 ± 8.49
10^{-3}	0.40	76.62 ± 0.00	87.21 ± 0.00	10.59	83 ± 16.21
10^{-4}	0.60	76.34 ± 0.00	80.55 ± 0.00	4.21	78 ± 2.87

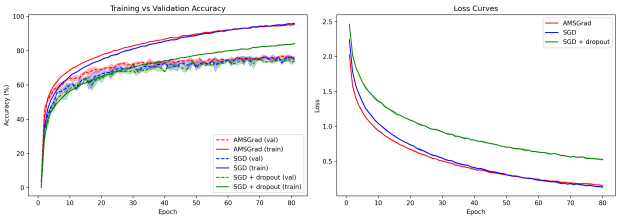

Figure 6: *ResNet-18, best runs for each experiment of Stage 1. Left: training–validation accuracy. Right:loss. Shaded areas show ±1 std across seeds.*

is $\approx 0.9\%$ shy of the absolute peak (78.0%), it trims the train–val gap to 8.8% while also avoiding the heavy capacity loss seen with $p \geq 0.5$ or $w \geq 10^{-3}$. Relative to Stage 1 the refined model lifts validation from 75.8%, a, increase of 1.34%. It maintains a similar val-train gap, and trains more efficiently (in less epochs).

We ran the chosen model thrice on the test set and obtained a final test accuracy of $77.91\% \pm 0.21$, confirming we did not overfit the validation set.

5.2 ResNet-18

5.2.1 Stage 1: Initial search

As with MaorNet we swept learning rates $\{10^{-4}, 10^{-3}, 10^{-2}\}$, batch sizes $\{128, 256, 512\}$ and both Gaussian and He initialisation, fixing weight-decay at 5×10^{-4} and training for 80 epochs. We initially omitted dropout—the canonical ResNet-18 does not use it [3]—and adopted **AMSGrad** as in the original MAMe paper [7]. The optimiser reached 76.7% validation but exhibited severe overfitting: gaps grew from 18% to 21% as the batch grew (Table 3, upper block).

Motivated by the idea that more gradient noise can act as implicit regularisation, we replaced AMSGrad with **SGD + momentum**. Unfortunately, Table 3 (middle block) shows the gap remains above 17%; we get however get, on average, better best validation accuracies, for which we opt to stick with SGD.

Adding dropout, in the way suggested by [5], finally controlled overfitting. The best run (75.3% val) slashed the gap to 5.5% while costing just one percentage point of validation accuracy (Table 3, bottom block). This "SGD + $p = 0.50$" setting is therefore adopted as the baseline for subsequent ablation and refinement. Figure 6 presents the training and validation accuracy, along with the corresponding loss curves, for the best-performing configurations of AMSGrad, SGD, and SGD with dropout.

As per overall trends we observed: interestingly, AMSGrad preferred higher lower learning initial learning rates than SGD—although this could be because it adapts it during training; larger batches appear widen AMSGrad's generalization gap more ($\approx 2\%$ BS 128 to 512); AMSGrad was not as sensitive to BS while in SGD we almost always find BS 128 on the top configurations; and AMSGrad seemed to prefer the He optimizer, although this was negligible for SGD.

5.2.2 Stage 2: Ablation studies

All experiments start from the *Stage-1 baseline* (SGD, $\eta = 10^{-2}$, momentum 0.9, batch 128, weight-decay 5×10^{-4}, dropout 0.5, BatchNorm on). Exactly one knob is changed at a time, except in the last column where every regulariser is disabled.

Removing dropout barely changes validation accuracy but almost doubles the train–validation gap, confirming its regularising effect. Disabling momentum slows convergence and costs about six percentage points on validation. Batch normalization surprisingly not as important: without it, the model overfit, but the effect was not as severe as we expected. The learning rate must sit in a narrow band; values of 10^{-6} under-fit, while 1 (obviously) sends training unstable. Weight decay shows a classic U-shape: turning it off increases overfitting, whereas setting it to 1 prevents learning altogether. It does seem to be less important, as with it set to 0 we still get decent validation results. Finally, eliminating *all* explicit regularisers hurts validation by 3–4 points despite higher training accuracy, and compounding this with the removal of BatchNorm catastrophically fails.

Comparing the results to MaorNet's, both architectures react similarly. BatchNorm is critical, dropout and weight-decay effectively control overfitting, momentum provides useful smoothing, and extreme learning rates derail optimisation. A small difference is that ResNet-18 tolerates the removal of dropout slightly better than MaorNet, hinting that residual connections already supply some implicit regularisation, whereas MaorNet relies more heavily on explicit dropout. Overall, the findings validate the design choices adopted for both networks.

5.2.3 Stage 3: Refinement

The Stage-1 grid search already held the *train–validation gap* to 5.5%, so we did not perform a grid search over regularisation parameters, and focused instead on *longer* and *smoother* training. Deeper residual networks are known to over-fit when

Table 3: *ResNet-18, Stage 1 — five best runs per optimisation strategy (mean ± std over 3 seeds), 'Gap' = Train - Val.*

Opt.	η	BS	Init	p	Val (%)	Train (%)	Gap (%)
AMSGrad	10^{-4}	128	He	–	76.74±0.81	94.55±0.16	17.82
	10^{-4}	256	He	–	76.37±0.42	96.40±0.22	20.03
	10^{-4}	512	He	–	75.56±0.48	96.64±0.17	21.08
	10^{-3}	128	He	–	73.48±0.70	93.27±0.38	19.79
	10^{-3}	256	Gauss	–	72.90±0.88	93.54±0.27	20.64
SGD	10^{-2}	128	He	–	76.23±0.66	95.60±0.41	19.37
	10^{-2}	128	Gauss	–	76.09±0.96	95.38±0.29	19.29
	10^{-2}	256	He	–	75.89±0.82	95.74±0.55	19.85
	10^{-3}	128	Gauss	–	75.72±0.55	92.95±0.21	17.22
	10^{-3}	128	He	–	75.49±0.37	94.42±0.06	18.92
SGD + dropout	10^{-2}	128	He	0.5	75.26±0.82	80.80±0.04	**5.54**

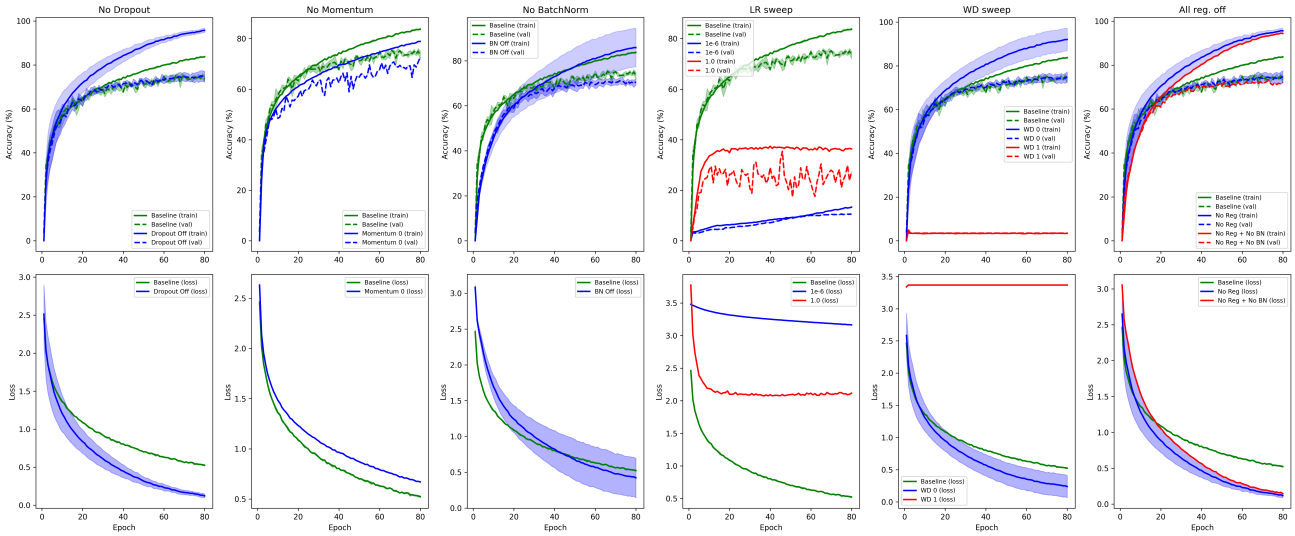

Figure 7: *ResNet-18 ablations (three seeds per curve). Top row: training / validation accuracy. Bottom row: cross-entropy loss. Shaded areas denote ±1 standard deviation.*

the epoch budget is extended (see, *e.g.*, the discussion in [1]), therefore we introduced some changes before increasing training length:

1. Label smoothing (0.1) to soften the targets [11];
2. a 5-epoch linear warm-up (from 2.5×10^{-8} to 0.25) followed by cosine annealing down to 0 at the end of training [2];
3. a lighter weight-decay ($w = 10^{-4}$), since the two previous techniques already act as strong implicit regularisers, and we saw on Stage 2 that weight decay was not *that important* for ResNet-18.

We first kept *early stopping* (patience = 10). It halted training after ≈ 76 epochs (not exact because we ran three seeds), just when the cosine tail was starting to converge. We believed the changes we introduced would permit more training, so we eliminated early stopping for a second experiment. Running the full 200 epochs allowed the optimiser to make small, useful refinements for another 120 epochs, pushing the validation accuracy up by almost four percentage points with *no increase in the generalisation gap*. Table 4 summarises the two runs averaged over three seeds.

Table 4: *Stage-3 results (three seeds). "Gap" = Train−Val.*

	Epochs	**Val (%)**	**Train (%)**	**Gap (%)**
Early Stop	75.7 ± 1.5	78.07 ± 0.51	86.77 ± 0.85	8.70
Full Run	200.0	$\mathbf{82.18 \pm 0.43}$	90.34 ± 0.26	8.16

Early stopping protected us from wasting compute, but it also clipped off the slow "refinement" phase of training. The full run adds ∼ 120 extra epochs, yet the train–val gap remains **constant** (about 8.2%), indicating that the additional optimisation steps improved both training and validation performance without causing extra overfitting. For this architecture and data regime, the cost of those extra epochs (roughly +120% more GPU-hours) translates into a sizeable **+4.1%** absolute gain in validation accuracy.

To confirm generalisation, we ran the final model on the test set three times, achieving $81.24\% \pm 0.23$, consistent with validation and showing no overfitting.

6 Discussion

We now benchmark our *test–set, top-1* results against the baselines of Parés et al. [7], contrast MaorNet and ResNet-18, and summarise the ablation take-aways. We close the section with limitations.

How do we fare against the published baselines? The Table 4 in Parés et al. [7] reports **83.33 %** top-1 for a *pre-trained* ResNet-18 and **81.35 %** for a *pre-trained* VGG-11 (R65k, FS) Table 5 juxtaposes those numbers with our scratch-trained results. Removing ImageNet weights costs **2.1** $\%$ on ResNet-18 and **3.4**$\%$ on the Standard Architecture, *i.e.*, we still recover $\approx 92\,\%$ and $\approx 89\,\%$ of the "transfer-learning bonus" without external data.

Table 5: *Test-set top-1 accuracy. "TF" indicates ImageNet transfer (**True**) or scratch (**False**); "Type": Standard (S) vs. Non-Standard (NS).*

Model	Type	TF	Params	Test (%)
MaorNet (ours)	S	False	30.4 M	77.91 ± 0.21
VGG-11 (paper)	S	True	132.9 M	81.35
ResNet-18 (ours)	NS	False	11.2 M	81.24 ± 0.23
ResNet-18 (paper)	NS	True	11.2 M	83.33

Equally striking is the efficiency gap: MaorNet uses *one-quarter* the parameters of the VGG-11 baseline, and our 11 M-parameter ResNet-18 matches the 133 M-parameter VGG within two points—underscoring that architectural design, not sheer size, is the main lever on MAME.

MaorNet *versus* ResNet-18. Although both models are trained from scratch, their parameterisation is radically different. MaorNet contains 30.4 M weights, of which 25.7 M $(84.6\,\%)$ lie in the two fully–connected layers; by contrast, ResNet-18 holds only 11.2 M weights, relies on global–average pooling and therefore spreads capacity more evenly across its residual blocks. This structural choice translates into better training: the residual network continues to gain accuracy for the full 200-epoch budget, whereas MaorNet saturates after roughly 60 epochs. It is however less robust to the removal of explicit regularisation. The net result is a test accuracy of $81.24 \pm 0.23\%$ for ResNet-18 against $77.91 \pm 0.21\%$ for MaorNet; a Welch unequal-variance t-test on the three matched seeds gives $t = 24.71$ and $p = 2.43 \times 10^{-4}$, confirming the 3.33-point advantage of ResNet-18 as statistically significant. Welch's test is appropriate here due to small sample size and potentially unequal variance. The effects parameters had in them was also different. We were very surprised to see BatchNorm having so little effect on ResNet-18, when it was indispensable for MaorNet's learning. The network was also more resilient to other parameter choices, such as an inadequate learning rate, *etc.*. In short, residual connections plus a slim head yield a model that is both more parameter-efficient and better-generalising than a Standard CNN Architecture.

Which parameters really mattered?

- **Batch Normalisation** is indispensable for MaorNet, (removing it resulted in severe underfitting), but was less important dropped both models to chance (severe underfitting).
- **Learning rate window** for SGD is tight $\{10^{-3}, 10^{-2}\}$; outside that range we saw either underfit or divergence.
- **Either** dropout *or* weight decay sliced $7 - 10\%$ off the train–test gap; using both improved at most one point further.
- **Optimiser.** AMSGrad hit the highest training accuracy but left a 19-21% generalisation gap; SGD + momentum plus a cosine schedule halved that gap and finished higher on val.

Limitations and future work. Our experimental design has two main caveats. First, the some training tricks employed for ResNet-18—five-epoch warm-up, cosine learning-rate decay and label smoothing—were not used for MaorNet; it is true that different architectures have different training requirements, but this opens the question on whether this impacted the improved performance. Second, the hyper-parameter exploration was modest to keep GPU time reasonable; a wider search could reveal stronger configurations for both architectures. Looking ahead, it would be interesting to extend the study to alternative backbones such as ConvNeXt, vision transformers or hybrid CNN-ViT models, and to revisit the task under a transfer-learning scenario, starting from ImageNet or from self-supervised models trained on the museum images themselves.

7 Conclusion

Our central question—*"how much does architecture matter when training from scratch on MAMe?"*—is answered in the affirmative: network design makes a decisive difference. With the same 256×256 inputs and no external data, an 11 M-parameter "Non-Standard CNN" ResNet-18 reaches **81.2 %** test accuracy, beating the 30 M-parameter "Standard CNN" MaorNet by 3.3% while recovering roughly 90 % of the transfer-learning boost reported for ImageNet-initialised models by [7]. Ablations confirm that Batch Normalisation is non-negotiable and that a single regulariser—dropout *or* weight decay—can shrink the train–test gap by up to 10%. Future work should broaden the hyper-parameter search, stabilise modern training tricks for Standard Architectures, and test newer backbones or self-supervised pre-training, but the present results already show that thoughtful architectural choice can offset much of the advantage usually ascribed to large-scale pre-training.

Bibliography

[1] Irwan Bello et al. *Revisiting ResNets: Improved Training and Scaling Strategies*. 2021. arXiv: 2103.07579 [cs.CV]. URL: https://arxiv.org/abs/2103.07579.

[2] Priya Goyal et al. *Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour*. 2018. arXiv: 1706.02677 [cs.CV]. URL: https://arxiv.org/abs/1706.02677.

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

[4] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: 1502.03167 [cs.LG]. URL: https://arxiv.org/abs/1502.03167.

[5] Bum Jun Kim et al. "How to use dropout correctly on residual networks with batch normalization". In: *Proceedings of the Thirty-Ninth Conference on Uncertainty in Artificial Intelligence*. UAI '23. Pittsburgh, PA, USA: JMLR.org, 2023.

[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[7] Ferran Parés et al. "The MAMe Dataset: On the relevance of High Resolution and Variable Shape image properties". In: *arXiv preprint arXiv:2007.13693* (2020).

[8] Farheen Ramzan et al. "A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks". In: *Journal of Medical Systems* 44.2 (2019), p. 37. ISSN: 1573-689X. DOI: 10.1007/s10916-019-1475-2. URL: https://doi.org/10.1007/s10916-019-1475-2.

[9] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV]. URL: https://arxiv.org/abs/1409.1556.

[10] Matthew D Zeiler and Rob Fergus. *Visualizing and Understanding Convolutional Networks*. 2013. arXiv: 1311.2901 [cs.CV]. URL: https://arxiv.org/abs/1311.2901.

[11] Chang-Bin Zhang et al. "Delving Deep Into Label Smoothing". In: *IEEE Transactions on Image Processing* 30 (2021), pp. 5984–5996. ISSN: 1941-0042. DOI: 10.1109/tip.2021.3089942. URL: http://dx.doi.org/10.1109/TIP.2021.3089942.